Dynamische elektronische Bücher

Dynamische elektronische Bücher

Untersuchungen zur Modellierung wissenschaftlicher
Lehrwerke als elektronische Publikationen am Beispiel eines
Lehrbuchs der Experimentalphysik

Der Fakultät für Mathematik und Informatik
der Universität Leipzig
eingereichte

Habilitationsschrift

zur Erlangung des akademischen Grades

doctor philosophiae habilitatus

(Dr. phil. habil.)

vorgelegt
von

Dr. phil. Christian Wolff

geboren am 14. September 1966 in München

Leipzig, den 3. April 2000

Inhaltsübersicht

Inhaltsverzeichnis

1 Einleitung

Die vorliegende Arbeit befasst sich im Rahmen des Gegenstandsbereichs *elektronisches Publizieren* mit der Entwicklung dynamischer elektronischer Bücher. Unter dynamischen elektronischen Büchern versteht man Hypermedia-Informationsbestände und die mit ihnen verbundenen Softwaresysteme, die interaktive Komponenten enthalten und durch Informationsdienste dynamisch ergänzt werden können. Dieses Konzept unterscheidet sich vom state-of-the-art elektronischer Bücher als digitalen Entsprechungen gedruckter Werke in drei Punkten:

1. Die Inhalte des elektronischen Buchs werden mit Hilfe von Standards auf der Basis der *Extensible Markup Language* (XML) ausgezeichnet. Dabei erfolgt eine Unterscheidung verschiedener Beschreibungsebenen (Struktur, Inhalte, Metadaten).
2. Dynamische elektronische Bücher enthalten interaktive multimediale Komponenten wie Animationen und Simulationen, die zusätzliche Nutzungsformen und Lernstrategien ermöglichen.
3. An ein dynamisches elektronisches Buch können *Dienste* angeknüpft sein. Durch sie werden die textuellen und multimedialen Inhalte des Buchs dynamisch erweitert, d. h. dem Benutzer stehen durch Zuordnung von Diensten an die deklarativ repräsentierten Inhalte des Buchs zusätzliche Informationsquellen und Weiterverarbeitungsmöglichkeiten zur Verfügung.

Im Zentrum dieser Arbeit steht die Frage, wie dynamische elektronische Bücher aus einer *leserorientierten Perspektive* zu gestalten sind. Aufgrund der Heterogenität des Gegenstands wird ein holistischer Ansatz verfolgt, der einerseits einen Überblick über verfügbare Technologien (Standards, Implementierungswerkzeuge etc.) gibt und andererseits versucht, ein Modell dynamischer elektronischer Bücher anhand eines Prototypen umzusetzen. Die Arbeit entstand im Umfeld des Forschungsprojektes *Multimediales Physikalisches Praktikum*, das im Rahmen des Verbundvorhabens *Weiterentwicklung des wissenschaftlich-technischen Lehrbuchs zur multimedialen Wissensrepräsentation* des Bundesministeriums für Bildung, Wissenschaft, Fohrschung und Technologie (BMB+F)[1] in den Jahren 1997-1999 in Kooperation mit

- der Fakultät für Physik und Geowissenschaften der Universität Leipzig (Prof. GESCHKE),
- dem Oldenburger Kuratorium OFFIS e. V. (Prof. APPELRATH) und
- dem B. G. Teubner Verlag, Stuttgart und Leipzig

an der Abteilung Automatische Sprachverarbeitung des Instituts für Informatik der Universität Leipzig (Prof. HEYER) durchgeführt wurde. Die in diesem Projekt erarbeiteten Ergebnisse bilden als *Anwendungsbeispiel* den Hintergrund dieser Arbeit und sind in die Konzeption dynamischer elektronischer Bücher eingegangen. Das Projekt wird nachfolgend als *Referenzprojekt* bezeichnet.

---

[1] Betreut vom Projektträger Fachinformation (PTF), vgl. *http://www.darmstadt.gmd.de/PTF/Ausschreibungen/Multimed.htm* und unten Kap. 3.1.2; Förderkennzeichen 08C5837 0 (Leipzig) bzw. 08C5832 0 (Stuttgart/Oldenburg).

Das elektronische Publizieren (EP) ist ein Arbeitsfeld, für das nicht nur verschiedene Teilgebiete der Informatik, sondern auch andere Disziplinen einen Beitrag liefern. Das elektronische Publizieren stellt Technologien und Verfahren bereit, die für eine Vielzahl von Anwendungen (*Hypermediasysteme, Distance Learning, Web Publishing*) von Bedeutung sind. Die Entwicklung und Realisierung eines Modells dynamischer elektronischer Bücher hat daher zahlreiche Berührungspunkte mit Themengebieten der praktischen und angewandten Informatik.

Ausgehend von den im Referenzprojekt gesammelten Erfahrungen lassen sich die wesentlichen Merkmale dynamischer elektronischer Bücher herausarbeiten. Dabei spielt der Aspekt einer generalisierbaren Architektur für elektronische Bücher eine zentrale Rolle. Sie umfasst auch eine Klassifikation elektronischer Dienste, die für unterschiedliche elektronische Bücher anwendbar ist. Einige Beispiele aus dem Kontext des Referenzprojekts sollen dies verdeutlichen:

- Das Verständnis eines physikalischen Phänomens lässt sich durch *Simulationen* unterstützen, mit denen der Benutzer bei der Lektüre des dynamischen elektronischen Buchs arbeiten kann.
- Die Durchführung physikalischer Versuche kann dem Benutzer mit Hilfe von interaktiven multimedialen *Animationen* verdeutlicht werden. Sie dienen dadurch der vorbereitenden Unterstützung der Lehrveranstaltung.
- Das Internet und insbesondere das World Wide Web (WWW) bieten eine Vielzahl von Diensten an, die gezielt an die Inhalte eines dynamischen elektronischen Buchs geknüpft werden können. Z. B. kann die Integration von Suchmaschinen in das dynamische elektronische Buch dem Benutzer bei der Ermittlung relevanter Inhalte helfen.
- Durch die Speicherung zusätzlicher Informationen, wie Anmerkungen oder Verknüpfungen zu Ressourcen im World Wide Web, passt der Benutzer das dynamische elektronische Buch für sich an.
- Der Einsatz eines elektronischen Lehrbuchs im Rahmen einer Lehrveranstaltung kann dadurch unterstützt werden, dass zusätzliche *Inhalte* (Texte, multimediale Komponenten) oder *Dienste* (E-Mail-Adresse eines Tutors, Speicherungsdienst für Aufgabenlösungen) nur für die Teilnehmer der Lehrveranstaltung bereitgestellt und in das dynamische elektronische Buch eingebunden werden.

Einige methodische Prämissen und technologische Randbedingungen haben die Entstehung dieser Arbeit maßgeblich beeinflusst: In methodischer Hinsicht steht die Überlegung im Vordergrund, dass komplexe Informationssysteme wie elektronische Bücher unter umfassender Berücksichtigung von *Standards* entwickelt werden sollten. Es werden vor allem Standards und Standardisierungsvorschläge aus dem Umfeld des *World Wide Web Consortium* (W3C), der *Internet Engineering Task Force* (IETF) sowie der Programmiersprache Java einbezogen bzw. vorgestellt. Ausgehend von dieser Überlegung haben drei technologische Gesichtspunkte diese Arbeit beeinflusst (vgl. RÖSNER 1999: 90 f.):

a) Die Entwicklung des World Wide Web zu einer *universellen Softwareplattform*, die sich zur Realisierung von Informationssystemen im Allgemeinen und elektronischen Büchern im Besonderen eignet und für die eine entsprechende Softwareinfrastruktur verfügbar ist (Server; Browser; Middleware).
b) Die Ausdifferenzierung der auf der *Standard Generalized Markup Language* (SGML) basierenden Standards für die Strukturierung, Transformation und Präsentation von Dokumenten (insbesondere XML und Tochterstandards).

c) Die Weiterentwicklung der objektorientierten Programmiersprache Java (vgl. ARNOLD & GOSLING 1997, WOLFF 1999A) zu einer Implementierungsplattform, die sich für ein dienstebasiertes dynamisches elektronisches Buch eignet.

SGML-basierte Standards und die Infrastruktur von Java bilden zusammen mit WWW-Technologien das Handwerkszeug zur praktischen Umsetzung der in dieser Arbeit entwickelten Konzepte. Sie reflektieren in *technischer* Hinsicht das *konzeptuelle* Ziel, dynamische Bücher als Anwendungen im Schnittfeld der in der Literatur oftmals getrennt diskutierten Bereiche *elektronisches Publizieren* und *interaktive Multimediaanwendungen* zu verstehen.

Die Untersuchung gliedert sich in drei Abschnitte: Teil I (2-4) führt in den Gegenstandsbereich *elektronisches Publizieren* ein, zeigt den state-of-the-art elektronischer Bücher auf und entwickelt darauf aufbauend ein Modell für dynamische elektronische Bücher. Teil II (Kap. 5-9) gibt einen Überblick über deklarative Standards der Informationsauszeichnung. In Teil III (Kap. 10-15) wird beschrieben, wie sich das Modell dynamischer elektronischer Bücher auf der Basis solcher Standards umsetzen lässt.

Das nachfolgende Kapitel 2 stellt das Arbeitsgebiet *elektronisches Publizieren* vor und ordnet elektronische Bücher in den Kontext von Hypertextsystemen, Multimediaanwendungen und Lehr- und Lernsystemen ein. Ein Überblick über den gegenwärtigen Stand der Forschung zu elektronischen Büchern erfolgt unter Bezug auf den Projektverbund, in dem das in dieser Arbeit diskutierte Referenzprojekt entstand (Kap. 3). Darauf aufbauend wird ein Kriterienraster für die Beschreibung elektronischer Bücher erarbeitet. Kap. 4 stellt das Konzept dynamischer elektronischer Bücher dar und entwickelt seine wesentlichen inhaltlichen und funktionalen Anforderungen. Dabei entsteht ein dienstebasiertes Architekturmodell. Dieses Modell wird auf einer allgemeinen Ebene eingeführt und schließt den ersten Teil der Arbeit ab.

Der zweite Teil der Arbeit ist der Problematik der Strukturierung und Beschreibung von Information in elektronischen Büchern gewidmet. Er gibt unter Betonung von Standards auf der Basis der *Standard Generalized Markup Language* einen Überblick unterschiedlicher Standards der Informationsmodellierung. Dabei orientiert sich die Darstellung an der *Funktion* der verschiedenen Sprachen und ihrer Anwendung:

- Kap. 5 stellt SGML und XML als Metasprachen der Informationsstrukturierung vor,
- Kap. 6 diskutiert ihre Anwendung in standardisierten Dokumentgrammatiken,
- die Transformation und Präsentation deklarativ kodierter Information ist Gegenstand von Kap. 7,
- die Problematik der Aufbereitung multimedialer Information erörtert Kap. 8 und
- schließlich wird in Kap. 9 der Einsatz von Metadaten hinsichtlich geeigneter Formalismen und anwendungsbezogener Schemata beschrieben.

Die Verwendbarkeit der verschiedenen Standards wird mit Blick auf das Referenzprojekt diskutiert und dient als Grundlage für die Anwendung des in Kap. 4 entwickelten Modells dynamischer elektronischer Bücher.

Der dritte Teil der Arbeit stellt ausgehend von dem im Referenzprojekt entwickelten Prototyp eines elektronischen Buchs die Umsetzung des in Kap. 4 beschriebenen Architekturmodells vor. Dabei kommen die in Teil II dargestellten Standards und Werkzeuge zur Anwendung. Kap. 10 konkretisiert die in Kap. 4.2 entwickelte Methode der Informationsstrukturierung. Anschließend führt Kap. 11 eine Auswahl relevanter Entwicklungswerkzeuge ein. Den Aufbau des Buchverwaltungssystems, wie ihn das Konzept dynamischer elektronischer Bücher voraussetzt, diskutiert Kap. 12. Das Buchbetrachtungssystem als Gegenstück zum Buchverwaltungssystem wird unter Berücksichtigung software-

ergonomischer Überlegungen in Kap. 13 vorgestellt. Schließlich erörtert Kap. 14 die Integration von Komponenten und Diensten in dynamische elektronische Bücher an Beispielen. Besonderes Augenmerk gilt dabei der Informationserschließung und ihrer Grundlagen im Information Retrieval. Kapitel 15 schließt die Arbeit mit Überlegungen zur Generalisierbarkeit des hier vorgelegten Ansatzes ab. Dabei wird auch untersucht, wie sich dynamische elektronische Bücher in digitale Bibliotheken integrieren lassen.

Teil I: Elektronische Publikationen und elektronische Bücher

2 Elektronisches Publizieren

Das elektronische Publizieren umfasst alle Prozesse und Verfahren, bei denen Publikationen mit Unterstützung durch Rechentechnik erstellt, aufbereitet, präsentiert oder distribuiert werden und damit sowohl die Unterstützung der traditionellen Publikationsformen (gedruckte Bücher, Zeitschriften etc.) als auch elektronische Publikationen (vgl. RIEHM et al. 1992, SANDKUHL & KINDT 1996: 45 ff.). Als Konzeption für die Aggregation und Distribution menschlichen Wissens kann man eine Vorwegnahme wesentlicher Grundgedanken des elektronischen Publizierens in visionären Werken wie H. G. WELLS' *World Brain*, das den Entwurf einer *Permanent World Encyclopedia* enthält (vgl. WELLS 1938, RAYWARD 1999: 560f.) oder Vannevar BUSHs Konzeption des universellen Informationssystems *Memex* (vgl. BUSH 1945, KLING & LAMB 1996: 26 ff.) finden.

Als Instrument der Kommunikation in der Informationsgesellschaft bzw. der Informationswirtschaft kommt dem elektronischen Publizieren eine große Bedeutung zu (vgl. ENDRES et al. 1999). EISENBERG 1994: 64 fasst die Möglichkeiten elektronischen Publizierens in ihren Auswirkungen auf die Gesellschaft wie folgt zusammen:

> From an electronic publishing view, we will increasingly see changes in:
>
> - individual behaviors, particularly in terms of information seeking, access, and use;
> - group behaviors including new forms of electronic network-decision-making, and cooperative learning; and
> - institutional structures and behaviors as schools, businesses, libraries, and others move beyond the wall and clock.

Als Schlüsselbegriff für die Kennzeichnung elektronischer Publikationen wird vielfach ihr *informationeller Mehrwert* angeführt; von einem *value-added publishing* erwartet man entscheidende Unterschiede zum traditionellen Publizieren (BERGHEL 1999), wobei die konkrete Realisierung auf unterschiedliche Arten erfolgen kann, wie die nachfolgenden Beispiele (nach BERGHEL 1999: 20f.) zeigen:

- Review-Systeme für elektronische Publikationen
- Integration von bibliometrischen Wertungsindikatoren (Zitierhäufigkeit, Auszeichnungen, Empfehlungen) in elektronischen Publikationen
- Integration von Such- und Indexierungswerkzeugen
- Anwendung von Algorithmen für inhaltliches Dokumentenclustering
- Versionsverwaltung für dynamische Dokumentinhalte
- Speicherung und Dokumentenlieferung in digitalen Bibliotheken
- Virtualisierung des Autoren- und Publikationsprozesses durch Kooperation und dynamische Dokumentgenerierung
- Individualisierung von Publikationsprodukten

Die Realisierung eines Mehrwerts durch elektronische Publikationen lässt sich nicht auf einzelne Komponenten oder Verfahren reduzieren. Als kleinster gemeinsamer Nenner liegt ihnen aber die Idee der Dynamisierung nicht nur der Publikationsinhalte, sondern auch der mit ihnen assoziierten Kommunikationsvorgänge zugrunde. Eines der zentralen Anliegen dieser Arbeit ist es, aufzuzeigen, mit welchen Methoden, nach welchen Stan-

dards und in welcher Infrastruktur dynamische elektronische Bücher entwickelt werden können.

Aus der Perspektive des Endprodukts kann man zwischen *computer aided publishing* (CAP) oder *desktop publishing* (DTP, vgl. PAPE & SANDKUHL 1990) einerseits und elektronischem Publizieren als Erstellung *elektronischer Publikationen* andererseits unterscheiden.[2] Ein wesentlicher Gesichtspunkt des elektronischen Publizierens im traditionellen Sinn ist die Betonung der Aspekte, die es ermöglichen, aus einem gegebenen Datenbestand in unterschiedlichen – elektronischen wie papiergebundenen – Medien publizieren zu können. In diesem Sinn ist das elektronische Publizieren ein zusammenfassender Begriff für den langfristigen technologischen Wandel im Druck- und Verlagswesen. Dies betrifft vor allem die *medienneutrale Datenhaltung* z. B. des Datenbestands für ein Lexikon, das sowohl im Druck als auch elektronisch publiziert werden kann (vgl. HEYER 1995). Allgemein spricht man in solchen Fällen von *cross-media publishing* (vgl. KAMPS et al. 1999). In den letzten Jahren haben sich die Schwerpunkte des elektronischen Publizierens allerdings verstärkt in Richtung auf elektronische Publikationen verlagert, wenn auch nach wie vor eine 1:1-Abbildung von Printpublikationen im elektronischen Medium typisch ist. Dies hat schon BLACKWELL 1983: 288 prognostiziert: „For the time being, as we see it, electronic publishing will provide mainly electronic versions of existing publications, with only a limited number of experiments; tradition and parallel publishing should dominate much of the scene for several years." Gliedert man die einzelnen Prozessschritte des elektronischen Publizierens in die

- Produktion von Publikationen,
- ihre Distribution und
- ihre Präsentation (vgl. RIEHM et al. 1992: 9),

so betrifft die Unterscheidung zwischen papiergebundenen und elektronischen Publikationen im *traditionellen Sinn* vor allem die Produktion, weniger die Präsentation und Distribution: In vielen Fällen handelt es sich bei elektronischen Publikationen lediglich um elektronische Pendants gedruckter Fassungen, für die im elektronischen Medium ein zusätzlicher Präsentations- und Distributionskanal erschlossen wird, ohne aber Konzeption, Umfang oder Strukturierung für das elektronische Medium signifikant zu modifizieren. Hierfür kann man eine Reihe von Beispielen anführen:

- die Präsentation und Distribution wissenschaftlicher Literatur über elektronische Bibliotheken[3],
- die Zusammenfassung umfangreicher Publikationsbestände von Zeitungen auf CD-ROM,
- die Publikation elektronischer Lexika, die sich von der jeweiligen papiergebundenen Ausgabe nur durch verbesserte Suchfunktionalität und die Einführung von Hypertextverknüpfungen unterscheiden.

---

[2] Im Umfeld des elektronischen Publizieren gibt es eine Vielzahl verwandter Begriffsbildungen z. B. *AutoPublishing*, *Intermedia Transmutation* oder *Publications Processing*, vgl. RESSLER 1993: 2; zu den Bedeutungsvarianten des Publikationsbegriffs (bekanntmachen vs. ein Printprodukt distribuieren) vgl. MORROW & MUMFORD 1994, bes. S16f.

[3] Die digitalen Bibliotheken von Fachgesellschaften im Bereich der Informatik geben ein gutes Beispiel ab, etwa die ACM oder IEEE *digital library* (vgl. *http://www.acm.org/dl* bzw. *http://www.computer.org/publications/dlib/*), die die von den Gesellschaften herausgegebenen Fachzeitschriften im elektronischen Format anbieten und durch Recherchemöglichkeiten (strukturierte Suche bzw. Volltextrecherche) erschließen.

Zur Unterstützung einer solchen Vorgehensweise stehen eine Vielzahl geeigneter Datenformate wie das *Portable Document Format* (PDF) der Fa. Adobe zur Verfügung. Aus der Sicht des *Nutzers* ist in solchen Fällen davon auszugehen, dass vor allem der Distributions- und weniger der Präsentationsaspekt den Unterschied zur traditionellen Publikation konstituiert, da in vielen Fällen vor der eigentlichen Nutzung ein Ausdruck z. B. eines Aufsatzes angefertigt wird.

Für ein Konzept dynamischer elektronischer Bücher steht im Unterschied zu solchen elektronischen Publikationsprodukten der Gedanke im Vordergrund, durch *quantitative* wie *qualitative Erweiterungen* zu neuen Publikationsformen zu gelangen, die für den Benutzer einen substantiellen Mehrwert bedeuten können. Bevor versucht wird, einen Kriterienkatalog für die Kategorisierung elektronischer Publikationen zu entwickeln, werden nachfolgend Modelle für das elektronische Publizieren betrachtet.

2.1 Modellbildungen für das elektronische Publizieren

Ausgehend von der Grobaufteilung in Produktion, Präsentation und Distribution stellen RIEHM et al. 1992: 2 ff. eine Reihe von Prozessmodellen für das elektronische Publizieren vor, wie die elektronische Erstellung druckfertiger Satzvorlagen durch Autoren oder die Mehrfachverwertung elektronisch gespeicherter Inhalte im Verlag. Sie ordnen die Beteiligten bzw. die Prozessstufen schematisch folgenden vier Bereichen zu:

- Autor
- Verlag
- Distribution
- Benutzer

Abbildung 1 zeigt schematisch die Prozesskette für die Erstellung elektronischer Satzvorlagen:

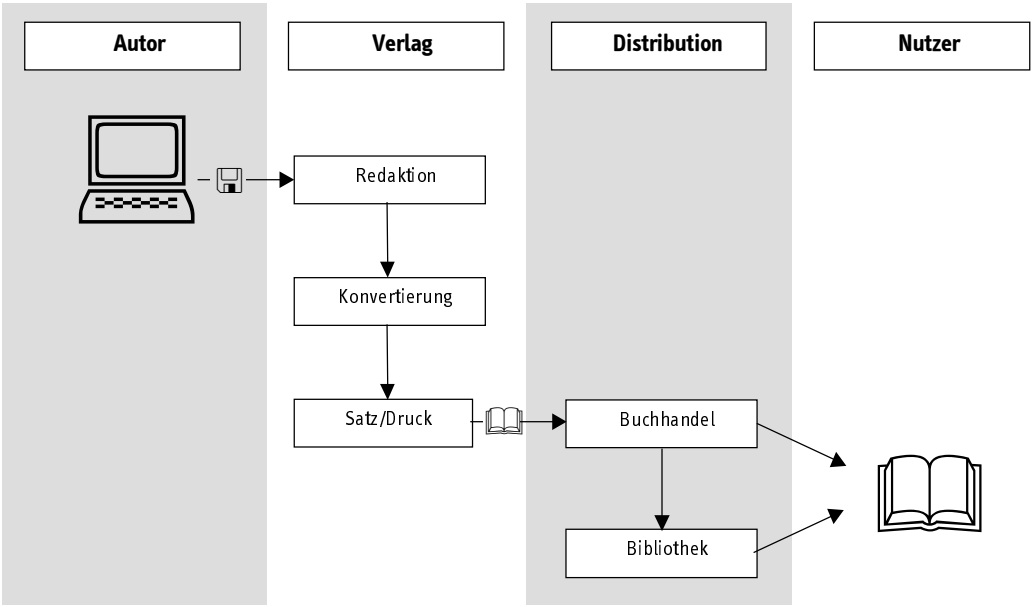

Abbildung 1: Elektronische Satzvorlagenerstellung als klassisches Beispiel elektronischen Publizierens (nach RIEHM et al. 1992: 2, Abb. 1)

Im Rahmen dieses Modell sind zahlreiche Aufgaben bei der Erstellung einer Publikation von Verlag oder Setzerei auf den Autor verlagert: Es wird erwartet, dass der Autor in der Lage ist, eine druckfertige Vorlage seiner Publikation mit – meist recht einfachen technischen Mitteln – selbst zu erstellen. Dieses Modell stößt es in Hinsicht auf elektronische Bücher dort an Grenzen, wo zu den Arbeitsschritten der Vorlagenerstellung die vielfältigen konzeptuellen wie technologischen Anforderungen der Hyper- bzw. Multimediatechnologie hinzukommen. Diese sind nur in den wenigsten Fällen vom einzelnen Autor zu bewältigen.

Ein differenzierteres Modell für die Prozesse im elektronischen Publizieren stellen Sandkuhl & Kindt mit dem *Referenzmodell Telepublishing* vor, das am Fraunhofer-Institut für Software- und Systemtechnik (ISST) entwickelt wurde (vgl. SANDKUHL & KINDT 1996, v. a. Kap. 8). Dieses Modell kann als wohl ausführlichstes Entwicklungsschema gelten. Es umfasst wie die Modellbildungen von RIEHM et al. 1992 grundsätzlich alle Prozesse und Produkte des elektronischen Publizierens. SANDKUHL & KINDT geben als Ziel die Ordnung der am elektronischen Publizieren beiteiligten komplexen Prozesse bzw. heterogenen Produkte und Dienste an, die durch das Modell beschrieben, erklärt und strukturiert werden sollen; aus dem Modell sollen unmittelbar Implementierungsstrategien und Migrationskonzepte ableitbar sein. Dazu ist es in ein Architekturmodell (Strukturen) und ein Phasenmodell (Prozesse) gegliedert (vgl. Tabelle 1).

Architekturmodell	*Phasenmodell*
Dienstetypen	Informationsmodellierung
Anwendungen	Prozessmodellierung
Businessmodelle	Wertschöpfung

Tabelle 1: Strukturierung von Architektur- und Phasenmodell

Das Architekturmodell unterscheidet die folgenden Dienstetypen:

- *Teledienste*: Generische elektronische Basisdienste wie Internetprotokolle (z. B. *file transfer protocol* (FTP), *telnet*, das *World Wide Web*, vgl. SCHELLER et al. 1994), deren Anwendung nicht auf das elektronische Publizieren beschränkt ist,
- *Telepublishing-Dienste*, die für das elektronische Publizieren spezifisch sind (z. B. SGML-Datenbanken oder Satzsysteme),
- *Telepublishing-Anwendungen*, d. h. komplexe Softwareanwendungen im Bereich des elektronischen Publizierens wie *Printing-on-Demand*, *Database-Publishing* oder *Just-in-Time-Publishing* (vgl. SANDKUHL & KINDT 1996: 278) und
- *Businessmodelle*, d. h. Geschäftsmodelle für das elektronische Publizieren.

Das Architekturmodell lässt sich durch die Herausarbeitung von Diensteprofilen für einzelne elektronische Anwendungen systematisch konkretisieren; es dient als Leitfaden bei der Erarbeitung eines Szenarios für die Erstellung elektronischer Publikationen. Am Beispiel eines elektronischen Buchs ist demnach zu klären,

- welche Basisdienste erforderlich sind,
- welche spezifischen EP-Dienste genutzt werden müssen,
- welche Anwendungen des elektronischen Publizierens benötigt werden,
- wie ein geeignetes Geschäftsmodell aussehen könnte.[4]

[4] Der Dienstebegriff des Telepublishing-Modells bezieht sich auf elektronische Dienste für die *Produktion* und *Distribution* elektronischer Publikationen, während das im Rahmen dieser Arbeit entwickelte Konzept der Ergänzung elektronischer Bücher durch Dienste (vgl. unten Kap.

Wichtiger als diese generische Sicht erscheint das *Phasenmodell* für das elektronische Publizieren, das mit einem Informationsmodell verbunden ist und jeden Einzelschritt des Publikationsprozesses durch einen festen Kriterienkatalog näher beschreibt. Abbildung 2 zeigt den Zusammenhang der einzelnen Phasen und der dabei entstehenden informationellen Strukturen.[5] Die Beschreibungskriterien für die einzelnen Phasen ergeben ein einheitliches Darstellungsraster, mit dem eine Vielzahl unterschiedlicher Publikationsprodukte und deren Erstellung eingeordnet werden können. Es handelt es sich im Einzelnen um die folgenden Kriterien:

- Beschreibung
- Prozesse
- Verrichtungsträger
- Wertschöpfung
- Input
- Output/Ergebnis
- Steuerungsinformation
- Werkzeuge
- Genutzte Telepublishing- und Teledienste
- Trends

Eine unmittelbare Interpretation des Phasenmodells als *sequentieller* Abfolge verschiedener Stufen wird der Komplexität des Entwicklungsprozesses *elektronischer* Publikationen allerdings nicht gerecht und zwar aus folgenden Gründen:

- Zahlreiche Einzelschritte bei der Erstellung elektronischer Publikationen verlaufen parallel.
- Der hohe Aufwand für die Erstellung multimedialer Komponenten in elektronischen Büchern rechtfertigt eine langfristige Entwicklungsperspektive mit mehreren Iterationsschritten (sukzessive Anreicherung des Publikationsbestandes durch multimediale Komponenten).

Insofern ist ähnlich wie bei der Softwareentwicklung der Aspekt *zyklischer* Prozesse zu verdeutlichen; dies gilt nicht nur für die Erstellung einer einzelnen Publikation (z. B. Erstellung von Software für eine Multimediapublikation nach dem *rapid prototyping*-Verfahren, vgl. THOMPSON & WISHBOW 1992), sondern darüber hinaus für den gesamten Lebenszyklus eines Werkes (Überarbeitung; Erweiterung; Versionsverwaltung etc.). Bevor notwendige Erweiterungen und Modifikationen dieses Modells für multimediale elektronische Publikationen diskutiert werden, soll seine Anwendbarkeit auf das Referenzprojekt *Multimediales Physikalisches Praktikum* geprüft werden. Tabelle 2 gibt die im Referenzprojekt erfolgten Arbeitsschritte nach den Beschreibungskriterien des Phasen- und Informationsmodells wieder. Dabei bleiben nicht anwendbare Kriterien ausgespart, einige Phasen (Redaktion und Veredlung; Replikation und Distribution) sind zusammengefasst.

4.4 und Kap. 14.2) die Erweiterung der *Nutzungsmöglichkeiten* für dynamische elektronische Bücher zum Ziel hat.

5 Die verschiedenen Phasen des Modells entsprechen partiell den von RESSLER 1993: 3 ff. diskutierten Sichten auf das elektronische Publizieren (*visual*, *logical*, *design*, *communications*, *engineered*, *database* und *specialized views*).

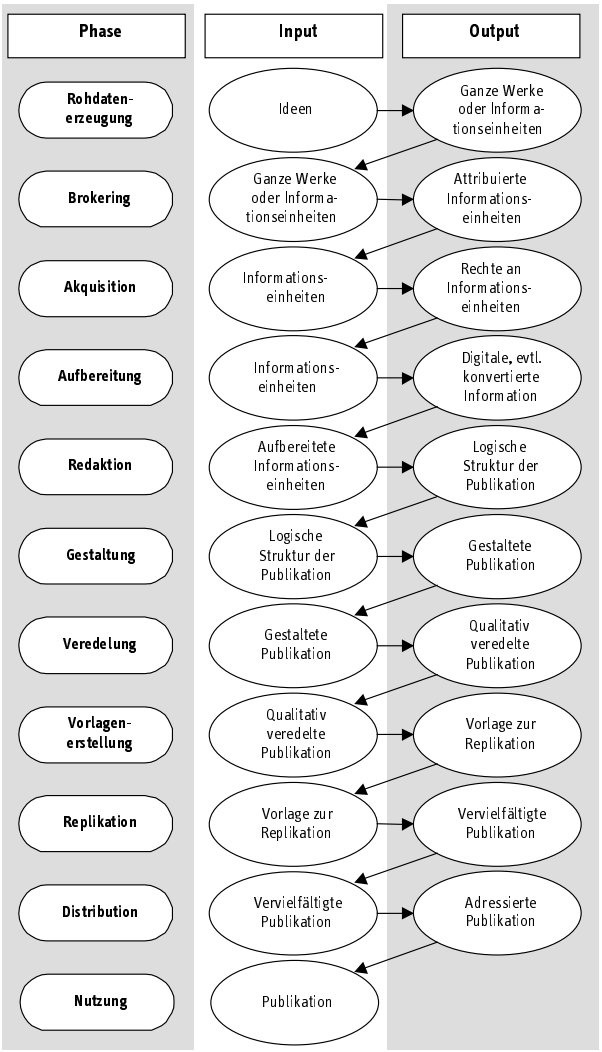

Abbildung 2: Phasen- und Informationsmodell Telepublishing (SANDKUHL & KINDT 1996: 295, Abb. 8.9)

Rohdatenerzeugung	
Beschreibung	Erstellung der Ausgangsmaterialien; klassischer Autorenprozess; zusätzlich Konzeption multimedialer Ergänzungen (Drehbücher für Animationen und Simulationen)
Prozesse	Texterstellung; Softwareimplementierung; Medienerstellung (Photographie, Erstellung von Diagrammen, Schemata etc.)
Verrichtungsträger	Autoren; Multmedia-Entwickler; externe Fachkräfte
Input	Grundkonzeption des Werkes; hier: gedrucktes Buch als Ausgangspunkt; Ideen für multimediale Ergänzungen
Output/Ergebnis	Rohdaten (Texte, Bilder, atomare Medienelemente für Multimediakomponenten)
Steuerungsinformation	Projektplan; Kommunikation und Koordination im Projekt
Werkzeuge und Telepublishing- und Teledienste	Editoren und Entwicklungsumgebungen
Ausblick	Heterogene Prozesse (unterschiedliche Beteiligte; extrem differenzierte Struktur der verwendeten Technologien)

Brokering und Akquisition	
Beschreibung	Ermittlung zusätzlich relevanter externer Datenquellen
Prozesse	Recherche, Kostenermittlung, inhaltliche und formale Überprüfung; Abschätzung des Bearbeitungs- und Integrationsaufwands
Verrichtungsträger	Autoren
Input	Aus der Grundkonzeption des Werkes durch Detailanalyse des vorliegenden Datenbestands ermittelte Bedarfsliste (im Referenzprojekt z. B. Liste physikalischer Versuche, zu denen Multimedia-Ergänzungen sinnvoll sind)
Output/Ergebnis	Sammlung zusätzlich zu integrierender Materialien (z. B. Multimedia-Simulationen)
Steuerungsinformation	Zielvorgaben der Bedarfsliste; Kostenrahmen
Werkzeuge und Telepublishing- und Teledienste	Recherchewerkzeuge; allg. Internetdienste; gezielte Nutzung von Medienagenturen
Ausblick	Herausbildung von Datenbanken mit wiederverwendbaren Medien- und Softwarekomponenten in Analogie zu Agenturkonzepten; bisher kaum strukturierter, nur schwer zu erschließender Markt; das World Wide Web als hauptsächliche genutzte Präsentations- und Marketingplattform

Aufbereitung	
Beschreibung	Überarbeitung der Informationseinheiten für die Nutzung im elektronischen Medium
Prozesse	Konvertierung und Bearbeitung, z. T. Nacherfassung für das elektronische Medium (Textbearbeitung, Scannen, Bildbearbeitung, Integration von Einzelmedien mit Multimedia-Autorensystemen etc.)
Verrichtungsträger	V. a. Multmedia-Entwickler; externe Fachkräfte
Input	Informationseinheiten in unterschiedlichen Medien als Rohmaterial
Output/Ergebnis	Elektronisch aufbereitete und überarbeitete Informationseinheiten; teilweise in integrierter Form (z. B. als Multimediakomponente mit komplexer Binnenstruktur)
Steuerungsinformation	Zielvorgaben für den Output; Plan zur Integration der einzelnen Einheiten
Werkzeuge und Telepublishing- und Teledienste	Konverter, Editoren, Software zur Medienbearbeitung, Autorensysteme
Ausblick	Herausbildung einheitlicher Standards für die elektronische Repräsentation der Medien (Zeichen- und Textkodierung; Bildformate; Multimediastandards), z. B. XML-basierte Beschreibungs-, Speicherungs- und Referenzierungsformate für beliebige Medientypen; Weiterentwicklung von Autorenumgebungen auf der Basis solcher Standards

Redaktion und Veredelung	
Beschreibung	Integration der aufbereiteten Informationseinheiten in das elektronische Buch; inhaltliche Überarbeitung und Korrektur
Prozesse	Hypertextualisierung der Informationseinheiten; Integration in ein Multimedia-Betrachtungssystem
Verrichtungsträger	Autoren und Verlag (inhaltliche Überprüfung und Korrektur), Multmedia-Entwickler (Integration)
Input	Aufbereitete Informationseinheiten in elektronischer Form
Output/Ergebnis	Elektronisches Buch als integriertes Hypermediasystem
Steuerungsinformation	Master-Drehbuch; Gesamtkonzeption; Ablaufplan für die Integrationsschritte
Werkzeuge und Telepublishing- und Teledienste	Diverse Einzelwerkzeuge im Rahmen der Präsentationsplattform; in Ermangelung eines integrierenden Autorensystems Skripte, Parsing- und Formatierungsroutinen für SGML/XML etc.
Ausblick	Herausbildung unterstützender Planungs- und Integrationssoftware, deren Unterstützung bei der Modellbildung beginnt (analog zu software-gestützten Modellierungsverfahren z. B. bei der objektorientierten Analyse)

Gestaltung	
Beschreibung	Layout und Gestaltung des Gesamtprojekts
Prozesse	Erarbeitung eines Designhandbuchs auf der Basis inhaltlicher und technischer Vorgaben; Anwendung des Design auf die integrierte Publikation (elektronisches Buch) im Anschluss auf die nach den Vorgaben erstellten Informationseinheiten
Verrichtungsträger	Gestalter und Multimediaentwickler
Input	Integrierte Publikation nach Redaktion, Korrektur und Veredelung mit bereits partiell berücksichtigten Designvorgaben
Output/Ergebnis	Gestaltete Publikation mit entsprechendem Layout; gestaltete Nebenprodukte (z. B. CD-ROM-Einleger, Website-Einstiegsseite; Benutzerhandbuch etc.)
Steuerungsinformation	Designhandbuch mit technischen Randbedingungen
Werkzeuge und Telepublishing- und Teledienste	Prinzipiell alle präsentationsorientierten Werkzeuge (Editoren, Bildbearbeitung, Satzsysteme etc.)
Ausblick	Bisher keine einheitliche Methodologie für die Integration des Designprozesses; als “Einzelphase” aufgrund der Heterogenität der Prozesse und der zu gestaltenden Informationseinheiten nicht sequentiell einzuordnen

Replikation und Distribution	
Beschreibung	Soweit ein physischer Datenträger (CD-ROM, DVD) benötigt wird, Erstellen eines Masters und Durchführen der Replikation; ansonsten Bereitstellung des elektronischen Buchs als online-Material; u. U. Druck begleitender Materialien; ggf. Integration geeigneter Abrechnungsverfahren
Prozesse	Mastererstellung (CD-ROM- oder DVD-Master) Vervielfältigung durch CD-ROM-Brenner bei Kleinserie, sonst durch externen Dienstleister (Presswerk) Distribution über traditionelle (Verlag; Buchhandel; Bibliotheken) und elektronische Kanäle (E-Commerce; WWW-Server)
Verrichtungsträger	Verlag und externe Dienstleister
Input	Abgeschlossenes elektronisches Buch
Output/Ergebnis	Online-Angebot und/oder Datenträger; ggf. als Media-Bundle (Buch und CD-ROM und/oder Online-Zugangsfreischaltung)
Steuerungsinformation	Ggf. gewünschte Auflagenhöhe
Werkzeuge und Telepublishing- und Teledienste	Replikationssoftware; Server (FTP-, HTTP-Server etc.)
Ausblick	Abnehmende Bedeutung physikalischer Datenträger zu Gunsten von Internet-Diensten

Nutzung	
Beschreibung	Alle Prozesse der Nutzung elektronischer Materialien einschließlich der Umwandlung in eine physische Repräsentation durch Erstellung eines Ausdrucks o. ä.
Prozesse	Zugriff auf CD-ROM und/oder World Wide Web; Interaktion mit dem elektronischen Buch und seinen Komponenten
Verrichtungsträger	Benutzer
Input	Gestaltete und integrierte Informationseinheiten und interaktive Softwarekomponenten und –dienste
Output/Ergebnis	Abhängig von der primären Intention der Publikation, z. B. Erweiterung des Wissens bei Lehr- und Lernmedien
Werkzeuge und Telepublishing- und Teledienste	Viewing-System des elektronischen Buchs (z. B. ein WWW-Browser)
Ausblick	Einführung innovativer Endgeräte (netzbasierte Buchviewer, allg. *digital appliances*)

Tabelle 2: Anwendung des Telepublishing-Modells auf das Referenzprojekt

Wie Tabelle 2 zeigt, lassen sich Phasen und Beschreibungskategorien des Modells für die Erstellung eines elektronischen Buchs anwenden. Es werden aber auch Defizite des Modells deutlich:

- Die sequentielle Ordnung der Arbeitsschritte trifft nicht die Realität komplexer Multimediaproduktionen, zu denen elektronische Bücher gehören. Diese verlaufen partiell zyklisch, einzelne Phasen überlappen sich. Dies wird indirekt durch die Tatsache unterschiedlicher Verrichtungsträger in den einzelnen Phasen unterstrichen.
- Der Aspekt der Integration externer Substanzen und Dienste muss eine höhere Bedeutung erlangen, da nur in Ausnahmefällen alle wünschenswerten Materialien für eine bestimmte Produktion neu erstellt werden können.
- Die Rolle der Grundkonzeption, die auf die verschiedenen Phasen einwirkt (sowohl in inhaltlicher, als auch in technischer und gestalterischer Hinsicht) ist stärker herauszuarbeiten: So beeinflusst ein Gestaltungsleitfaden nicht nur das abschließende Layout einer elektronischen Publikation, sondern bei der Wahl bestimmter Datenformate, Farbtabellen, Schriftarten etc. auch die Informationsaufbereitung. Makroskopische Konzepte wie Designleitfaden, Drehbuch und Storyboard haben in diesem Modell keinen angemessen Platz (vgl. dazu SCHIFMAN, HEINRICH & HEINRICH 1999: 101 ff.).
- Schließlich ist neben den Entwicklungszyklen bei der Erstellung einer elektronischen Publikation der *langfristige Versionierungsaspekt* zu berücksichtigen: Aufgrund des hohen Ressourcenaufwands bei der Erstellung einer Multimediaanwendung ist es sinnvoll, eine sukzessive Ergänzung und Erweiterung des elektronischen Buchs mit einkalkulieren. Gute Beispiele für ein solches Vorgehen sind Multimediaenzyklopädien wie Microsoft Encarta, die eine langfristige Anreicherung durch Multimediakomponenten zeigen; stand am Anfang noch die elektronische Erschließbarkeit durch Recherchewerkzeuge als Mehrwert im Mittelpunkt, so sind in den vergangenen Jahren sukzessive Bild-, Film- Ton- und 3D-Material sowie multimediale Simulationen und Animationen in die Enzyklopädie aufgenommen worden.

Die Versionierung und Erweiterbarkeit der Inhalte eines elektronischen Buchs berührt die Frage nach der Differenzierung zwischen elektronischen Büchern und offenen Hypermediasystemen (OHS, vgl. HAMMWÖHNER 1997), die in dieser Hinsicht als eher *gradueller* Unterschied erscheint: Für das Konzept dynamischer elektronischer Bücher wird zwar ein *Kerninformationsbestand* angenommen. Dieser ist jedoch sukzessive zu erweitern, wobei nach unterschiedlichen Strategien vorgegangen werden kann:

- Gezielte und geplante Ergänzung durch fest definierte Typen inhaltlicher Ergänzungen (z. B. nach einem bestimmten Drehbuch entwickelte multimediale Simulationen).
- Dynamische adhoc-Ergänzung durch Informationseinheiten, die z. B. durch einen Recherchedienst ermittelt und in das elektronische Buch fallweise integriert werden können.

Das Gegensatzpaar *abgeschlossen – offen* ist nur bedingt hilfreich, da einerseits *Abgeschlossenheit der Ausgangsinhalte* das dynamische elektronische Buch konstituiert, Offenheit gegenüber Ergänzungen durch zusätzliche Materialien oder Dienste aber andererseits ein Wesensmerkmal eines dynamischen elektronischen Buches ist. Zwei weitere Aspekte der Durchführung von Multimediaproduktionen betreffen die zugrunde gelegte Methodologie bzw. die benötigte Fachkompetenz:

- Wie SCHIFMAN, HEINRICH & HEINRICH 1999: 83 ff. deutlich machen, ist es sinnvoll, sich bei der Konzeption und Projektplanung von Multimediaproduktion an etablierten

Standards aus dem Software-Engineering zu orientieren (Erstellung von Lasten- und Pflichtenheft etc.).
- Im Vergleich mit traditionellen Publikationsprojekten, die eine Bündelung unterschiedlicher Arbeitsschritte beim Autor vorsehen, ist bei einer Multimediaproduktion der Bedarf an interdisziplinärem Einsatz qualifizierten Fachpersonals sehr hoch und eine Abbildung so unterschiedlicher Arbeitsschritte wie Drehbucherstellung, Gestaltung, Implementierung multimedialer Komponenten auf den Autor sachlich und zeitlich kaum denkbar.

Bevor anschließend unterschiedliche Typen elektronischer Publikationen diskutiert und der Begriff dynamischer elektronischer Bücher präzisiert werden kann, ist auf das konzeptuelle Umfeld dynamischer elektronischer Bücher einzugehen.

2.2 Das Umfeld des elektronischen Publizierens

Elektronische Bücher und elektronische Publikationen als Gegenstände der angewandten Informatik sind rechnergestützte Informationssysteme (IS). Dieser zentrale Begriff der Informatik lässt sich nach unterschiedlichen Gesichtspunkten definieren. STEINMÜLLER 1993: 218 unterscheidet

- die *informationswissenschaftliche* Definition, die jedes Information verarbeitende System als IS betrachtet, also auch Systeme, die nicht computergestützt sind,
- die *modelltheoretische* Definition, die den Schwerpunkt auf das zugrunde liegende Informationsmodell legt,
- die Definition eines IS als *Informations- und Kommunikationssystem* (IuK-System im Unterschied zu Informationssystemen in der Nachrichtentechnik),
- das *formale* Informationssystem als Modellbildung der formalen Informatik und
- das Konzept eines Informationssystem im Sinne der angewandten Informatik, das er wie folgt definiert:

  > Informationssystem ist jedes reale dynamische System mit Informationstechnik-Unterstützung einschließlich seiner sozialen räumlichen und zeitlichen Bezüge unter dem Aspekt der Informationsverarbeitung. [STEINMÜLLER 1993: 217].

Elektronische Bücher als Informationssysteme im letztgenannten Sinn stehen in enger Beziehung zu

- Hypertext und Hypermediasystemen,
- Multimediasystemen sowie zu
- elektronischen Lehr- und Lernsystemen.

Diese drei Typen von Informationssystemen sind von besonderer Bedeutung für elektronische Bücher: Sie beinhalten durch Hypertextverknüpfungen Elemente von Hypertextsystemen, multimediale Bestandteile rechtfertigen die Erstellung elektronischer Bücher jenseits der einfachen elektronischen Distribution von Texten und Lehr- und Lernsysteme sind eines der wichtigsten Anwendungsgebiete elektronischer Bücher.

Bei diesen drei Kategorien (Hypertextsysteme, Multimediasysteme, elektronische Lehr- und Lernsysteme) handelt es sich nicht um disjunkte Bereiche; sie überlappen sich vielmehr in vielfältiger Form, wenn Hypertextsysteme multimediale Inhalte umfassen oder elektronische Lehrsysteme ihre Inhalte über Multimediakomponenten präsentieren und als Hypertext strukturiert sind.

Die drei Bereiche der Entwicklung von Informationssystemen sind in dieser Arbeit eingebettet in den allgemeineren Kontext der Web-basierten Informationssysteme (vgl. grundlegend zum World Wide Web BERNERS-LEE et al. 1994, BERNERS-LEE 1999B, zu webbasierten Informationssystemen ISAKOWITZ, BIEBER & VITALI 1998 und unten Kap. 2.2.4). Es gilt eine grundlegende technologische Beschränkung auf elektronische Bücher, die mit den für das World Wide Web verfügbaren Standards und Werkzeugen realisiert sind und im World Wide Web genutzt werden können. Diesen Zusammenhang verdeutlicht die nachfolgende Abbildung:

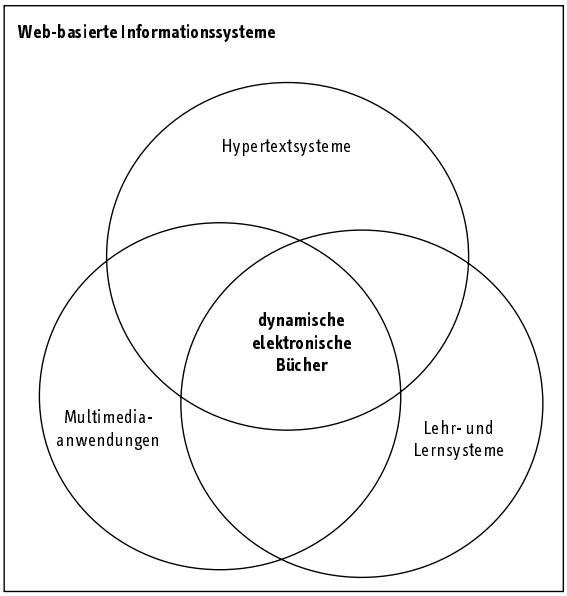

Abbildung 3: Einordnung dynamischer elektronischer Bücher als Systemklasse

Die Einordnung der drei Systemtypen in den allgemeinen Kontext Web-basierte Informationssysteme ist nicht als Unterordnung zu verstehen; man kann argumentieren, Anwendungen im World Wide Web seien – einfach ausgeprägte – Hypermediasysteme, s. u. Kap. 2.2.1 zu Hypertext und NÜRNBERG & ASHMAN 1999. Ein verwandter Ansatz ist das Konzept der *computational hypermedia applications* (CHA), das die Erweiterung von Hypertextsystemen um „traditionelle“ Softwarefunktionalität beschreibt, vgl. ROSSI, SCHWABE & GARRIDO 1999: Kap. 1.

2.2.1 Hypertext und Hypermedia

Unter Hypertext versteht man die *nichtlineare* Verknüpfung informationeller Einheiten; beinhalten diese nicht nur textuelle, sondern auch multimediale Elemente, so spricht man von Hypermedia (vgl. SHNEIDERMAN 1998: 552 ff.). Elektronische Bücher und elektronische Publikationen, die nicht nur ein maschinenlesbares Äquivalent ihres gedruckten Pendants darstellen, sind eng mit dem Begriff Hypertext verbunden: Lediglich die einfache (Ab-)Speicherung von Dokumenten verzichtet auf die Einführung von Hyperlinks. Selbst in diesen Fällen ist aber das nicht als Hypertextnetz kodierte Dokument als Knoten in ein Hypertextsystem eingebettet (z. B. im World Wide Web oder in einer elektronischen Bibliothek).

Hypertext macht aus dem rein linearen einen strukturierten und grundsätzlich beliebig vernetzten Informationsbestand (vgl. SCHNUPP 1992: 11, HOFMANN & SIMON 1995: 2f.).

die folgende Definition von KUHLEN 1991: 13 ergänzt den Hypertextbegriff um die Dimension der *Nutzung* bzw. Manipulation von Hypertexten:

> Die Grundidee von Hypertext besteht darin, dass informationelle Einheiten, in denen Objekte und Vorgänge des einschlägigen Weltausschnitts auf textuelle, graphische oder audiovisuelle Weise dargestellt werden, flexibel über Verknüpfungen manipuliert werden können.

Dem Hypertextgedanken liegt der Versuch einer einfachen Analogie zwischen der assoziativen Arbeitsweise des menschlichen Gehirns und einer entsprechenden Repräsentationsform für Information zugrunde, wie TOMEK et al. 1991: 63 deutlich machen:

> *Hypermedia* is a philosophy of representation and access of information. Its conceptual basis is the model of the information space as a graph whose nodes store information and whose arcs represent semantic relationships [...].
> The concept of hypermedia owes much to the fascination of information scientists with the functioning of the human brain and the associative model of its memory. It has been argued that while human memory operates on the principle of multidimensional association of pieces of information, conventional representation of information (for example, in books) is linear and provides only an imperfect match with human information processing.

Aus der Annahme eines Isomorphismus zwischen externer Repräsentation von Strukturierung von Information und der internen mentalen Speicherung (assoziatives Denken) lässt sich aber nicht unmittelbar ein Vorteil für die Rezeption von Information über Hypertext ableiten: Die Strukturähnlichkeit allein reicht nicht aus, da nicht klar ist, ob solch isomorphe Strukturen 1:1-rezipiert und gespeichert werden können. Vielmehr fragt sich, ob nicht die richtige Linearisierung eines gegebenen Informationsbestands bzw. darin enthaltener Teilbestände die besten Voraussetzungen für die mentale Speicherung liefert. Dies geschieht in elektronischen Büchern: Hypertext-Elemente ergänzen die lineare Grundstruktur, wo dies sinnvoll erscheint; es entsteht aber kein grundsätzlich offenes Hypertextnetz.

HOFMANN & SIMON 1995: 6 versuchen, den Hypertextbegriff durch Analyse seiner wesentlichen Strukturkomponenten zu definieren und stellen die Merkmale *Struktur*, *Operationen*, *Medium* und *Interaktion* heraus:

> **Struktur**: Hypertext ist ein Netzwerk von Knoten, die Informationen beinhalten oder repräsentieren.
> **Operationen**: Anlegen eines Hypertextes durch Autoren sowie das Lesen durch Benutzer sind nichtsequentielle Tätigkeiten. Der Leser steuert selbst, seinem Wissen und seiner Motivation entsprechend durch das Netz, dabei unterstützt von entsprechenden Werkzeugen wie graphischen Browsern und anderen Navigationswerkzeugen.
> **Medium**: Hypertexte sind nur maschinengestützt, d. h. auf Basis des Computers sinnvoll. Inhalte können durch statische Medien (Fließtext, Graphiken, Bilder, Tabellen) oder durch dynamische Medien (Audio, Video, Animationen) repräsentiert werden.
> **Interaktion**: Last but not least bestehen Hypertexte aus Informationen, auf die interaktiv zugegriffen werden kann. Im allgemeinen geschieht das über eine direkt manipulierbare graphische Benutzungsoberfläche.
> **Hypermedia** ist ein Begriff für Hypertext, der multimediale Datentypen enthält.

Diese Einteilung lässt sich auf elektronische Bücher übertragen: Ihnen liegt ebenfalls eine Struktur zugrunde, die sich als Netz informationeller Einheiten beschreiben lässt, die durch Kanten miteinander verbunden sind, wobei allerdings der Strukturaspekt weniger Freiheitsgrade aufweist als in einem Hypertextmodell: Die vom Autor definierte Struktu-

rierung des Inhalts betont lineare (Teil-)Sequenzen, zu denen Hypertextverknüpfungen als zusätzliches Strukturierungs- und Erschließungsmittel hinzukommen (s. u. Kap. 4.5.2).

2.2.1.1 Entwicklungsgeschichte des Hypertextgedankens

Der Hypertextgedanke geht aus der Perspektive der Entwicklung der Informationstechnologien auf Vannevar BUSHs 1945 vorgestelltes Konzept für das System *Memex* als assoziatives Speichersystem für das Wissen des Menschen mit leistungsfähiger Inhaltserschließung zurück. Auch wenn BUSH die konkrete Realisierung seiner Idee mit Hilfe der damals verfügbaren Technik plante, also mit elektromechanischen/photographischen Repräsentationsformaten und Werkzeugen, so sind seine Grundkonzepte und vor allem die kognitionstheoretische Begründung der nichtlinearen Repräsentation von Information doch äußerst fruchtbar für die Entwicklung der Hypertextsysteme gewesen. Die Begriffe „Hypertext“ bzw. Hypermedia“ prägte in den 60er Jahren NELSON, der seitdem sein System *Xanadu* als universales verteiltes Wissensverwaltungs- und Informationssystem auf der Basis der *Transklusion* entwickelt, das die Möglichkeit der *Einbettung* externer Inhalte in Hypermediadokumente realisiert (vgl. NELSON 1994, 1995, 1999A). NELSON definiert Transklusion als „virtual inclusion across a document boundary“ (NELSON 1994: 262) bzw. als „reuse with original context available through embedded instance sharing (rather than duplicate bytes)“ (NELSON 1995: 32). Externe Inhalte können durch transkludierende Verknüpfungen (ohne Duplikation) in ein Dokument eingeschlossen werden; eine technische Realisierung ist in der einbettenden Linkaktivierung zu sehen, die XLink, die Kodierungssprache für Hypertextverknüpfungen in XML, enthält, vgl. unten Kap. 5.2.4.

Eine Renaissance erlebte die Hypertextforschung ab Ende der 80er Jahre, als mit der Einführung graphischer Benutzerschnittstellen und einfacher Autorensysteme für Hypertext die Erstellung von Hypertexten für eine hohe Zahl von Anwendungsgebieten möglich wurde (vgl. NIELSEN 1999: 71). Mit der Verbreitung des World Wide Web als Internet-Informationssystem für mittlerweile fast alle denkbaren Anwendungen hat der Hypertextgedanke generelle Akzeptanz gefunden. In wissenschaftlicher Hinsicht findet dabei eine Konvergenz zwischen seit langem bekannten theoretischen Konzepten der Hypertextforschung wie NELSONs Transklusion und der Weiterentwicklung des ursprünglich einfach strukturierten World Wide Web zu einem leistungsfähigen Hypertextwerkzeug statt (vgl. NÜRNBERG & ASHMAN 1999).

2.2.1.2 Strukturelemente von Hypertexten

Prinzipiell lässt sich ein Hypertext in folgende Strukturelemente gliedern:

- Die *Knoten* als atomare informationelle Einheiten des Hypertextnetzes, die eine unterschiedliche Binnenkomplexität aufweisen können und nicht notwendigerweise in einem einzelnen Medium repräsentiert sein müssen.
- Die Verknüpfungen zwischen Knoten, die aus einzelnen Informationseinheiten einen Hypertext erzeugen.
- Das Makrokonzept des Hypertexts selbst als Graph bzw. Netz informationeller Einheiten (Hypertextnetz[6]).

---

[6] Vgl. BERNSTEIN 1998, der versucht, für die Binnenstrukturen innerhalb eines Hypertextgraphen ein geeignetes Beschreibungsvokabular einzuführen (z. B. *Cycle* für die Rückkehr an einen Ausgangspunkt bei der Navigation oder *Counterpoint* für dialogische Strukturmuster).

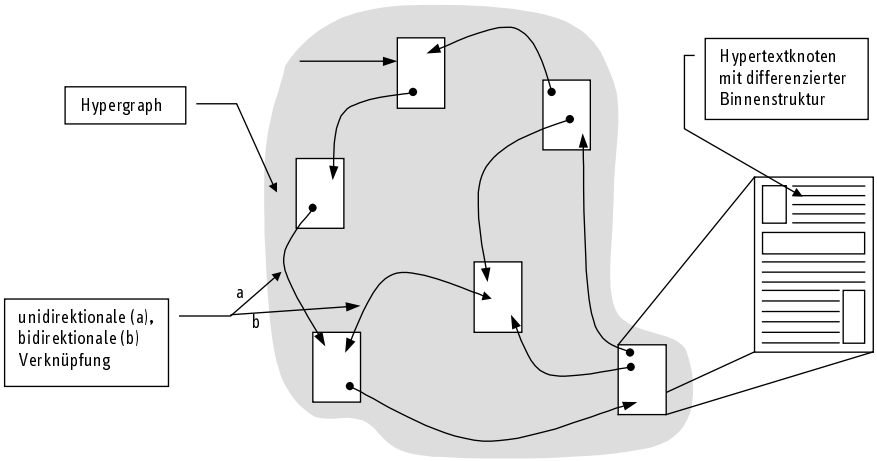

Abbildung 4: Schematische Darstellung eines Hypergraphen

Hypertexte lassen sich anhand der Eigenschaften der in ihnen enthaltenen Informationseinheiten kategorisieren:

- *Binnenstruktur* einer Informationseinheit (d. h. zulässige Inhalte (Medientypen) und Komplexität der Binnenstruktur einer Informationseinheit),
- *Umfang* einer Informationseinheit (logisch/technisch (Datenmenge)),
- Bezug der logischen Einteilung zur Einteilung in *Präsentationseinheiten*,
- *Speicherort* einer Informationseinheit (lokal oder über Netzwerke erreichbar),
- *Medientypen*, die in den Informationseinheiten enthalten sind.

Die Eigenschaften der Hypertextverknüpfungen haben Einfluss auf die Nutzungsmöglichkeiten eines Hypertexts. Zu unterscheiden sind

- die *Richtung* der Verknüpfungen (unidirektional, bidirektional),
- die *Granularität* der Adressierung (jeweils Hypertextknoten als Gesamtheit oder Teile innerhalb von Hypertextknoten),
- die *Kardinalität* der Verknüpfung (1:1-, 1:n-, n:1-, und n:m-Verknüfungen, vgl. LEGGETT 1994: 78, Fig. 1),
- die *Typisierung* der Verknüpfung, d. h. die Unterscheidung unterschiedlicher Verknüpfungsarten (z. B. durch semantische Relationen oder als medienabhängige Typisierung) und
- die *technische Realisierung*, z. B. über ein standardisiertes Adressierungsschema wie die *Uniform Resource Locators* (URL) des World Wide Web (vgl. KHARE 1999A, 1999B).

Bei der *Nutzung* eines Hypertexts, d. h. der *Traversierung* der Kanten zwischen den Knoten des Hypertexts ist zu entscheiden, auf welche Weise der Zielpunkt einer Hypertextverknüpfung durch das System dargestellt wird. Die Traversierung einer Folge von Hypertextverknüpfungen durch den Benutzer bezeichnet man als *Pfad* durch den Hypergraphen. Man unterscheidet drei Realisierungsformen der Kantentraversierung (vgl. KUHLEN 1991: 16):

- Darstellung durch *Ersetzung*, d. h. die neue Informationseinheit verdrängt im Darstellungssystem den Knoten, von dem die Verknüpfung ausgeht.

- Darstellung *in separatem Betrachtungssystem*, d. h. das Ziel der Verknüpfung wird entweder in einem zusätzlichen (neuen) Fenster dargestellt oder es verdrängt eine andere Informationseinheit in einem anderen Fenster.
- Darstellung durch *Einbettung* bzw. *Expansion*, wobei der verknüpfte Inhalt an die Stelle der Verknüpfung (d. h. ihrer Repräsentation) tritt und in den Ausgangsknoten eingebettet wird.

Diese Unterscheidung von Traversierungstypen ist auch in der für die XML vorgesehen Hypertextsprache XLink vorgesehen (vgl. unten Kap. 5.2.4). Neben diesen Basiseigenschaften können Hypertexte eine Reihe weiterer Eigenschaften aufweisen:

- *Regeln* für die Konstruktion von Hypertexten bzw. für die Navigation in einem Hypertextbestand,
- Strukturierungskonzepte, die innerhalb eines Hypertextnetzes bestimmte *Pfade* herausheben (*guided tours*, *overview*, vgl. unten Kap. 2.2.1.3).
- Zusätzliches Wissen für die dynamische Generierung von Hypertextstrukturen oder die Verwaltung von Wissen über den Benutzer (Dialoggeschichte, Pfade des Benutzers durch das Hypertextnetz).

2.2.1.3 Formale Beschreibung von Hypertextgraphen

Der Aufbau von Hypertexten lässt sich sowohl auf der Ebene der formalen Modellierung als auch bei der Entwicklung eines Architekturmodells für Hypertextsysteme beschreiben. In formaler Hinsicht kann man einen Hypertext in Anlehnung an SCHNUPP 1992: 92 ff. als einen Hypergraphen G beschreiben, der aus einem 5-Tupel $G=<K,E,F,I,A>$ besteht, wobei K eine endliche Menge von Knoten, E eine endliche Menge von gerichteten, markierten Kanten, I eine endliche Menge von Teilinformationen, $F: K \rightarrow I$ die Zuordnung einer Teilinformation zu jedem Knoten und A eine endliche Menge von Kantentypen (Attribute oder Markierungen) ist.

Die Menge E der Kanten lässt sich durch das Produkt $K \times K \times A$ beschreiben, d. h. eine Kante ist ein 3-Tupel, bestehend aus Ausgangsknoten, Zielknoten und einem Kantenattribut. Gibt es keine Typisierung der Kanten, so ist die Menge A leer und man kann die Kantenmenge als 2-Tupel aus $K \times K$ beschreiben. Der nach diesem Modell aufgebaute Graph kann Verweise (Hypertextverknüpfungen) enthalten, die entweder zwei Knoten innerhalb des Graphen verbinden oder auf einen Knoten in einem externen Graphen verweisen:

$e_i = <k_1,k_2,a>$ *(interner Verweis)*
$e_x = <k_1,G'.k_2,a>$ *(Verweis in externen Graphen G')*

Die Interaktion mit einem Hypergraphen lässt sich als *Hypertextmaschine* formalisieren; dabei handelt es sich um einen endlichen Automaten, dessen Zustände die aktuelle Position im Hypergraphen kennzeichnen und die sich wie folgt darstellen lassen: $G_Z = <K, E, F, I, A, k_{akt}>$ mit k_{akt} = aktueller Knoten.

Die Operationen über dem Graphen, insbesondere seine Traversierung durch den Benutzer des Hypertexts lassen sich als Zustandsübergänge von Zustand Z nach Zustand Z' des Automaten beschreiben: $G_Z \rightarrow G_{Z'}$. Auf dieser Basis kann eine formale Definition der Semantik von Hypertextoperationen angegeben werden. Eine Operation *follow*($<k_{akt}, k_i, a>$), die das Traversieren vom aktuellen Knoten k_{akt} zu einem Zielknoten k_i entlang der Kante a beschreibt, kann wie folgt dargestellt werden: Gelten $G_Z = <K, E, F, I, A, k_{akt}>$

und $< k_{akt}, k_i, a> \in E$, dann bewirkt *follow*($<k_{akt}, k_i, a>$) den Übergang $G_Z \rightarrow G_{Z'} = <K, E, F, I, A, k_i>$.

Die Formalisierung eines Hypertexts kann nicht nur Ausgangspunkt einer Übersetzung in ein konkretes Hypertextsystem sein, sondern auch dazu dienen, komplexitätstheoretische Untersuchung von Hypertexten durchzuführen bzw. eine anwendungsneutrale Spezifikation von Hypertextmodellen ohne konkreten System- oder Anwendungsbezug zu erreichen. Weitergehende Formalismen (vgl. TOCHTERMANN 1995, STOTTS, FURUTA, CABARRUS 1998) versuchen, über eine Differenzierung der elementaren Strukturelemente von Hypertexten – Knoten und Kanten – die Beschreibung von Makrostrukturen (z. B. sog. *guided tours*) in die Formalisierung einzubeziehen und abzuklären, wie sich solche differenzierteren Beschreibungsmechanismen für den Vergleich tatsächlich existierender Hypertextsysteme eignen. Das bisher einflussreichste Modell zur Beschreibung von Hypertextsystemen, das *Dexter Reference Model*, operiert im Vergleich dazu auf einer stärker an Konzepten der Softwaretechnologie orientierten Ebene. Da es nicht nur Ausgangspunkt zahlreicher Weiterentwicklungen war, sondern explizit zur Standardisierung der Diskussion des Hypertextprinzips beitragen will und insofern für die Entwicklung des Modells dynamischer elektronischer Bücher hilfreich ist, wird es in Kap. 2.2.1.5 näher diskutiert.

2.2.1.4 Linktypen in Hypertext

Um einen Hypertext aufzubauen, bedarf es der Bestimmung der möglichen Zielstrukturen. Eine Vielzahl unterschiedlicher Verknüpfungstechnologien ist in der Literatur vorgestellt worden (vgl. AGOSTI 1997: 134 f., ALLAN 1997). Ihnen gemeinsam ist das Konzept der Verknüpfung zweier unterschiedlicher Textpositionen (Ausgangs- und Zielknoten). Die Menge aller Knoten konstituiert den Hypertextgraphen, durch den der Benutzer sich bewegen kann. Die Links selbst lassen sich dann auf unterschiedliche Art klassifizieren:

1. Unterscheidung nach *inhaltlichem Bezug* zwischen Link und Material
- *Strukturelle* Links spiegeln die Struktur des Dokuments wieder (Kapitelgliederung etc.).
- *Referentielle* Links greifen Verweisstrukturen zwischen Elementen des Ausgangsdokuments auf (Verweise auf Literatur, die selbst im Volltext elektronisch dargestellt werden kann etc.).
- *Assoziative* Links verknüpfen ansonsten nicht unmittelbar miteinander verbundene, aber inhaltlich zusammengehörige Knoten.

Für die entstehenden Linktypen haben sich verschiedene Klassifikationsmodelle herausgebildet (vgl. STEINMETZ 1999: 714 f., BOTAFAGO, RIVLIN & SHNEIDERMAN 1992), ohne dass aber eine allgemein akzeptierte Linktypologie existierte. Aufgrund der vielfältigen Inhalte und Funktionen elektronischer Bücher ist es sinnvoll, von einem *mehrstufigen* Konzept der Linktypisierung auszugehen, das sowohl verallgemeinerbare strukturelle Links (sequentielle Navigation, hierarchischer Aufbau der Inhalte) als auch durch den spezifischen Inhaltskontext und seine deklarative Auszeichnung durch inhaltsorientiertes Markup bedingte Verknüpfungen umfasst (vgl. dazu Kap. 4.2 und Kap. 10).

2. Unterscheidung nach *Entstehung* der Links (vgl. ALLAN 1997)
- Manuelle Links: Vom Autor selbst erstellt (Hypertextaufbau mit Hilfe eines Autorensystems und Editoren).

- *Pattern matching* Links: Links, die durch einfachen Mustervergleich entstehen (gleiche Muster für die Referenzierung von Inhaltselementen im Text, Strukturmerkmale wie Überschriften).
- Automatische Links: Diejenigen Links, die sich nicht durch einfaches *pattern matching*, sondern nur durch aufwendigere Verfahren, z. B. Dokumentenclustering, erkennen lassen.

3. Unterscheidung nach der *Präsentations*form
- Anbindung des Hypertext-Ankers an Textelemente eines Knotens und Hervorhebung durch ein graphisches Attribut (Schriftfarbe, -schnitt, -größe etc.).
- Darstellung des Links durch abstrakte oder piktorielle visuelle Elemente (Ikonen).
- Verzicht auf eine explizite Darstellung der Verknüpfung (implizite Links).

4. Unterscheidung nach der Eingliederung der Verknüpfungen in den Hypertextgraphen
- Darstellung der Anker innerhalb der Ressource des Knotens (*inline*-Verknüpfungen).
- Getrennte Darstellung und Verwaltung von Verknüpfungen und Knoteninhalten wie sie manche Hypermedia-Server, z. B. *HyperWave* (vgl. MAURER 1996 und *http://www.hyperwave.com*) anbieten.

Folgt man den Unterscheidungsmerkmalen nach Ziff. 1, so lassen sich nur die ersten beiden Linktypen durch Strukturanalyse gewinnen. Assoziative Links können nur aufgrund einer IR-basierten Inhaltsanalyse aufgefunden werden. Eine weitere Unterscheidung betrifft den *Erstellungs-* bzw. *Darstellungs*zeitpunkt einer Verknüpfung: Ein *expliziter* Link kennzeichnet dauerhaft eine Verbindung zwischen zwei Knoten, während ein *impliziter* Link erst auf Verlangen des Benutzers generiert und dargestellt wird. Er ist gewissermaßen *implizit* im Informationsmaterial enthalten, aber nicht *explizit* dargestellt. Man kann in einer Kombination aus Typisierung von Links und deren dynamischer Erzeugung ein großes Potenzial für leistungsfähige Hypertextsysteme sehen, die nicht ein bestimmtes Informationsnetz festlegen, sondern nach Bedarf bestimmte Hypertextverknüpfungen herausfiltern (nach unterschiedlichen Kriterien wie Benutzerfaktoren, gewählter Art der Verknüpfung etc.). Offen bleibt bei dem Verfahren, wie gut sich der entstehende Hypertext für das Browsing eignet.

2.2.1.5 Hypertextmodelle

Zur Beschreibung von Hypertextsystemen und ihrer Funktionalität haben sich eine Reihe von Modellen herausgebildet, von denen das bekannteste das sog. *Dexter Reference Model* (HALASZ & SCHWARTZ 1994) ist. Es hat die Entwicklung einer Modellierungssprache für die Evaluierung und den Vergleich unterschiedlicher Hypertextsysteme zum Ziel. Dabei soll eine standardisierte Terminologie für die Strukturelemente von Hypertexten und Hypertextsystemen angewandt werden (*node, anchor, link*). Das Modell unterscheidet drei wesentliche Ebenen eines Hypertextsystems:

- Der *runtime layer,* der die Präsentation von Inhalten durch das System beschreibt.
- Der *storage layer,* der die Speicherung der im Hypergraphen enthaltenen Daten formalisiert.
- Der *within-component-layer*, der dazu dient, die Binnenstruktur der einzelnen Knoten (Komponenten) zu beschreiben.

Die Übergänge zischen den Ebenen sind in diesem Modell durch

- *presentation specifications* für die Abbildung der gespeicherten Inhalte auf eine konkrete Präsentationsform im *runtime layer* bzw. durch

- *anchoring*, d. h. die Einführung unterschiedlicher Verknüpfungstypen innerhalb der Komponenten beschrieben.

Hinsichtlich der Modellierung der Inhalte eines Hypergraphen geht das Modell vom Begriff einer Komponente (*component*) aus, was der kleinsten geschlossenen Darstellungseinheit verschiedener Hypertextsysteme entspricht (*card, page, document, frame*). Eine Komponente ist demnach ein generischer Behälter, dessen innere Struktur durch den *within-component layer* beschrieben werden kann. Komponenten sind rekursiv definiert: Component := Link | Atom | composite entity.

Unter einem Link sind zwei oder mehr *endpoint specifications* zu verstehen, die auf vorhandene Komponenten Bezug nehmen. Ein Atom ist ein Knoten, d. h. ein Synonym für eine einfache Inhaltskomponente. Eine *composite entitity* muss als gerichteter azyklischer Graph modelliert werden können und darf sich nicht selbst als Subkomponente enthalten.

Das Modell führt für die Adressierung der Inhaltskomponenten ein Adressierungsschema ein (*unique* ID, UID), über das alle Komponenten eindeutig referenzierbar sind. Sie lassen sich in dieser Hinsicht mit dem Konzept der *uniform resource locators* (URLs) vergleichen. Die Auflösung von Verknüpfungen (*link resolving*) erfolgt unter Verwendung der UIDs. Um innerhalb einzelner Komponenten referenzieren zu können, existiert zusätzlich das Konzept eines Ankers mit einer ihm zugeordneten ID (*anchor value & anchor ID, AID*). Neben der Beschreibung der Strukturbausteine eines Hypergraphen gibt das Modell an, welche Funktionen den drei Ebenen zugeordnet sind. Auf der Ebene des *storage layer* gehören dazu Basisfunktionen für die Manipulation von Hypertexten wie:

- *Adding* für die Addition von Komponenten in den Hypergraphen,
- *Deleting* für das Entfernen von Komponenten,
- *Modifying* für die (generische) Änderung einer Komponente,
- *Retrieving* für Ermitteln einer Komponente und
- *Link resolution* für die konkrete Auflösung des Zielpunkts einer Verknüpfung.

Zusätzlich zu diesen Grundfunktionen muss ein Hypertextsystem Möglichkeiten für die Abbildung von Adressen (UIDs) auf Komponenten (*accessor*-Funktion) und die Auflösung aller gültigen UIDs bereithalten (*resolver*-Funktion).

Betrachtet man nur *storage* und *within-component layer*, so lässt sich damit jeder statische Hypertext beschrieben, der unabhängig vom Nutzungskontext immer die gleiche Struktur aufweist. Erst durch Hinzunahme der *presentation specifications* werden dynamische Sichten auf Hypertexte möglich, die aus derselben Hypertext-Datenbasis generiert werden, wie das nachfolgende Beispiel belegen soll: Je nachdem, ob Lehrer oder Schüler einen Link aktivieren, wird eine zu ladende Animation in einem Editor oder in einem Viewer präsentiert. Die unterschiedliche Darstellungsform muss über eine Präsentationsspezifikation kodiert sein.

Die Vorteile des Dexter-Modells sind vor allem in der Tatsache zu sehen, dass es nicht nur ein Strukturierungsschema für nicht-lineare Informationspräsentation definiert, sondern auch die *dynamischen* und *interaktiven* Aspekte der Darstellung von Hypertexten in unterschiedlichen Anwendungen behandelt (vgl. LEGGETT 1994). Als abstrakte Beschreibung ist es nicht nur ein geeigneter Ausgangspunkt für die Entwicklung einer Architektur für elektronische Bücher, es ist auch innerhalb der Hypertextforschung auf vielfältige Weise weiterentwickelt worden. Zwei Beispiele verdeutlichen dies:

Die im Dexter-Modell nicht betrachtete Sonderproblematik der Modellierung multimedialer zeitabhängiger Inhalte versuchen HARDMAN et al. 1994 mit ihrem *Amsterdam Hypertext Model* zu beseitigen und erweitern das Dexter-Modell um den Aspekt der Kontextualität und der Zeitabhängigkeit. HARDMAN et al. gehen von der Überlegung aus,

dass einfache Hypertextmodelle sich überwiegend an textuellen, nicht kontinuierlichen Daten orientieren und die oft verwendete Gleichung „Hypertext + Multimedia = Hypermedia“ zusätzliche Herausforderungen an die Modellbildung stellt. DE BRA, HOUBEN & WU 1999 erweitern in ihrem Dexter-basierten *adaptive hypermedia application model* (AHAM) vor allem die Speicherungsebene des Ausgangsmodell um Komponenten für die Beschreibung von Benutzer- und Domäneneigenschaften sowie die Gegebenheiten einer Lernumgebung. Sie verfolgen das Ziel, durch benutzerbezogene Adaption die flexible Präsentation von Inhalten zu erreichen, eine Problemstellung, die in der unterschiedlichen Generierung von Komponenten des *runtime layer* auf der Basis verschiedener *presentation specifications* im Dexter-Modell nur ansatzweise vorgesehen ist.

Die neuere Hypertextforschung hat sich in Richtung *offener Hypertextsysteme* weiterentwickelt,[7] die nach GRØNBÆK & WIIL 1998: Kap. 2.1 durch folgende Merkmale charakterisiert werden können:

- *Integration* als die Fähigkeit eines Hypertextsystems, Komponenten und Dienste zu integrieren,
- *Interoperabilität*, d. h. die Möglichkeit der Kooperation eines OHS mit anderen OHS bzw. externen Applikationen,
- *Strukturierung*, d. h. alle Merkmale der nichtlinearen Strukturierung und Verknüpfung
- *Navigation*, d. h. neben der Traversierung von Verknüpfungen alle weiteren Hilfsmittel der Navigation (wie gezielte Suche, Pfade, Übersichten etc.),
- *Distribution* als Möglichkeit der plattformunabhängigen Verteilung eines Hypertextsystems und
- *Kollaboration* als diejenigen Merkmale eines OHS, die die Kooperation von Benutzern ermöglichen.

Ausgehend von einer Analyse offener Hypertextarchitekturen schlagen sie ein Modell vor (das sog. *flag model*, vgl. GRØNBÆK & WIIL 1998: Kap. 3.1), das wie das Dexter-Modell eine Aufteilung der Architektur in die Ebenen

- Inhalte (*content layer*),
- Dienste (*service layer*) und
- Struktur (*structure layer*)

vorsieht. Damit steht es in Einklang mit einem weiteren Vorschlag für ein Hypermedia-Referenzmodell, in dem NÜRNBERG & LEGGETT 1998: Kap. 4 für die vermittelnde Ebene zwischen Anwendung (Client) und Speicherung (*storage layer*) folgende Komponenten vorsehen:

- Einen *server information manager* (*sim*), der Informationen über verfügbare Hypermedia-Server bereitstellt und
- einen *structure processor* (*sproc*), der als ein generalisierter Linkserver verstanden werden kann.

Neben der Architekturspezifikation verfolgt die Forschung zu offenen Hypermediasystemen auch das Ziel der Standardisierung von Protokollen für Hypermediasysteme: Das *Open Hypermedia Protocol* (OHP, vgl. ANDERSON, TAYLOR & WHITEHEAD 1998, MILLARD, REICH & DAVIS 1999) für das Spezifikationen in SGML und XML sowie prototypische Implementierungen vorliegen (vgl. REICH 1999). Hinsichtlich der Entwicklung

[7] Hier sind vor allem die Aktivitäten der *Open Hypermedia Systems Working Group* und deren Workshops zu nennen, vgl. zuletzt WIIL 1999.

dynamischer elektronischer Bücher sind die Hypertextmodelle ein wichtiger Bezugspunkt bei der Entwicklung eines Architekturmodells: Einerseits sind die voranstehend genannten Merkmale offener Hypertextsysteme für dynamische elektronische Bücher relevant (z. B. der Aspekt der Integration), zum anderen verwendet das Modell elektronischer Bücher soweit als möglich bereits existierende Netzwerk- und Kodierungsstandards wie HTTP als Netzwerkprotokoll und XML als Beschreibungssprache. Es hat im Gegensatz zu offenen Hypermediasystemen mit seinem geschlossenen Primärdatenbestand des elektronischen Buchs einen anderen Ausgangspunkt. Schließlich wird in dieser Arbeit ein dediziertes Betrachtungssystem für ein dynamisches elektronisches Buch entwickelt, während die *Offenheit* eines Hypermediasystems primär als die Möglichkeit verstanden wird, unterschiedliche Clients anbinden zu können: „[...] open hypermedia systems allow an open set of clients of the hypermedia services provided by the system“ (NÜRNBERG & LEGGETT 1998: Kap. 2.1).

2.2.1.6 Entwicklungsverfahren für Hypertext

Die bisher diskutierten Modelle beschreiben den *Aufbau* von Hypertexten und Hypertextsystemen, nicht aber geeignete *Verfahren* für die Erstellung eines Hypertexts. Neben der allgemeinen Systemarchitektur stellt diese Frage aber ein zentrales Problem der Realisierung von Hypertextanwendungen dar. Es sind nicht nur inhaltsbezogene Fragen nach

- der Einteilung der Inhalte in geeignete Komponenten, die sich vom Benutzer adäquat rezipieren lassen (u. a. Fragen der Kohärenz der Inhalte, der kognitiven Adäquatheit verschiedener Präsentationsformen, *cognitive design issues*, vgl. THÜRING, HANNEMANN & HAAKE 1995) und
- der Auswahl eines bestimmten Verknüpfungskonzepts mit ggf. typisierten und unterschiedlich dargestellten Verknüpfungen

zu klären, sondern vor allem ist ein geeignetes Entwurfsverfahren anzugeben, nach dem sich ein Ausgangsdatenbestand systematisch in eine Hypertextanwendung überführen lässt. Zwei Verfahrensmodelle seien nachfolgend vorgestellt.

HOFMANN & SIMON 1995: 95 f. geben für die Entwicklung von Hypertexten ein an der objektorientierten Analyse ausgerichtetes Stufenmodell an, in dem – ausgehend von einem logischen Grundgerüst (Schemata, Klassen, Instanzen) – die Konkretisierung des Hypertexts durch nutzungsbezogene „Leseschemata“ erfolgt.[8] Der Entwurf einer Hypertextanwendung erfolgt demnach durch die Schritte

- Definition von Zugriffsstrukturen,
- Festlegen und Zuordnen der Inhalte (mit starker Rückkopplung zu den Grundstrukturen),
- Festlegen von Elementen für die Navigationsunterstützung,
- Entwurf zusätzlicher Funktionen und
- Berücksichtigung externer Entwicklungsfaktoren (Umwelt; Benutzer, Nutzungsort, Vorwissen etc.).

Dabei wird der Informationskorpus für den Hypertext einer Medienanalyse zur Bestimmung der Strukturierungsgranularität unterworfen, die Rückschlüsse auf die passende Präsentationsform zulässt; allgemein lässt sich festhalten, dass diese Überführung bei

[8] Neben diesem Ansatz ist die OOHDM (*Object-Oriented Hypermedia Design Method*) ein weiteres bekanntes Beispiel für die Verwendung von OO-Methoden zur Modellierung von Hypermedia, vgl. SCHWABE, DE ALMEIDA PONTES & MOURA 1999.

hochstrukturierten Daten (z. B. eines Lexikons) einfacher zu bewältigen ist, als bei Substanzen mit einem weniger deutlich erkennbarem „Datenmodell“ (z. B. bei unstrukturierten Texten in einem elektronischen Buch).

Das von ISAKOWITZ et al. 1995 vorgestellte *Relationship Management Model* (RMM) ist als Entwicklungsverfahren deshalb bemerkenswert, weil es bewährte Verfahren aus der Datenbanktechnologie bzw. der Datenmodellierung auf den Hypertextentwurf überträgt und in ein allgemeines Entwicklungsmodell für Hypertextanwendungen eingliedert. Es liegt das Verständnis zugrunde, Hypertext bzw. Hypermedia als Mittel zur Darstellung von Beziehungen zwischen Informationsobjekten zu modellieren und zu verwalten. Hinreichend strukturierte Inhalte vorausgesetzt, lassen sich aus der Datenbanktechnik bekannte Verfahren wie *entity-relationship*-Modelle auf die Hypertextproblematik anwenden und für sie erweitern.

Legt man ein solches formales Datenmodell als Abstraktion darstellbarer Weltausschnitte für die Beschreibung der Daten in einem Hypertext zugrunde, so zeigt sich, dass zusätzliche Konzepte zur Beschreibung von Zugangsprimitiven (Navigationsmodell) bzw. von Makrosichten auf den Hypertext erforderlich sind. Die Modellprimitive in RMM umfassen daher nicht nur die bekannten Elemente eines *entity-relationship*-Modells, sondern darüber hinaus weitere Elemente, die spezifisch auf die Hypertextanwendung zugeschnitten sind, wie Domänenprimitive und Zugangsprimitive (Verknüpfungstypen, Makrokonzepte wie Übersichten und *guided tours*, vgl. ISAKOWITZ, STOHR & BALASUBRANIAN 1995: 35, Abb. 1).

Die Erarbeitung eines Hypertext-Datenmodells mit RMM ist in eine allgemeine Entwicklungsmethode eingebettet, die nicht nur Konzepte aus dem Software-Engineering integriert (z. B. Machbarkeitsstudie und *requirements analysis* im Vorfeld der Anwendungsentwicklung), sondern die einzelnen Verfahrensschritte für die Datenkonvertierung, die Gestaltung der Benutzerschnittstelle und den Test des Anwendungssystems enthält.

Das Ziel des Modells ist die durch ein vorgegebenes Entwicklungsmodell mögliche Effizienzsteigerung. Seine Anwendung ist abhängig von der logischen Struktur der Daten: Je besser sich diese durch ein formales Datenmodell beschreiben lassen, um so einfacher ist die (semi-automatische) Umsetzung in eine konkrete Hypertextanwendung. Abbildung 5 zeigt die einzelnen Entwicklungsschritte im Rahmen der RMM im Überblick.

Vorteile des Modells liegen in seinem holistischen Ansatz, der den gesamten Hypertextentwicklungsprozess berücksichtigt, sowie in der Übernahme bewährter Datenmodellierungsverfahren. Es ist auch bereits für die Entwicklung einer Methode zur Erstellung webbasierter Informationssysteme herangezogen worden (vgl. TAKAHASHI & LIANG 1997). Wesentliche Bestandteile des Modells lassen sich auf die Entwicklung dynamischer elektronischer Bücher übertragen, wobei zusätzliche Aspekte – die Integration komplexer Komponenten sowie externer Diente – hinzukommen. CONALLEN 1999: 64 weist darauf hin, dass zu den Schwächen von RMM die fehlende Berücksichtigung von Applikationen bzw. Komponenten zählt; diese an der Modellierung von *Web applications* v. a. für den E-Commerce orientierte Argumentation gilt aber auch für dynamische elektronische Bücher und die in ihnen realisierten Komponenten und Dienste. Gerade auf der Ebene der Informationsstrukturierung und Informationsmodellierung ergeben sich Parallelen, wenn auch davon auszugehen ist, dass sich das Informationsmodell eines elektronischen Buchs nicht mit beliebig feiner Granularität auf ein erweitertes E-R-Modell übertragen lässt. Die Ursache hierfür ist darin zu sehen, dass sich unstrukturierter Text nicht direkt in ein (formales) Datenmodell überführen lässt.

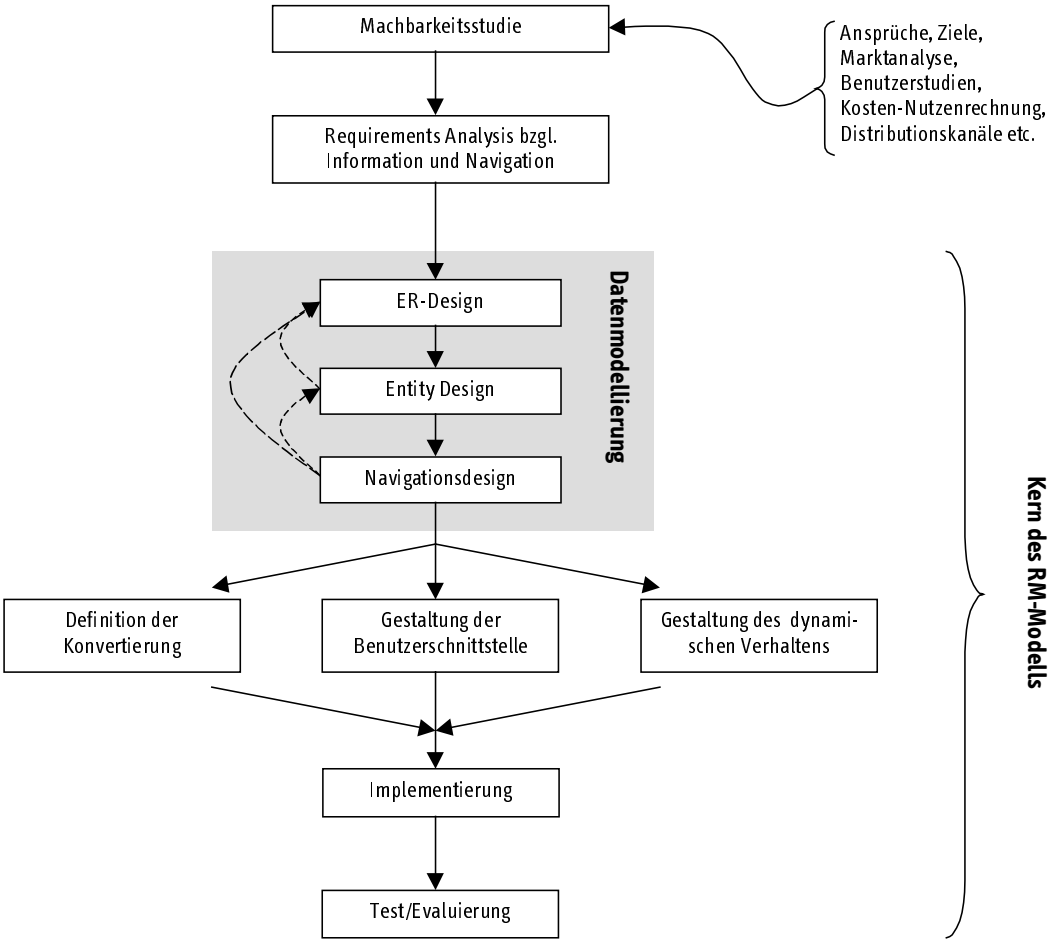

Abbildung 5: RMM-Entwicklungsmodell (nach ISAKOWITZ, STOHR & BALASUBRANIAN 1995: 38)

2.2.1.7 Hypertextsysteme

Im Anschluss an die Präsentation verschiedener Modellierungs- und Entwicklungsmethoden stellt das vorliegende Kapitel eine kurze Übersicht bekannter Hypertextsysteme vor. Einen umfassenden Überblick zu Hypertextsystemen findet man bei KUHLEN 1991, NIELSEN 1996, LENNON 1997 oder HAMMWÖHNER 1997.

Das erste Konzept eines Hypertextsystems ist in dem von BUSH 1945 entworfenen System *Memex* zu sehen, das in technischer Hinsicht und durch seine Entstehungszeit bedingt noch auf traditionellen Technologien beruhte, aber bereits wesentliche Elemente eines Hypertextsystems als Sammlung vernetzter Informationseinheiten einführte.

Das von Ted NELSON seit den 60er Jahren entwickelte System *Xanadu* ist wohl das älteste kontinuierlich weiterentwickelte Hypertextsystem.[9] Es kann als konzeptueller Vorläufer des World Wide Web angesehen werden, da es als weltweites verteiltes Publikationssystem modelliert ist und u. a. auf dem Prinzip der *Transklusion*, d. h. der dynami-

[9] Vgl. *http://www.xanadu.net/TECH/xuTech.html* und *http://www.udanax.com*. Seit 1999 ist Xanadu als *open source*-Software verfügbar, vgl. dazu VANNINI 1999 mit einer kritischen Würdigung der Entwicklung von Xanadu vor dem Hintergrund der Entwicklung des World Wide Web zum allgemein verfügbaren Hypertextsystem.

schen Einbettung verteilter Publikationsinhalte aufgebaut ist, ein Konzept, das im Sinne der WWW-Standards erst mit den weitgehend *technisch* noch nicht realisierten Konzepten der *Extensible Link Language* (XLink) seine Entsprechung findet (vgl. unten Kap. 5.2.4). Schematisch lässt sich das Prinzip der Transklusion wie in Abbildung 6 darstellen, in der Teile von Dokument B per Transklusion in Dokument A eingebettet sind:

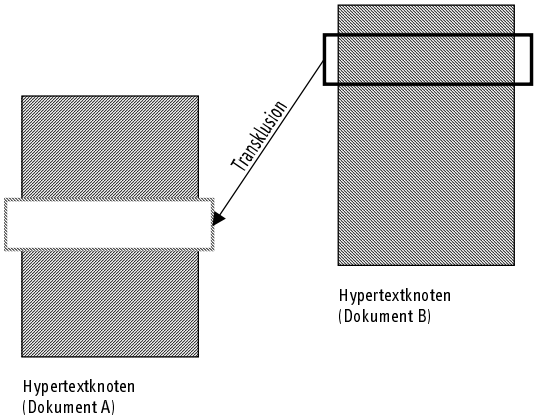

Abbildung 6: Prinzip der Transklusion (nach NELSON 1994: 263, Abb. 9)

Die Datenhaltung erfolgt in *Xanadu* nicht mit Hilfe eines Knotenkonzepts, sondern über Verweise auf Zeichenfolgen. Neben einem Adressierungsschema, das weitgehend hardwareunabhängig („architekturneutral“) arbeitet, verfügt *Xanadu* bereits über ein Abrechnungsmodell, mit dessen Hilfe der Abruf von Informationen über das Hypertextsystem in Rechnung gestellt werden kann und nimmt jüngere Entwicklungen auf dem Gebiet elektronischer Zahlungssysteme vorweg. In Verbindung mit einem Modell der Urheberrechte für Hypermedia-Publikationen entwirft NELSON ein differenziertes Abrechnungs- und Lizensierungverfahren (*Transpublishing*; *Transcopyright*, vgl. NELSON 1999B und SAMUELSON & GLUSHKO 1991: 41 ff.). Zur semantischen Klassifikation und Strukturierung der in Xanadu adressierten Dokumente dient eine Dezimalklassifikation.

Xanadu hat sich als Hypertextsystem allerdings nicht durchsetzen können und seinen Status als Forschungssystem mit Implementierungen auf UNIX- und VAX-Rechnern nie verloren, was allerdings nichts an der Tatsache ändert, dass von diesem System wesentliche Anregungen für die Hypertextforschung, die Entwicklung des World Wide Web und auch des elektronischen Publizieren ausgingen.

Das *Knowledge Management System* (KMS) wurde seit den 70er Jahren an der Carnegie-Mellon University von Robert AKSCYN und seinen Mitarbeitern entwickelt und stellt ein frühes Beispiel für ein hypertextbasiertes, groupware-orientiertes Autorenwerkzeug mit vielfältigen Anwendungen in

- *electronic publishing*,
- Software-Dokumentation,
- Projektverwaltung und
- kooperativem Arbeiten (*computer supported cooperative work*, CSCW).

dar (vgl. AKSCYN, MCCRACKEN & YODER 1988). Das System baut auf einem Datenmodell auf, das sich an „Bildschirm-Arbeitsblättern“ mit beliebigem Inhalt orientiert. Es stellt eine eigene Programmiersprache zur Verfügung und ermöglicht das verteilte kooperative Erstellen und Integrieren von Texten. Als Forschungssystem stehen Implementierungen auf Unix-Rechnern zur Verfügung. KMS ist bemerkenswert als ein frühes hypertextbasiertes System mit Möglichkeiten der verteilten Kooperation (*groupware*-Aspekt).

Das von Hermann MAURER entwickelte Hyper-G/HyperWave-System versucht, verschiedene Internetdienste wie WAIS,[10] Gopher und das World Wide Web durch eine einheitliche modellierte Hypertextumgebung zusammenzuführen. Das System legt ein einheitliches Datenmodell zugrunde und führt eine konzeptuelle Trennung der Inhalte und ihrer Hypertextverknüpfungen ein, wodurch sich als Konsequenz verbesserte Verwaltungsmöglichkeiten und differenzierte Zugriffsformen ergeben. Damit sind Entwicklungen vorweggenommen, die sich erst in jüngeren WWW-Spezifikationen wiederfinden, wie die erweiterten Spezifikationsmöglichkeiten für externe Verknüpfungen in XLink, die sich in modifizierter Form im Modell dynamischer elektronischer Bücher finden (vgl. unten Kap. 5.2.4 und Kap. 10.3).

Schließlich sei das Konstanzer Hypertextsystem (KHS, HAMMWÖHNER 1997) erwähnt, das eine Umsetzung der Konzepte *offener Hypertextsysteme* darstellt und verschiedene Verfahren zur dynamischen und wissensbasierten Integration von Inhalten in Hypertexte sowie seine Anwendung in unterschiedlichen Nutzungsszenarien beschreibt. Dazu gehören unter anderem verschiedene Möglichkeiten für die Präsentation des Hypergraphen z. B. durch interaktive Übersichtsnetze und Modelle für die Beschreibung von Makrostrukturen innerhalb des Hypertexts. Wie HyperWave hat sich KHS zu einer Anwendung im World Wide Web weiterentwickelt, d. h. seine Konzepte lassen sich mit den im World Wide Web verfügbaren Standards umsetzen (HTML als Dokumentformat; HTTP als Übertragungsprotokoll; Webbrowser als Viewing-System, vgl. HAMMWÖHNER 1997 und *http://www.inf-wiss.uni-konstanz.de/FG/IV/KHS/*). Es steht hier prototypisch für eine neuere Generation offener Hypertextsysteme (OHS).[11]

2.2.1.8 Fazit

Hypertexttechnologie hat sich als ein generelles Konzept der Organisation von Information durchgesetzt; ein elektronisches Buch, das nicht wenigstens die Grundkonzepte der nicht-linearen Verknüpfung von Informationseinheiten aufgreift, ist kaum mehr vorstellbar. ENGELBART verallgemeinert den Hypertextgedanken, wenn er feststellt:

> Everything in the Work Environment is Live Hypertext Stuff: All knowledge products, such as email, notes, source code, to-do lists, work breakdown structures, status reports, design documents, user guides, trouble reports, and other are inherently hyperdocument structures. [...]. [ENGELBART 1995: 30].

Auf ähnliche Weise charakterisiert Tim BERNERS-LEE seine Motivation bei der Entwicklung des World Wide Web: „The vision I have for the web is about anything being potentially connected with anything“ (BERNERS-LEE 1999B: 1). Das World Wide Web als eine Art universelles Hypertextsystem greift bisher nicht alle in der Hypertexttheorie entwickelten Konzepte auf. Vergleicht man die Standardarchitektur eines Web-basierten Informationssystems mit dem Dexter-Referenzmodell oder den mächtigeren (offenen) Hypertextsystemen wie HyperWave oder KHS, so muss man feststellen, dass im World Wide Web nur einfache Hypertextkonzepte verwirklicht sind. Tatsächlich stellen die Arbeiten zu offenen Hypertextsystemen und Informationssystemen im World Wide Web zwei bis-

---

[10] *Wide Area Information Server*, ein vor Verbreitung des World Wide Web gebräuchliches IR-System im Internet, vgl. SCHELLER et al. 1994: 227 ff.

[11] HAMMWÖHNER 1997: 105 ff. gibt einen Überblick zu Systemen und Modellen im Umfeld offener Hypertextsysteme; vgl. auch NÜRNBERG & LEGGETT & WIIL 1998, die Desiderata für die Weiterentwicklung von OHS zu komponentenbasierten Systemen formulieren (*component-based open hypermedia systems*, CB-OHS).

lang zumindest partiell parallel verlaufende Entwicklungsstränge dar, die sich erst schrittweise annähern. NÜRNBERG & ASHMAN 1999: 84 ff., vergleichen die Grundkonzepte offener Hypertextsysteme (OHS) mit den Charakteristika von WWW-Anwendungen und kommen zu dem Schluss, dass der entscheidende Unterschied in dem Ausmaß der expliziten Strukturrepräsentation (*structure awareness*) sowie den besser ausgebildeten Middleware-Konzepten der OHS liegen. Insofern soll der Versuch, eine Architektur für dynamische elektronische Büchern mit den Mitteln der Standards des World Wide Web zu definieren, einen Beitrag dazu leisten, Konzepte aus der Hypertextforschung aufzugreifen und sie auf Web-basierte Informationssysteme zu übertragen.

2.2.2 Multimediasysteme

Ein zweiter Aspekt dynamischer elektronischer Bücher ist ihr Bezug zu Multimediatechnologie. Von Multimediasystemen spricht man, wenn in einem Informationssystem Daten in unterschiedlichen Medien repräsentiert sind. Den Begriff *Medium* kann man in verschiedener Hinsicht verstehen. STEINMETZ 1999: 7 ff. unterscheidet den Medienbegriff nach den Dimensionen

- *Perzeption*, d. h. Wahrnehmung durch den Menschen,
- *Repräsentation*, d. h. interne Kodierung im Rechner,
- *Präsentation*, d. h. für die Darstellung von Information erforderliche Hilfsmittel,
- *Speicherung*, d. h. Datenträger für die Informationsspeicherung und
- *Übertragung*, d. h. die für den Informationsaustausch verwendeten Medien (z. B. Netze).

Für die Modellierung eines elektronischen Buchs als Anwendungsfall für Multimediatechnologie sind diese Dimensionen relevant. Im Rahmen dieser Arbeit sollen jedoch vor allem Repräsentation und Präsentation mit den Mitteln der deklarativen Informationsauszeichnung Beachtung finden. Weitergehend definiert STEINMETZ ein Multimediasystem wie folgt:

> Ein Multimediasystem ist durch die rechnergesteuerte, integrierte Erzeugung, Manipulation, Darstellung, Speicherung und Kommunikation von unabhängigen Informationen gekennzeichnet, die in mindestens einem kontinuierlichen (zeitabhängigen) und einem diskreten (zeitunabhängigen) Medium kodiert sind. [STEINMETZ 1999: 13].

Diese Definition ist insofern restriktiv, weil sie die Kombination kontinuierlicher und diskreter Medien fordert; als Multimediasysteme werden im landläufigen Sinne aber auch zahlreiche Anwendungen bezeichnet, deren Schwerpunkt auf der Speicherung, Distribution oder Darstellung (ausschließlich) kontinuierlicher Medien liegt (z. B. Multimedia-Datenbanken und streaming-Server für Audio- und Videodateien; Information Retrieval in Filmsequenzen; elektronische Bücher, die zusätzlich zum Text lediglich verschiedene diskrete Bildformate enthalten). SCHULMEISTER führt in Erweiterung der oben genannten Definition von Multimediasystemen zusätzlich den Begriff der Interaktivität als Anforderungskriterium ein:

> Durch die Digitalisierung und die Rechnermanipulation wird die Sequenzialität der verschiedenen Medien aufgehoben, ihre Reihenfolge kann willkürlich manipuliert, und sie können aufgrund der Integration im Computer über eine Benutzerschnittstelle interaktiv zugänglich gemacht werden. Dies weist der Interaktivität zwischen Benutzer und System eine hervorgehobene Rolle zu. [...] Ohne diesen Aspekt der Interaktion ist die Definition

> von Multimedia unzureichend. Wir sollten von Multimedia stets als von einem interaktiven Medium sprechen. [SCHULMEISTER 1997: 21 f.].

Auch wenn man darauf hinweisen muss, dass sich die Definition von SCHULMEISTER einschränkend auf Multimedia*anwendungen* bezieht, während STEINMETZ Multimediasysteme allgemeiner definiert, kann man ihr durchaus folgen: Eine nicht interaktive Multimediaanwendung böte dem Benutzer weniger Freiheitsgrade bei ihrer Bedienung und wäre letztlich auf eine rein sequentielle Präsentation (eines Films, einer Audiodatei) reduziert, was keine Neuerung gegenüber traditionellen Medienpräsentationsformen darstellte. Ist Interaktivität nicht konstitutiv für Multimediasysteme im Allgemeinen, so ist sie doch ein unverzichtbares Qualitätsmerkmal für elektronische Bücher mit multimedialen Inhalten. In der vorliegenden Arbeit zur Entwicklung eines dynamischen elektronischen Buchs sind die Kriterien, die an ein Multimedia-Informationssystem gestellt werden müssen, dadurch erfüllt, dass Animationen und Simulationen physikalischer Phänomene als *interaktive* und *zeitabhängige* Informationsvisualisierungen unter den Begriff Multimediakomponente fallen. Dabei werden allerdings die *Grundlagen* der Multimedia-Technologie (Datenformate, Kompressionsverfahren, Synchronisationstechniken etc.) nicht näher betrachtet; Deklarative Kodierungssprachen für multimediale Inhalte werden in Teil II dieser Arbeit im Rahmen der Diskussion SGML-basierter Standards für die Informationsaufbereitung vorgestellt.

2.2.3 Elektronische Lehr- und Lernsysteme

Die dritte relevante Systemkategorie neben Hypertexttechnologie und Multimediasystemen sind elektronische Lehr- und Lernsysteme, ein Bereich, dessen Entwicklung sich lange zurückverfolgen lässt (vgl. SCHULMEISTER 1996: 87 ff.) und der durch die Möglichkeit, Lehr- und Lernsysteme im World Wide Web zu realisieren, zusätzlich an Aufmerksamkeit gewonnen hat.[12] Diese Systeme sind für die Entwicklung dynamischer elektronischer Bücher relevant, da die Verwendung elektronischer Bücher als Lernmaterialien, d. h. als ein Wissensspeicher in einem Lernprozess eine der wichtigsten Anwendungen für elektronische Publikationen darstellt. Elektronische Bücher sind als *Wissensspeicher* von *Lehr-* und *Lernsystemen* abzugrenzen, die die vollständige Durchführung eines Lernprozesses einschließlich der hierfür erforderlichen administrativen Funktionen ermöglichen. Hierzu gehören u. a.

- Tele-Learning-Anwendungen (vgl. ZIMMER 1997) und virtuelle Universitäten (vgl. CHANG, HASSANEIN & HSIEH 1998)
- Kursverwaltungssysteme und Computer-Based Training-Systeme (vgl. GOLDBERG, SALARI & SWOBODA 1996),
- Anwendungen für die automatische Auswertung von Aufgaben (vgl. KISS 1998).

Im Unterschied zu solchen Anwendungen dienen elektronische Bücher primär der *Präsentation* von Lerninhalten und der *Interaktion* mit ihnen, nicht der Abwicklung des gesamten Lernprozesses mit elektronischen Mitteln. Sie sind als Lernmaterial eine Voraus-

[12] Zahlreiche Beispiele für WWW-basierte Instruktionssysteme finden sich in den Proceedings der WWW-Konferenzen; dabei handelt es sich bisher – in Ermangelung geeigneter Standards – in der Regel um Insellösungen, vgl. GOLDBERG, SALARI & SWOBODA 1996, HENZE et al. 1999 und den Überblick zu adaptiven Lernsystemen im World Wide Web bei BRUSILOVSKY 1999 [bes. 23 f., Tab. 1-3].

setzung für einen solchen Lernprozess; deshalb spielt der Zusammenhang mit Lehr- und Lernsystemen in diesem Rahmen eine Rolle. Es muss untersucht werden,

1. inwiefern durch die Auszeichnung der Lerninhalte mit Metainformation ihre Integration in Lernsysteme ermöglicht werden kann und
2. inwiefern über geeignete Schnittstellen Dienste eines Lernsystems in das elektronische Buch integriert werden können, ohne es selbst zu einem Lernsystem im oben angedeuteten Sinn weiterzuentwickeln.

Elektronische Bücher können Teil einer Infrastruktur für computergestütztes Lernen sein, realisieren aber nur einige der dafür erforderlichen Funktionen. Dies wird deutlich, wenn man elektronische Bücher in ein Gesamtmodell für die Entwicklung von Lerntechnologie einordnet: Ein allgemeines Modell für elektronische Lehr- und Lernsysteme ist im Rahmen der Standardisierungsbemühungen des IEEE *Learning Technology Standards Committee* (IEEE 1484) entwickelt worden (vgl. FARANCE & TONKEN 1998 und *http://ltsc.ieee.org*); es ist in folgende Ebenen gegliedert (vgl. Abbildung 7):

1. Allgemeines Modell des Lerners und seiner Interaktion mit der Umwelt,
2. Annahmen über das menschliche Lernen mit Hilfe von elektronischen Lernsystemen,
3. abstrakte Systemarchitektur für Lehr- und Lernsysteme,
4. Zielszenarien für die Anwendung (Untermengen von Ebene 3) und
5. technische Realisierung.

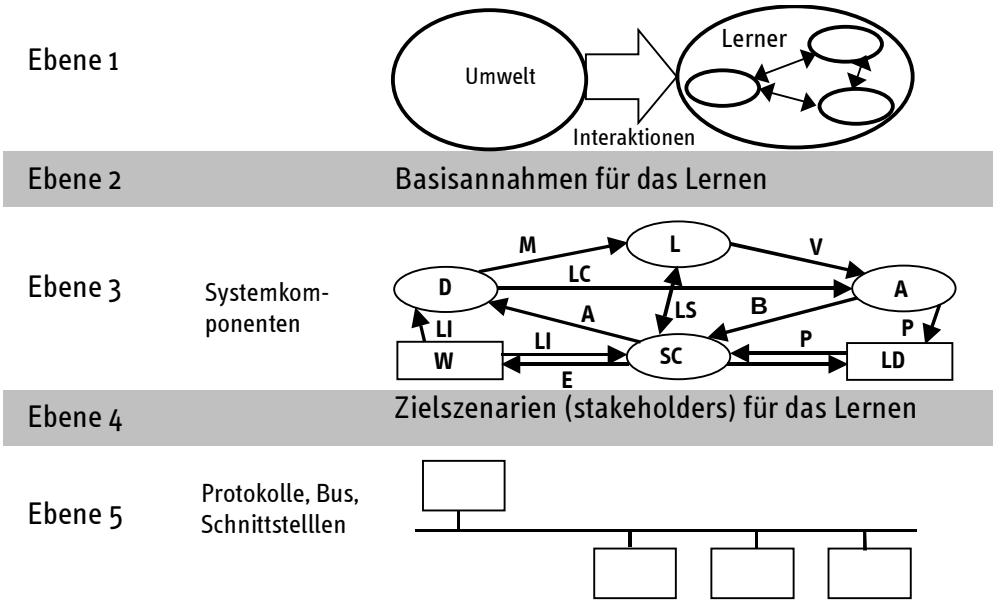

Abbildung 7: Ebenen der Learning Technology Systems Architecture (LTSA, nach FARANCE & TONKEL 1998: 31, Abb. 12)

Für die Einordnung elektronischer Bücher ist vor allem Ebene drei relevant, die die Komponenten und Informationsflüsse in einem elektronischen Lernsystem spezifiziert (vgl. Abbildung 8).

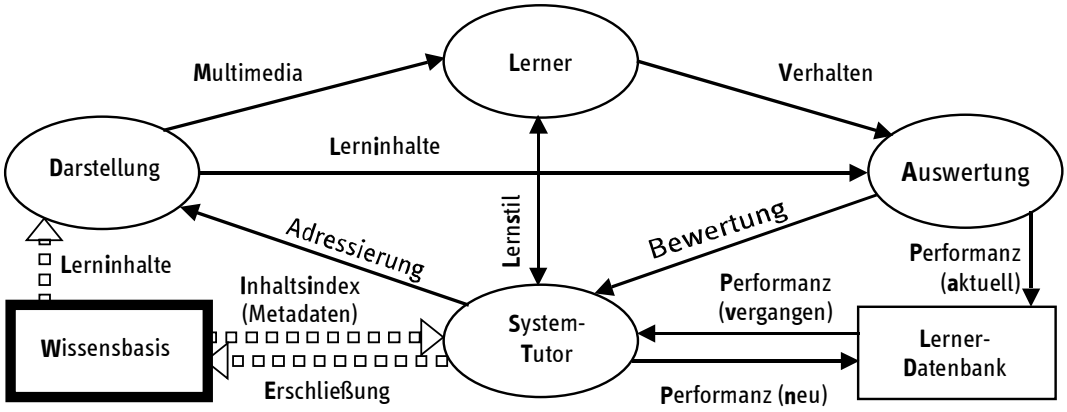

Abbildung 8: Systemkomponenten der Learning Technology Systems Architecture (LTSA, nach FARANCE & TONKEL 1998: 62, Abb. 41)

Das elektronische Buch realisiert die Komponenten *Wissensbasis* und *Darstellung* (Informationsfluss *Multimedia*). Durch die im elektronischen Buch enthaltenen Datenbeschreibungen (deklaratives Markup und Metadaten) sowie die Spezifikation von Schnittstellen zu externen Diensten (d. h. in diesem Fall zu anderen Komponenten eines Lernsystems) lässt es sich in das Gesamtmodell einordnen. Dabei kann es im Sinne dieses Modells (Ebene 4) in unterschiedlichen Szenarien bzw. Lehr- und Lernsituationen zum Einsatz kommen; das hier entwickelte Modell trägt dem insofern Rechnung, als nicht ein vollständiges, auf eine bestimmte Lernsituation zugeschnittenes System entworfen wird, sondern Schnittstellen zur Integration von Diensten angegeben werden, die es erlauben, ein elektronisches Buch als Wissensspeicher in unterschiedlichen Lehr- und Lernkontexten einzusetzen und durch unterschiedliche Dienste zu ergänzen.

Neben dieser Einordnung in eine Lehr- und Lerninfrastruktur muss die Frage beantwortet werden, durch welche Eigenschaften und Funktionen sich dynamische elektronische Bücher als Wissensspeicher für computergestütztes Lernen von ihren Pendants im Printmedium unterscheiden bzw. welche Mehrwerte sie für den Lerner erzielen können. Hierzu zählen die folgenden Annahmen:

1. Die elektronische Bereitstellung von Lernmaterialien erlaubt neue *Zugangs*- und *Distributionsmöglichkeiten* (*teleteaching*; vereinfachter Zugang über Datennetze) und kann auch ohne Betrachtung alternativer Aufbereitungs- und Darstellungsformen des Wissens einen Mehrwert erreichen.
2. Es gilt die *Vermutung*, dass durch multimediale Inhalte eine motivationsfördernde Alternative zu traditionellen Präsentationsformen für Wissen geschaffen wird. WEIDENMANN 1997: 69 ff., kennzeichnet die einfache Annahme eines prinzipiell positiven, da motivierenden Effekts durch Multimedia als „naive“ Basishypothese und gibt einen Überblick zu den Erkenntnissen über Einfluss multimedialer bzw. multimodaler Präsentation auf den Wissenserwerb gibt. Die Forschung steht bei der pädagogischen Bewertung von Multimedia aber noch am Anfang, vgl. SCHULMEISTER 1997: 335 ff., STEINMETZ 1999: 822 ff.
3. Die verschiedenen Formen multimedialer Ergänzungen in dynamischen elektronischen Büchern, insbesondere interaktive Animationen und Simulationen, stellen kein einfaches *Äquivalent* einer textuellen Darstellung dar, sondern konstituieren einen neuen Typus von Lernmaterial, der in integrierter Präsentation die klassischen Formen Text und Bild sinnvoll ergänzen und erweitern kann. Wie die Vielfalt multimedialer Ler-

ninhalte gerade im Bereich der Informatik zeigt, ist dies ein weit verbreiteter Ansatz, vgl. GLOOR 1997: 226 ff. (Visualisierung von Algorithmen), BURGER, MECKLENBURG & ROTHERMEL 1998: 181 ff. (animierte Darstellung von Kommunikationsprotokollen) oder PAPE & SCHMITT 1998: 104 ff. (interaktive Simulation von Automaten).

4. Die Vernetzung der Inhalte durch Hypertextverknüpfungen stellt ein die Interaktion vereinfachendes Strukturierungsmittel dar, da durch sie traditionelle Verweissysteme gedruckter Bücher wie Fußnoten oder Querverweise einfacher handhabbar sind. Darüber hinaus erlauben Hypertextverknüpfungen vielfältigere Nutzungsformen des Wissensmaterials (*browsing*).

Evaluierungsstudien, die verschiedene Darstellungsmodi für Lernmaterial in multimedialen Anwendungen miteinander vergleichen, deuten auf Vorzüge einer Integration von Multimediakomponenten hin (vgl. MORENO & MAYER 1999). Interaktive und WWW-basierte Lernmaterialien sind ein relativ junges Gebiet, für das Evaluierungskriterien erst noch entwickelt werden müssen und über dessen pädagogische Vor- und Nachteile bisher kein abschließendes Urteil möglich ist.[13] Die folgende Einschätzung, die auf der Basis einer Analyse von mehr als hundert Websites entstand, ist für den gegenwärtigen Entwicklungsstand webbasierter Lernmaterialien zutreffend:

> Although there are a large variety of techniques such as simulations using Java [...] they are sparingly applied [...]. We conclude that the web is not yet being used to its full potential with respect to the conversational framework. [...] Educational multi-media web content developers are probably of two categories:
> - the educationalist who is technologically naive; and
> - the technologist who is educationally naive. [ADHVARYU & BALBIN 1999: 53].

Eine Diskussion der Motivation für die Entwicklung unterschiedlicher Typen multimedialer Komponenten und ihrer Einordnung in den Lernprozess im Rahmen des Referenzprojekts findet sich in Kap. 14.1.

Im Rahmen dieser Arbeit geht es nicht darum, ein vollständiges Lehr- und Lernsystem zu spezifizieren. Vielmehr wird das elektronische Buch als Wissensspeicher modelliert, der Schnittstellen zu Lehr- und Lernanwendungen aufweist und durch die dynamischer Integration unterschiedlicher Dienste in solche Anwendungen integriert werden kann. Diese bewusste Einschränkung folgt der Überlegung, dass es im Gegensatz zu ausschließlich für einen ausgewählten Informationsbestand entwickelten Lehr- und Lernsystemen (z. B. interaktives Lehrmaterial zu einer Lehrveranstaltung) für ein elektronisches Buch unterschiedliche Einsatzszenarien und Verwendungsformen gibt, z. B.

- Einsatz des elektronischen Buchs als reines Referenzmaterial,
- als Lehrwerk für das Selbststudium,
- im Rahmen einer *telelearning*-Anwendung (*distance learning*),
- als direkte Grundlage für die Durchführung eines Kurses.

Der jeweils erforderliche Nutzungskontext unterschiedlicher Verwendungsszenarien kann von den Autoren bzw. dem Distributor (z. B. dem Verlag oder dem Betreiber einer digitalen Bibliothek) nicht *ex ante* vollständig, d. h. in der Gesamtheit der jeweils erforderlichen oder wünschenswerten Dienste modelliert werden.

---

[13] Vgl. etwa die in *IEEE Computer* jüngst geführte Debatte, ob das World Wide Web „bereits soweit gereift“ sei, dass es sinnvollerweise als Plattform für Lernsysteme dienen kann, vgl. BORK & BRITTON 1998, CRAVENER 1998, KESSLER, ROSENBLAD & SHEPARD 1999.

2.2.4 Web-basierte Informationssysteme

Das World Wide Web hat sich in den letzten Jahren von einem hypertextbasierten Kommunikationsmedium für den Informationsaustausch zwischen Wissenschaftlern zu einer Softwareplattform weiterentwickelt, die die Realisierung grundsätzlich beliebiger Informationssysteme zuläßt. Dies machen ISAKOWITZ, BIEBER & VITALI 1999: 78 deutlich:

> Thus, a Web platform has transformed itself in the past few years from a mere marketing presence to a platform that can support all facets of organizational work. As a result, important information system efforts are geared increasingly toward exploiting the benefits of this platform, leading to the development of information systems based on Web technology, which we call „Web-based information systems (WISs)."

Elektronische Publikationen und Bücher stehen den ursprünglichen Zielen des World Wide Web naturgemäß sehr nahe (vgl. LIE & SAARELA 1999: 95); die jüngere Entwicklung zu einer Plattform für Informationssysteme hat eine Parallele zu elektronischen Büchern in der Tatsache, dass sich diese durch die Anreicherung mit interaktiven Komponenten und die Integration von Diensten von der reinen Informations*präsentation* zu interaktiven Softwaresystemen weiterentwickeln: Ein elektronisches Buch ist in diesem Sinn kein *statischer Wissensspeicher*, der digital gespeichert ist und ggf. auch über eine digitale Bibliothek abgerufen werden kann, sondern es ist ein selbständiges Informationssystem mit eigener Benutzerschnittstelle, einer eigenen Softwarearchitektur und eingebetteten Komponenten.[14]

2.3 Typen elektronischer Publikationen

Neben der Einordnung in den Zusammenhang verschiedener Klassen von Informationssystemen stellen die unterschiedlichen Typen *elektronischer Publikationen* ein weiteres Vergleichskriterium dar, nach dem elektronische Bücher beurteilt werden können. Elektronische Publikationen konstituieren denjenigen Teilbereich des elektronischen Publizierens, bei dem als Endprodukt eine elektronisch zu distribuierende und zu nutzende Publikation als Medium für den Zugang zu Information steht,[15] wie es folgende Definition zum Ausdruck bringt:

> We define an *electronic publication* as a document distributed primarily through electronic media. The distribution medium is the defining factor, because an electronic publication may well be printed to be read, and may be circulated postpublication in printed form. [KLING & MCKIM 1999: 891]

Der Dokumentenbegriff ist für elektronische Zeitschriften, der am weitesten verbreiteten Form wissenschaftlichen Publizierens mit elektronischen Mitteln, adäquat. Für komplexere elektronische Bücher mit multimedialen Inhalten tritt der Aspekt der Distribution aber zurück, da keine Äquivalenz zwischen elektronischen und digitalen Inhalten mehr besteht; interaktive Elemente als Mehrwert elektronischer Bücher lassen sich im Printmedi-

---

[14] Schon BIER & GOODISMAN 1990 machen auf die Tatsache aufmerksam, dass elektronische Dokumente durch die Anreicherung mit interaktiven Elementen (Hypertextverknüpfungen, Controls) neben der Informationspräsentation auch die Funktion einer Benutzerschnittstelle zu einem Softwaresystem (dem Buchbetrachtungssystem) haben.

[15] Zur Problematik des Informationszugangs und seiner Ausprägungsformen vgl. MCCREADIE & RICE 1999A und 1999B, bes. 46 u. 48, die in ihrer Klassifikation von Informationsmodellen Bücher als praktische Umsetzung von Information als *Repräsentation von Wissen* ansehen.

um nicht vollständig darstellen (oder ausdrucken). Zu den Typen elektronischer Publikationen zählen:

- Elektronische Zeitschriften. Dabei ist zu unterscheiden zwischen genuinen elektronischen Zeitschriften, für die keine Printversion existiert (z. B. *j.ucs – Journal of Universal Computer Science*, vgl. *http://www.iics.edu/jucs*, *JoDI – Journal of Digital Information* (*http://jodi.ecs.soton.ac.uk/*) oder das *Dlib Magazine* (*http://www.dlib.org/dlib.html*),[16] und elektronischen Parallelfassungen von Printjournals. Elektronische Zeitschriften zeichnen sich durch folgende Merkmale aus:
  - Periodische Erscheinungsweise
  - Standardisierte Aufbereitung und inhaltliche Gestaltung
  - Ergänzung einer Printversion um zusätzliche Hyperlinks und Recherchemöglichkeiten
  - Orientierung am traditionellen Dokumentenbegriff und
  - Nutzung über digitale Bibliotheken.[17]

  Exemplarisch zeigt die folgende Abbildung die Startseite eines Artikels aus der elektronischen Zeitschrift Journal of Digital Information (*http://jodi.ecs.soton.ac.uk/Articles/v01/i02/editorial.shtml*):

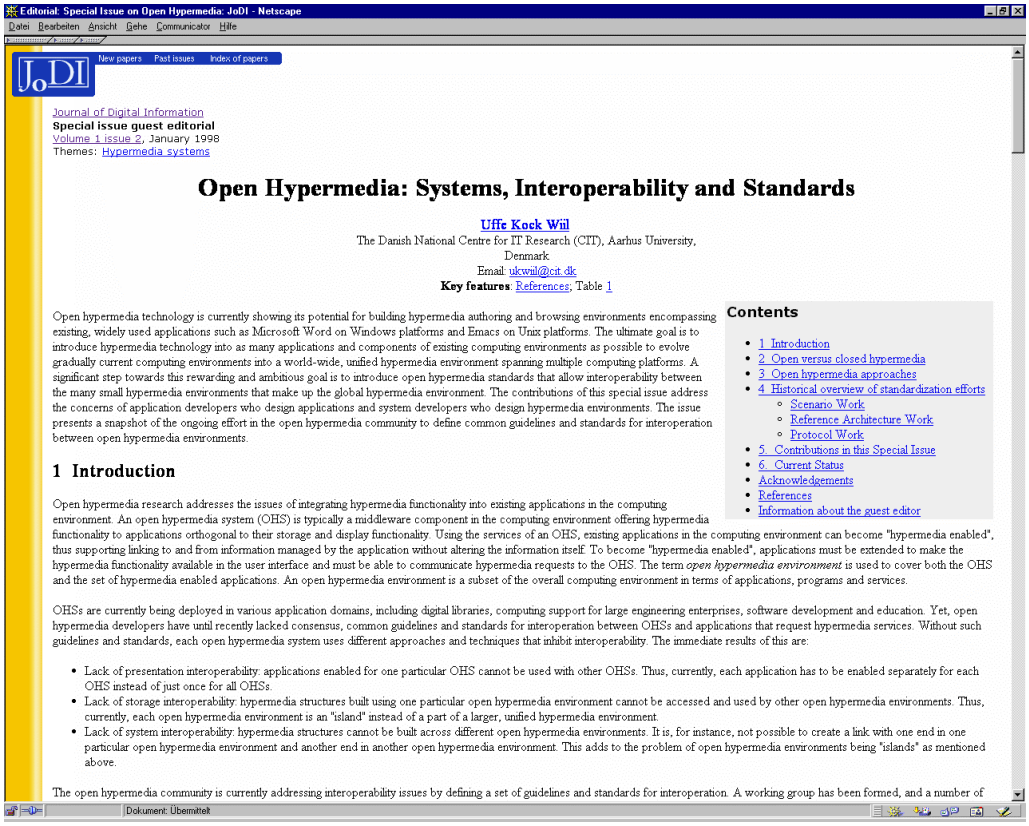

Abbildung 9: Beispielseite aus der elektronischen Zeitschrift Journal of Digital Information

16 Eine Übersicht zur Migration wissenschaftlicher Zeitschriften in das World Wide Web geben PEEK, POMERANTZ & PALING 1998.

17 Dies gilt besonders für die elektronischen Versionen von Printjournals, die oft (z.B. *ACM Digital Library* oder *Wiley Interscience Digital Library*) in an der Satzvorlage orientierten elektronischen Formaten wie dem *Portable Document Format* erscheinen.

- Online-Datenbanken, die z. B. als bibliographische Nachweissysteme seit langem als Netzwerkdienste der Fachinformationszentren ein wesentlicher Bestandteil der Infrastruktur des wissenschaftlichen Publizierens sind. Sie zeichnen sich durch einheitliche, an einer festen Dokumenten- bzw. Metadatenstruktur orientierte Einträge, eine kontinuierliche Erscheinungsweise, unterschiedliche Zugangswege (z. B. *telnet*, World Wide Web) und formale Abfragesprachen (z. B. *messenger*) aus.
- Volltextarchive periodischer Publikationen (Zeitungen, Zeitschriften) mit entsprechenden Erschließungsmechanismen zur gezielten Recherche.
- Elektronische Bücher als primär geschlossener Hypertext-Informationsbestand mit Rechercheschnittstelle und ggf. multimedialen Erweiterungen; sie existieren meist in Anlehnung an oder abgeleitet von gedruckten Fassungen.
- Informationssysteme im World Wide Web mit unterschiedlichen Verfahren für die Strukturierung und Gestaltung von informationellen Einheiten bzw. deren automatischer Erzeugung (z. B. unter Einbeziehung datenbankbasierter Dokumentenspeicher). Ein gutes Beispiel für diese Klasse elektronischer Publikationen sind Dokumentenserver an Hochschulen, die Lehr- und Forschungsmaterial (Skripte, Preprints, Graduierungsschriften etc.) verwalten und über eine einheitliche Benutzerschnittstelle zugänglich machen. Beispiele für diesen Typus sind die Dokumentenserver der Universitäten Leipzig und Karlsruhe (vgl. *http://www.dol.uni-leipzig.de*, *http://www.ubka.uni-karlsruhe.de/eva/*) sowie der Dissertationsserver der Humboldt-Universität zu Berlin (*http://www.ub.hu-berlin.de/cdrom/disspub.html*). Durch eine fachübergreifende Verfahrensregel kann dort die elektronische Publikation an die Stelle der gedruckten Fassung treten, vgl. *http://www.hu-berlin.de/presse/amb/amb98_14.html*.

Für diese nicht abschließende Liste von Publikationstypen gibt es nicht je ein einzelnes Kriterium, das die unterschiedlichen Typen voneinander unterscheidet. Beispielsweise kann man Hypertextsysteme als verallgemeinernde Kategorie betrachten, die WWW-Informationssysteme als typischen Vertreter enthalten. Außerdem ist anzumerken, dass sich im Umfeld unterschiedlicher Typen elektronischer Publikationen im wissenschaftlichen Bereich eine ganze Reihe von Kommunikations- und Publikationsverfahren etabliert hat, die die Verbreitung elektronischer Publikationen (z. B. *Technical Reports* oder *Preprints*) unterstützen (vgl. KLING & MCKIM 1996: 901 ff., bes. 902, Tab. 2). Nachfolgend soll eine Eingrenzung der Untersuchungen auf elektronische Bücher erfolgen.

3 Elektronische Bücher

> Buch, im Lateinischen Liber (d. i. Bast, weil man in der ältesten Zeit auf Bast schrieb), heißen mehrere zu einem Ganzen verbundene Blätter oder Bogen Papier. [Allgemeine deutsche Real-Encyklopädie für die gebildeten Stände. 11. Auflage, Leipzig: F. A. Brockhaus 1864, Bd. 3, S. 803]

Unter einem *elektronischen* Buch soll ein im elektronischen Medium repräsentierter Informationsbestand verstanden werden, dessen Inhalte von einem oder mehreren Autoren verfasst und/oder redaktionell bearbeitet werden, und der durch ein geeignetes Betrachtungssystem mit Hilfe eines Computers genutzt werden kann. In der Regel sind die Inhalte eines elektronischen Buches durch einen in sich geschlossenen und strukturierten Text repräsentiert, der durch Ergänzungen unterschiedlichen Inhalts, Umfangs oder Formats erweitert sein kann: „Elektronische Bücher sind vom Konzept her Hypertexte mit "constraints" (Beschränkungen: weniger, reduzierte oder schematisierte Links)." (SCHULMEISTER 1997: 299). Diese weite Definition schließt eine große Bandbreite von Typen elektronischer Bücher ein und unterscheidet sich grundsätzlich von der am physischen Trägermedium orientierten Definition eines Buchs:[18] Von einer einfachen Textrepräsentation eines belletristischen Werks ohne jegliche Zusatzfunktion bis hin zu aufwendigen Multimediaproduktionen, bei denen einem ggf. wenig umfangreichen Basistext zahlreiche Ergänzungen (z. B. Bilder, Filme, Simulationen) beigegeben sein können.

DE DIANA & WHITE 1994 weisen darauf hin, dass die in einem elektronischen Buch enthaltenen Inhalte nicht formalisiert sind: „The term Electronic Book [...] refers to organized sets of datafiles (either in a database or in a hyperstructure), containing ‚informal knowledge', readable and understandable by humans" (DIANA & WHITE 1994: 95), während BARKER & MANJI 1991: 273 f. das Buchkonzept unabhängig vom Medium durch Verweis auf die Strukturierung der Inhalte definieren: „A book may be informally defined as being a corpus of textual or graphical material that is logically subdivided into chapters [...]. Chapters are, in turn, divided (both logically and physically) into pages. Pages may contain just text, sound or pictures; alternatively, they may embed a combination of these communicative forms."

Aus diesen Definitionen werden wesentliche Eigenschaften elektronischer Bücher deutlich: Sie enthalten nicht vollständig formalisierbare Inhalte, verfügen i. d. R. über eine hierarchische Strukturierung, könnten multimediale Elemente beinhalten und sind stärker strukturiert als Hypertexte. In der Regel verfügen elektronische Bücher zusätzlich über Erschließungs- und Recherchehilfsmittel (intellektuelle Beschlagwortung/Register, Suchfunktion).

Man kann elektronische Bücher – wie bereits ausgeführt – sowohl unter den Begriff Informationssystem als auch unter den Begriff Hypertextsystem (bzw. Anwendung eines Hypertextsystems) subsumieren. Es stellt sich jedoch die Frage, inwieweit sich das elektronische Buch als Phänomen überhaupt vom Hypertext im Allgemeinen abgrenzen lässt. Offensichtlich ist, dass eine Überlappung in den meisten Kriterien vorliegt (vgl. unten

[18] Diese scheint als so selbstverständlich zu gelten, dass etwa RÖHRING (1997) in seinem Standardwerk „Wie ein Buch entsteht" den Begriff als solchen nicht thematisiert oder definiert.

Kap. 3.2).[19] Im Vergleich mit der von der UNESCO verabschiedeten Definition des gedruckten Buchs als „Druckerzeugnis von mehr als 49 Seiten Umfang“[20] kann man feststellen, dass diese an der Aufteilung der Informationseinheiten auf das physische Trägermedium orientierte Definition schon für das gedruckte Buch wenig hilfreich ist; eine analoge Definition, „ein elektronisches Buch ist ein im elektronischen Medium repräsentiertes Publikationsprodukt, das aus wenigstens 49 Darstellungseinheiten besteht“ ist aufgrund des noch unbestimmteren Begriffs *Darstellungseinheit* noch weitaus unbefriedigender. Im Verlagswesen spricht man i. d. R. von einem *Werk*, wenn auf den Inhalt eines Buches Bezug genommen wird, von einem *Buch* nur dann, wenn das konkrete physische Objekt gemeint ist.

Soweit ein elektronisches Buch auch im elektronischen Medium *genutzt* wird, kann es seine Existenzberechtigung nur über die Erzeugung eines *Mehrwertes* erlangen, da aufgrund der technischen Einschränkungen der Betrachtungssysteme die Lesesituation im Vergleich mit dem gedruckten Buch deutlich schlechter ist (schlechte ergonomische Randbedingungen: etwa 72 dpi Auflösung bei Darstellung von Text am Bildschirm; geringer Informationsumfang pro Darstellungseinheit Bildschirmseite; Bindung an ein wenig flexibles Betrachtungssystem (Bildschirm)).[21] Dem stehen allerdings auch Nachteile des *gedruckten* Buchs gegenüber, die BARKER 1992: 139 auflistet:

- difficult to reproduce
- expensive to disseminate
- difficult to update
- single copies cannot be easily shared
- easily damaged and vandalised
- bulky to transport
- embedded material is static and unreactive
- cannot utilise sound
- cannot utilise animation or moving pictures
- unable to monitor reader’s activity
- cannot assess reader’s understanding
- unable to adapt material dynamically

Ein Mehrwert kann in den oben genannten elementaren *searching*- und *browsing*-Funktionen liegen und schließt eine einfache 1:1-Übertragung gedruckter Bücher in das elektronische Medium aus.[22] Der Mehrwert kann sich qualitativ und/oder quantitativ nach folgenden Kriterien ergeben:

---

[19] SEILER 1992: 20 f., 27 f. löst dieses Definitionsproblem durch Einführung des Konzeptes „digital electronic medium“ als Oberbegriff für unterschiedliche Erscheinungsformen elektronischer Publikationen und prognostiziert: „The ascendance of DEM signals the end to print in general and printform books in particular.“

[20] Das Buch wird als "a non-periodical printed publication of at least 49 pages, exclusive of the cover pages" definiert (UNESCO 1964: 18), vgl. CUMMINGS et al. 1992: Kap. 5, Fn. 24.

[21] Vgl. VALAUSKAS 1994: 45: „It will be some time before a computer will be as physiologically and aesthetically pleasing as paper. We will never hear of someone curling up with a good monitor to read in bed, or learn of an electronic book that’s a real ‘window turner’.“ Ähnlich SEILER 1992: 19: „What threatens most is the loss of the physical object: not being able to hold books, to turn their pages, and to walk and search among shelves of them stacked in arranged fashion.“ Vgl. BÜTTEMEYER 1995: 150.

[22] Vgl. PETERS 1998: 88: „Sehr viele CD-ROMs, die Verlage in ihrer ersten Technologiebegeisterung konzipiert haben, sind objektiv Fehlgriffe. Wer ein Buch auf eine CD transferiert, muss

- Medien und Darstellung,
- Umfang/Quantität,
- Erschließbarkeit (*browsing and searching*),
- Interaktivität und differenzierte Nutzungsformen,
- Offenheit für externe Ressourcen und Dienste (dynamische Bücher),
- funktionale Erweiterung z. B. durch integrierte Softwarekomponenten und
- vereinfachte Distribution und Integration in elektronische Bibliotheken und Nachweissysteme.

Diese Unterschiede zwischen elektronischen und gedruckten Texten verdeutlicht REINKING 1992:

- „Electronic texts can control readers‘ access to text during independent reading“ (REINKING 1992: 15);
- „Electronic texts permit readers and texts to interact“ (REINKING 1992: 17);
- „Computer-mediated texts may be structured differently than printed texts“ (REINKING 1992: 18);
- „Electronic texts make available a wide range of symbolic elements that can be integrated with written prose.“ (REINKING 1992: 19).

Bevor nachfolgend in Kap. 3.2 ein Begriffsraster für elektronische Bücher erstellt werden kann, soll der Blick auf den state-of-the-art elektronischer Bücher eine Vorstellung von der Vielfalt der zugrunde gelegten Konzepte vermitteln. Da eine normative Festlegung der Merkmale elektronischer Bücher ex ante nicht sinnvoll ist, wird ein induktives Verfahren angewandt: Aus der Fülle der Beispiele leitet sich ein Beschreibungsrahmen ab, in den sich unterschiedliche Typen elektronischer Bücher einordnen lassen. Er ist gleichzeitig der Ausgangspunkt für die Anforderungen an die Entwicklung dynamischer elektronischer Bücher, wie sie in den nachfolgenden Kapiteln zu diskutieren sein werden.

3.1 Elektronische Bücher – state-of-the-art

Die Darstellung des state-of-the-art elektronischer Bücher gliedert sich in drei Teile: Zunächst werden einige repräsentative Beispiele kommerzieller, d. h. von Verlagen in elektronischer Form publizierter elektronischer Bücher vorgestellt. Es schließt sich eine Übersicht über den Projektverbund „Weiterentwicklung des wissenschaftlich-technischen Lehrwerks zur multimedialen Wissensrepräsentation“ als konkretes wissenschaftliches Umfeld dieser Arbeit an; schließlich wird das *Referenzprojekt* mit seinen wichtigsten Charakteristika diskutiert.

3.1.1 Beispiele elektronischer Bücher

Ein vollständiger Überblick über elektronische Publikationen, die sich als elektronische Bücher i. e. S. bezeichnen lassen, verbietet sich aufgrund von Zahl und inhaltlicher Bandbreite solcher Medienprodukte. Es ist daher eine Eingrenzung auf thematisch und technologisch verwandte Ansätze, mithin auf elektronische Lehrbücher in technischen und naturwissenschaftlichen Disziplinen erforderlich. Eine grundlegende Einteilung kann dabei nach der Art der Erstellungswerkzeuge und der Distributionsweise erfolgen:

---

einen Zusatznutzen anbieten, der zwei Nachteile der CD-ROM mehr als ausgleicht: schlechtere Handhabbarkeit [...] und schlechtere Auflösung [...]“.

- CD-ROM-basierte elektronische Bücher, die mit Hilfe eines integrierten Autorensystem erstellt wurden und
- elektronische Bücher als Hypertexte im World Wide Web.

Nachfolgend ist die Betrachtung auf elektronische Bücher im World Wide Web eingeschränkt, da damit eine gute Vergleichsgrundlage mit dem hier vorgestellten Konzept gegeben ist und diese in den letzten Jahren sehr stark an Bedeutung zugenommen haben. Elektronische Bücher, die wie das Referenzprojekt die Digitalisierung ursprünglich als Printfassung erschienener Werke beinhalten, finden sich im deutschsprachigen Bereich in der im Rahmen der Fördervorhaben MeDoc bzw. InterDoc (vgl. BOLES et al. 1998A, HABER, MEYER & WEBER 1998) in den vergangenen Jahren aufgebauten digitalen Bibliothek der Informatik. Die dort verfügbaren Lehrbücher der Informatik, Naturwissenschaften (Physik, Chemie) und der Mathematik können als repräsentativ für den state-of-the-art bei der Hypertextkonversion von Lehrbüchern angesehen werden. Sie sind durch folgende Merkmale gekennzeichnet:

- Ihre Kodierungsgrundlage bildet die Hypertext Markup Language (HTML) als Ergebnis einer Transformation aus dem Erfassungsformat der Autoren, als Betrachtungssystem dient ein Webbrowser.
- Die Strukturierung erfolgt in der Regel auf der Basis der kleinsten Struktureinheit (unterste Gliederungsebene) des Buchs und teilweise mit Hilfe einfacher Konvertierungswerkzeuge, die standardisierte Textformate (MS Word, TEX) in HTML überführen. Eine stärker präsentationsbezogene Aufteilung (seitengerechtes Layout) liegt normalerweise vor.
- Sie weisen ein einfaches Navigationsmodell auf, das die lineare Abfolge betont und daneben die Inhaltshierarchie als mit Hypertextverknüpfungen versehene Einstiegs- und Navigationsmöglichkeit anbietet.
- Hypertext als Navigationskonzept ist nur unvollständig realisiert, d. h. es existieren Querverweise, die aber nicht vollständig durchgeführt sind und keine Typisierung aufweisen.
- Interaktive Komponenten fehlen oder sind nur als einfache Medienkomponenten (Film) enthalten; Dienste sind – von sehr einfachen Ausnahmen wie mailto-Verknüpfungen abgesehen – nicht in die elektronischen Bücher integriert.
- Eine Gestaltung auf der Basis eines klar erkennbaren *screen design*-Konzeptes fehlt, von wenigen Ausnahmen abgesehen (etwa die Referenzwerke des Harri Deutsch Verlags, vgl. STÖCKER 1997 und unten Abbildung 10);[23] die Gestaltungsmöglichkeiten der Webbrowser werden kaum oder in unzulänglicher Weise ausgenutzt (z. B. Fensterverwaltung, Einsatz von HTML-Frames).
- Die Suchfunktionalität als Volltextrecherche ist in die umgebende Infrastruktur der digitalen Bibliothek verlagert; für die Einzelwerke stehen Register mit Hypertextverknüpfungen zur Verfügung.
- Es bestehen keine Erweiterungsmöglichkeiten, d. h. die elektronischen Bücher stellen einen statischen Datenbestand dar.

Einige der genannten Defizite sind durch die beschränkten Ressourcen bei der Erstellung der elektronischen Bücher bedingt, da es sich bei ihnen anders als im nachfolgend zu

[23] Es ist sicher kein Zufall, dass es sich bei dieser Ausnahme um Werke handelt, die auch unabhängig von der Nutzung im World Wide Web als Multimedia-CD-ROM bzw. als Medienbündel (Kombination unterschiedlicher Medientypen wie (gedrucktes) Buch und CD-ROM) kommerziell vertrieben werden.

beschreibenden Projektverbund *Multimediabuch* nicht um Vorhaben mit umfangreichen technischen und personellen Ressourcen für die Entwicklung von Inhalten oder die Gestaltung der Bücher handelte. Abschließend zeigen die folgenden beiden Abbildungen zwei repräsentative Beispiele für in InterDoc enthaltene elektronische Bücher.

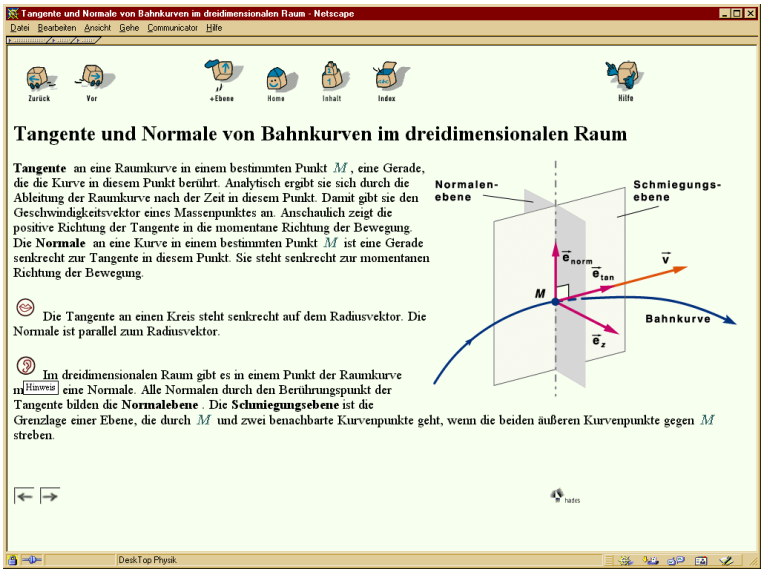

Abbildung 10: Beispielseite aus STÖCKER 1997 (InterDoc-Server TU München)

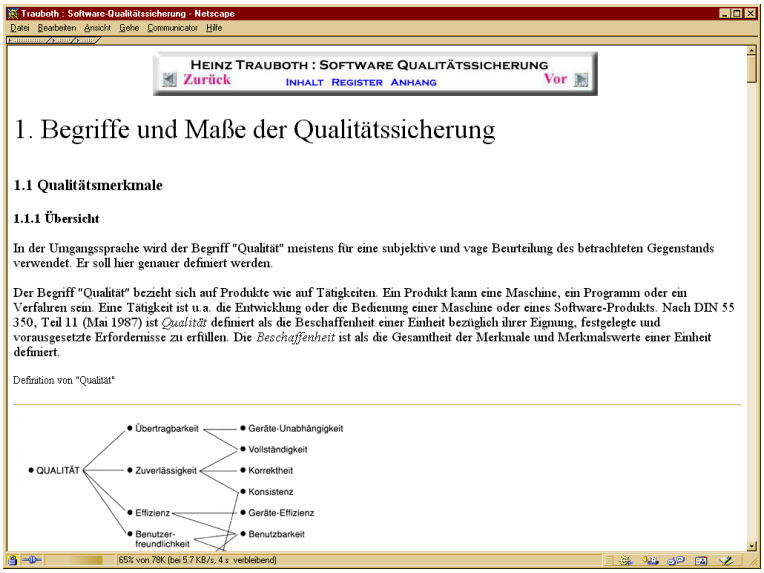

Abbildung 11: Beispielseite aus TRAUBOTH 1996 (InterDoc-Server Universität Leipzig)

3.1.2 Projektverbund Multimediabuch

Als wissenschaftlicher Kontext für die vorliegende Arbeit dient im engeren Sinn das Referenzprojekt *Multimediales Physikalisches Praktikum*, im weiteren Sinn der Projektverbund *Weiterentwicklung des wissenschaftlich-technischen Lehrwerks zur multimedialen Wissensrepräsentation* des Projektträgers Fachinformation beim Bundesministerium für Bildung, Forschung und Technologie, dessen Förderzeitraum sich von 1997 bis 2001

erstreckt (Projektverbund *Multimediabuch*). In diesem Projektverbund werden in Zusammenarbeit von Universitäten, Verlagen und weiteren wissenschaftlichen Infrastruktureinrichtungen Referenzprojekte aus den Bereichen Technik, Naturwissenschaften und Medizin entwickelt. Das Förderprogramm hat das Ziel, anhand ausgewählter Vorhaben die Möglichkeiten der Multimediatechnologie und internetbasierter Informationssysteme für das elektronische Publizieren, insbesondere den Bereich wissenschaftlicher Lehrwerke, zu untersuchen, wobei jeweils produktionsnahe Prototypen Ziel der Forschungsprojekte sind, vgl. dazu *http://www.darmstadt.gmd.de/PTF/Ausschreibungen/Multimed.htm*. Dies wird unter anderem durch eine Einbeziehung von Verlagspartnern in fast alle Projekte gewährleistet. Im Unterschied zu den im voranstehenden Kapitel diskutierten elektronischen Büchern ist nicht die *Konversion von Printprodukten*, sondern die Entwicklung neuer Konzepte für interaktive multimediale Lehrmaterialien das primäre Ziel. Das Referenzprojekt nimmt insofern eine Zwischenposition ein, als es beide Zielstellungen in Einklang zu bringen versucht: Konversion eines Buchs und Entwicklung interaktiver Elemente. Im Rahmen dieser Arbeit wird diese Konzeption durch die Integration von Diensten weitergeführt.

Die Eingrenzung des weit definierten Gegenstandsbereichs *elektronisches Publizieren* auf den Bereich *scholarly publishing* bzw. wissenschaftliche Lehrwerke einerseits, auf in sich abgeschlossene Einzelprojekte andererseits steht im Einklang mit der in dieser Arbeit verfolgten Sichtweise, die ebenfalls vom Einzelfall ausgeht und die so gewonnenen Erkenntnisse zu generalisieren versucht.[24] Tabelle 3 gibt einen Überblick der vierzehn Projekte, ihrer Beteiligten, zeigt auf, ob ein gedrucktes Werk als Ausgangspunkt den Projekten zugrunde liegt und nenn die zugehörigen Informationen im World Wide Web.

Titel	*Projektpartner*	*Referenzwerk*	*Informationen im WWW*
Training *Entwurf anwendungsspezifischer Mikrochips*	Informedia GmbH, Stuttgart	—	*http://www.ims-chips.de/education/Team2000_Internet/index.htm*
Weiterentwicklung des wissenschaftlichen und technischen Buches "Instandhaltung von Kanalisationen" zur multimedialen Wissensrepräsentation	Prof. Dr.-Ing. Stein & Partner GmbH, Bochum	STEIN 1999	*http://www.leitungsbau.de*
Das interaktive multimediale Multimediabuch	Springer-Verlag, Heidelberg, TH Darmstadt GmbH, FernUni Hagen	STEINMETZ 1999	*http://www.multibook.de*
Entwicklung eines interaktiven multimedialen Lernsystems Physikalisches Praktikum (*Referenzprojekt*)	B.G. Teubner Verlag, Stuttgart & Leipzig, Universität Leipzig, Kuratorium OFFIS e. V., Oldenburg	GESCHKE 1998	*http://aspra9.informatik.uni-leipzig.de/Teubner*
Bildbibliothek biologischer Makromoleküle	Institut für Molekulare Biotechnologie e.V., Jena	—	*http://www.imb-jena.de/IMAGE_PROJECT.html*

[24] Stärker an Infrastruktur für elektronische Publikationen orientierte Fördervorhaben sind *GlobalInfo* oder *Nutzung des weltweit verfügbaren Wissens für Aus- und Weiterbildung. GlobalInfo* ist eine Fördermaßnahme des BMB+F, deren erste Teilprojekte erst Ende 1999 begonnen haben, so dass aus diesem Kontext noch keine Forschungsergebnisse vorliegen, vgl. *http://www.global-info.org*, insb. *http://www.global-info.org/doc/was.html*. Unter dem Titel *Nutzung des weltweit verfügbaren Wissens für Aus- und Weiterbildung* sind fünf Leitprojekte zusammengefasst, die sich mit der Entwicklung elektronischer Lehr- und Lernsysteme sowie der Weiterentwicklung der elektronischen Informationsinfrastruktur für ausgewählte Wissenschaftsgebiete befassen, vgl. *http://www.bmbf.de/deutsch/arbeit/aufgaben/leitproj/wissen.htm*.

GenLAB A: Entwicklung eines gentechnischen Praktikums als Basis für eine multimediale Version B: Entwicklung einer elektronischen multimedialen Arbeits- und Experimentierumgebung für die Bio- und Gentechnologie	Kuratorium OFFIS e.V., Oldenburg, Heinrich-Heine-Universität Oldenburg	—	*http://offis.uni-oldenburg.de/projekte/dibo/projekt_mgp.htm*
Multimediale mathematische Arbeitsumgebung für die Sekundarstufe II und im universitären Bereich	SciFace Software GmbH & Co. KG, Paderborn	—	*http://www.mupad.de/PROJECTS/GERMAN/bmbf.shtml*
Multimediale Weiterentwicklung der Biochemical Pathways A: Verarbeitung chemischer Strukturen und Reaktionen B: Basisversion als CD-ROM mit Internet-Kopplung C: Visualisierung D: Datenverwaltung	Spektrum Akademischer Verlag, Universität Erlangen-Nürnberg, Universität Passau, Universität Mannheim	Plakat Biochemical Pathways (MICHAL 1999)	*http://www.biochemical-pathways.de/*
Die multimediale Wissensrepräsentation "Zellbiologie" auf CD-ROM und Online	Institut für den Wissenschaftlichen Film g.GmbH, Göttingen	—	*http://www.iwf.de/iwfger/1wir/11/111/online_in.html, http://www.cells.de*
INTERBRAIN – multimediale Visualisierung des menschlichen Gehirns	iAS interActive Systems Gesellschaft für interaktive Medien, Marburg, Springer Verlag, Heidelberg	—	*http://www.springer.de/newmedia/medicine/interbr/interbr.htm, http://www.brainmedia.de*
Sobotta interaktiv – ein multimediales Anatomieprogramm	Urban & Fischer Verlag für Medizin GmbH, München	SOBOTTA 1999	*http://www.urbanfischer.de/Medizinstudium/42880.htm*
MOVE – Multilinguale Wissensrepräsentation der Topographie, Statik und Dynamik des menschlichen Bewegungsapparates	Georg Thieme Verlag, Stuttgart	—	*http://www.thieme.de*
Computerbasiertes integriertes Lern-, Lehr- und Präsentationssystem für die Mammadiagnostik	MeVis g.GmbH an der Universität Bremen	—	*http://www.mevis.de/~guido/MammaVision.html*
Multimediales Informationssystem zur Herzchirurgie („Galerie der Herzchirurgie“, CARDIO-OP)	Hüthig Verlag, Heidelberg, Universität Ulm/Klinikum, Universität Heidelberg/Klinikum	—	*http://www.informatik.uni-ulm.de/dbis/f&l/forschung/mminfosystems/ftext-HerzGalerie_e.html*

Tabelle 3: Übersicht zum Projektverbund Multimediabuch

Schon die von den Teilnehmern des Verbundes gewählten Bezeichnungen für die Einzelprojekte sind hinsichtlich der Vielfalt elektronischer Bücher bzw. Informationssysteme aufschlussreich und spiegeln die Wortwahl der Ausschreibung wieder („multimediale Wissensrepräsentation“):

- computerbasiertes integriertes Lehr-, Lern- und Präsentationssystem
- elektronische multimediale Arbeits- und Experimentierumgebung
- interaktives multimediales Buch
- interaktives multimediales Lernsystem
- Multilinguale Wissensrepräsentation
- Multimediale Weiterentwicklung
- Multimediales Informationssystem
- Programm
- Training
- Visualisierung

Eine Kurzcharakteristik der Vorhaben soll Forschungsschwerpunkte verdeutlichen und Besonderheiten hervorheben.[25] Die Systematisierung der wesentlichen Eigenschaften erfolgt anschließend: Es wird ein Analyse- und Evaluierungsschema für elektronische Bücher und Multimediapublikationen erstellt.

3.1.2.1 Bereich Technik

Das Vorhaben *Training Entwurf anwendungsspezifischer Mikrochips* zielt unter Entwicklung eines Autorensystems bzw. Redaktionssystems auf den Aufbau einer umfassenden Informationsbasis zur Mikrochip-Entwicklung, wobei diese nicht unmittelbar als Lehranwendung, sondern als Wissensspeicher angelegt ist und neben der Integration von Multimediakomponenten Softwarewerkzeuge für den Schaltungsentwurf (Entwurf, Programmierung) bereitstellen soll. Die Realisierung als online-Server mit regelmäßigen Updates soll den schnellen technischen Fortschritt auf diesem Gebiet adäquat reflektieren.

Das Projekt Weiterentwicklung des wissenschaftlichen und technischen Buches "Instandhaltung von Kanalisationen" zur multimedialen Wissensrepräsentation setzt auf dem gleichnamigen umfangreichen Referenzwerk auf (vgl. STEIN 1999) und erstellt eine systematisch strukturierte Multimediaversion, die sich vor allem durch folgende Merkmale von der Druckfassung unterscheidet:

- Zahlreiche Vorgänge sind durch Animationen interaktiv visualisiert. Dabei kommt – dem Gegenstand angemessen – vor allem der 3D-Visualisierung von Maschinen bzw. Arbeitsvorgängen der Kanalinstandhaltung besondere Bedeutung zu.
- Die im Buch bereits vorhandene strukturierte Information (z. B. Klassifikation von Schadenstypen anhand eines normierten Schemas) wird systematisch für die datenbankbasierte Erschließung genutzt (Analysekomponente); die Funktionen eines Nachschlagewerks werden so im elektronischen Medium durch zusätzliche Nutzungsformen (z. B. Auswahl von Lösungsverfahren für ausgewählte Schadensklassen) unterstützt.
- In technologischer Hinsicht setzt das Projekt auf Internetstandards (HTML, JavaScript, WWW-Browser als Viewing-System) auf und präsentiert diese in einer einheitlich gestalteten Benutzerschnittstelle, die im Ausgangsmaterial enthaltenen Strukturen und semantischen Relationen (Thesaurus) werden konsequent für die Präsentation genutzt.

Das Projekt *MultiBook* hat zum Ziel, ein interaktives, adaptives und multimediales Lernprogramm für das umfangreiche Standardwerk Multimedia-Technologie (STEINMETZ 1999, vgl. auch REICHENBERGER & STEINMETZ 1999) zu erstellen. Folgende Merkmale sind hier zu nennen:

- Der Aufbereitung des Ausgangsmaterials liegt ein einheitliches Modell der Aufteilung des Wissens auf sog. *media bricks* (Medienbausteine) zugrunde, die nach Bedarf dynamisch zusammengestellt werden können.
- Das Projekt arbeitet wissensbasiert, d. h. die einzelnen Lehr- und Lerneinheiten stehen durch ein explizit repräsentiertes Wissensnetz bzw. einen Thesaurus untereinander in Beziehung.

[25] Dies geschieht unter Auswertung von elektronisch verfügbarem Material (Arbeitsberichte, WebSites), der Ergebnisse mehrerer Projektworkshops des Projektträgers in Darmstadt bzw. Ulm sowie von Kommunikation mit den Projektpartnern.

- Bei der Nutzung soll ein Benutzermodell Verwendung finden, das Lerneinheiten in Abhängigkeit vom Wissensstand des Benutzers dynamische zusammenstellt und konfiguriert (Umfang und Struktur des präsentierten Wissens).
- Multimediale Anwendungen (z. B. Visualisierungen unterschiedlicher Verfahren der Datenkompression) stellen Phänomene und Technologien im Multimediabereich interaktiv dar.
- In technologischer Hinsicht arbeitet das Projekt datenbankbasiert und verwendet Internet-Standards für die Repräsentation von Information (HTML, JavaScript) bzw. die Entwicklung von Multimediakomponenten (Macromedia Director, Java).

3.1.2.2 Bereich Naturwissenschaft

Die *Bildbibliothek biologischer Makromoleküle* präsentiert eine umfangreiche Datenbank mit Biopolymerstrukturen als interaktives multimediales Informationssystem. Dabei stehen die Integration unterschiedlicher Datenquellen (z. B. externe Datenbanken mit verwandtem Inhalt), die Visualisierung und Analyse von Informationen über Moleküle und die Erschließung über Suchverfahren im Mittelpunkt, wobei verschiedene Softwarewerkzeuge und Standards insbesondere aus dem Bereich der 3D-Viualisierung (z. B. die *Virtual Reality Modeling Language*, VRML) zum Einsatz kommen. Die Ergebnisse sind als Informationssystem im World Wide Web verfügbar und haben sich zu einem wichtigen Arbeitswerkzeug für dieses Fachgebiet entwickelt.

Der Aufbau eines 3D-Modells für Laborexperimente steht im Mittelpunkt des Projektes *GenLAB*. Ausgehend von einem objektorientierten Modell für die Strukturierung genetischer Versuche enthält GenLAB verschiedene Komponenten, u. a. eine lexikonorientierte Nachschlagekomponente mit multimedialer Information zu allen Aspekten im Umfeld der Versuchsausführung sowie eine interaktive 3D-Experimentierumgebung, in der in verschiedenen Experimentiermodi virtuelle genetische Versuche ausgeführt werden können. Eine datenbankgestützte Wissensbasis unterstützt den Lerner bei der Durchführung der Experimente. Besonderes Augenmerk liegt bei diesem Projekt auf einer konsequenten Entwicklungsmethode unter Einsatz bewährter Verfahren aus der objektorientierten Modellierung. Die verschiedenen Module sind in einer einheitlich gestalteten Benutzerschnittstelle integriert und mit Hilfe eines Multimedia-Autorensystems realisiert.

Das Projekt *Multimediale Weiterentwicklung der Biochemical Pathways* stellt insofern einen Sonderfall dar, als das gedruckte Ausgangswerk ein Plakat (!) mit einer äußerst detaillierten Visualisierung biochemischer Stoffwechselprozesse ist. Bei dessen elektronischer Umsetzung, die sich in die Bereiche Datenverwaltung, Visualisierung und Recherche bzw. Erschließungsverfahren gliedert, sind sowohl die eingrenzenden (Darstellungsraum am Bildschirm) als auch die erweiternden Möglichkeiten des elektronischen Mediums besonders deutlich ausgeprägt: Durch die gegenüber stärker textorientierten Projekten vorhandene Dominanz strukturierter Daten (Formeln, chem. Gleichungen, Prozessbeschreibungen) ist eine weitgehende Formalisierung der Ausgangsdaten (Formel- und Prozessrepräsentation) sowie die Übersetzung in eine visuelle Sprache möglich. Das Projekt hat eine hybride Lösung für das Trägermedium zum Ziel, d. h., dass eine CD-ROM die Basisversion enthält, die durch Internetzugang erweitert werden kann.

Das Projekt *Multimediale mathematische Arbeitsumgebung für die Sekundarstufe II und im universitären Bereich* setzt auf dem an der Universität Paderborn entwickelten und im

akademischen Bereich weit verbreiteten Algebra-System MuPad auf (vgl. *http://www.mupad.de* und FUCHSSTEINER, DRESCHER & KEMPER 1996) und definiert einen mathematikorientierten Explorationsraum für den Benutzer, der das elektronische Buch konzeptuell weiterentwickeln soll, ein Aspekt, der auch für diese Arbeit eine Rolle spielt: Die Annahme, dass generische Dienste wie ein Formelmanipulations- und Auswertungssystem in dynamischen elektronischen Büchern zur Verfügung stehen sollten, wird in Kap. 4.4 näher untersucht; Kap. 14.2.3 zeigt eine exemplarische Realisierung eines solchen Dienstes.

3.1.2.3 Bereich Medizin

Im Vorhaben *MOVE – Multilinguale Wissensrepräsentation der Topographie, Statik und Dynamik des menschlichen Bewegungsapparates* wird eine interaktive Multimedia-CD-ROM entwickelt, die unterschiedliche Medien und Informationstypen (klinische Aufnahmen, Schemata, Animationen, Videoaufnahmen, Texte, eine Terminologie-Datenbank) integriert und in einer einheitlich gestalteten Benutzerschnittstelle darstellt.

Das Projekt *MammaVision* entwickelt ein interaktives Lehr-, Lern- und Präsentationssystem für den Bereich der Mammographie, das aus den Komponenten Falldatenbank, unterschiedlich differenzierten Trainingsprogrammen (*MammaTrainer Pro*; *MammaTrainer Home*, vgl. *http://www.mevis.de/~guido/MammaVision.html*), einer elektronischen Zeitschrift und einem elektronischen Informationsnetzwerk besteht. Im Mittelpunkt steht die Möglichkeit, Informationen multimedial zu präsentieren, dem Benutzer unterschiedliche Interaktions- und Darstellungsmodi anzubieten (Multimodalität) und die Interaktivität aller Komponenten.

Das Projekt *Multimediales Informationssystem zur Herzchirurgie* (*Galerie der Herzchirurgie/CARDIO-OP*) hat die „Entwicklung eines netzwerkfähigen, datenbankbasierten und multimedialen Informationssystems mit Ausbildungs- und Beratungsinhalten sowie zugehörigen Nutzungsszenarien für Ärzte, medizinische Lehrer, Studenten und Patienten in der Herzchirurgie“ zum Ziel. Der Kern des Systems ist eine Datenbank mit heterogenen Multimediainhalten, die um verschiedene Wissenskomponenten mit Lehrbuchcharakter (z. B. Operationslehre, Fallsimulationen) ergänzt wird. Dabei kommt ein im Umfeld des Projekts entwickeltes Multimedia-Datenmodell zum Einsatz (ZYX-Model, vgl. BOLL, KLAS & WESTERMANN 1999A, 1999B, 1999C und unten Kap. 8). Für den Benutzer erfolgt der Zugang über eine Benutzerschnittstelle, die nach einer Architekturmetapher aufgebaut ist.

3.1.3 Referenzprojekt Multimediales Physikalisches Praktikum

Zuletzt soll das dieser Arbeit zugrunde liegende Referenzprojekt *Multimediales Physikalisches Praktikum* vorgestellt werden, da es im Folgenden zur Herausarbeitung von Konzepten für dynamische elektronische Bücher als Bezugspunkt dient; eine ausführliche Darstellung seiner wesentlichen Merkmale findet sich in Teil III dieser Arbeit. Bei dem Projekt handelt es sich um die multimediale Umsetzung eines bewährten Lehrbuchs für das Anfängerpraktikum der Experimentalphysik (GESCHKE 1998).

Das Lehrwerk hat einen homogenen Aufbau, da es für Versuche aus Mechanik, Wärmelehre, Elektrik, Optik und Atomphysik die theoretische Herleitung des untersuchten Phänomens, die Beschreibung des Versuchsaufbaus sowie Handreichungen zu Versuchs-

durchführung und –auswertung enthält. Insofern bietet es eine gute Voraussetzung für eine inhaltsorientierte Informationsaufbereitung mit Hilfe deklarativer Markupsprachen; zur Frage der systematischen Modellierung des Ausgangsdatenbestands s. u. Kap. 4.2. Schon im gedruckten Lehrwerk sind unterschiedliche Medienelemente enthalten, an denen eine multimediale Umsetzung anknüpfen kann, u. a.

- schematische Darstellungen von Versuchsaufbauten,
- tabellarisches Datenmaterial,
- Funktionsgraphen oder
- Schaltpläne.

Hinzu kommt der in naturwissenschaftlichen Werken hohe Anteil mathematischer Symboldarstellungen, sowohl im laufenden Text als auch abgesetzt.

Die elektronische Fassung als multimediales Lehrwerk ist im Referenzprojekt als „*off-line add-on*“ konzipiert, d. h. das elektronische Buch dient als CD-ROM-*Ergänzung* des gedruckten Werks und soll dem Studenten der Physik Hilfestellung bei der Vor- und Nachbereitung von Versuchen leisten. Dieser Charakter eines *Begleitmediums* für den eigentlichen Studienablauf hat die Auswahl und Typisierung der multimedialen Elemente des Buches bestimmt. Das elektronische Buch umfasst aber den vollständigen Text des Ausgangswerks und ist um Navigations-, Browsing und Recherchefunktionalität angereichert (vgl. unten Kap. 12.5.1).

Der Entwicklungsablauf für das Projekt lässt sich grob in die Phasen Konzeption, Entwicklung und Optimierung gliedern und umfasst die in der folgenden Tabelle gezeigten Einzelschritte:

Konzeptionsphase	• Materialerschließung und exemplarische Aufbereitung anhand eines Subsets des Ausgangsmaterials (im Sinne eines *rapid prototyping* mit quantitativer Einschränkung) • Konzeption des Betrachtungssystems, darauf aufbauend Ausarbeitung eines gestalterischen und technischen Anforderungsprofils sowie eines Storyboards als makroskopisches Modellierungskonzept • Definition von Materialergänzungstypen und deren Auswahl
Entwicklungsphase	• Materialkonvertierung (Text, Bild, Formeln) • Integration des konvertierten Materials auf der Basis eines Gestaltungsleitfadens (*screen design*) • Sukzessive Entwicklung multimedialer Ergänzungen (sowie Medienerstellung bzw. Medienakquise)
Optimierungsphase	• Erstellung und Test eines Prototyps • Quantitative und qualitative Erweiterung des Prototyps

Tabelle 4: Übersicht zum Entwicklungsablauf des Referenzprojekts

Um multimediale Elemente und Buchinhalte optimal koppeln zu können, wurde der vollständige Text des Buches elektronisch umgesetzt und dafür eine geeignete Benutzerumgebung entworfen. In Analogie zum gedruckten Buch liegt der Gestaltung des Betrachtungsmodells eine modifizierte Buchmetapher zugrunde, d. h. eine vertikale Zweiteilung des Betrachtungssystems, die ein Weiterblättern zulässt (vgl. SCHNEIDER & WOLFF 1998A, 1998B; vgl. zur Motivation dieser Metapher LANDONI & GIBB 1999). Die jeweils rechte Seite trägt den aktuellen Lesefokus („Fokusseite“, vgl. unten Kap. 13.3.1), während die jeweils linke Seite als *Kontextbereich* entweder (*default*) die vorangegangene Seite oder multimediale Ergänzungen oder entfernte Buchseiten darstellt. Abbildung 12 zeigt schematisch den Aufbaus des Viewers bzw. der Benutzerschnittstelle, Abbildung 13 gibt ein Beispiel aus der elektronischen Fassung.

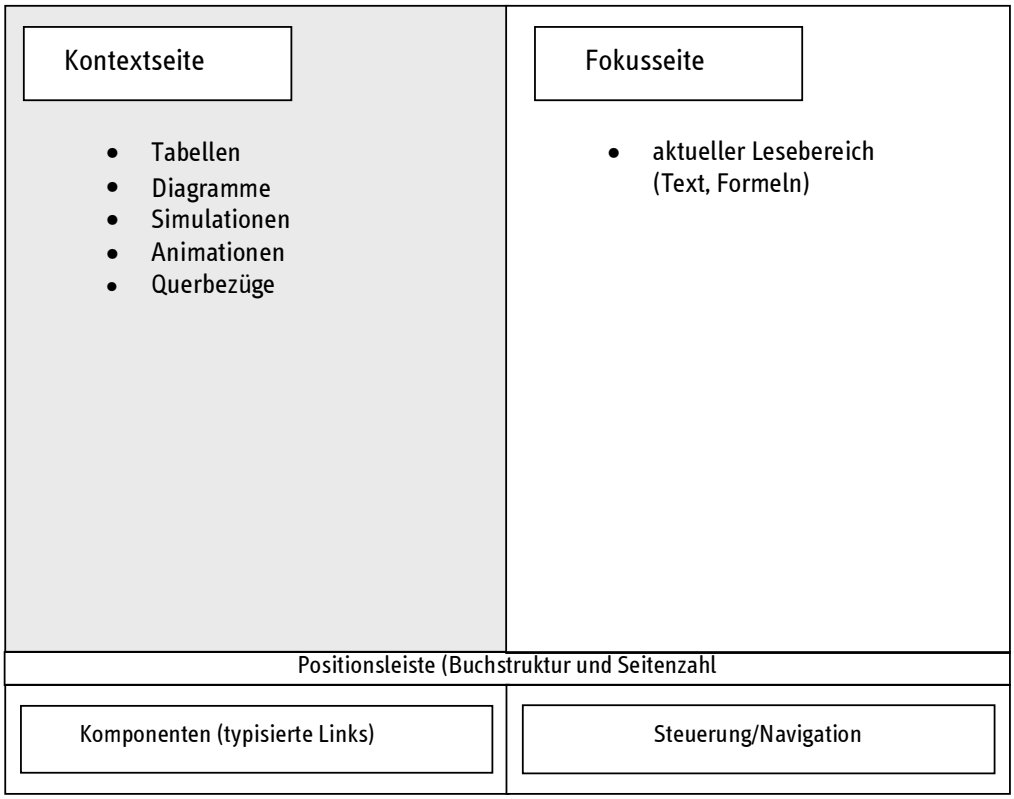

Abbildung 12: Schematischer Aufbau des Buchviewers

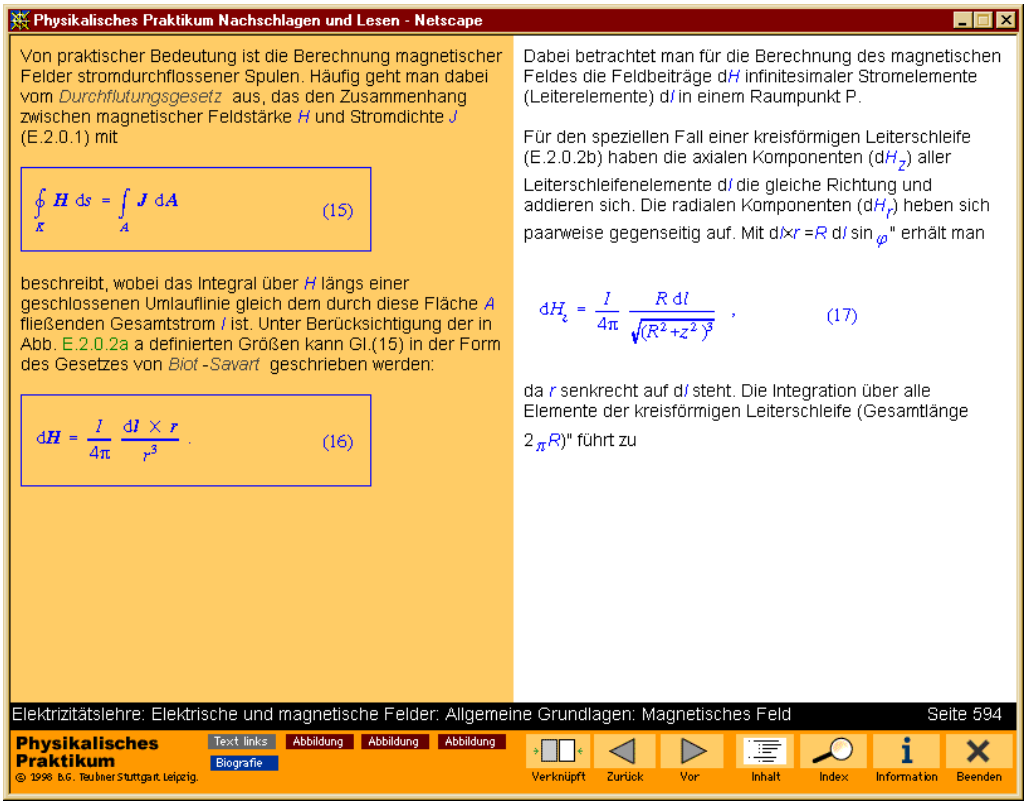

Abbildung 13: Beispielseite der elektronischen Fassung des Multimedialen physikalischen Praktikums

Die Umsetzung des Buchtextes ist nur eine einfache Nutzungsvoraussetzung für die sinnvolle Einbettung von Multimediaelementen oder allgemeiner Ergänzungen des Basisdatenbestands. Dabei handelt es sich sowohl um quantitative Ergänzungen (zusätzliche Texte, Exkurse), die kein neues Medium einführen, sondern nur den Ausgangsbestand ergänzen, als auch um qualitative Ergänzungen, die den Status als Multimediaanwendung konstituieren (Simulationen und Animationen). Die Auswahl erfolgte als Ergebnis einer Detailanalyse durch eine Arbeitsgruppe von Physikern unter Berücksichtigung unterschiedlicher didaktisch motivierter Zielstellungen sowie eines eingangs definierten Katalogs von Ergänzungstypen. Eine ausführliche Diskussion der Motivation für ihre Einführung sowie zahlreiche Beispiele finden sich unten in Kap. 14.

Die Integration aller zusätzlichen Elemente erfolgt immer durch Bezugnahme auf den jeweils relevanten Bereich des Buches, d. h. in Korrelation zu einzelnen *Inhalt*sbereichen, nicht zu *Darstellungseinheiten* (Bildschirmseiten). Sie wird dem Benutzer durch Hypertextverknüpfungen im Text sowie durch die Multimediaelement-Leiste im Steuerungsbereich des Buchs angezeigt. Zur Integration zusätzlicher generischer Funktionalität durch die Integration von Diensten, die im Referenzprojekt nicht berücksichtigt werden konnte vgl. unten Kap. 4.4 bzw. 14.2.

Wegen der kurzen Projektlaufzeit von 18 Monaten war keine flächendeckende Entwicklung multimedialer Ergänzungen möglich, d. h. nicht alle wünschenswerten Animationen und Simulationen konnten entwickelt werden. Es existiert aber ein gleichmäßig über die inhaltlichen Schwerpunkte des Buches verteilter Grundbestand an multimedialen Ergänzungen, der im Sinne einer längerfristigen Entwicklung dieses elektronischen Buchs ergänzt werden kann. Dies ist in der Praxis bereits durch Softwarepraktika geschehen; eine vollständige Abdeckung ist nur mittelfristig zu erreichen.

Die Umsetzung des dargestellten Konzeptes ist nicht mit *einer* bestimmten Technologie oder einem einzelnen Entwicklungswerkzeug zu bewältigen, da die Heterogenität der geforderten technischen Merkmale (typographischer Satz, Formeln, Bildintegration, Animationen, Simulationen, Hypertextverknüpfungen) bei einer ersten Evaluierungsstudie für jedes der drei untersuchten Szenarien als Ausschlusskriterium wirkte (vgl. BRÖRKENS EL AL. 1998):

- Realisierung auf HTML-Basis mit Integration von Java-Applets,
- Realisierung als Kombination von *Portable Document Format* und Autorensystem (plug-in) und
- Realisierung ausschließlich mit Autorensystem (Macromedia Director).

Die technischen Mindestanforderungen sind mit Rücksicht auf die technische Infrastruktur der zukünftigen Benutzer niedrig definiert, d. h. max. 256 Farben und 800 × 600 Pixel Bildschirmauflösung, 16 MB RAM, vgl. dazu unten Kap. 13.3.2, Abbildung 50. Nach Abschluss der technischen Evaluierung entstand folgendes Umsetzungsszenario:

Realisierungsaspekt	*Technologie*
Aufbau des Viewers, Realisierung der Navigation	HTML-Browser (Netscape) und Javascript/*dynamic HTML*
Satz und Hypertextverknüpfungen	*dynamic HTML* und *Cascading Style Sheets* (Layoutspezifikation)
Bilder, Graphiken	Bitmapformate (GIF, JPEG)
Formelsatz	Konvertierung in Bitmapformate, Hypertextualisierung
Animationen und illustrierte Versuchsaufbauten	Multimedia-Autorensystem (Macromedia Director), integriert durch plug-in-Software (Macromedia Shockwave)
Simulationen	Java

Tabelle 5: Entwicklungstechnologien für das Physikalische Praktikum

Die Entscheidung für HTML und somit für einen SGML- bzw. XML-basierten Standard beruht auf der Überlegung, dass trotz des offline-Formats CD-ROM eine Integration ins World Wide Web sinnvoll ist und für zukünftige Versionen mächtigere Standards auf XML-Basis leicht zu integrieren sind. Gleichzeitig stehen die Modellierungsmöglichkeiten einer Markupsprache zur Verfügung – ein zentraler Aspekt für die systematische Konzeption dynamischer elektronischer Bücher (vgl. unten Kap. 4.2). Gegenüber der Alternative des *Portable Document Format* bedeutet dies zwar einen erhöhten Entwicklungs- und Produktionsaufwand und eine schlechtere Darstellungsqualität (z. B. nicht skalierbare Formeln), aber einen Gewinn an Flexibilität bei der Spezifikation von Links, bei der Textpartitionierung und der Anpassung an die vorgegebene Viewerumgebung (vgl. MERZ 1997B: 7 f. mit ähnlichen Ergebnissen).

Die zunächst nach den Standards HTML V. 3.2 bzw. HTML V. 4 aufbereiteten Ausgangsmaterialien konnten unter Beachtung der Einschränkungen an eine XML-basierte HTML-Dokumententypbeschreibung angepasst werden (XHTML, vgl. unten Kap. 6.2), die von WWW-Browsern unterstützt wird (vgl. KORPELA 1998, LIE & SAARELA 1999 und unten Kap. 5.2.7). Der Prototyp ist für die Distribution als CD-ROM-Anwendung konzipiert. Dabei wird auf der CD-ROM die benötigte Software bereitgestellt (WWW-Browser; Macromedia-Shockwave-plug-in). Die Installation erfolgt mit Hilfe eines Installationsprogramms (*InstallShield*). Parallel dazu existiert eine Online-Version, die laufend aktualisiert wird und bei vorhandener Zugangsberechtigung genutzt werden kann.

Die im Rahmen des Projektes erstellte produktnahe Fassung des Multimedialen Physikalischen Praktikums kann als typisch für den state-of-the-art elektronischer Bücher angesehen werden; es zeigt sich eine Übereinstimmung mit zahlreichen anderen Projekten des Projektverbunds, was grundlegende Designentscheidungen angeht:

- Die Realisierung informationellen Mehrwerts durch quantitative und qualitative Ergänzung eines Ausgangsdatenbestands, insbesondere durch Integration (interaktiver) Multimediakomponenten als primärer Legitimation für die Realisierung elektronischer Bücher bzw. Multimediaanwendungen,
- die Verwendung kommerzieller Multimedia-Autorensysteme, insbesondere Macromedia Director, für die Erstellung interaktiver multimedialer Komponenten,[26]
- Nutzung SGML-/XML-basierter Standards für die Dokumentenstrukturierung und
- die Verwendung eines WWW-Browsers als Präsentationssystem bzw. allgemein das Aufgreifen von Standards und Technologien des World Wide Web (HTTP, HTML).

Es treten dabei z. B. folgende Probleme auf:

- Es fehlt ein integrierendes Autorensystem, das alle Aspekte des *multimedia authoring* abdeckt. Dies führt zu einer heterogenen Menge erforderlicher Softwarewerkzeuge.
- Methodologien für die Erstellung von Multimediasystemen oder multimedialen elektronischen Büchern sind bisher noch kaum entwickelt.

Insgesamt ist das Referenzprojekt durch die Vielzahl der Darstellungs- und Präsentationsformen (Text, Formel, Tabelle, Diagramm, schematische Zeichnung, Animation, Simulation, Photographie/Film) geeignet, Gegenstand weiterführender Überlegungen zu sein. Die zunächst nur kursorische Präsentation wesentlicher Merkmale eines solchen elektronischen Buchs wird in den folgenden Kapiteln vertieft. Auf der Basis der in der Literatur

[26] Gleichzeitig werden aber die Grenzen dieser Systeme deutlich, da der Mangel standardisierter Austauschformate für Multimediadaten (vgl. unten Kap. 8) die Wiederverwendbarkeit der mit hohem Aufwand realisierten Multimediakomponenten stark einschränkt.

diskutierten Beispiele elektronischer Lehrbücher sowie aufbauend auf den im Projektverbund Multimediabuch gemachten Erfahrungen entstand ein Bewertungs- oder Evaluierungsleitfaden, der den gegenwärtigen state-of-the-art der Erstellung multimedialer Publikationen reflektiert und als Orientierungsrahmen für das Modell dynamischer elektronischer Bücher dient (Kap. 3.2).

3.2 Merkmale elektronischer Bücher

Die Darstellung des wissenschaftlichen Kontexts sollte die konzeptuelle und technologische Bandbreite des state-of-the-art elektronischer Bücher verdeutlichen. Um die Einordnung einzelner Projekte und der Konzepte dynamischer elektronischer Bücher zu vereinfachen, wird nachfolgend ein Kriterien- und Merkmalskatalog für elektronische Bücher vorgestellt. Er entstand im Umfeld des oben vorgestellten Projektverbunds als Orientierungshilfe für die Beurteilung des state-of-the-art multimedialer elektronischer Bücher und versucht, relevante Beschreibungsmerkmale strukturiert darzustellen; vergleichbare Anforderungskataloge für elektronische Bücher oder Multimediaanwendungen finden sich bei STEINMETZ 1999: 837 ff., LYNCH & HORTON 1999: 4-10 oder in den *Leipziger Empfehlungen für das Elektronische Publizieren* (BÖRSENVEREIN 1994). Die relevanten Kriterien werden kurz eingeführt und typische Ausprägungen genannt sowie zum Referenzprojekt in Bezug gesetzt. Die Kriterien sind auf folgende Kategorien aufgeteilt:

- Voraussetzungen und Randbedingungen,
- Grundlegende technische Aspekte,
- Modellierungs- und Entwicklungsmethoden,
- Gestaltung und Präsentation,
- Buchfunktionalität,
- Hypertext-Eigenschaften,
- Gestaltung und Präsentation,
- Multimediale Inhalte,
- Didaktische Aspekte,
- Informationserschließung und Einbindung externer Ressourcen,
- Abrechnung und Sicherheit.

3.2.1 Voraussetzungen und Randbedingungen

Zu den Voraussetzungen und Randbedingungen bei der Erstellung elektronischer Bücher zählen organisatorische Aspekte wie

- die Aufgabenverteilung bei den Projektbeteiligten,
- die Einordnung des elektronischen Publikationsprodukts in den Publikations- und Distributionsprozess und
- die Einordnung des Projekts bezüglich des Ausgangsdatenbestands (Bezug zu einem Printprodukt).

3.2.1.1 Projektstruktur

Der Entstehungsprozess elektronischer Bücher, insbesondere wenn sie interaktive Multimediakomponenten enthalten, ist im Vergleich mit Printprodukten stärker ausdifferenziert; technische Entwicklungsaufgaben spielen eine zentrale Rolle und neben die traditionelle Koordinationsfunktion von Verlagen treten Aufgaben wie das Projektmanage-

ment für die Entwicklungsprozesse. Der wichtigste Aspekt dabei ist die veränderte Rolle der Autoren, die in der Regel anders als oben in Kap. 3.2.1.1, Abbildung 1 gezeigt, den Entwicklungsprozess bis zur Fertigstellung des (elektronischen) Publikationsprodukts nicht vollständig kontrollieren (können).

Typisch sind räumlich verteilte interdisziplinäre Kooperationen, wie dies Tabelle 3 für den Projektverbund des Referenzprojekts deutlich macht. Beispielhaft zeigt dies auch die Verteilung von Aufgaben und Rollen im Referenzprojekt:

Rolle	*Funktion*
Autoren	Bereitstellung des Ausgangsmaterials, inhaltliche und didaktische Spezifikation für multimediale Inhalte
Produzent	Grundkonzeption, Projektkoordination, Redaktionsaufgaben, Produktion
Entwickler	Materialkonvertierung Aufbau des Buchnutzungssystems Realisierung von Multimediakomponenten
Dienstleister	Gestaltungsvorgaben
Verlag	Koordination, Distribution, Bereithaltung von Inhalten

Tabelle 6: Rollen und Aufgaben im Referenzprojekt

Die verschiedenen Rollen sind dabei unabhängig von der konkreten Ausprägung des Projekts; während im *Projektverbund Multimediabuch* universitäre Einrichtungen dominieren, sind die Rollen (Entwicklung, Produzenten) bei kommerziellen Entwicklungen analog Funktionsträgern bei Verlagen und Multimediaproduktionsfirmen angesiedelt. Eine scharfe Trennung zwischen Forschungsprojekten und kommerziellen Entwicklungen ist nicht möglich, da einerseits fast alle Projekte dieses Verbunds die erstellten elektronischen Bücher durch Verlage („kommerziell“) publizieren und andererseits für den wissenschaftlichen Fach- und Lehrbuchbereich Hochschulen ohnehin eine Schlüsselrolle spielen. Der wesentliche Unterschied zu Printprodukten liegt in den Bereichen Entwicklung und Produktion, die dort keine oder eine sehr untergeordnete Funktion haben.

3.2.1.2 Distribution

Elektronische Bücher haben im Vergleich zu gedruckten Büchern erweiterte Möglichkeiten der Distribution. Dies betrifft sowohl das Distributions*medium* (Diskette, CD-ROM, DVD, Internet/WWW) als auch den Distributions*kanal* (Buchhandel, Softwarehandel, E-Commerce, Verlag, digitale Bibliothek, Autoren (z. B. WebSite des Autors)), wobei sich die Merkmale überschneiden können, wenn z. B. ein Verlag oder eine Organisation selbst auch Betreiber einer digitalen Bibliothek ist oder über E-Commerce den Direktvertrieb seiner elektronischen Bücher übernimmt.

War bis vor kurzem noch das Medium CD-ROM vorherrschend für multimediale elektronische Bücher, so wird mit besserer Verfügbarkeit großer Netzbandbreiten der Anteil direkt oder vermittelt über digitale Bibliotheken im World Wide Web angebotener elektronischer Bücher deutlich zunehmen. Diese Entwicklung kennzeichnet auch das Referenzprojekt, das zwar als Begleit-CD-ROM realisiert ist, aber gleichzeitig als WWW-basiertes Informationssystem genutzt werden kann. Das Konzept dynamischer elektronischer Bücher, wie es in dieser Arbeit entwickelt wird, ist von vornherein als WWW-basiertes Informationssystem angelegt. Ein weiteres Kriterium ist die *Granularität der Distribution*, d. h. die Frage ob die Distribution eines elektronischen Buchs

- geschlossen oder aufgeteilt in kleinere informationelle Einheiten (z. B. Kapitel, Aufsätze eines Sammelbands, einzelne Medienelemente) und
- ggf. im Kombination mit einem Printprodukt (Medienbündel).

3.2.1.3 Bezug zum Printmedium

Ein wesentliches Unterscheidungsmerkmal für elektronische Bücher ist die Frage, ob ein entsprechendes gedrucktes Buch als Ausgangspunkt der Erstellung einer elektronischen Fassung vorliegt. Ist dies der Fall, ergeben sich eine Reihe von Folgefragen für die Umwandlung und Anreicherung der vorliegenden Materialien (z. B. Strategien der Konvertierung und Strukturierung, vgl. STORRER 1988, 1999 und unten Kap. 3.2.2.3).

3.2.2 Grundlegende technische Aspekte

Zu den technischen Aspekten elektronischer Bücher zählen

- die für das Distributionsmedium geltenden Randbedingungen, die in der Regel auch den Entwicklungsprozess als Zielvorgaben beeinflussen,
- die Verwendung von Standards bei der Realisierung des elektronischen Buchs,
- die verwendeten Datenformate für die verschiedenen Medientypen,
- Strategien und Techniken der Materialkonvertierung und
- die Erstellungswerkzeuge für das elektronische Buch.

3.2.2.1 Randbedingungen

Technische Randbedingungen legen fest, welche Mindestanforderungen an die Nutzungsinfrastruktur für das elektronische Buch gestellt werden und welche Vorgaben sich daraus für die Entwicklung ergeben. Bei sehr umfangreichen Produktionen (z. B. im Projekt CARDIO-OP, vgl. oben Tabelle 3 und Kap. 3.1.2.3) kann es auch zu einer parallelen Vorhaltung von Inhalten für unterschiedliche Nutzungsinfrastrukturen kommen (z. B. Bilder und Filme mit verschiedenen *Quality of Service*-Chrakteristika (QoS)). Die folgende Tabelle zeigt relevante Kriterien bei der Festlegung technischer Randbedingungen:

Merkmal	*Beispiele typischer Ausprägungen*	*Referenzprojekt*
Plattform und Rechnerleistung	MS-Windows, UNIX, MAC, plattformneutral	MS-Windows (CD-ROM-Fassung)/Pentium, plattformneutral (WWW-Fassung)
Für die Nutzung erforderliche Software	WWW-Browser, plug-in-Software, Viewer eines Autorensystems, Java-Interpreter	WWW-Browser, Macromedia Shockwave
Speicher	Größe des erforderlichen Arbeitsspeichers, benötigter Festplattenplatz	16 MB RAM max. 40 MB (je nach Installationsversion)
Bildschirmauflösung und Farbtiefe	Auflösung 800 × 600, 1024 × 768 Pixel, 256 Farben, High Color, True Color	800 × 600, 256 Farben, High Color empfohlen
Weitere Hardwareanforderungen	CD-ROM- oder DVD-Laufwerk, Soundkarte	CD-ROM-Laufwerk, Soundkarte (empfohlen)
Online-Verbindung	dauerhaft, fallweise, für Updates	nur für WWW-Fassung, dauerhaft

Tabelle 7: Beispiele technischer Randbedingungen für die Nutzung elektronischer Bücher

3.2.2.2 Verwendung von Standards

Ein wesentliches Kriterium für die Realisierung elektronischer Bücher ist die Verfügbarkeit von Standards für alle Aspekte der Kodierung von Information. Bei der Festlegung

von Standards sind unterschiedliche Typen zu unterscheiden; das formal wichtigste Kriterium ist der Status eines Standards. Man unterschiedet zwischen

- „Industriestandards“ wie Microsoft *Object Linking and Embedding* (OLE), Adobe *Postscript* und Adobe *Portable Document Format*, die durch ihre weite Verbreitung am Softwaremarkt den Status eines de facto-Standards erreicht haben, ohne von einem offiziellen Standardisierungsgremium verabschiedet worden zu sein, und
- offiziellen de jure-Standards, die von einer Standardisierungsinstanz auf nationaler (z. B. ANSI-, DIN, EU-Normen) oder internationaler Ebene (*International Organization for Standardization*, ISO, *http://www.iso.ch*) kodifiziert sind wie SGML, DSSSL oder HyTime im Bereich der Dokumentenstrukturierung und Texttechnologie.

Die Unterscheidung zwischen Industriestandards und offiziellen Normen spiegelt in vielen Fällen eine unterschiedliche Entstehungsgeschichte wieder: Offizielle normierte Standards wie DIN-Normen oder ISO-Standards können an sich ohne starke Korrelation mit ihrer Auswirkung auf die am Markt verfügbare Softwareinfrastruktur zustande kommen, wofür gerade die von SGML abgeleiteten Standards wie HyTime oder DSSSL gute Beispiele sind (vgl. dazu unten Kap. 7.1.1 und Kap. 8.1), während sich Industriestandards in der Regel im Umfeld bestimmter Softwarewerkzeuge herausbilden und eine größere Praxisnähe aufweisen können (z. B. das von Microsoft entwickelte *Rich Text Format* oder das präsentationsnahe *Portable Document Format*). Zu den weiteren Beurteilungskriterien für EP-relevante Standards Status und Entstehung gehören:

- der Sachbezug des Standards („was wird standardisiert?“),
- seine Beziehung zu anderen Standards („lässt sich eine Gruppe von Standards für mehrere Aspekte der Informationskodierung nutzen?“),
- die verfügbaren Werkzeuge,
- die Verbreitung bzw. die Durchsetzungschancen des Standards („ist der Standard in absehbarer Zeit praxisrelevant?“),
- die verfügbaren Alternativen („welche anderen Standards stehen zur Verfügung ?“).

Hinsichtlich der im Projektverbund Multimediabuch realisierten Vorhaben lässt sich bezüglich der Verwendung von Standards folgendes festhalten:

1. Die Durchsetzung von Standards ist stark von den Entwicklungswerkzeugen abhängig: Wo für die Umsetzung Autorensysteme mit proprietären Datenformaten Verwendung finden, fehlt es am Einsatz entsprechender Standards; es finden sich aber auch Entwicklungen für neue Kodierungsstandards bzw. Datenformate (vgl. BOLL, KLAS & WESTERMANN 1999A, 1999B und 1999C). Dies gilt gerade für den Bereich interaktiver multimedialer Komponenten in elektronischen Büchern, da es zu Autorensystemen mit proprietären Formaten kaum Alternativen gibt und relevante Standards wie HyTime oder SMIL (vgl. unten Kap. 8) nicht unterstützt werden.
2. Ein zweiter Faktor sind proprietäre Ausgangsformate der Konversion aus dem Printprodukt ins elektronische Medium, die – auch bedingt durch beschränkte Ressourcen – verhindern, dass von vornherein auf Standards (z. B. SGML/XML und Ausarbeitung entsprechender Dokumentgrammatiken) zurückgegriffen werden kann.
3. Soweit eine Realisierung als webbasiertes Informationssystem (elektronisches Buch im World Wide Web) vorgesehen ist, werden nicht nur standardisierte Medienformate (HTML für Texte, JPEG für Bilder, AVI/Quicktime für Bewegtbilder etc.) verwendet, sondern auch das für den Datenzugriff erforderliche Protokoll (HTTP) und die Zugangssoftware (Webbrowser) sind (*de facto*) standardisiert.

4. Schließlich ist zu beachten, dass die Architektur elektronischer Bücher auf unterschiedlichen Ebenen unterschiedliche Standards für die Verwaltung und Präsentation der Medien erforderlich macht, wenn z. B. ein Datenbanksystem zur Datenverwaltung Verwendung findet und erst bei Anforderung einer informationellen Einheit durch den Benutzer die Daten mit Hilfe von Standards (z. B. HTML/XML für Hypertexte, Streamingformate für zeitabhängige Daten) aufbereitet und präsentiert werden.

Die für die Kodierung atomarer Medienelemente verfügbaren Standards (Bitmaps, Vektorgraphik, Film, Audio etc.) sollen hier nicht diskutiert werden, vgl. dazu STEINMETZ 1999. Die für die Kodierungsebenen deklarativer Informationsmodellierung verfügbaren Standards der SGML-/XML-Familie werden in Teil II dieser Arbeit vorgestellt.

3.2.2.3 Materialaufbereitung, Konvertierung und Informationsstrukturierung

Bei der Konvertierung elektronisch repräsentierter Ausgangsmaterialien, im Referenzprojekt z. B. die mit einem Textverarbeitungssystem erstellten Satzvorlagen der gedruckten Buchfassung, sind in technischer Hinsicht in *Vorgehensweise* und *Werkzeuge* zu unterscheiden:

Merkmal	Ausprägung	Beispiel im Referenzprojekt
Konvertierungsmethode(n)	manuell automatisch gemischt	Zahlreiche Einzelschritte der Konvertierung (Textexport, automatische Einführung von Verknüpfungen, Nacherfassung von Medien (Abbildungen), manuelle Überarbeitung und Korrektur, Einführung von Markup etc.)
Konvertierungswerkzeuge	Exportroutinen, UNIX- bzw. OS-Tools, z. B. grep, Search & Replace, Skripte SGML-Konverter Eigenentwicklungen	Exportmakros, Skripte, Ersetzungsroutinen zur Nachbearbeitung exportierter Daten, Formatierungswerkzeuge zur Aufteilung in einzelne Hypertextknoten/Buchseiten

Tabelle 8: Methoden und Werkzeuge der Konvertierung

In konzeptueller Hinsicht muss die Konvertierung geeignete Strategien verfolgen, um den linearen Ausgangstext in ein für das elektronische Buch geeignetes Hypertextformat umzuwandeln. Dabei unterscheidet STORRER 1998: 34 ff. in Anlehnung an KUHLEN 1991: 161 ff. folgende Strategievarianten:

- Die *einfache Konversion*, bei der der Ausgangstext in seiner Gesamtheit zu einem Hypertextknoten wird, eine Verfahrensweise, die für Aufsätze innerhalb digitaler Bibliotheken typisch ist.
- Die Segmentierung nach *formalen Texteigenschaften* wie Überschriften oder Absätzen; dies entspricht dem in dieser Arbeit diskutierten Konzept der strukturellen Informationsauszeichnung (Strukturmarkup, vgl. unten Kap. 4.2.1).
- Die Konversion anhand von *Kohärenzkriterien*, d. h. die Anwendung textlinguistischer Analysen für die Umwandlung linearen Texts in Hypertextknoten; dazu findet sich in dieser Arbeit eine Analogie in der Einführung inhaltsorientierten Markups und dessen Berücksichtigung bei der Umwandlung des Ausgangsmaterials, vgl. unten Kap. 4.2.2).
- Die *intertextuelle Konversion*, bei der zunächst nicht miteinander in Verbindung stehende Texte (oder allgemeiner informationelle Einheiten) systematisch miteinander verknüpft werden, was in dieser Arbeit durch die Einbindung zusätzlicher (externer) Komponenten erfolgt.

Für nicht-textuelle Elemente (Bilder, Diagramme) kommen entsprechende Konvertierungsvorgänge hinzu, soweit die für das elektronische Buch definierten Zielformate dies erforderlich machen. Eine besondere Bedeutung kommt der Verwendung deklarativer Auszeichnungsformate zu, die vor allem für Texte, aber zunehmend auch für multimediale Daten von Belang ist. Dabei ist zu unterscheiden,

- auf welcher Basis die deklarative Informationsstrukturierung erfolgt (z. B. XML),
- ob vordefinierte Dokumentgrammatiken verwendet werden (z. B. HTML, DocBook),
- ob Dokumentgrammatiken neu definiert werden,
- welche Verarbeitungswerkzeuge zum Einsatz kommen und
- auf welcher Ebene im Erstellungsprozess die deklarativen Formate eine Rolle spielen (z. B. zentraler Datenpool in einer SGML-Datenbank, Produktion eines elektronischen Buchs in einem proprietären Format eines Autorensystems).

Im Referenzprojekt wurde zunächst HTML als vordefinierter Dokumenttyp eingesetzt, im Rahmen dieser Arbeit wurden dann nach der in Kap. 4.2.5 vorgestellten Methode dedizierte XML-Dokumentgrammatiken entwickelt (vgl. Anhang 17.1). Neben die Frage der primären Informationsauszeichnung tritt die Beschreibung von Metadaten, für die zu präzisieren ist,

- welche Kodierungsstandards zum Einsatz kommen (z. B. das *Resource Description Framework* (RDF)),
- für welche Medientypen die Beschreibung erfolgt (Metadaten für Texte, digitale Bild- und Videodaten etc.) und
- welchen Anwendungsbezug die Metadatenbeschreibung hat (z. B. bibliographische Beschreibung (Dublin Core), Indexierung, Beschreibung von Lehrmaterialien).

Eine Beschreibung von Metadatenformaten und ihrer Einsatzgebiete findet sich in Kap. 9, ihr Einsatz in dynamischen elektronischen Bücher wird in Kap. 10.5 erörtert.

3.2.2.4 Erstellungswerkzeuge

Bei den Erstellungswerkzeugen, die im Rahmen der Entwicklung elektronischer Bücher Verwendung finden, lassen sich im wesentlichen drei Typen unterscheiden:

- Bearbeitungswerkzeuge für individuelle Medientypen (Editoren für Texte, Bildbearbeitungsprogramme, 3D-Software, Video- und Audioeditoren etc.),
- Autorensysteme als integrierte Entwicklungsumgebungen,
- die Verwendung höherer Programmiersprachen für einzelne Entwicklungsaufgaben.

Für CD-ROM-basierte multimediale Bücher dominiert die Verwendung von Autorensystemen, insbesondere Macromedia Director, was allerdings den Nachteil proprietärer Datenformate und damit u. U. auch eingeschränkter Wiederverwendbarkeit der erzeugten Medienkomponenten („Datenkonserven“) mit sich bringt. Entwicklungswerkzeuge für elektronische Bücher werden im Rahmen der Diskussion des hier entwickelten dynamischen elektronischen Buchs näher betrachtet (vgl. unten Kap. 11).

3.2.2.5 Speicherungs- und Verwaltungstechnologie

Bei umfangreichen Datenbeständen elektronischer Bücher, die entweder eine sehr hohe Anzahl informationeller Einheiten umfassen oder sehr speicherintensive multimediale Elemente wie Bewegtbilder beinhalten, ist das Speicherungsverfahren ein wesentlicher Aspekt, der in Zusammenhang mit der zyklischen Entwicklung elektronischer Bücher

gesehen werden muss (Kreislauf der Erstellung, Verwaltung, Distribution/Präsentation und Überarbeitung der Inhalte). Tabelle 9 fasst wesentliche Merkmale zusammen:

Merkmal	*Ausprägung/Unteraspekte*	*Referenzprojekt*
Speicherungstechnologie	• Dateisystem • Datenbanksystem (relationales, objekt-relational, objekt-orientiert, multimedial) • *Content Management-System* • Mischform (z. B. Textbasis in objekt-orientierter Datenbank, Benutzerprofile in einer relationalen Datenbank, Multimediaelemente im Dateisystem)	Dateisystem; im Rahmen des Prototyps eines dynamischen elektronischen Buchs Verwendung von Datenbanken für administrative Aufgaben
Speicherort	• Distributionsmedium • auf zentralem Server • verteilt im Netzwerk • Mischformen	Verwaltung auf zentralem Server; Distribution über CD-ROM bzw. als WWW-Informationsdienst
Bezugsebenen der Speicherung	• Erfassung und Produktion • Distribution/Präsentation/Nutzung	Erfassung, Produktion: Webserver Distribution: CD-ROM
Update- und Erweiterungsmöglichkeit	Durchführung des Updates (automatisch, manuell) Umfang des Updates (vollständig oder nur Addition von neuem Material) Hybride Lösung	im Referenzprojekt keine gesonderte Updatemöglichkeit

Tabelle 9: Merkmale der Inhaltsspeicherung

3.2.3 Modellierungs- und Entwicklungsmethoden

Elektronische Bücher mit multimedialen Inhalten sind, wie bereits dargelegt, durch die Vielzahl der an ihrer Erstellung Beteiligten sowie durch die nicht-triviale Verbindung traditioneller Publikationsprozesse mit denen der Softwareentwicklung gekennzeichnet. Eine allgemein verbreitete oder gar standardisierte Entwicklungsmethode hat sich für elektronische Bücher bislang nicht herausgebildet; bei der Entwicklung elektronischer Bücher finden sich daher unterschiedliche Methoden:

- Verfahren der Texttechnologie, insbesondere Methoden für die Entwicklung von Dokumentgrammatiken (vgl. MALER & EL ANDALOUSSI 1996 und Kap. 4.2.5),
- Entwicklungsmethoden aus dem Software-Engineering (z. B. *rapid prototyping*) und dem objekt-orientierten Design (Analysemethoden, graphische Darstellungsformalismen wie die *unified modeling language* (UML, vgl. FOWLER & SCOTT 1998)),
- Ansätze für die systematische Entwicklung webbasierter Informationssysteme (*web engineering*, vgl. MURUGESAN 1999, CONALLEN 1999)
- strukturierte Darstellungsformate für den Multimediabereich zur Entwicklung von Drehbüchern und Storyboards (vgl. BURGER 1996, GARRAND 1997).

Eine diese verschiedenen Aspekte integrierende Entwicklungsmethode, die Gestaltung, Software-Engineering und texttechnologische Informationsmodellierung in einem einheitlichen Rahmen integriert, existiert bisher nicht bzw. nur in einzelfallbezogenen Ansätzen, vgl. das von DEGEN 1996 diskutierte Konzept, Multimediaentwicklung als Integration der drei Ebenen *Technikdesign*, *Mediendesign* und *Interaktionsdesign* aufzufassen; bei diesem integrierenden Ansatz fehlt aber der für elektronische Bücher zentrale texttechnologische Aspekt.

3.2.4 Buchfunktionalität

Der Bezug elektronischer Bücher zu analogen Printprodukten wirft die Frage auf, ob und wie ein elektronisches Buch funktionelle Merkmale gedruckter Bücher aufgreift. Dazu gehören

- die Möglichkeit, Lesezeichen anzulegen,
- Annotationen durch den Leser,
- die Identifikation von Präsentationseinheiten (z. B. durch Seitenzahlen),
- die Verwendung von Fuß- oder Endnoten

Zusätzlich stellt ein Beurteilungskriterium die Korrelation des elektronischen Buchs mit einer ggf. vorhandenen Printversion durch explizite Verweise dar.

3.2.5 Hypertext-Eigenschaften

Wie bereits gezeigt wurde, lassen sich elektronische Bücher als restringierte Hypertexte auffassen. Die Beurteilung der Hypertextmerkmale eines elektronischen Buchs gehört daher auch zu den wesentlichen funktionalen Aspekten. Die folgende Tabelle gibt die wichtigsten Kriterien wie oben in Kap. 2.2.1 eingeführt wieder:

Hypertextmerkmal	*Typische Ausprägungen*	*Referenzprojekt*
Architekturmodell des Hypertextsystems	z. B. Dexter, OHS	mehrschichtiges Architekturmodell in Analogie zum Dexter-Modell
Kleinste Modellierungseinheit für Hypertextknoten	inhaltlich, darstellungsorientiert	orientiert an Struktur- und Inhaltsmarkup sowie auf unterster Ebene an den Gegebenheiten des Präsentationssystems
Erstellung der Hyperlinks	automatisch manuell interaktiv durch geeignete Editoren	automatische Erkennung und Generierung struktureller Links
Markup für Verknüpfungen	HTML-Anker, XLink-Links, proprietäres Format eines Autorensystems	HTML-Anker
Darstellung der Hyperlinks	typographisch, durch Symbole, durch Farben	typographisch und durch Symbole
Kardinalität der Verknüpfungen	1:1, 1:n, n:m	1:1, 1:n, n:m
Typisierung von Verknüpfungen	nach Funktion, nach Inhalten	sowohl nach Inhalten (Komponententypen) als auch nach Funktion (Querverweis, Navigation)

Tabelle 10: Hypertext-Eigenschaften elektronischer Bücher

3.2.6 Gestaltung und Präsentation

Die Gestaltung und Präsentation spielt für elektronische Bücher eine in mehrfacher Hinsicht eine Rolle, da nicht nur die typographische Gestaltung wie in Printprodukten, sondern auch die Gestaltung der Benutzerschnittstelle für das elektronische Buch sowie der eingebetteten Multimedia- und Softwarekomponenten zu regeln ist. Die folgende Tabelle zeigt wesentliche Merkmale der Gestaltung. Software-ergonomische Anforderungen an die Gestaltung von elektronischen Büchern werden in Kap. 13.2 diskutiert.

Merkmal	*Ausprägung/Unteraspekte*
Globale Aspekte des Screen Design	Einheitlicher Aufbau der Benutzerschnittstelle durch Farbleitlinie Gestaltung von • Typographie • Layout • Steuerelemente • Multimediakomponenten
Präsentationsmetapher	Visuelle Metapher (z. B. Labormetapher) Funktionale Metapher (z. B. Buchmetapher) Realweltbezug der Metapher
Navigationsmodell	Zugriff über Übersichten/Verzeichnisse Sequentielle Navigation/“Blättern“ Hierarchienavigation Technische Realisierung der Navigation (Hypertextverknüpfungen, Funktionen des Autorensystems etc.)
Präsentationssystem	Viewer eines Autorensystems (Director, ToolBook) WWW-Browser, SGML-Browser Eigenentwicklung
Präsentationsplanung	Verwendung multimodaler Informationsdarstellung Dynamische Generierung unterschiedlicher Präsentationsformate

Tabelle 11: Gestalterische Aspekte elektronischer Bücher

3.2.7 Multimediale Inhalte

Neben der nicht-linearen Strukturierung von Inhalten durch Hypertextverknüpfungen ist die Einbettung interaktiver multimedialer Inhalte ein weiteres Element elektronischer Bücher, das informationellen Mehrwert gewährleisten kann. Sie lassen sich nach den in der folgenden Tabelle gezeigten Kriterien einordnen:

Merkmal	*Ausprägung*	*Referenzprojekt*
Typ	Bild (Fotografien, Diagramme, schematische Darstellungen, Funktionsgraphen) Bewegtbild (Filme, (nicht-interaktive) Animation) Ton (Sprachdokumente, Musik, Geräusch) Interaktive Elemente (Animationen, Simulationen, 3D-Modelle)	schematische Abbildungen Photographien Animationen Simulationen
Entstehung	Integration von Altbeständen Neuentwicklung Übernahme aus externen Ressourcen	Übernahme aus dem Printprodukt Neuerfassung und Entwicklung
Integration	gestalterische Integration Einbindung in das Gestaltungs- und Navigationsmodell (eingebettet in den Text des elektronischen Buchs, als eigenständige Gestaltungskomponente) technische Integration • durch plug-ins • als eigenständige Softwarekomponente (z. B. ActiveX, Java Beans, OLE) • als MIME-Typ mit zugeordneter Betrachtungssoftware	integriert als vom Buchtext getrennte Komponenten, inhaltliche Einbindung durch typisierte Verknüpfungen; technische Integration durch eingebetteten Interpreter (Java, Simulationen), plug-in-Software (Shockwave, Animationen), bzw. Darstellungsmöglichkeiten des Webbrowsers (Bilder)

Tabelle 12: Merkmale multimedialer Komponenten elektronischer Bücher

3.2.8 Didaktische Aspekte

Soweit es sich bei einem elektronischen Buch um Lehr- bzw. Lernmaterialien handelt, sind Aspekte der Didaktik zu beachten, die die folgende Tabelle darstellt:[27]

Merkmal	*Ausprägungen/Aspekte*	*Referenzprojekt*
Theoretische Basis	Konstruktivismus Behaviorismus exploratives Lernen *instructional design*	Exploratives Lernen durch Interaktion mit multimedialen Animationen und Simulationen; Ergänzung traditioneller Lernformens
Übungsformen	Freitextaufgaben formalisierbare Übungstypen (Multiple-Choice-Aufgaben, Lückentext, Zuordnungsaufgaben, Berechnungen) komplexe interaktive Abläufe	wenige Übungsbeispiele; Animationen und Simulationen zur Exploration und Vorbereitung
Auswertung	automatisch/wissensbasiert intellektuell	- (ggf. durch integrierte Dienste)
Standardisierte (Metadaten-) Beschreibung der Lehr- und Lernmaterialien	IMS Ariadne IEEE Learning Objects Model (IEEE LOM), vgl. unten Kap. 9.2.2	Beschreibung multimedialer Komponenten mit Hilfe von Metadatenformaten.
Integration	Integration in Unterrichtsabläufe Didaktischer Einsatzort Kommunikationskanäle Technische Realisierung	Einsatz im Umfeld eines Physikpraktikums zu Vor- und Nachbereitung

Tabelle 13: Didaktische Aspekte elektronischer Bücher

3.2.9 Informationserschließung und Einbindung externer Ressourcen

Verfahren zur Informationserschließung sind für elektronische Bücher in zweierlei Hinsicht relevant: Zum einen für die Erschließung der Inhalte eines elektronischen Buchs selbst, zum anderen hinsichtlich der Erschließung zusätzlicher Informationen aus externen Quellen. Eine vertiefte Darstellung der Informationserschließung und ihrer Anwendung für dynamische elektronische Bücher findet sich in Kap. 14.2.2 im Kontext der Integration von Diensten. Die folgende Tabelle zeigt die Hauptkriterien für die Beurteilung der Informationserschließung bzw. der verwendeten Information Retrieval-Verfahren:

Merkmal	*Ausprägungen/Aspekte*	*Beispiel/Referenzprojekt*
Information Retrievalmodell	• boolesch • vektorbasiert/statistisch • probabilistisch • Sonderform (z. B. direkte Stringsuche)	automatisch erstelltes Register, Volltextrecherche mit boolescher Logik und statistischem Ranking
Indexierung des Ausgangsmaterials	• automatisch, • intellektuell	automatische Extraktion gekennzeichneter Fachbegriffe, automatische Volltextindexierung

[27] Eine umfassende Darstellung der pädagogischen Grundlagen des computerbasierten Lernens findet sich bei SCHULMEISTER 1997 sowie in dem Sammelband ISSING & KLIMSA 1997; STEINMETZ 1999: 816 ff. führt knapp in die wichtigsten Lerntheorien ein und gibt eine Übersicht zu unterschiedlichen Typen von Lehr- und Lernsoftware.

Merkmal	*Ausprägungen/Aspekte*	*Beispiel/Referenzprojekt*
Indexierung von Multimediadaten	• inhaltliche Analyse • Auswertung von Beschreibungen • Auswertung von Metadaten	Auswertung von Beschreibungen (Bildbeschriftungen etc.) und Metadaten
Anfragesprache	• Stichwortsuche/ Register • boolesche Logik • freie Formulierung	Stichwortsuche durch Register; boolesche Anfragesprache für die Volltextsuche
Bezugspunkt der Erschließung	• Inhalte des elektronischen Buchs • Externe Ressourcen	Im Referenzprojekt nur unmittelbare Inhalte; im Prototyp dynamischer elektronischer Bücher externe Ressourcen über Erschließungsdienste

Tabelle 14: Merkmale der Informationserschließung in elektronischen Büchern

Soweit in ein elektronisches Buch zusätzliche Ressourcen eingebunden werden sollen, die in technischer Hinsicht oder unter dem Gesichtspunkt des Urheberrechts (andere Autoren) nicht zum Grundbestand des elektronischen Buchs gehören, ist zu unterscheiden,

- um welchen *Ressourcentyp* es sich handelt (Texte, Datenbanken, z. B. Online-Nachweisdatenbanken), Multimediaelemente (z. B. Bilder, Video- oder Audioelemente)),
- wie die Einbindung erfolgt (implizit durch Erschließungsdienste, explizit durch Hypertextverknüpfungen) und ob sie statisch (feste Einbindung durch statische Verknüpfung) oder dynamisch (automatische Generierung durch Erschließungsdienste) ist.

3.2.10 Abrechnung und Sicherheit

Ein letzter relevanter Aspekt sind schließlich Verfahren für die Abrechnung von Informationsdienstleistungen und Fragen der Sicherheit bzw. der Vermeidung nicht lizensierten Gebrauchs der Daten elektronischer Bücher.

Merkmal	*Ausprägung/Unteraspekte*
Abrechnungs- und Lizenzverfahren	Nutzungslizenz • nur für das elektronische Buch, • für ein Medienbündel (gedrucktes Buch und CD-ROM) Nutzung über ein Abonnementsystem, z. B. einer digitalen Bibliothek Einsatz von *micro payment standards* für die kleinteilige Abrechnung der Nutzung von Buchinhalten (*pay per view*)
Sicherheitsaspekte	Kopierschutz Identifikation/Autorisierung (Paßwortschutz/*access control lists*) Authentisierung der Inhalte (digitale Signaturen, *message digests*, Wasserzeichen)

Tabelle 15: Sicherheits- und Abrechnungsaspekte elektronischer Bücher

3.2.11 Fazit

Die Darstellung des state-of-the-art elektronischer Bücher und die Beschreibung ihrer wesentlichen Merkmale verdeutlicht die Vielfalt der Erscheinungsformen elektronischer Bücher. Der Kriterienkatalog ist dabei bewusst weit gefasst und geht über die im Rahmen dieser Arbeit behandelten Aspekte dynamischer elektronischer Bücher hinaus. Dabei wird deutlich, dass ein einheitliches Vorgehen bei der Entwicklung elektronischer Bücher bisher nicht zu erkennen ist. Dies gilt sowohl für die verwendeten Technologien (z. B. Einsatz von Kodierungsstandards) als auch für konkrete gestalterische Umsetzung elektronischer Bücher.

3.3 Digital Appliances und Betrachter für elektronische Bücher

Der nachfolgende Exkurs dient der Einordnung des Konzepts dynamischer elektronischer Bücher in den weiteren Kontext der technologischen Infrastruktur digitaler Anwendungen. Seit einigen Jahren sind Entwicklungen zu beobachten, die auf eine Ablösung des Personalcomputers als arbeitsplatzbezogener universeller Rechenmaschine durch aufgabenbezogene digitale Geräte (*appliances*) abzielen (vgl. LEWIS 1998, HALLA 1998). Eine ausführliche Untersuchung dieser Entwicklung hin zu sog. *information appliances* liefert NORMAN 1998. Zu ihnen zählen nicht nur Netzcomputer als vereinfachte[28] Zugangsgeräte für Internetdienste und insbesondere das World Wide Web oder die Integration von Netzdiensten in die Funktelephonie (z. B. durch das *Wireless Application Protocol* (WAP), vgl. WAPFORUM 1999), sondern auch für den Kontext dieser Arbeit relevante dedizierte Buchbetrachtungssysteme. Erste elektronische Geräte (*appliances*) stehen mittlerweile für elektronische Bücher bzw. für die Entwicklung von Bibliotheksdiensten zur Verfügung. Die verschiedenen Produkte bzw. Prototypen (u. a. *Xlibris*, *EveryBook*, *The Lectrice*, *NewsPad*, *Rocket ebook*, *Millenium Ebook*)[29] führen das Konzept der *personal digital assistants* (PDAs) weiter und haben folgende gemeinsame Charakteristika:

- Seitengerechte Darstellung, d. h. Darstellung in Analogie zu regulären Buchseiten,
- LCD-Display über interaktiven Touchscreen mit der Möglichkeit handschriftlicher Annotation (Marginalien, Highlighting etc.),
- Verzicht auf zusätzliche Interaktionsinstrumente wie Tastatur oder Maus (Zeigegerät) und
- Austauschbare Inhalte, die z. T. über das Internet geladen werden können.

Der bekannteste Vertreter solcher Buchbetrachtungssysteme ist das *Rocket eBook*, für das Anfang des Jahres 2000 bereits über 2000 elektronische Bücher als Inhalte zur Verfügung standen (vgl. ROUSH 1999B, LADWIG 1999 und *http://www.rocket-ebook.com*). Das *Rokket eBook* ist durch die folgenden Merkmale zu charakterisieren:

- Bildschirmgröße ca. 11,5 × 7,5 cm (4,5‘‘ × 3‘‘),
- Auflösung ca. 110 dpi, d. h. ca. 500 × 300 darstellbare Pixel,
- zwischen 4 und 48 MB Speicherkapazität,
- Möglichkeit zur Annotation durch den Benutzer,
- Betriebsdauer 20-45 Stunden pro Akkuladung,
- einfache Benutzerschnittstelle mit symbolischer Darstellung der Navigations- und Interaktionsfunktionen und
- Laden von Inhalten über das Internet in einem proprietären Format bzw. HTML (vermittelt durch PC mit Netzzugang).

Abbildung 14 zeigt zwei Beispiele für hardwarebasierte Buchbetrachtungssysteme:

---

[28] Vereinfacht insofern, als sie auf lokale Speichermedien weitgehend verzichten, ein reduziertes Betriebssystem aufweisen und sich auf wenige Anwendungen, insb. einen Webbrowser beschränken.

[29] Vgl. PRICE, GOLOVCHINSKY & SCHILIT 1998, SCHILIT, PRICE & GOLOVCHINSKY 1998, SCHILIT et al. 1999 sowie *http://www.ebooknet.com* mit einer Übersicht zu den am Markt verfügbaren sowie geplanten eBook-Viewern.

Abbildung 14: Die Buchbetrachter Rocket eBook (links) und EveryBook (rechts)

Bemerkenswert ist, dass nicht nur die Analogie zum gedruckten Buch ein wesentliches Beurteilungskriterium für *eBooks* ist (vgl ROUSH 1999A, 1999B, 1999C), sondern die Weiterentwicklung solcher Buchbetrachtungssysteme auch noch zusätzliche Elemente des Printmediums aufgreift, wie der in Abbildung 14 gezeigte Prototyp des für Mitte 2000 am Markt erwarteten *EveryBook* mit seiner Darstellung von Doppelseiten wie im gedruckten Buch zeigt (vgl. ROUSH 1999C).

Im Rahmen dieser Arbeit sind dedizierte Hardwarelösungen für elektronische Publikationen insofern relevant, als die Architektur dynamischer elektronischer Bücher, wie sie in Kap. 4 entwickelt wird, grundsätzlich auf der Inhaltskodierung mit Hilfe deklarativer Markupstandards aufbaut und die Möglichkeit der Transformation in unterschiedliche Ausgabeformate zulässt. Damit ließe sich das Modell dynamischer elektronischer Bücher an hardwarebasierte Buchbetrachter anpassen. Zwar wird im hier diskutierten Prototyp ein Webbrowser (Netscape) als clientseitige Betrachtungssoftware verwendet; diese Lösung wäre somit vor allem mit *Web Browsing Appliances* kompatibel, ließe sich aber auch an die Nutzungsmöglichkeiten spezifischer eBooks anpassen, soweit sie deklarative Markupstandards verwenden, wie den in Kap. 6.3 diskutierten *Open eBook*-Standard, eine Weiterentwicklung von HTML für die Anwendung in hardwarebasierten Buchbetrachtern.

4 Dynamische elektronische Bücher

Die Darstellung des state-of-the-art elektronischer Bücher sowie des zuletzt erörterten Merkmalskatalogs zeigt, dass bisher kein einheitliches Modell elektronischer Bücher existiert, dass aber bei einer Vielzahl von Ansätzen gemeinsame Strategien, z. B. die Integration von Multimediakomponenten in elektronische Publikationen oder die Verwendung kommerzieller Multimedia-Autorensysteme wie Macromedia Director erkennbar sind. Nachfolgend soll das Konzept *dynamischer elektronischer Bücher* als Weiterführung des bisherigen Stands elektronischer Bücher eingeführt und diskutiert werden. Es wird ein Architekturmodell für elektronische Bücher auf der Basis allgemein verfügbarer Standards entwickelt. Dieses Modell zeichnet sich durch folgende grundlegende Merkmale aus:

- Die Dokumentenstrukturierung mit Hilfe XML-basierter Standards als Voraussetzung der *Dynamisierung* elektronischer Publikationsinhalte. Dies wird durch die Verwendung von deklarativen Auszeichnungen auf den Ebenen *logische Struktur*, *Inhaltsaufbereitung* und *Präsentation* erreicht und durch die Beschreibung von *Metadaten* ergänzt.
- Die *Einbettung interaktiver multimedialer Komponenten* als integraler Bestandteil dynamischer elektronischer Bücher sowie die Beschreibung der Schnittstellen zwischen dem elektronischen Buch und seinen eingebetteten Komponenten.
- Die Erweiterbarkeit der Inhalte und der Funktionalität elektronischer Bücher durch *Integration externer Dienste* und *Ressourcen*.
- Die einheitliche Entwicklung von Benutzerschnittstellen durch Verwendung von *style standards* und Gestaltungsleitlinien (Navigation, Viewerfunktionalität).

Die genannten Aspekte bilden den Kern des Konzepts dynamischer elektronischer Bücher. Ein solcher Ansatz lässt sich mit dem Vorhaben vergleichen, aktive oder mehrwertige (*multivalent*) Dokumente zu modellieren (vgl. PHELPS & WILENSKY 1996). Allerdings ist hier das Augenmerk auf die komplexere Informationsmenge eines elektronischen Buchs gerichtet. Es sind aber auch zusätzliche, weiterführende Nutzungsmöglichkeiten denkbar:

- Die kontextabhängige Generierung informationeller Einheiten auf der Basis unterschiedlicher Auswertung inhaltsorientierten Markups, woraus sich verschiedene Nutzungsmodelle ergeben können (z. B. die Wiederverwendbarkeit eingebetteter Komponenten durch einheitliche (Metadaten-)Beschreibung als Lehr- und Lernmaterialien (vgl. LOBIN 1999B)).
- Das Aufgreifen von Nutzerdaten durch Auswertung der Interaktion des Benutzers mit dem elektronischen Buch, wie dies etwa im *Projekt MultiBook* für die elektronische Fassung von STEINMETZ 1999 geschieht (vgl. *http://www.multibook.de*).
- Die Integration dynamischer Bücher in elektronische Bibliotheken, ggf. unter Berücksichtigung von Verfahren zur Identifizierung elektronischer Inhalte und der Verwendung unterschiedlicher Abrechnungsmodi (vgl. unten Kap. 15).

In den folgenden Abschnitten werden die Merkmale dynamischer elektronischer Bücher betrachtet; aus ihnen ergibt sich dann ein Architekturmodell, das die konzeptuelle

Grundlage des in Teil III zu diskutierenden Prototyps darstellt. Dabei steht eine *problemorientierte Sichtweise* im Mittelpunkt, die von technischen Realisierungsmöglichkeiten weitgehend abstrahiert. Eine vertiefende Diskussion der zur Umsetzung des Architekturmodells verfügbaren Standards der Informationsaufbereitung bzw. der Implementierungstechnologien findet sich in Teil II dieser Arbeit (Informationskodierung) sowie in den Kap. 11-13 in Teil III (Softwaretechnologien).

4.1 Dynamisierung von Publikationsinhalten

Dem Konzept *dynamisches* elektronisches Buch steht der Begriff *statisches* elektronisches Buch gegenüber, das zwar das elektronische Medium als Präsentationsplattform nutzt, aber informationelle Einheiten in einer fest vorgegebenen Struktur präsentiert und ggf. einfache Formen der Hypertextverknüpfung als zusätzlichen Navigationsmodus (*browsing*) neben der sequentiellen Präsentation anbietet. Im Gegensatz dazu sei unter der Dynamisierung von Publikationsinhalten die Erweiterung von Publikationsinhalten durch Komponenten und Dienste verstanden, die dem Benutzer verschiedene Interaktionsmöglichkeiten bieten und deren Umfang veränderbar ist. Grundsätzlich Normalfall ist davon auszugehen, dass eine solche Dynamisierung zu unterschiedlichen Zeitpunkten stattfinden kann und von einem informationellen Kernbestand ausgeht, der als Bezugspunkt dient (Basis- oder Primärdatenbestand des Buchs). In einer erweiterten Definition kann man unter der Dynamisierung von Publikationsinhalten auch die Möglichkeit verstehen, aus gegebenem Material unterschiedliche Publikationsprodukte zusammenzustellen (Perspektive von Substanzanbieter/Verlag bzw. Autor); dies soll hier aber nicht weiter untersucht werden. Es steht vielmehr die Dynamisierung aus der Perspektive des Benutzers im Zentrum der Überlegungen. Das dynamische Verhalten elektronischer Bücher lässt sich drei Ebenen zuordnen:

- Der Ebene der *Inhalte*, d. h. der Frage, inwiefern sich ein Ausgangsdatenbestand durch Autoren bzw. Benutzer verändern und erweitern lässt und zu welchem Zeitpunkt dies geschehen kann,
- der Ebene der *Interaktion* bzw. der eingebetteten Komponenten. Darunter sind in diesem Kontext vor allem *eingebettete Multimediakomponenten* zu verstehen, die interaktives dynamisches Verhalten ermöglichen, und
- der Ebene der *Präsentation*, d. h. der Möglichkeit, Inhalte aus einem vorgegebenen Ausgangsdatenbestand auf unterschiedliche Weise zu präsentieren, wobei lokale Moduswechsel für eine informationelle Einheit (Wahl unterschiedlicher Präsentationsweisen, z. B. tabellarisch vs. diagrammatisch) von globalen Präsentationsfiltern zu unterscheiden sind.

Folgende Faktoren können bei der dynamischen Präsentation eine Rolle spielen:

- Medientypus und Präsentationsmodus,
- gezielte bzw. ad-hoc-Generierung einer informationellen Einheit,
- vollständige Integration ergänzender Materialien in den Ausgangsbestand bzw. Parallelstellung, vor allem bei Ermittlung inhaltlich passender externer Ressourcen zur Laufzeit,
- Entstehung durch Integration von Modulen einer erweiterten Informationsinfrastruktur (Dienste).

Der Aspekt der Dynamisierung von Publikationsinhalten durch die *Interaktion* mit dem Dokument durch den Leser macht deutlich, dass sich der Dokumentbegriff – unabhängig

von der statischen oder dynamischen Generierung – in elektronischen Büchern einem allgemeinen Begriff eines *interaktiven Informationssystems* annähert: In elektronischen Büchern können Softwarekomponenten unmittelbar eingebettet oder aus dem Dokument erreichbar sein (Interaktion mit externen Modulen): Dieser Gedanke sieht die elektronische Publikation nicht als *Zusatz* zu einer Softwarekomponente (z. B. einer Simulation), wie das etwa bei Hilfesystemen bezüglich der unterstützten Softwarekomponente der Fall wäre. Statt dessen steht das Publikationskorpus im Zentrum. Die Annäherung von *electronic publishing* bzw. Hypertext und *software engineering* findet mittlerweile in der Literatur Beachtung, insbesondere hinsichtlich der Fragestellung, ob ggf. methodisches Wissen aus dem *software engineering* auf die Hypertextentwicklung übertragbar ist, vgl. BRERETON, BUDGEN & HAMILTON 1998.

Damit kehrt sich das gewöhnliche Verhältnis elektronisch verfügbarer Dokumentation zu Softwarekomponenten um: Während textuelle Beschreibungen, Hilfesysteme, online-Lexika wie man-pages etc. nur einen Nebenaspekt der Softwarenutzung darstellen, steht bei elektronischen Büchern die Informationspräsentation über ein geeignetes Viewingsystem im Mittelpunkt. Als Ergänzung kommt die Nutzung spezifischer Komponenten oder generischer Dienste hinzu. Dazu braucht man Verfahren, die über die reine Paralellstellung von Publikationsinhalten und zugeordneten Komponenten hinausgehen: Das Beispiel instantiierbarer Formeln[30] kann dies verdeutlichen: Die mathematische Herleitung eines physikalischen Zusammenhangs verlangt in vielen Fällen an ihrem Endpunkt nach einer exemplarischen Instantiierung, d. h. mit einer im Text dargestellten Formel soll interagiert (gerechnet) werden können.[31] Solche Interaktionsmöglichkeiten durch Integration von Komponenten bzw. Diensten sind ein Beispiel für die Dynamisierung von statischen Dokumentinhalten. Wie der state-of-the-art elektronischer Bücher zeigt, sind sie das am weitesten verbreitete Merkmal, das die Weiterentwicklung elektronischer Bücher über einfache Hypertextanwendungen hinaus charakterisiert.[32]

Ein Beispiel aus dem Kontext des Multimedialen Physikalischen Praktikums kann dies verdeutlichen: Ein Versuch mit theoretischer Herleitung und Erläuterung eines physikalischen Phänomens sowie einer Beschreibung zur Versuchsdurchführung dient als Ausgangspunkt der Interaktion, d. h. der Leser rezipiert die allgemeine Darstellung eines physikalischen Phänomens und arbeitet sich in den Versuchsaufbau und die Anleitung zur Versuchsdurchführung ein.

Parallel zur Versuchsbeschreibung kann der Leser eine Versuchssimulation aktivieren, die als Komponente (z. B. als ein Java-Applet) in die Publikation integriert ist; ggf. können bereits die Startparameter der Simulation aus dynamisch gewählten oder errechneten Werten aus dem Text heraus in die Simulation übernommen werden. Sind geeignete Schnittstellen und Dienste verfügbar, so stehen eine Vielzahl von Interaktions- und Verarbeitungsschritte offen, die auf die Interaktion des Benutzers mit der eingebetteten Multimediakomponente folgen:

---

[30] Zur prototypischen Realisierung mit Hilfe der *Mathematical Markup Language* (MathML, vgl. Kap. 6.6) siehe unten Kap. 14.2.4.

[31] Vgl. dazu den im Statistiklernpaket LernSTATS realisierten Ansatz, SCHULMEISTER 1997: 346 ff. sowie das oben in Kap. 3.1.2.2 angesprochene Projekt einer mathematischen Arbeitsumgebung.

[32] Beispiele sind etwa die Simulation physikalischer Versuche im Referenzprojekt, die Simulation von Algorithmen aus dem Multimediabereich im Projekt *MultiBook* oder die interaktive Darstellung komplexer chirurgischer Operationsabläufe in der Multimedia-Galerie der Herzchirurgie, vgl. oben Kap. 3.1.2.

1. Bei Durchführung der Simulation werden neue Daten erzeugt (z. B. eine Messwertreihe).
2. Sie sollen anschließend gespeichert, ausgewertet oder dargestellt werden.
3. Der Benutzer instantiiert die mathematische Beschreibung des Phänomens zur Überprüfung der empirisch ermittelten Werte.
4. Er wählt für die neu generierten Daten eine geeignete Visualisierungsform (z. B. einen geeigneten Diagrammtypus).
5. Der Benutzer übergibt die Daten an ein externes Auswertungsprogramm, das sie z. B. auf Vorliegen eines bestimmten statistischen Verteilungsmodells hin überprüft.

Neben diesen am Sachproblem im engeren Sinn orientierten Interaktionsschritten kann in ähnlicher Weise eine Schnittstelle zur *Kommunikationsinfrastruktur* der elektronischen Publikation erforderlich sein, wenn die Arbeitsschritte Teil einer Übungsaufgabe sind. In diesem Fall ergibt sich aus der Aufgabenstellung die geforderte Struktur des Lösungsdatensatzes (Messreihe, Versuchsprotokoll, Auswertung etc.), der in den Kommunikationsprozess eingebracht werden soll, z. B. durch Versenden per e-Mail an einen Tutor, dessen Adresse über das elektronische Buch verfügbar ist.

Eine weitere Schicht in diesem Szenario könnte durch die dynamische Generierung von Daten zur Lesezeit bzw. deren Integration über externe Schnittstellen (Schnittstelle zum Versuchskontrollrechner, externe Sensoren, z. B. Geiger-Müller-Zählrohr mit Schnittstelle) hinzukommen.

Das Beispiel zeigt zwei wesentliche Aspekte der Dynamisierung elektronischer Bücher:

- Zum einen die *Erweiterung* durch für das elektronische Buch spezifische Komponenten (Versuchssimulationen und Animationen physikalischer Experimente),
- zum anderen die Integration *generischer Dienste*, die für eine Vielzahl elektronischer Bücher relevant sein können, die aufgrund ihrer Komplexität aber nicht für den Einzelfall entwickelt oder mit einem einzelnen Buch distribuiert werden können.

Die Realisierung des oben geschilderten *Einzelfalls* stellt aufgrund der Verfügbarkeit geeigneter Realisierungswerkzeuge kein prinzipielles Problem dar, solange man die Frage nach seinem Generalisierungspotential außer Acht lässt; neben der konkreten Umsetzung eines über den state-of-the-art hinausgehenden Szenarios interaktiver Dokumente und angesichts der engen ökonomischen und organisatorischen Randbedingungen bei der Produktion elektronischer Publikationen kommt der Generalisierung aber eine zentrale Rolle zu. Sie kann erreicht werden, wenn

- ausschließlich offene Standards verwendet werden, insbesondere für die Auszeichnung von Information (vgl. Teil II),
- eine Basissoftwareinfrastruktur für die Nutzung dynamischer Bücher zur Verfügung steht und
- die Ableitung des Einzelfallmodells (Datenmodell, Interaktionsformen, Datentransfer, Kommunikationsmodell) für den Autor/die Autoren mit angemessenem Aufwand zu spezifizieren ist (z. B. mit Hilfe eines *Autorensystems*).

Es ist zu klären, wie sich eine inhaltsorientierte Informationsauszeichnung realisieren lässt und welche Möglichkeiten sich für die Integration von Komponenten und Diensten in dynamischen elektronischen Büchern anbieten.

4.2 Deklarative Auszeichnung von Information

Unter der deklarativen Auszeichnung von Information sei die Anreicherung der Primärdaten eines elektronischen Buchs durch Auszeichnungsmarken verstanden, die

- die logische Struktur des Buches kodieren,
- Informationen über die kodierten Inhalte liefern,
- festlegen, wie eine bestimmte informationelle Einheit präsentiert werden soll, bzw.
- zusätzlich Metainformation liefern.

In Bezug auf die Auszeichnung von Texten definieren SPERBERG-MCQUEEN & BURNARD 1993 den zugrundeliegenden Begriff Markup ohne detaillierte Differenzierung der Markupebenen wie folgt:

> By markup we mean all the information contained in a computer file other than the text itself, by means of which computer programs are able to manipulate texts in useful ways; the term is borrowed from the history of printing, where markup referred to the notations made in the margins of a text to guide the compositor in the layout of the text. Markup intended to specify the proper layout or presentation of a text is still the most common type of markup in computer files.

Die Auszeichnung ist *deklarativ*, wenn sie Kodierungsmerkmale verwendet, die allgemein verfügbaren Standards entstammen, nicht an ein bestimmtes Softwaresystem gebunden sind und ihren Auszeichnungszweck in einer für den Leser nachvollziehbaren Weise kodieren (vgl. BRYAN 1988: 5 ff., MALER & EL ANDALOUSSI 1996: 7 ff., CONNOLLY, KHARE & RIFKIN 1997: 120 ff.). So soll gewährleistet werden, dass die Kodierung im Ergebnis wiederverwendbare informationelle Einheiten liefert. Der Ausgangspunkt ist die Überlegung, dass sich in Texten elektronischer Bücher regelmäßige Strukturen finden lassen, die durch eine geeignete Grammatik beschrieben werden können. LOBIN 2000: 3 spricht in diesem Zusammenhang von „textueller Informationsmodellierung“ – durch die Besonderheiten dynamischer elektronischer Bücher, vor allem die Einbindung multimedialer Information sowie durch ihren Status als Softwareanwendung wird der Problembereich der Modellierung zwar erheblich erweitert, grundsätzlich lässt sich der Ansatz der deklarativen Informationsmodellierung aber für alle in einem elektronischen Buch enthaltenen Medientypen anwenden. Dies gilt auch auf der Ebene der Kommunikationsprozesse zwischen Softwarekomponenten, wobei die deklarative Auszeichnung von Information die Funktion eines interoperablen Datenaustauschformats annehmen kann, vgl. unten Kap. 12.4 zur Informationskodierung in den Steuerungskomponenten des elektronischen Buchs.

Diese Überlegungen verfeinern die begriffliche Trennung zwischen *Struktur* und *Präsentation*, wie sie dem Ansatz der *Standard Generalized Markup Language* (SGML) zugrunde liegt (vgl. dazu MALER & EL ANDALOUSSI 1996: 95 f.): Der Begriff *Struktur* ist für die inhaltsorientierte Aufbereitung dynamischer elektronischer Bücher nicht ausreichend, da er allgemeine Textstruktureigenschaften mit inhaltlichen Merkmalen vermischt, wie es in HTML als weit verbreiteter SGML-Dokumentgrammatik zum Ausdruck kommt: Elemente, die Überschriften bzw. Kapiteluntergliederungen (`<H1>`, `<H2>`) oder Binnenstrukturen des Textes (`<P>`, `<OL>`, `<UL>` etc.) auszeichnen, sind Strukturmerkmale im engeren Sinn, sagen aber nichts über die *Inhalte* aus, die sie kodieren, während die HTML-Marken `<ADDRESS>` oder `<INTRODUCTION>` bereits einen inhaltlichen oder wenigstens textfunktionalen Bezug haben. *Struktur* ist in diesem Sinn ein Gegenbegriff zu *Präsentation* bzw. *Layout*, da die SGML-Strukturmerkmale nichts über das Präsentati-

onsmedium und seine Charakteristika aussagen, die entsprechende Information erst durch Transformationssprachen wie DSSSL (s. u. Kap. 7.1.1) oder *Cascading Style Sheets* (CSS) dem Strukturmarkup zugeordnet werden muss. Für dynamische Publikationen mit inhaltsorientiertem Markup ist diese Zweiteilung daher durch die inhaltsbezogene Markupebene zu einer Dreiteilung (Struktur, Inhalte, Präsentation) zu erweitern.

4.2.1 Strukturmarkup

Zur logischen Struktur gehören Merkmale, die nicht an den *Einzelfall* eines bestimmten elektronischen Buchs gebunden sind, sondern generell für eine Vielzahl von Publikationsprodukten verwendbar sind. Dazu gehören Gliederungsebenen und die hierarchische Einteilung von Inhalten, die einfache sequentielle Abfolge informationeller Einheiten (z. B. Absätze als logische Gliederung eines längeren Texts) und untypisierte Hypertextverknüpfungen, die einen nicht näher spezifizierten Zusammenhang zwischen zwei informationellen Einheiten kodieren und dem Benutzer eine zusätzliche Navigationsmöglichkeit eröffnen. Strukturmarkup kann je nach Realisierung einer elektronischen Publikationsplattform eine Doppelfunktion übernehmen, wenn es zusätzlich als Präsentationsformat für mit inhaltsorientiertem Markup versehene Ausgangsdaten verwendet wird: Die Überführung von inhaltsorientierter Kodierung in reines Strukturmarkup bildet die inhaltsorientierte Auszeichnung (z. B. <DEFINITION>, <BEWEIS>, <VERSUCHSBESCHREIBUNG>) auf einfachere Textstrukturmerkmale wie Paragraphen, Überschriften etc. ab. In dieser Form ist das Publikationsprodukt mit Standardwerkzeugen wie einem Webbrowser ohne weiteres nutzbar, ein gebräuchliches Beispiel für diese Transformation, wie es im Referenzprojekt verwendet wird, ist die Abbildung einer inhaltsorientierten Beschreibung von SGML oder XML auf (X)HTML. Damit das inhaltsorientierte Markup als Bezugspunkt für die dynamische Ergänzung durch Komponenten und Dienste für den Benutzer nicht verloren geht, ist der Zeitpunkt der Transformation kritisch: Soll eine Materialzusammenstellung bereits vor der Distribution erfolgen (z. B. ausführlicher vs. einfacher Eintrag in einem elektronischen Lexikon), kann die Transformation bereits auf der Seite der Distribution erfolgen, d. h. als Umwandlung eines Materialbestands mit inhaltsorientierter Auszeichnung in einen Materialbestand, der weniger umfangreich ist und eine einfachere Binnenstruktur aufweist, aber nach wie vor inhaltsorientierte Marken enthält. Die eigentliche Übersetzung in Strukturmarkup mit zugeordneten Präsentationsanweisungen wird dann erst durch den Viewer auf der Seite des Nutzers erfolgen. Dieser Schritt ist unerlässlich, da gerade die Offenheit des Entwurfs unterschiedlicher Dokumentstrukturen bzw. Inhaltsauszeichnungen verhindert, dass ein Viewer a priori *Wissen* über die korrekte Präsentation aufweisen kann. Der Ansatz, eine *inhaltsorientierte* Auszeichnung für ihre Präsentation in ein Zielformat ohne Inhaltsmarkup zu transformieren, bietet die Möglichkeit, von der Zielplattform zu abstrahieren: Ein detailliertes Inhaltsmarkup bei der Aufbereitung der Daten kann erfolgen, ohne dass das *front end* bzw. der Benutzer dessen Details verstehen müssen. Die Aufbereitung durch Markup mit einer generischen logischen Struktur entspricht dem state-of-the-art elektronischer Publikationen, insbesondere in Form von in HTML kodierten Hypertextdokumenten.

4.2.2 Inhaltsorientierte Auszeichnung

Unter inhaltsorientiertem Markup wird die Einführung von Beschreibungsmarken verstanden, die anders als Strukturmarkup nicht eine allgemeine logische Struktur, sondern die konkreten Inhalte eines dynamischen elektronischen Buchs beschreiben.

Inhaltsorientiertes Markup ist als eine Form der Informationsanreicherung mit zusätzlichem Aufwand verbunden; seine Einführung muss durch einen entsprechenden *Mehrwert* gerechtfertigt sein. Ein solcher Mehrwert kann erzeugt werden, wenn

- in Abhängigkeit von den kodierten Inhalten unterschiedliche *Präsentationsformen* möglich sind (Beispiel: Präsentation von inhaltsorientiert kodierten Messdaten in tabellarischer oder graphischer Form, unter Zuhilfenahme entsprechender Transformationsverfahren),
- durch inhaltsorientierte Filter eine unterschiedliche Materialzusammenstellung erreicht werden kann,
- aufgrund der Analyse inhaltsorientierten Markups eine Zuordnung von Komponenten und Diensten zu den jeweiligen Inhalten möglich ist (Beispiel: Zusammenführung von Versuchsbeschreibung und Versuchssimulation über inhaltsorientiertes Markup; Anbindung externer Dienste an durch inhaltsorientiertes Markup gekennzeichnete Textpassagen, z. B. Zuordnung von Recherche- oder Nachschlagediensten zu inhaltlich gekennzeichneten Fachbegriffen) oder
- inhaltsorientiertes Markup für die Informationserschließung genutzt wird und der Wiederverwendbarkeit von Ressourcen bzw. deren Integration in eine Informationsinfrastruktur (z. B. eine digitale Bibliothek) dienen kann; dies gilt vor allem für die Beschreibung von Komponenten und Multimediadaten, bei denen für die Primärdaten (Bitmap, Videodatei) keine leistungsfähigen Erschließungsverfahren vorliegen.

Ein weiteres Beispiel für inhaltsorientiertes Markup stellen typisierte Verknüpfungen dar: In HTML als dem am weitesten verbreiteten Kodierungsstandard für Hypermedia-Daten werden nur einfache syntaktische Verknüpfungen verwendet. Sie sind sowohl in ihrer formalen Mächtigkeit (unidirektionale 1:1-Verknüpfungen) als auch in ihrer semantischen Aussagekraft eingeschränkt. Es gibt keine Möglichkeit, durch die Spezifikation der Verknüpfung eine Aussage über ihre Semantik bzw. die semantische Relation der verknüpften informationellen Einheiten zu machen. Dies geschieht i. d. R. unsystematisch durch entsprechende Ausgestaltung von Ankern: Reine Navigationslinks, die z. B. dem linearen Weiterblättern innerhalb des Informationsbestands dienen, werden durch geeignete Beschriftungen oder Graphiken kenntlich gemacht, während bei Links mit nichtnavigationaler Semantik die Beschriftung des Ankers meist einen Hinweis auf die semantische Rolle der Verknüpfung gibt. Auch wenn man einräumen muss, dass es nicht möglich ist, eine abschließende Menge möglicher Relationen zu definieren[33] oder ggf. automatisch zu erzeugen, ist die Möglichkeit, Links mit semantischer Zusatzinformation auszustatten, ein geeignetes Werkzeug, um die Dynamisierung elektronischer Publikationen zu realisieren und im Interface durch geeignete graphische Gestaltung kenntlich zu machen.

Neben der Frage nach der *Funktion* inhaltsorientierten Markups ist die *Granularität* ein wesentliches Unterscheidungsmerkmal. Hinsichtlich des Umfangs der als inhaltsorientiertem Markup kodierten (zusätzlichen) Information existiert eine offene Skala: An deren einem Ende steht *vollständige Transformation* aller beschriebenen Inhalte in deklarative Beschreibungen, d. h. die explizite *Wissensrepräsentation*: Auch wenn ein solcher Ansatz derzeit für elektronische Bücher nicht unmittelbar umsetzbar ist, da sich die komplexen multimedialen und sprachlichen Inhalte allenfalls in eng eingegrenzten, weitge-

[33] Die semantische Typisierung von Hypertextverknüpfungen ist ein Sonderfall der Klassifikation semantischer Beziehungen zwischen informationellen Einheiten; eine ausführliche Übersicht zu verschiedenen Ansätzen geben KUHLEN 1991: 106 und STEINMETZ 1999: 714 f.

hend formalisierbaren Domänen deklarativ beschreiben lassen, sei darauf verwiesen, dass sich deklarative Standards der Informationsauszeichnung zur Wissensrepräsentation eignen.[34]

Bei einer weniger ausgeprägten Beschreibungstiefe – inhaltsorientiertes Markup als echte Zusatzinformation – bedeutet dies, dass zu vorgegebenen informationellen Einheiten Attribute vergeben werden, deren Semantik durch das Publikationssystem in einer der oben dargelegten Formen bestimmt wird. Die Granularität der Zuordnung kann unterschiedlich sein:

- *Beschreibung* von Komponenten, z. B. einer Versuchssimulation durch Bezeichnung, Kurzbeschreibung, Bedienungsanleitung, Schlagworte etc.
- Angabe der *Funktion* von strukturellen (Text-)Einheiten, z. B. Kennzeichnung als Versuchsbeschreibung, Herleitung, Aufgabe, Beispiel etc.
- Kennzeichnung auf *Wort*- und *Phrasenebene* (z. B. als Fachbegriff, als Verweis auf eine Komponente etc.).

Die konkrete Auswahl von Funktion und Granularität des inhaltsorientierten Markup ist dem Einzelfall vorbehalten – gerade der Bezug zu den konkreten Inhalten einer Publikation schließt eine einfache Generalisierung des Einsatzes inhaltsorientierten Markups aus; in Kap. 4.2.5 wird versucht, eine *Methode* zur Ermittlung von Art und Umfang inhaltsorientierten Markups anzugeben; Kap. 10 stellt die Verwendung inhaltsorientierten Markups im Referenzprojekt vor. Es wird deutlich, dass die anwendungsspezifische Definition von inhaltsorientiertem Markup die Gefahr birgt, nicht-standardisiertes oder standardisierbares Wissen für die Aufbereitung der Information einzusetzen. Dem kann allerdings durch Einführung sekundärer Strukturierungs- bzw. Restringierungsformate entgegengewirkt werden, wie sie etwa die *Architectural Forms* als Erweiterung von SGML bereitstellen, vgl. LOBIN 2000: 85 ff. und unten Kap. 5.1.3. Zusammenfassend kann man den Versuch, durch inhaltsorientierte Auszeichnung von Information die Anknüpfung von Komponenten und Diensten zu ermöglichen, als einen Beitrag zur Entwicklung eines *semantic web* auffassen, das Tim Berners-Lee 1999A als wesentliche Herausforderung für die Weiterentwicklung des World Wide Web ansieht.

4.2.3 Präsentationsmarkup

Der dritte Aspekt der Informationsauszeichnung ist das Präsentationsmarkup, d. h. die Kodierung darstellungsrelevanter Information. Während generisches Strukturmarkup wie HTML mit Defaultregeln für die Präsentation versehen ist, die eine Darstellung in jedem Fall gewährleisten, erfordert inhaltsorientiertes Markup die zusätzliche Angabe von Präsentations- und Layoutinformation: Zu jedem Auszeichnungselement muss angegeben werden, wie es darstellt werden soll. Auf der Mikroebene betrifft dies die Information über Text- und Absatzformate, die Darstellung von Bildern und Tabellen, d. h. allgemein Regeln für die Darstellung der einzelnen Medientypen in einem elektronischen Buch.

Auf der Makroebene ist damit die Frage angesprochen, was die kleinste zusammenhängende Präsentationseinheit ist, welche Möglichkeiten das Viewing-System bietet und inwiefern die tatsächliche Darstellung in die logische bzw. inhaltliche Auszeichnung eingreift. Die Einbeziehung unterschiedlicher Präsentationsregeln für denselben inhaltsori-

[34] Ein Entwurf für einen Standard zur deklarativen Kodierung von Wissen ist die *Ontology Markup Language* (OML, vgl. *http://wave.eecs.wsu.edu/CKRMI/OML.html*); dort auch Beispiele für die Übersetzung von textuell repräsentiertem Wissen in OML.

entiert kodierten Informationsbestand stellt ein Beispiel für die Nutzung der Zusatzinformation in inhaltsorientiertem Markup dar: Je nach Nutzungssituation erfolgt eine unterschiedliche Transformation derselben Information in ein Präsentationsformat.

Die Kodierung präsentationsbezogener Information ist ein Werkzeug der Umsetzung gestalterischer und ergonomischer Anforderungen an elektronische Bücher; diese Anforderungen werden im Rahmen der Diskussion des Prototyps eines dynamischen elektronischen Buchs in Kap. 13.2 diskutiert. Kap. 7 erläutert verschiedene SGML-basierte Standards für die Kodierung von Präsentationsinformation.

4.2.4 Metadatenmarkup

Eine vierte, ergänzende Ebene der deklarativen Informationsauszeichnung ist die Kodierung von Metainformation. Metadaten sind von inhaltsorientiertem Markup in funktionaler und formaler Hinsicht abzugrenzen: Unter Metadaten sei diejenige deklarativ zu kodierende Information verstanden, die anders als inhaltsorientiertes Markup nicht in *unmittelbarem* Zusammenhang mit der konkreten Nutzungssituation steht, z. B.

- bibliographische Angaben zu einer Komponente,
- die Einstufung einer informationellen Einheit hinsichtlich ihrer Funktion im Rahmen eines Lehr-/Lernsystems oder
- die Verwaltung von Versionsinformation.

Lässt sich auf diese Weise eine *funktionale* Unterscheidung treffen, so kommt die formale Unterscheidung hinzu: Die Kodierung von Metadaten erfolgt mittels dedizierter Metadatenschemata wie dem *Resource Description Framework* (RDF, vgl. unten Kap. 9.1), das als generisches Format für unterschiedliche Metadaten-Funktionen angewandt werden kann und vom primären Markup unterschieden werden kann: Der allgemeine Ansatz des *Resource Description Framework* gibt nur ein *syntaktisches* Schema für die Kodierung von Metainformation vor, ohne Inhaltsszenarien zu spezifizieren oder Auswertungsmechanismen vorzuschreiben. Der Begriff Metadaten ist ein allgemeiner Containerbegriff, der weder einer der oben eingeführten Ebenen zugeordnet ist, noch einen Hinweis auf die funktionale Anwendung enthält. Dieser Ansatz geht über die „klassischen“ bibliographischen Metadatenformate hinaus, die wie *DigiMarc* oder *Dublin Core* im wesentlichen die Erfassung bibliographischer Daten und deren Nutzung bei der Informationserschließung zum Ziel haben.

Da RDF aber keinerlei Hinweise auf die Operationalisierung (Schlussregeln, *Knowledge Engine* etc.) enthält, ist in diesem Bereich eine Standardisierung mittelfristig kaum zu erwarten. Daher stellt sich für die Implementierung die Frage, ob für unterschiedliche Einsatzszenarien von Metadaten wie

- die Speicherung und Auswertung von Interaktionsdaten,
- die Bereitstellung funktional mächtiger Schnittstellen zwischen Publikationsinhalten und eingebetteten Multimediakomponenten oder externen Diensten oder
- die Kodierung von zusätzlichem Wissen über alternative Dokumentzusammenstellungen

in jedem Fall auf das generische Format von RDF zurückzugreifen ist oder ob es im Einzelfall nicht sinnvoller sein kann, durch die Definition zusätzlicher *XML*-DTDs Funktionsbereiche dynamischer elektronischer Bücher abzudecken.

4.2.5 Methodik der Inhaltsauszeichnung

Die voranstehenden Kapitel haben nur allgemein beschrieben, welche Anforderungen an die deklarative Auszeichnung von Information zu stellen sind. Sie haben aber keinen Hinweis auf die Ermittlung eines konkreten Markupmodells gegeben. Daher ist nun eine Methode für die Ermittlung eines Markupmodells zu entwickeln. Dabei ist von folgenden Prämissen auszugehen:

1. Ausgangspunkt der Analyse ist der vorhandene bzw. geplante *Informationsbestand* eines elektronischen Buchs. Im Referenzprojekt *Multimediales Physikalisches Praktikum* lag zunächst ein Teil des Informationsbestands als Druckfassung vor, während alle weitergehenden Komponenten (Multimediasimulationen, Dienste etc.) lediglich als abstrakte Anforderung spezifiziert waren. Eine solche „Mischsituation" ist typisch für die Entwicklung elektronischer Bücher, soweit sie von einem gedruckten Referenzwerk ausgehen.
2. Dedizierte Markupstandards stehen als Leitfäden bzw. Orientierungspunkte zur Verfügung (vgl. unten Kap. 6 zu einzelnen vordefinierten Markupmodellen).
3. Die verschiedenen Ebenen der Informationsauszeichnung (logische Struktur, Inhaltsbeschreibung, Präsentation, Metadaten) sind grundsätzlich erweiterbar, d. h. das Markupmodell muss für Modifikationen auf den verschiedenen Ebenen (Beschreibung zusätzlicher Inhalte, Änderungen der logischen Struktur, Anpassung an verschiedene Präsentationssysteme) offen sein.[35]
4. Der Entwurf des Markupmodells ist zunächst noch unabhängig von der Architektur des Nutzungssystems für ein dynamisches elektronisches Buch, vgl. dazu unten Kap. 4.5 und 12.5.1.

Zur Ermittlung eines konkreten Markupmodells werden folgende Arbeitsschritte vorgeschlagen:

1. top-down-Analyse: Ermittlung der logischen Struktur
2. bottom-up-Analyse: Ermittlung der unterschiedlichen Inhaltskomponenten und der erforderlichen Beschreibungskategorien
3. Definition von Verknüpfungstypen zwischen den Strukturbestandteilen
4. Festlegen der Präsentationsmerkmale für alle Elemente
5. Definition von Metainformation und Schnittstellen zu Diensten und ihre Kodierung

Die Ermittlung der logischen Grundstruktur (Hierarchieebenen) bildet den Ausgangspunkt der Modellierung, zunächst noch unabhängig von der Präsentationsform im Viewersystem. In sie werden die in Schritt zwei ermittelten Inhaltskomponenten eingeordnet, wobei sich bei der im Referenzprojekt gewählten Lösung eine Unterscheidung in

- Komponenten des Basistexts, im wesentlichen die Textgrundlage des Buchs sowie die darin enthaltenen Formeln, und in
- ergänzende Komponenten (nicht nur Multimediakomponenten, sondern auch Abbildungen, Schemata, Tabellen) ergeben hat.[36]

[35] Ein wesentliches Werkzeug hierfür ist die Definition von Namensräumen, die es erlaubt, verschiedene Markupmodell konfliktfrei zu integrieren, vgl. dazu unten Kap. 5.2.3.

[36] Diese Unterscheidung ist auch durch eingeschränkte Layoutmöglichkeiten, insb. den eingeschränkten Darstellungsbereich des Viewingsystems bedingt.

Ergänzende Komponenten können jeweils mehreren Strukturknoten des Basistexts zugeordnet sein. Zwei Beispiele aus dem Referenzprojekt sollen dies verdeutlichen:

a) Die Simulation eines Elektronenstrahloszilloskops kann unter verschiedenen didaktischen Gesichtspunkten mehreren Struktureinheiten bzw. den in ihnen enthaltenen Inhaltskomponenten zugeordnet werden:
   – Erlernen der Bedienung eines *Arbeitsgerätes*: Zuordnung zu *Einführung* bzw. *allgemeine Grundlagen.*
   – Einsatz als Teil einer Versuchssimulation (z. B. RC-Glied): Zuordnung zu den Inhaltskomponenten *Versuchsaufbau* bzw. *Versuchsdurchführung.*
b) Ähnliches gilt für *generisches Wissen*, das wie die *Tabelle physikalischer Grundkonstanten* im Referenzprojekt zwar nicht unmittelbar Teil des Basistextes ist (im gedruckten Werk als Tabellenanhang realisiert), aber einer Mehrzahl von Strukturbestandteilen zugeordnet werden kann und auch über allgemeine Erschließungs- und Navigationsmöglichkeiten auffindbar sein muss.

Die beiden ersten Schritte

- bauen einen Baum als hierarchische Repräsentation der Struktur des Dokuments auf,
- füllen die Knoten des Baums mit den verschiedenen Inhaltskomponenten und
- erweitern den Baum durch Zuordnung ergänzender Komponenten zu einem Netzwerk.

Diese Strategie orientiert sich grundsätzlich an den Annahmen, dass

- sich für elektronische Bücher eine hierarchische Gliederung finden lässt und
- für die Inhalte grundsätzlich eine lineare Abfolge von Teilsequenzen als primärer Nutzungspfad gefunden werden kann.

Wie die Erläuterung der für das Referenzprojekt gewählten Gestaltungsmetapher zeigen wird (vgl. unten Kap. 13.3.1), orientieren sich diese beide Annahmen stark am Vorbild des gedruckten Buchs. Hinsichtlich der inhaltsorientierten Auszeichnungselemente ist dabei folgendes zu beachten:

- In vielen Fällen überlappen sich Struktur und inhaltliche Funktion, wenn z. B. eine Versuchsbeschreibung gleichzeitig eine Gliederungsebene zugewiesen bekommt. Diese Doppelfunktion ist durch das Markup entsprechend zu berücksichtigen.
- Die Menge inhaltsorientierter Marken ergibt sich aus einer textfunktionalen Analyse: Dabei werden im Primärdatenbestand wiederholt auftretende Bestandteile auf Wort- und Abschnittsebene ermittelt. Im Beispiel des Referenzprojekts ergibt sich die bereits erwähnte Aufgliederung von Versuchsbeschreibungen.

Der dritte Schritt legt fest, welche Verknüpfungstypen eingeführt werden – zunächst an den Inhaltskomponenten orientiert, d. h. Verknüpfungen, die inhaltlich motiviert und nicht von den Erfordernissen des Präsentationssystems bedingt sind (wie reine Navigationsverknüpfungen, die die sequentielle Abfolge von Präsentationseinheiten ermöglichen).

Im vierten Schritt werden die Präsentationseigenschaften für die eingeführten Elemente bzw. – bei einem generischen Ausgabeformat wie HTML – die Übersetzungsregeln für die inhaltsorientierten Marken angegeben. Eine Übersetzung ist dann erforderlich, wenn das Betrachtungssystem nur für bestimmte Dokumenttypen wie HTML zu nutzen ist und andere Dokumenttypbeschreibungen auf dieses Zielformat abgebildet werden müssen. Die Festlegung der Präsentationseigenschaften gehört zwar formal zur Markupspezifikation, soweit die Präsentationsinformation durch einen deklarativen Standard erfolgt (z. B. XSL oder CSS, vgl. unten Kap. 7); ihre Ausprägung ist aber auf der Basis

eines Gestaltungsleitfadens zu ermitteln (vgl. unten Kap. 13.3.2). Sie sagt zudem noch nichts über die Gestaltung des Viewingsystems auf der Makroebene (Aufbau der Benutzerschnittstelle, Fensterorganisation) aus. Je nach den technischen Möglichkeiten des Betrachtungssystem kann eine weitere Aufteilung der in den Schritten 1 und 2 definierten logischen bzw. inhaltlichen Komponenten in kleinste Präsentationseinheiten erforderlich sein – im Referenzprojekt die einzelnen Bildschirmseiten als kleinste Einheit der Darstellung.

Der letzte Schritt betrifft die Definition der Metadatenformate und die Festlegung der Schnittstellen zu externen Diensten (siehe unten Kap. 4.4.3 und 14.2). Dabei ist nicht nur zu definieren, welche Arten von Metainformation (z. B. bibliographische Angaben, Nutzungsinformation für Lehr-/Lernkomponenten), sondern auch, wie diese Metadaten zu nutzen sind (Übersetzung in ein Darstellungsformat; Nutzung als Parameterwerte für die Anforderung von Diensten etc.). Wie bei der Einführung von inhaltsorientiertem Markup ist die Granularität der Beschreibung bzw. die Zuordnung der Metadaten zu den Inhaltskomponenten zu bestimmen: Sie ergibt sich aus der Funktion der Metadaten hinsichtlich der beschriebenen Ressourcen – z. B. kann die Zuordnung einer bibliographischen Beschreibung auf der obersten Ebene der Inhaltshierarchie erfolgen, für die das Merkmal (Autor; Entwickler etc.) noch zutreffend ist, soweit ein impliziter Schlussmechanismus vorausgesetzt werden kann, der bei für Inhaltskomponenten unterhalb dieser Ebene auf die höheren Hierachiestufen zurückgreift. Auf der Ebene der Metadaten kann eine Wiederholung von Schritt 4 erforderlich sein, um die Metainformation mit einem Präsentationsformat zu versehen.

Durch die Integration von Multimediakomponenten und Diensten und die elektronische Präsentation weicht die Methodik von aus dem SGML-Umfeld bekannten Strukturierungsmodellen ab, die sich stärker an textuellen Formaten orientieren.[37] Dies betrifft weniger die logische Struktur und die Identifikation von Ansatzpunkten für die inhaltsorientierte Auszeichnung, als vielmehr die Fragen der Hypertextverknüpfungen, der Präsentationsspezifikation und der Beschreibung der Schnittstellen zwischen Komponenten. Die Schritte 1 – 5 sind nicht als einmalige und abschließende Arbeitsschritte zu verstehen: Durch Hinzunahme neuer Dienste oder die Ermittlung zusätzlicher Information oder die Integration neuer Komponenten(typen) in ein dynamisches elektronisches Buch kann eine sukzessive Erweiterung der inhaltsorientierten Beschreibung und der Metadateninformation erforderlich sein.

Abschließend sei bemerkt, dass die Definition eines Markupmodells ähnlich der Ermittlung eines Datenmodells für eine Datenbankanwendung zwar eine zentrale Rolle spielt, aber bei der Entwicklung eines dynamischen elektronischen Buchs durch eine Reihe weiterer Schritte zu ergänzen ist – die Definition des Markupmodells besagt noch nichts über Verfahren zur tatsächlichen Einführung der Auszeichnungselemente, die in der Regel sowohl durch intellektuelle als auch automatische Verfahren erfolgt. Der konkrete Arbeitsablauf hängt stark davon ab, inwiefern schon bei der Produktion von Inhaltselementen mit Werkzeugen gearbeitet wird, die deklaratives Markup unterstützen oder ob

[37] Man muss anmerken, dass sich für die Verwendung deklarativer Markupstandards, z. B. im Verlagswesen als Vorstufe der Buchproduktion bisher anders als im Softwarebereich kein einheitliches Modellierungsverfahren herausgebildet hat; die von MALER & EL ANDALOUSSI 1996 vorgestellte Methodik bildet insofern eine Ausnahme.

Ausgangsdaten aus proprietären Formaten in eine Kodierung mit deklarativem Markup überführt werden müssen.[38]

4.3 Integration von Komponenten

Unter der Integration von Komponenten im Kontext dynamischer elektronischer Bücher ist die Einführung von Funktionselementen zu verstehen, die einfache textuelle Inhalte um Multimedia- oder allgemein Softwarekomponenten ergänzen und dadurch wesentlich zum Mehrwert des elektronischen Buchs beitragen. Dies entspricht einen Sonderaspekt der Inhaltsaufbereitung und -integration (Schritt 2 der Entwicklung des Markupmodells). Im Gegensatz zur Anreicherung von elektronischen Büchern durch atomare multimediale Datenelemente wie Filme oder Audiodateien sind vor allem solche Komponenten von Interesse, die eine komplexe Binnenstruktur aufweisen, mit der der Benutzer interagieren kann und die daher über eine eigene Benutzerschnittstelle verfügen. Dabei kann man Komponenten von Diensten durch folgende Merkmale abgrenzen:

- Anders als ein Dienst ist eine Komponente *unmittelbarer Inhaltsbestandteil* des elektronischen Buchs.
- Die Funktion einer Komponente ist auf die Inhalte des elektronischen Buchs zugeschnitten und die Komponente ist i. d. R. gezielt für ein einzelnes elektronisches Buch entwickelt worden.
- Eine Komponente bietet anders als ein Dienst keine generische Dienstleistung an, die für eine Vielzahl inhaltlich unterschiedlicher elektronischer Bücher genutzt werden kann.

Der Komponentenbegriff kann unter verschiedenen Aspekten verstanden werden:

- Als *Modellierungseinheit* für Knoteninhalte im Sinne eines Hypertextmodells (vgl. oben Kap. 2.2.1.3),
- als *softwaretechnologisches* Konzept im Sinne des *component ware*-Ansatzes (vgl. unten Kap. 14.1.1),
- als *präsentationsorientierte* Darstellungseinheit, die abgesetzt vom Basisdatenbestand eines elektronischen Buchs bestimmte Inhaltstypen in gesonderter Weise präsentiert (vgl. unten Kap. 13.3).

Alle drei Sichtweisen spielen für das Modell dynamischer elektronischer Bücher eine Rolle: Als *Modellierungseinheit* sind Komponententypen bei der Entwicklung des Markupmodells zu berücksichtigen und die für sie erforderlichen deklarativen Beschreibungselemente sind zu definieren. Softwarekomponenten als Sonderfall eines in einem dynamischen Buch enthaltenen *Inhaltstyps* sind deshalb von Bedeutung, weil sich mit ihnen der Mehrwert dynamischer elektronischer Bücher realisieren lässt und eingebettete Softwaremodule sowohl hinsichtlich ihrer Interaktivität und Integration in die Nutzungsinfrastruktur des elektronischen Buchs als auch hinsichtlich ihrer Gestaltung eine herausgehobene Rolle spielen. Schließlich ist die Betrachtung als gesonderter *Darstellungstyp* für den Aufbau der Benutzerschnittstelle wichtig: Komplexe Komponenten lassen sich nicht nahtlos in einen Buchbetrachter integrieren, sie müssen aber auf der Basis einheitli-

[38] Vgl. dazu die Hypertext-Entwicklungsmethode RMM (Kap. 2.2.1.6) und das allgemeine Entwicklungsmodell für das elektronische Publizieren, das in Kap. 2.1) eingeführt wurde, sowie die Übersicht zu XML-Werkzeugen in Kap. 5.2.7.2.

cher Gestaltungsanforderungen realisiert werden; es stellt sich gewissermaßen das Problem, eine „Benutzerschnittstelle innerhalb einer Benutzerschnittstelle“ zu realisieren.

Im Referenzprojekt wurden – unter der weitgefassten Bezeichnung „Multimediale Ergänzungen“ eine Vielzahl unterschiedlicher Inhaltstypen – von einfachen Photographien bis hin zu komplexen Simulationsprogrammen – als Komponenten gesondert gestaltet und in das elektronische Buch integriert (vgl. ausführlich unten Kap. 14.1).

4.3.1 Merkmale von Komponenten

Unabhängig von ihrer technischen Realisierung lassen sich folgende Merkmale von Komponenten unterscheiden:

- Der *inhaltliche Typ* der Komponente (im Referenzprojekt z. B. Simulation eines physikalischen Phänomens, Animation eines Versuchsablaufs, interaktive Versuchsdarstellung),
- die *Funktion* der Komponente hinsichtlich des Publikationszwecks (z. B. didaktische Funktion bei Lehr- und Lernwerken),
- die Art der *Einbettung* der Komponente in das elektronische Buch (Art der Präsentation, Erschließung der Komponente über Verknüpfungen, den hierarchischen Zugang oder die Suchfunktionen),
- die *technische Realisierung* der Komponente (Multimedia-Datenelement, z. B. eine Videosequenz, ein eingebettetes Programm, eine Simulation oder Animation),
- die *Schnittstellen* einer Komponente zu externen Diensten (z. B. Speicherung von durch die Komponente erzeugten Daten) und
- Art und Umfang der für die Nutzung der Komponente notwendigen *Zusatzinformation* (Beschreibung, Anleitung, Metadaten).

Der *state-of-the–art* elektronischer Bücher, wie er sich im Referenzprojekt widerspiegelt und wie er in zahlreichen Partnerprojekten des Projektverbunds „Multimediabuch“ beobachtet werden kann, umfasst nur einfache Schnittstellen zwischen Multimediakomponenten und elektronischen Büchern (z. B. bei der Einbettung von Java-Applets, wo eine Parameterübergabe-Schnittstelle existiert, die i. d. R. *statisch*, d. h. mit fest vorgegebenen Werten genutzt wird). Stellt man die – pragmatisch kaum hoch genug einzuschätzenden – Aspekte des Kodierungsaufwands zurück, so gilt es auszuloten, mit welchen Kodierungsstandards und Werkzeugen eine verbesserte Integration erreichbar ist. Dies gilt nicht nur für „fest“ integrierte multimediale Bestandteile einer elektronischen Publikation sondern auch für dynamisch einzubindende externe Dienste (s. u. Kap. 4.4). Denkbare Beispiele aus dem Kontext des Referenzprojekts sind:

- Übergang zwischen formaler Ableitung eines Sachzusammenhangs und dessen beispielhafter Instantiierung in einer Komponente,
- Instantiierung einer Simulation mit Werten, die statisch vorgegeben sind oder dynamisch aus dem aktuellen Nutzungskontext bezogen werden,
- Speicherung der Daten einer Messung oder
- Übergabe von Messwerten an externe Module zur Durchführung einer Berechnung; Zuordnung von Animations- oder Filmsequenzen zu Textabschnitten und Anforderung bei Bedarf durch den Benutzer.

4.3.2 Methodik der Komponentenentwicklung

Die Entwicklung von Multimediakomponenten ist arbeits- und kostenintensiv, eine vollständige Realisierung aller zu einem dynamischen Buch denkbaren und wünschenswerten Komponenten ist nur bei hohen personellen und finanziellen Investitionen möglich.[39] Es ist daher zu fragen, mit welcher Methodik sich die Komponentenentwicklung effizient gestalten lässt. Es zeigen sich zwei wesentliche Ansatzpunkte:

- Das Aufgreifen bewährter Methoden aus dem Software-Engineering, um den Entwicklungsprozess zu strukturieren, z. B. Modellierungsverfahren und -sprachen des objektorientierten Programmierens wie die *Unified Modeling Language* (UML, vgl. FOWLER & SCOTT 1998, BOOCH, RUMBAUGH & JACOBSON 1999) oder die Verwendung bewährter Entwurfsmuster (*design patterns*, vgl. GAMMA et al. 1996) und
- die Definition einer Grundkonzeption für die Komponentenentwicklung und ihre Integration in das elektronische Buch, die eine nachhaltige Nutzung und die langfristige Erweiterbarkeit sicherstellt.

Der erste Aspekt betrifft die systematische Erstellung von Entwicklungsschemata, die die Komponentenrealisierung unterstützen sollen. Im Referenzprojekt ergaben sich folgende Entwicklungsschritte:

1. Definition der didaktischen Zielsetzung für die Komponenten, darauf aufbauend Selektion geeigneter Inhalte für die Realisierung als Komponente.
2. Festlegen unterschiedlicher Komponententypen und ihrer technischen Realisierung (auf der Basis einer höheren Programmiersprache bzw. eines Multimedia-Autorensystems).
3. Entwicklung von Modellen bzw. *templates* als Programmgerüsten, die für einzelne Komponenten instantiiert und wieder verwendet werden können.
4. Entwicklung von Gestaltungsanforderungen für die Benutzerschnittstelle der Komponenten, die bei der Realisierung weitestgehend zu berücksichtigen sind.
5. Realisierung der Komponente (Implementierung).

Die Schritte 3 und 4 sind entscheidend für die Optimierung des Entwicklungsprozesses, da sie die Vorgaben für die konkrete Realisierung liefern. Im Referenzprojekt wurden diese Schritte wie folgt umgesetzt: Ausgehend von einer Einteilung der Komponententypen in

1. Simulationen physikalischer Experimente, Phänomene und Arbeitsgeräte,
2. interaktive Animationen von Versuchsabläufen bzw. physikalischer Phänomene
3. und interaktiver Versuchsaufbaudarstellungen

und ihrer technischen Realisierung als Java-Applets (1.) bzw. Macromedia Director-Anwendungen (2./3.) wurden entsprechende Programmschemata bzw. *templates* ausgearbeitet, die unabhängig von der konkreten Realisierung eine Entwicklungsvorgabe darstellen und deren Gestaltung bzw. Aufbau der Benutzerschnittstelle Teil des Gestaltungsleitfadens für das elektronische Buch ist.

---

[39] Im Referenzprojekt standen ca. 80 physikalische Experimente als Korpus für die Komponentenentwicklung zur Auswahl; davon ließen sich nur ca. 20 interaktive multimediale Komponenten tatsächlich realisieren; die Gesamtzahl der ergänzenden Komponenten liegt bei Berücksichtigung einfacher nicht multimedialer Komponenten allerdings deutlich höher (ca. 50 Ergänzungen).

4.4 Integration externer Dienste und Ressourcen

Der dritte grundlegende Aspekt dynamischer elektronischer Bücher neben dem Einsatz deklarativen Markups und der Realisierung eingebetteter Komponenten sind die für ein elektronisches Buch zu spezifizierenden Dienste. Unter einem Dienst wird eine generische Funktion zur Bereitstellung von Information verstanden, die über Netzwerke abgerufen werden und deren Inhalte bzw. Funktionen anders als Komponenten nicht spezifisch für ein einzelnes elektronisches Buch sind, sondern von einer Vielzahl unterschiedlicher Benutzer verwendet werden können.[40] Wie noch zu zeigen sein wird, schließt diese Definition eine Vielzahl unterschiedlicher Dienstetypen ein, neben

- generischen Diensten wie E-Mail oder ftp im Sinne des Dienstbegriffs der Netzwerktechnologie (vgl. SCHELLER et al. 1994) vor allem
- stärker inhaltlich determinierte Informationsdienste wie das Nachschlagen einer Definition zu einem Begriff im Text des elektronischen Buchs oder die Anbindung von Suchmaschinen und Nachweissystemen.

Aus der Perspektive des Benutzers ist nur die *Schnittstelle* des Dienstes und seine Integration in das elektronische Buch relevant, nicht aber seine *Arbeitsweise*, wie MILLARD, REICH & DAVIS 1999: 38 feststellen:

> A service is a black box of functionality, known only by name to a client, that can be invoked and its results understood, even though its workings are completely opaque.

Die Spezifikation externer Dienste für ein elektronisches Buch geht von folgenden Überlegungen aus:

- Zahlreiche Dienste, die für die Nutzung einer elektronischen Publikation (allgemein, eines (Web-)Informationssystems) herangezogen werden können, sind über Datennetze von Drittanbietern erreichbar (vgl. ENDRES et al. 1999: 137f.).
- Nicht jeder für ein dynamisches elektronisches Buch sinnvolle Dienst kann für eine einzelne Publikation oder eine Distributionsplattform wie eine digitale Bibliothek neu entwickelt und/oder integriert werden.
- Über die Integration von Diensten kann das einzelne elektronische Buch in eine Nutzungsinfrastruktur eingebettet werden, die der Benutzer anpassen kann bzw. die das elektronische Buch in einen konkreten Nutzungskontext z. B. einer Lehrveranstaltung oder eines Kurses stellt (individuelle Anpassung vs. gruppenbezogene Anpassung).
- Die Auszeichnung der Inhalte des elektronischen Buchs und seiner eingebetteten Komponenten mit inhaltsorientiertem Markup kann als Anknüpfungspunkt für die Integration von Diensten dienen.
- Die technische Realisierung der Dienste-Integration soll weitgehend unabhängig von der Realisierung des einzelnen elektronischen Buchs sein, um das Konzept der Dienste-Integration verallgemeinern zu können.

Die zu integrierenden Dienste sind von der Basisfunktionalität des elektronischen Buchs abzugrenzen: Die Bereitstellung der primären Buchinhalte, ihre Präsentation, die Naviga-

---

[40] Vgl. die allgemeine Definition für *Dienste* wie sie STEINMETZ 1999: 4 in Hinblick auf Dienste für die Bereitstellung multimedialer Inhalte gibt: „Dienste stellen dem Anwender, bzw. der Anwendung in sich zusammenhängende Funktionen zur Nutzung bereit." In SCHNEIDER 1997 finden sich zwar Begriffsbestimmungen für *Dienstleistung*, *Dienstanbieter* und *Dienstprotokoll*, eine Definition des zugrundeliegenden Konzepts *Dienst* fehlt aber, vgl. auch POPIEN 1995: 18 ff.

tions- und Erschließungsmöglichkeiten für die Buchinhalte und die Interaktion mit den im Buch enthaltenen Komponenten können zwar ebenfalls als *Informationsdienste* verstanden werden, wenn z. B. eine Buchseite als HTML- oder XML-Datei über das HTTP-Protokoll als Internetdienst abgerufen werden kann, von Dienste-Integration soll aber nur dann gesprochen werden, wenn ein Dienst nicht ursprünglicher Funktionsbestandteil des elektronischen Buchs ist. Das Dienstekonzept soll einerseits die Einbindung prinzipiell beliebiger Informationsdienste ermöglichen, ohne dies andererseits zur alleinigen Aufgabe von Autoren und Produzenten zu machen – je nach Nutzungskontext sollen für das elektronische Buch unterschiedliche Dienste zur Verfügung gestellt werden können.

Diese Vorgehensweise stellt die Interaktion mit dem elektronischen Buch und die sich daraus ergebende Lesesituation in den Mittelpunkt: Das Ziel ist es dabei, für jede Lesesituation sinnvolle, über das Buch hinausgehende Dienstleistungen unmittelbar aus der Benutzerschnittstelle des Buches heraus anbieten zu können, ohne dass der Benutzer zu einem Systemwechsel gezwungen wird: Prinzipiell stehen dem Benutzer über das World Wide Web und andere internetbasierte Kommunikationskanäle beliebige Dienstleistungen zur Verfügung; die Integration ausgewählter Dienste in das elektronische Buch optimiert den unmittelbaren Nutzungskontext.

4.4.1 Typen von Diensten

Anhand einiger konkreter Beispiele sollen die durch Dienste-Integration erweiterten Nutzungsmöglichkeiten elektronischer Bücher illustriert und typisiert werden. Die Kategorisierung der Dienstetypen erfolgt nach einer Unterscheidung der Dienstefunktionen und auf der Basis der oben eingeführten Prämissen. Zusätzlich ist als technische Voraussetzung die Realisierbarkeit der Dienste im World Wide Web bzw. mit Kommunikationsprotokollen auf Internetbasis ein Kriterium, das die Einführung von Dienten im Kontext einer allgemeinen Architektur elektronischer Bücher (vgl. unten Kap. 4.5) und unter Verzicht auf proprietäre Softwarelösungen bzw. die einzelfallorientierte Realisierung von Diensten auf der Nutzerseite (Buchbetrachter) vorsieht. Nachfolgend sollen einige wesentliche Funktionen von Diensten zur Dienstetypisierung herangezogen werden:

- Dienste für die gezielte Informationsanreicherung,
- Dienste zur Informationserschließung,
- Weiterverarbeitungsdienste,
- Dienste zur Individualisierung des elektronischen Buchs und
- Kommunikationsdienste, vor allem für die Nutzung in einem gruppenbezogenen Kontext.

4.4.1.1 Dienste für die Informationsanreicherung

Hierzu gehören Dienste, die eine *bekannte* Informationsquelle zur Erweiterung der im elektronischen Buch enthaltenen Information anbinden, z. B. das Nachschlagen von Begriffen in einem elektronisch verfügbaren (Fach-)Lexikon oder die Übersetzung von Begriffen. Prototypisch wird ein solcher Dienst als Integration der im *Projekt Deutscher Wortschatz* (vgl. QUASTHOFF & WOLFF 1999, 2000) entwickelten lexikalischen Datenbank realisiert. Die Integration eines solchen Dienstes muss man von der Einbindung fester Verknüpfungen (z. B. Sammlungen von Links ins World Wide Web) unterscheiden: Diese gehören als vom Autor zur Verfügung gestellte Information/Funktion zum Primärdatenbestand des elektronischen Buchs (Suche im *Inhalt des Buchs* durch Volltextrecher-

che, Register etc.) und sind insofern statisch, als sich die Anforderung zusätzlicher Information nicht ad hoc aus einem beliebigen Lesekontext durch den Benutzer ergibt.

4.4.1.2 Dienste zur Informationserschließung

Unter *Informationserschließung* ist die Auswahl einer geeigneten Informationsmenge aus einem potentiell sehr großen Korpus informationeller Einheiten (Dokumentenkollektion) zu verstehen, d. h. das Problem des Information Retrieval. Im Kontext elektronischer Bücher stellt sich die Frage, wie im World Wide Web vorhandene Inhalte aus dem Lesekontext heraus erschlossen werden können und welche zusätzlichen Dienste der Benutzer zur Optimierung der Informationserschließung heranziehen kann (vgl. dazu ausführlich unten Kap. 0). Weniger generisch ist die Integration der Nutzung elektronischer Bibliotheken, die relevantes Material zum Inhalt des elektronischen Buchs enthalten können. Die Informationserschließung mit Bezug auf den *Primärdatenbestand* muss man von der Informationserschließung in *externen Ressourcen* abgrenzen; nur Letzteres ist ein Dienst im hier diskutierten Sinn, wenn auch beide Problemstellungen Gemeinsamkeiten aufweisen. Als Beispiel eines solchen Dienstes wird die Integration der Funktionalität von Suchmaschinen sowie die Unterstützung des Nutzers bei der Anfrageformulierung durch Auswertung des Lesekontexts diskutiert (vgl. unten Kap. 14.2.2.5).

4.4.1.3 Dienste für die Weiterverarbeitung der Primärdaten des Buchs

Dies betrifft Dienste, mit denen es möglich ist, im elektronischen Buch enthaltene Daten weiterzuverarbeiten, die aber selbst nicht Teil der Grundfunktionalität des Buchs sind; im Fall des Referenzprojekts sind denkbare Beispiele

- die Symbolmanipulation,
- die Auswertung von Versuchsdaten und
- die Datenvisualisierung.

Für sie existiert jeweils eine Vielzahl von Softwarelösungen (Computeralgebrasysteme, Datenauswertungsprogramme, Visualisierungspakete), die aber weder spezifisch für ein elektronisches Buch entwickelt werden können noch im state-of-the-art elektronischer Bücher direkt in das Buch eingebunden sind. Inhaltsorientiertes Markup bietet eine Möglichkeit, solche Dienste unmittelbar in die Nutzungssituation des elektronischen Buchs zu integrieren.

4.4.1.4 Dienste für die Individualisierung des elektronischen Buchs

Unter Diensten für die Individualisierung des elektronischen Buchs durch den Benutzer sind alle Verfahren zu verstehen, die dem Benutzer helfen, das Buch für seine Zwecke anzupassen (Prinzip der Adaption als wesentlicher Anforderung der Software-Ergonomie, vgl. unten Kap. 13.2.1); dabei ist vor allem an die persistente Anreicherung des Buchs durch benutzergenerierte Daten (z. B. Notizen) und Zusatzinformation, die aus einem Informationsanreicherungs- oder Informationserschließungsdienst gewonnen werden, zu denken: Allgemein soll der Benutzer die Möglichkeit erhalten, seine „elektronische Kopie“ des Buchs durch Informationen zu erweitern und an seine Nutzung anzupassen. Die Realisierung erfolgt exemplarisch über einen benutzerbezogenen Speicherungsdienst, mit dessen Hilfe sich Daten speichern lassen. Der Bezug zum Ausgangsdatenbestand des dynamischen elektronischen Buchs erfolgt über das Struktur- und Inhaltsmarkup.

4.4.1.5 Dienste innerhalb eines gruppenbezogenen Nutzungskontexts

Ein letzter Aspekt der Dienstefunktionalität betrifft die Frage der Zuordnung eines elektronischen Buchs zu einer Benutzergruppe mit einem gemeinsamen Nutzungskontext, z. B. im Rahmen der Verwendung des elektronischen Buchs als Textbuch einer Lehrveranstaltung.

Dienste für die Integration des elektronischen Buchs in eine *Kommunikationsinfrastruktur* sind dabei diejenigen Verfahren, die es erlauben, das elektronische Buch in ein konkretes, auf Kommunikationsdiensten im Internet bzw. World Wide Web basierendes Nutzungsszenario einzuordnen, in dem das Buch für Kurse, *distance learning* etc. genutzt werden kann. Es ist zu beachten, dass die hierfür benötigte Information nicht *ex ante*, also durch den Verlag oder den Distributor bereitgestellt werden kann und sich Art und Anzahl der Dienste erst aus der *konkreten Nutzungssituation* ergeben. Anders als bei den Diensten zur Weiterverarbeitung von Daten steht die Kommunikation, d. h. der Austausch von Information mit anderen Akteuren im Kontext der Buchbenutzung im Vordergrund. Das elektronische Buch wird durch die Integration solcher Kommunikationsdienste Teil einer Lehr- und Lerninfrastruktur, wenn der Benutzer Aufgabenlösungen, die durch Interaktion mit dem elektronischen Buch entstanden sind (z. B. Messwertgenerierung in einer Simulation), über einen Kommunikationsdienst weiterleitet oder Fragen an einen Betreuer übermittelt.

Die Schaffung entsprechender Nutzungskontexte und die durch sie vermittelten Dienste (und Inhalte) entspricht der Erweiterung des elektronischen Buchs um Aspekte der *computer supported cooperative work* (CSCW). Sie sind ein generischer Bestandteil von Lehr- und Lernsystemen. Von ihnen unterscheidet sich das hier diskutierte Dienstekonzept, als zunächst vom elektronischen Buch und seinen Inhalten ausgegangen wird, ohne dabei spezifische CSCW-Funktionalität zu berücksichtigen; für sie können aber durch inhaltsorientierte Auszeichnungen Schnittstellen bereitgestellt werden und durch Funktionen des Buchverwaltungssystems integriert werden. Solche Konzepte lassen sich in elektronische Bücher integrieren,

- wenn ein geeignetes Modell für einen gruppenbezogenen Nutzungskontext und seine Beschreibung existiert,
- die Architektur elektronischer Bücher entsprechende Verwaltungsfunktionen bereitstellt und
- Dienste und Inhalte für den gruppenbezogenen Nutzungskontext zur Verfügung stehen und sich mit vertretbarem Aufwand in das elektronische Buch in seiner spezifischen Nutzungssituation integrieren lassen.

Das in Kap. 4.5 entwickelte Architekturmodell berücksichtigt entsprechend eine Verwaltungsebene für gruppenbezogene Dienste und Komponenten, Kap. 14.2.5 stellt exemplarisch ihre Realisierung vor.

4.4.2 Interoperabilität von Diensten

Betrachtet man die für die einzelnen Kategorien von Diensten genannten Beispiele, so wird deutlich, dass die verschiedenen Dienste untereinander interoperabel sein müssen, da bestimmte Dienstetypen nur dann sinnvoll genutzt werden können, wenn für die in ihrem Kontext entstandenen Informationen die Möglichkeit der Speicherung besteht. Erzeugt der Benutzer z. B. aus einer Versuchssimulation einen Datensatz (Nutzung einer eingebetteten Multimediakomponente des Buchs) und verarbeitet diesen Datensatz an-

schließend mit einem Statistikprogramm (externer Dienst), so muss er die Möglichkeit haben, die entstandenen Daten zur Erweiterung des Buchs (Annotation) zu verwenden und diese Ergebnisse abzuspeichern (Speicherungsdienst als Verfahren zur Individualisierung des elektronischen Buchs). Gleiches gilt für die durch gezielte Informationsanreicherung oder Informationserschließung ermittelte Information: Hat der Benutzer z. B. zu einem physikalischen Versuch zusätzliche Information (z. B. eine weitere Versuchssimulation aus dem World Wide Web; Fachliteratur zu einer spezifischen Variante des Versuchs aus einer digitalen Bibliothek), so muss er die Möglichkeit haben, die zusätzliche Information (bzw. Verweise auf sie) in das Buch zu integrieren und es so für sich zu individualisieren.

Es bestehen Wechselwirkungen nicht nur zwischen dem Primärdatenbestand und der Basisfunktionalität des elektronischen Buchs und den angebotenen Diensten, sondern auch zwischen den zusätzlichen Diensten untereinander. Die Realisierung dieses Modells setzt die deklarative Auszeichnung von Information als Mittel der Interoperabilität[41] von Information und Dienst und die Verfügbarkeit offener Kommunikationsstandards voraus. Wie solche Dienste zu modellieren und strukturieren sind, wird im folgenden Kapitel untersucht.

4.4.3 Struktur und Modellierung von Diensten

Nachdem unterschiedliche Funktionen von Diensten diskutiert wurden, ist zu klären, wie – heterogene – Dienste zu strukturieren und zu modellieren sind, damit sie dynamisch in elektronische Bücher integriert werden können. Dies geschieht anhand der bereits angedeuteten Beispiele auf einer konzeptuellen Ebene (für die Dienste*realisierung* erforderliche Technologien werden in Kap. 12.3.1 ff. untersucht, die prototypische Umsetzung im Referenzprojekt in Kap. 14.2 vorgestellt). Um einen Dienst integrieren zu können, müssen dem Buchbetrachtungssystem unterschiedliche Informationen zur Verfügung stehen, die sich auf

- den Dienst selbst,
- die inhaltliche Struktur des Buchs, in die der Dienst eingebunden werden soll,
- den Benutzer,
- die aktuelle Benutzungssituation (Lesekontext, „Zustand“ des elektronischen Buchs)

beziehen. Zunächst ist es erforderlich, allgemeine Informationen über den Dienst bereitzustellen, um ihn technisch einbinden und ggf. dem Benutzer Informationen über den Dienst liefern zu können. Zu diesen Informationen gehören

- die *Dienstfunktion*,
- die *Adressierung* des Dienstes,
- die *Schnittstelle(n)* zwischen Buch bzw. Buchbestandteil und Dienst,
- mögliche Folgedienste, an die von einem Dienst gelieferten Daten übergeben werden können,
- die *Modellierung* der Daten (Format, Attribut-Wert-Strukturen), die an den Dienst geliefert werden müssen, bzw. die der Dienst selbst generiert sowie ggf.

---

[41] Zur Frage der Realisierung interoperabler Dienste im World Wide Web vgl. VAN OSSENBRUGGEN, HARDMAN & RUTLEDGE 1999, die als Ansatzpunkte für die Interoperabilität neben generischen Diensten im World Wide Web wie die Erstellung von Informationsprofilen oder die Modularisierung von WWW-Anwendungen vor allem das Problem der Integration unterschiedlicher Anwendungen sehen.

- plattformspezifische Eigenschaften.

Stehen diese Informationen zur Verfügung, so stellt sich weiter die Frage, an welchen Stellen eines elektronischen Buchs, d. h. an welche Elemente seines Inhaltsmodells ein Dienst angebunden werden soll und wie diese Anbindung syntaktisch spezifiziert werden kann. Das Problem, Dienste so zu beschreiben, dass sie in ein elektronisches Buch dynamisch integriert werden können und miteinander interagieren können, weist konzeptuelle Ähnlichkeit mit der Problematik der Spezifikation von Agentensprachen auf. Daher werden zunächst Standards aus dem Bereich der Agentensprachen diskutiert, um im Anschluss daran ein konkretes Modell der Dienstebeschreibung vorzustellen.

4.4.3.1 Sprachen für Agentensysteme

Ein modulares Konzept, das elektronische Publikationen in eine heterogene dynamische Softwareinfrastruktur einordnet, ist mit dem Konzept von Softwareagenten eng verwandt. Z. B. ist die *Informationserschließung* nicht nur ein Beispiel für zu integrierende Dienste in elektronischen Büchern, sondern auch ein klassisches Anwendungsgebiet für Softwareagenten (vgl. BRENNER, ZARNEKOW & WITTIG 1998: 21 ff., 225 ff.), wobei man einräumen muss, dass in vielen Fällen schon einfach strukturierte Informationssammlungsalgorithmen unter den Begriff intelligenter Softwareagenten subsumiert werden[42]. Die Konzepte und Sprachen, die für Agentensysteme entwickelt wurden (*agent communication Languages* – ACL, vgl. LABROU, FININ & PENG 1999), können bei der Spezifikation der Schnittstellen und des Verhaltens elektronischer Publikationen eine wichtige Hilfestellung leisten. SINGH definiert den Agentenbegriff wie folgt:

> In the true sense of the word, an agent is a persistent computation that can perceive its environment and reason and act both alone and with other agents. The key concepts in this definition are interoperability and autonomy. [SINGH 1998: 40]

Unter einen Softwareagenten ist ein Programmsystem zu verstehen, das eigenständig Handlungen ausführt und mit einer Informationsinfrastruktur interagiert (vgl. CAGLAYAN & HARRISON 1998: 9 ff., BRENNER, ZARNEKOW & WITTIG 1998: 21 ff.). Die Interoperabilität und die Wahrnehmung der Umgebung ist durch die Kommunikationsfähigkeit des Agenten zu realisieren. Zur Spezifikation der Funktionalität von Agenten wurden verschiedene Sprachen bzw. Protokolle entwickelt: Ein in der Agentenliteratur schon seit längerem eingeführter Standardisierungsvorschlag ist die *Knowledge Query and Manipulation Language* (KQML, vgl. FININ et al. 1994, LABROU, FININ & PENG 1999). Sie geht von der theoretischen Basis der Sprechakttheorie (vgl. AUSTIN 1979, SEARLE 1986; HEYER & WOLFF 1998B) als Modellierung sprachlichen Handelns aus und entwirft ein Kommunikationsschema für Agenten. Nach diesem Ansatz beruht die Kommunikationsfähigkeit von Agenten auf einem dreischichtigen Modell, bestehend aus

- den zu kommunizierenden *Inhalten*, die in einer geeigneten Repräsentationssprache deklarativ kodiert vorliegen,
- der *Kommunikationsfunktion* (bzw. -logik), die in Analogie zu den Typen performativer Akte der Sprechakttheorie den Typ einer kommunikativen Handlung des Agenten nach einem vorgegebenen Muster klassifiziert und

[42] Vgl. dazu CHEONG 1996, SINGH 1998: 40, BRENNER, ZARNEKOW & WITTIG 1998: 225 ff. Die dort diskutierten Beispiele für (Meta-)Suchmaschinen lassen sich zwar unter den Agentenbegriff subsumieren, ein „intelligentes“ Verhalten ist aber nur in Ansätzen (z. B. bei der Individualisierung durch den Benutzer) zu erkennen.

- der *Kommunikationsebene* („Mechanik der Kommunikation“) selbst, auf der die technischen Details des Informationsaustauschs kodiert werden (Datenprotokoll etc.).

Eine KQML-Nachricht besteht aus

- einem performativen Bestandteil, der die kommunikative Intention vermittelt,
- der Spezifikation von Inhalten bzw. Platzhaltern für gewünschte Information (z. B. bei Abfragen) und
- einer Attribut-Wert-Liste mit Angaben zu weiterer Kontextinformation (z. B. der Spezifikation der aktuell gültigen Ontologie oder Repräsentationssprache).

Im folgenden Beispiel wird eine an den Kontext des Referenzprojekts angepasste KQML-Nachricht wiedergegeben, bei der eine Informationsabfrage für einen Messwertdatensatz erfolgt:[43]

```
(ask-one
     :sender BuchbenutzerXY
     :content (<Messergebnis versuch=”Fadenpendel”/>)
     :receiver (Speicherungsdienst_Messergebnisse)
     :reply-with Anfrage_5
     :Language XML
     :ontology Physikalische_Versuche
)

(tell
   :sender Speicherungsdienst_Messergebnisse
   :content (<Messergebnis versuch=”Fadenpendel”>
                  <Parameter>
                    <Daempfung>0</Daempfung>
                    <Gewicht>500 g</Gewicht>
                    …
                  </Parameter>
                  <Messwert>Dauer der Pendelschwingung</Messwert>
                  <Datensatz>
                    <Wert Nr=”1”>2,23</Wert>
                    <Wert Nr=”2”>2,12</Wert>
                    <Wert Nr=”3”>2,59</Wert>
                  </Datensatz>
                  </Messergebnis>)
     :receiver (BuchbenutzerXY)
     :in-reply-to: Anfrage_5
     :Language XML
     :ontology Physikalische_Versuche
)
```

Codebeispiel 1: Beispiel einer KQML-Kommunikation

Die Performativa (ask-one, tell etc.) in KQML sind sowohl hinsichtlich ihrer Standardisierung (*core performatives – standard performatives – extended performatives*) als auch hinsichtlich ihrer kommunikativen Funktion klassifiziert. Als Basisbestand werden folgende Subklassen performativer Akte eingeführt:

[43] Das Beispiel nimmt an, dass XML als Kodierungsformat für den Informationstransport verwendet wird und geeignete DTDs für die Kodierung der Daten bzw. der zugrunde gelegten Ontologie vorhanden sind – KQML ist so modelliert, dass beliebige Beschreibungssprachen für Inhalte eingesetzt werden können, vgl. LABROU, FININ & PENG 1999: 47.

- Einfache Anfrageperformativa (z. B. ask-if),
- Anfrageperformativa für multiple Antwortquellen (z. B. stream-in),
- Antwortperformativa (z. B. reply),
- Generische Informationsperformativa (z. B. tell, cancel),
- Generierungsperformativa (z. B. ready, next),
- Performativa zur Beschreibung von Fähigkeiten (z. B. import, export, monitor) und
- Netzwerkperformativa (z. B. forward, broadcast).

Ein jüngeres Protokoll, die ACL der *Foundation for Intelligent Physical Agents* (FIPA, FIPA-ACL), baut ebenfalls auf der Sprechakttheorie auf und kodiert eine sowohl syntaktisch als auch inhaltlich KQML eng verwandte Agentensprache. Abbildung 15 zeigt den prinzipiellen Aufbau von Nachrichten in FIPA-ACL (FIPA 1998: Kap. 6.3.1, S. 10 f.):[44]

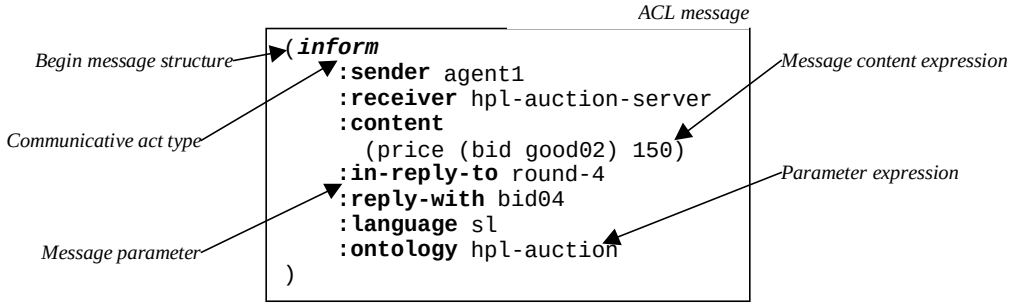

Abbildung 15: Nachrichtenstruktur nach dem FIPA-ACL-Modell

Die Anwendbarkeit von Agentenspezifikationssprachen für elektronische Publikationen ergibt sich

- aus ähnlichen Anwendungsproblemen, insbesondere der Möglichkeit der Anbindung von Diensten an Inhalte des elektronischen Buchs und
- aus dem konkreten Bedarf nach einheitlicher Spezifikation von Schnittstellen zwischen Softwarekomponenten (die u. a. nicht vollständig ex ante bekannt sind, sondern durch dynamische Zuschaltung interoperabel sein sollen).

Im Unterschied zu einem Agentensystem handelt es sich aber bei den Diensten, die für ein elektronischen Buch spezifiziert werden sollen, nicht um autonom miteinander interagierende Entitäten, sondern der Einsatz eines Dienstes setzt voraus, dass seine Integration gezielt durch Autor, Verlag/Distributor bzw. Benutzer erfolgt und auf diesem Weg die erforderliche Information über die Funktionsweise des Dienstes kodiert werden kann. Auf diesem Weg lässt sich pragmatisch die Problematik der Definition universeller Kommunikationsprimitive – ein Kernanliegen der Agentenkommunikationssprachen – umgehen. Die ACL sind mehr konzeptueller Rahmen oder Vorgabe denn konkret nutzbares Umsetzungswerkzeug. Die Orientierung an der technologisch konkret nutzbaren Infrastruktur schränkt den Gestaltungsspielraum ein.

4.4.3.2 Dienstekodierung

Die konkrete Kodierung der dienstebezogenen Information erfolgt im Rahmen dieser Arbeit auf der Basis deklarativen Markups, d. h. für jeden Dienst, der in ein elektronisches Buch integriert werden soll, wird ein Beschreibungsschema auf der Basis einer XML-DTD instantiiert, das im Rahmen der Dienstearchitektur durch das Buchverwal-

[44] Annex C in FIPA 1998 diskutiert den Zusammenhang zwischen KQML und der FIPA-ACL.

tungssystem genutzt wird, um den Dienst an die Inhalte des Buch anzukoppeln und dem Benutzer zur Verfügung zu stellen. Diese textuelle Repräsentation der Dienstespezifikation hat den Vorteil, dass sie sich mit einfachen Editoren von Autoren bzw. Benutzern erstellen und anpassen lässt; für ihre Nutzung ist allerdings die Überführung in Diensteobjekte erforderlich.[45]

Die Kodierung greift die eingangs genannten Eigenschaften von Diensten auf und dient als Anknüpfungspunkt der Diensteaktivierung und -ausführung. Die nachfolgenden Beispiele zeigen vereinfacht die Beschreibung zweier Dienste für ein elektronisches Buch. Die Dienstebeschreibung hält Name und Funktion eines Dienstes fest, gibt dessen Adresse an, spezifiziert, an welches Auszeichnungselement des Buchs er angebunden werden soll und welche Art von Daten übermittelt werden. Hinzu kommt optionale Information, z. B. zur Autorisierung des Benutzers und zum Abrechnungsmodus.[46]

Anders als bei einer Agentenkommunikationssprache kann die Dienstekodierung nicht von den zugrundeliegenden Netzwerktechnologien abstrahieren, d. h. um einen Dienst integrieren zu können, sind genaue Informationen über seine Adressierung, das zu verwendende Protokoll und die zum Datenaustausch benötigten Formate erforderlich.

```
<DIENSTELISTE>
  <DIENST>
    <NAME>Encyclopedia Britannica</NAME>
    <FUNKTION>Lexikon (Nachschlagen von Begriffen)</FUNKTION>
    <BEZUG>Fachbegriff</BEZUG>
    <ADRESSE>http://www.britannica.com/bcom/search/results/1,5843,,00.html</ADRESSE>
    <TYP>WWW</TYP>
    <PROTOKOLL>HTTP</PROTOKOLL>
    <INTERFACE typ=”HTML” adresse=”/HTML-Schemata/WWWDienst.html”/>
    <DIENSTPARAMETER name="p_query0" typ="string"/>
    <FOLGEDIENST>Uebersetzung</FOLGEDIENST>
    <FOLGEDIENST>Speicherung</FOLGEDIENST>
  </DIENST>
  <DIENST>
    <NAME>Altavista Babelfish</NAME>
    <FUNKTION>&Uuml;bersetzung Deutsch-Englisch</FUNKTION>
    <BEZUG>Fachbegriff</BEZUG>
    <ADRESSE>http://babelfish.altavista.com/cgi-bin/translate</ADRESSE>
    <TYP>WWW</TYP>
    <PROTOKOLL>HTTP</PROTOKOLL>
    <INTERFACE typ=”HTML” adresse=”/HTML-Schemata/WWWDienst.html”/>
    <DIENSTPARAMETER name="urltext" typ="string"/>
    <DIENSTPARAMETER name="doit" typ="string" wert="done"/>
    <DIENSTPARAMETER name="lp" typ="string" wert="en_de"/>
    <FOLGEDIENST>Speicherung</FOLGEDIENST>
  </DIENST>
  <!-- ... -->
</DIENSTELISTE>
```

[45] Vgl. MERZ & LAMERSDORF 1998: 9 ff., die für ihr allgemeines Dienstemodell die *Interface Definition Language* (IDL) von CORBA verwenden; HODES & KATZ 1999 stellen eine XML-basierte *Interface Specification Language* (ISL) vor, die dem hier verfolgten Ansatz sehr ähnlich ist, da mit ihr Dienste nicht nur ebenfalls durch XML spezifiziert, sondern auch in Dokumente eingebunden werden können.

[46] Das Kodierungsschema verwendet als Syntax XML, vgl. dazu unten Kap. 5.2; die vollständige Dokumententypdefinition (DTD) der Dienstebeschreibung ist in Anhang 17.1 wiedergegeben.

Codebeispiel 2: Beispiele für die Dienstekodierung

Problematisch ist dabei die *Benennung* der Dienstefunktion (z. B. *Speicherung, Informationserschließung, Darstellung* etc.) über die ein Dienst einem Inhaltselement des Buchs zugeordnet werden kann (s. u.). Für sie wäre ein verbindliches und standardisiertes Vokabular wünschenswert.

4.4.4 Koordination und Zuordnung von Diensten

Stehen Informationen über Dienste im oben skizzierten Format zur Verfügung, so kann das Buchverwaltungssystem sie dem Benutzer verfügbar machen. Dazu sind allerdings Informationen darüber erforderlich, an welcher Stelle eines dynamischen Buchs ein bestimmter Dienst zur Verfügung stehen soll (Zuordnung des Dienstes). Ein Diensteverwaltungsalgorithmus muss die Dienste ermitteln und in das Buch einbinden. Die Zuordnung von *Diensten* zu *Inhalten* setzt zweierlei voraus:

- die inhaltsorientierte Kodierung der Buchinhalte und
- ein einheitliches Beschreibungsformat für Dienste (s. o.).

Die Koordination besteht aus der Zuordnung von Diensten und Inhalten in einer *Dienstekoordinationsmatrix* sowie einem Algorithmus, der auf der Basis generischer, benutzer- oder gruppenkontextbezogener Dienstespezifikationen (s. o.) die Koordinationsmatrix aufbaut bzw. ergänzt. Die Dienstekoordinationsmatrix hält fest, welcher Dienst an welches Auszeichnungselement des Buchs angebunden werden soll und wird vom Buchverwaltungssystem zur dynamischen Anbindung der Dienste verwendet. Der Ansatz ist mit dem MIME-Standard (*Multipurpose Internet Mail Extensions*, vgl. BORENSTEIN & FREED 1993 [RFC 1521]) vergleichbar, hat aber einen anderen Ansatzpunkt: Nicht zu den Dokumenttypen (z. B. text/xml), sondern zu den *Markupelementen* innerhalb von Dokumenten werden Dienste zugeordnet. Drei Beispiele sollen dies veranschaulichen:

1. Um es dem Benutzer zu ermöglichen, zu Fachbegriffen des dynamischen elektronischen Buchs zusätzliche Information zu ermitteln, können Informationsdienste des World Wide Web eingebunden werden. Da ein solcher Dienst für alle Benutzer nützlich sein kann, erfolgt seine Anbindung durch Autoren oder Distributoren in einer generischen Diensteliste.
2. Zur Instantiierung einer mathematischen Gleichung ist ein Computeralgebrasystem erforderlich, das nicht Teil eines elektronischen Buchs ist. Dem Autor bzw. dem Distributor (Verlag; digitale Bibliothek) ist grundsätzlich nicht bekannt, ob bzw. welches Computeralgebrasystem dem Benutzer zur Verfügung steht. Der Benutzer hat aber durch Angabe einer Dienstespezifikation, die vom Buchverwaltungssystem in die Diensteliste aufgenommen wird, die Möglichkeit, entsprechende Informationen anzugeben.
3. Dasselbe gilt für die Eingliederung eines elektronischen Buchs in eine Kommunikationsinfrastruktur: A priori kann die Dienstekoordination nicht vorsehen, welcher Speicherungsdienst (Protokoll, Adresse) z. B. für eine Aufgabenlösung in einem konkreten Nutzungskontext verwendet wird. Eine Ausnahme stellt lediglich der Fall dar, dass eine direkte Kommunikation des Lesers mit dem Autor oder dem Distributor erfolgen soll; die dafür erforderliche Information zur Diensteanbindung könnte bereits bei der Produktion des elektronischen Buchs erfolgen.

Aus diesen Überlegungen folgt, dass die Dienstekoordinationsmatrix eine Liste einzelner Diensteanbindungen ist, deren Einträge jeweils ein Strukturelement des elektronischen Buchs mit einem Dienst verknüpfen. Beim Aufbau dieser Liste bietet es sich an, auf deklarative Kodierungsstandards zurückzugreifen. Das folgende Beispiel eine solche Matrix. Dabei steht jedes bezug-Attribut für ein Auszeichnungselement des dynamischen elektronischen Buchs, jedes dienst-Attribut für einen Dienst (eindeutiger NAME des Diensts, s. o.), den das Buchverwaltungssystem aus einer generischen, individuellen oder gruppenkontextbezogenen Dienstespezifikation ermittelt hat.

```
<DIENSTEKOORDINATION>
  <DIENSTEANBINDUNG bezug="PHYSIKPRAKTIKUM"   dienst="Annotation"/>
  <DIENSTEANBINDUNG bezug="FACHBEGRIFF"       dienst="Encyclopedia Britannica"/>
  <DIENSTEANBINDUNG bezug="GLEICHUNG"         dienst="Symbolmanipulation"/>
  <DIENSTEANBINDUNG bezug="ABSATZ"            dienst="Uebersetzung"/>
</DIENSTEKOORDINATION>
```

Codebeispiel 3: Beispiel einer Dienstekoordinationsmatrix

Der Aufbau dieser Matrix ist Aufgabe des Buchverwaltungssystems. Es setzt voraus, dass entsprechende Dienstespezifikationen vorliegen. Für den individuellen Benutzer oder einen Gruppenkontext muss wenigstens ein Speicherungsort bekannt sein, von dem die Dienstebeschreibungen geladen werden können.

Die Zuordnung eines Dienstes zu einem Auszeichnungselement macht auch den Unterschied zu Komponenten deutlich: Eine Komponente ist einem konkreten Inhaltsbestandteil eines Buchs zugeordnet, z. B. die Animation der Funktionsweise eines Lasers zur inhaltsorientierten Marke <GRUNDLAGEN> bei der Einführung der Prinzipien eines Lasers im Kapitel *Optik*. Ein Dienst ist dagegen generell *allen* Marken eines Typs zugeordnet. Die Zuordnung der Dienste erfolgt also auf *Klassenebene*, nicht auf *Objektebene*. Die tatsächliche Zuordnung der Dienste bei der Nutzung des elektronischen Buchs übernimmt die *Diensteverwaltung* als integraler Bestandteil der Architektur eines dynamischen elektronischen Buchs. Sie hat die Aufgabe,

1. Verfügbare Dienste zu ermitteln,
2. die Dienstekoordinationsmatrix zu verwalten und
3. sie dem Benutzer zur Verfügung zu stellen (Dienstanforderung).

Die ersten beiden Schritte, d. h. die Diensteermittlung, laufen nach dem folgenden Schema ab:

1. Start der Interaktion des Benutzers mit dem elektronischen Buch (*login*)
2. Analysieren von Informationen über die Nutzungssituation (z. B. benutzerbezogene Information, optional)
3. Ermitteln verfügbarer Dienste
   a) aus der generische Diensteliste für das elektronische Buch
   b) aus der Diensteliste des individuellen Diensteliste des Benutzers (optional)
   c) aus einer Diensteliste, die für einen spezifischen Nutzungskontext definiert ist (optional)
4. Aufbau der Dienstekoordinationsmatrix als Liste verfügbarer Dienste

Nach Abschluss dieser Schritte steht dem Buchbetrachtungssystem das erforderliche Wissen über Dienste zur Verfügung. Damit ist allerdings noch nichts über die konkrete Diensteanforderung durch den Benutzer und die Präsentation der Dienste im Kontext der Nutzung des elektronischen Buchs gesagt.

4.4.5 Anforderung von Diensten

Dienste können nicht nur spezifisch an einzelne Inhaltselemente (d. h. einen Textbestandteil des Buchs, einen bestimmten Medientyp oder eine Komponente) gebunden, sondern als generelle Dienste dem Buch global zugeordnet sein. Es stellt sich die Frage, aus welchen Situationen heraus ein Dienst aktiviert wird. Die Anforderung eines Dienstes aus dem Kontext der Nutzung eines elektronischen Buchs heraus kann auf folgende Weise erfolgen:

- Generische Diensteanforderung ohne weitere inhaltliche oder funktionale Zuordnung,
- Dienstezuordnung zu einer Grundfunktion des elektronischen Buchs (insbesondere Informationserschließung),
- Diensteanforderung aus der Interaktion mit konkreten Inhalten des Buchs und
- Diensteanforderung aus dem Navigationsverhalten.

4.4.5.1 Unspezifische Dienste-Integration

Die allgemeinste Form der Anforderung von Diensten ist die Aktivierung eines Dienstes, der lediglich als allgemein verfügbarer Dienst bekannt ist, aber weder einer bestimmten Inhaltskomponente noch einer Basisfunktion des elektronischen Buchs zugeordnet ist. In diesem Fall erfolgt die Dienstenutzung ohne Integration in das Buch im engeren Sinn, d. h. der Benutzer wählt einen verfügbaren Dienst aus einer Liste aus, die als allgemeiner Funktionsbestandteil des elektronischen Buchs bereitstehende Dienste auflistet und dem Benutzer Informationen über Name und Funktion des Dienstes vermittelt. Dies setzt voraus, dass eine Diensteliste vom Dienstemanager aufgebaut wurde und alle nicht in ein vorhandenes Funktionselement integrierten oder an eine Inhaltskomponente angebundenen Dienste enthält und erfordert weiter auf der Seite des Buchviewers (Client) ein eigenes Funktionselement für unspezifische Dienste. Dienste, die zusätzlich an bestimmte Inhaltselemente gebunden sind, können in dieser Liste enthalten sein, wenn ihre Aktivierung ohne genaue Zuordnung zu einem Inhaltskontext sinnvoll ist.[47]

Diese Form der Präsentation von Diensten hat den Vorteil, dass sie konzeptuell von der Softwareinfrastruktur, in der ein Benutzer mit dem elektronischen Buch interagiert, abstrahiert, d. h., die Dienste werden aus dem Kontext des Buchs angeboten, unabhängig davon, ob der Benutzer sie (z. B. lokal vorhandene Softwarepakete) ohne Interaktion mit dem Buchbetrachtungssystem aktivieren könnte. Sie bietet außerdem den Vorteil, dass bei Abwicklung einer Diensteanforderung diese über das Buchverwaltungssystem vermittelt abläuft und die Ergebnisse der Interaktion mit dem Dienst – z. B. der Inhalt einer Seite im World Wide Web, die der Benutzer über einen Dienst abgerufen hat – für andere Dienste zur Verfügung stehen (z. B. zur Speicherung als Annotation).

4.4.5.2 Dienste als Teil allgemeiner Buchfunktionen

Einige Funktionen elektronischer Bücher können den Zugriff auf externe Dienste sinnvoll erscheinen lassen und sollten von vornherein in den Aufbau der Benutzerschnittstelle des elektronischen Buchs einbezogen werden. Es handelt sich vor allem um die Funktionen

- Informationserschließung und
- Informationsspeicherung.

---

[47] Ein Beispiel für einen solchen Dienst ist die Übersetzung für einen ausgewählten Text, die als generischer Dienst nicht an eine einzelne Auszeichnungsmarke gebunden sein muss.

Beide Funktionen sind ein wichtiger Bestandteil elektronischer Bücher, für den entsprechende Elemente in der Benutzerschnittstelle vorzusehen sind (Retrievalschnittstelle bzw. Annotations- und Speicherungsschnittstelle). Die Aktivierung solcher Dienste setzt voraus, dass für die Nutzung erforderliche Information in einer Diensteliste bereitgestellt wurde und den entsprechenden Bestandteilen der Benutzerschnittstelle nach dem Start des Buchbetrachtungssystems zugeordnet wurden. Im Fall des Speicherungsdienstes sind dazu benutzerbezogene Daten erforderlich, da ein Speicherungsdienst als Werkzeug zur Individualisierung von Buchinhalten nur benutzerbezogen sinnvoll ist. Auch bei der Informationserschließung im Sinne der Auswahl externer Ressourcen sind Daten über den konkret zu verwendenden Dienst (z. B. eine WWW-Suchmaschine oder ein Metasucher) notwendig. Die Doppelfunktion der Informationserschließung als Aufbereitung und Erschließung der Primärdaten sowie als Schnittstelle, um externe, inhaltlich zum Buchinhalt passende Ressourcen zu ermitteln, wird in Kap. 14.2.2.5.1 näher untersucht.

Die Aktivierung erfolgt durch den Benutzer durch Auswahl einer allgemeinen Funktionskomponente des elektronischen Buchs.

4.4.5.3 Kontextspezifische Aktivierung von Diensten

Der dritte Typus der Aktivierung von Diensten stellt die stärkste inhaltliche Integrationsebene dar: Der Dienst ist einem Inhaltselement des Buchs zugeordnet – in diesem Fall wird er durch den Benutzer in einer konkreten Lesesituation aktiviert, indem er z. B. auf einen hervorgehobenen Fachbegriff klickt und so eine Diensteliste für diesen Inhaltstyp erhält (in Form einer Verknüpfungsliste), über die er gezielt eine Definition des Begriffs anfordern, verwandte Begriffe ermitteln oder den Begriff übersetzen lassen kann.

Gleiches gilt für eingebettete Komponenten, denen Dienste zugeordnet sind. Bei der Generierung einer die Komponente enthaltenden Buchseite muss über eine geeignete Schnittstelle (im Referenzprojekt die Parametertags der Applet-Marke) die Information über den verfügbaren Dienst integriert und bei Start der Komponente ausgewertet werden. Auf diese Weise wird dynamisch die Benutzerschnittstelle der eingebetteten Komponente modifiziert, z. B. durch Erzeugung einer Schaltfläche, die einen zugeordneten Dienst aufruft. Als Beispiele können die bereits erwähnten Speicherungs- oder Visualisierungsdienste für Messergebnisse von Versuchssimulationen dienen.

Die Zuordnung der Dienste zu einzelnen Inhaltselementen setzt voraus, dass ein entsprechender Eintrag in der Dienstekoordinationsmatrix existiert und bei der Seitengenerierung die Diensteinformation für die einzelnen Elemente kodiert wird. Technisch bedeutet dies die Anpassung der Transformationsregeln für die Buchinhalte auf der Basis der verfügbaren Dienste: Bei der Aufbereitung einer Buchseite muss der Transformationsprozessor für alle Inhaltselemente prüfen, welche Dienste zur Verfügung stehen und ggf. die notwendige Information (Hypertextverknüpfungen, Parameterdaten für eingebettete Applets etc.) dynamisch kodieren. Die tatsächlich generierten Inhalte einer Präsentationseinheit hängen von der in einer bestimmten Nutzungssituation verfügbaren Dienstemenge ab.

4.4.5.4 Aktivierung von Diensten aus dem Navigationsverhalten

Der letzte Typus der Diensteanforderung betrifft die Aktivierung von Diensten aus dem Navigationsverhalten. Darunter ist zu verstehen, dass die Aktivierung eines Dienstes auch durch Auslösen einer generischen Navigationshandlung durch den Benutzer erforderlich sein kann, z. B. Blättern, d. h. Anfordern der jeweils nächsten Präsentationseinheit oder

gezieltes Ansteuern einer Buchseite über die Funktionen der Informationserschließung (Suche, Inhaltsverzeichnis).

Hier besteht eine Analogie zur Integration von Komponenten in das elektronische Buch, wie sie im Referenzprojekt realisiert wurde: Dort werden die jeweils bezüglich einer *Präsentation*seinheit verfügbaren Komponenten jeweils bei Anforderung einer Buchseite ermittelt und dem Benutzer als Liste typisierter Symbole angeboten. Eine ähnliche Funktionsweise ist für die Präsentation von Diensten sinnvoll. Ein Beispiel für diese Form der Diensteintegration ist die Zuordnung von Annotationen, die über einen Speicherungsdienst persistent verwaltet werden, zu einzelnen *Präsentation*seinheiten des elektronischen Buchs: Bei Aktivierung einer Präsentationseinheit prüft das Buchverwaltungssystem, ob mit der Darstellung ein bestimmter Dienst verbunden ist, ermittelt die aktuellen Nutzerdaten und lädt ggf. vorhandene Annotationen (z. B. Notizen, Messergebnisse, Verweise auf externe Ressourcen, abgespeicherte Ergebnisse der Interaktion mit anderen Diensten).

4.4.6 Präsentation und Kapselung von Diensten

Die bisherige Modellierung der Dienste-Integration hat dargelegt, wie Dienste an Inhalts- und Strukturbestandteile elektronischer Bücher angebunden werden können und aus welchen Nutzungssituationen sie zu aktivieren sind. Bisher blieb aber die Frage offen, wie die verschiedenen Dienste präsentiert werden, d. h. wie die Benutzerschnittstellen für die verschiedenen Dienste selbst zu realisieren sind.

Im Unterschied zur Frage der Gestaltung der Dienste*aktivierung* in der Benutzerschnittstelle, die mit den für ein elektronisches Buch zur Verfügung stehenden Mitteln vergleichsweise einfach zu realisieren ist (Hyperlinks, dynamische Generierung von Interfaceelementen einer eingebetteten Komponente, Integration in allgemeine Funktionselemente des elektronischen Buchs), ist diese Frage nicht *generell* zu beantworten, da die eingebundenen Dienste über heterogene Benutzerschnittstellen verfügen können und über unterschiedliche Kommunikationsprotokolle mit ihnen interagiert wird.

Der Idealfall wäre eine Situation, in der alle benötigten Dienste mit denselben Technologien wie das elektronische Buch realisiert sind (u. a. generische Interoperabilitäts-Schnittstellen, Java-basierte Middleware, einheitliche Kodierung von Information mit deklarativen Auszeichnungssprachen, Integration in das World Wide Web). Eine solche Situation lässt sich aber nur für diejenigen Dienste erreichen, die spezifisch für ein elektronisches Buch entwickelt wurden; genau dies läuft aber dem Grundkonzept der Integration externer Dienste zuwider, das auch Dienste verfügbar machen soll, die in keinem unmittelbaren Entstehungszusammenhang zu einem einzelnen elektronischen Buch als diensteintegrierendes System stehen. Insofern wird man unterschiedliche Formen der Präsentation von Diensten unterscheiden müssen. Nach abnehmendem Grad der Integration in die Benutzerschnittstelle des elektronischen Buchs lassen sich folgende Alternativen unterscheiden:

1. Vollständige Integration eines Dienstes durch eine Benutzerschnittstelle, die Teil des Buchbetrachtungssystems ist,
2. Kapselung der Benutzerschnittstelle des Dienstes durch das Buchbetrachtungssystem und
3. Integration des Dienstes ohne Anpassung seiner Benutzerschnittstelle.

Die Realisierung dieser drei Varianten hängt von

- technologischen Aspekten (inwiefern kann ein Dienst programmtechnisch angepasst und in das Buch integriert werden ?) und
- von der Kosten-Nutzen-Abschätzung für den Integrationsaufwand ab, die klären muss, wie wichtig ein Dienst bzw. seine nahtlose Integration seiner Benutzerschnittstelle in das Buch im Verhältnis zum dadurch gewonnenen Mehrwert ist.

Die drei Typen der Präsentation von Diensten sollen nachfolgend näher untersucht und in das Dienstemodell eingeordnet werden.

4.4.6.1 Vollständig integrierte Präsentation von Diensten

Eine vollständige Integration der Präsentation von Diensten in das elektronisch Buch bedeutet, dass nicht nur die Aktivierung des Dienstes, sondern auch die Benutzerschnittstelle des Dienstes selbst vollständig Teil des Buchbetrachtungssystems ist, für den Benutzer daher während der Interaktion keine klare Trennung zwischen Buchbetrachtungssystem und genutztem Dienst besteht. Mit vertretbarem Aufwand ist eine solche Dienstereализierung nur für die diejenigen Dienste sinnvoll und möglich, die

- besonders häufig verwendet werden,
- Teil einer allgemeinen Funktion der Interaktion mit dem elektronischen Buch sind und
- ggf. zusätzlich eine Rolle als sekundärer Dienst übernehmen, d. h. die von einer Komponente oder einem anderen Dienst generierten Daten weiterverarbeiten.

Unter diesen einschränkenden Randbedingungen lassen sich Speicherungsdienst und – mit Einschränkungen – Dienste der Informationserschließung in diese Kategorie einordnen: Ein Speicherungsdienst kann für eine Vielzahl von Nutzungssituationen sinnvoll eingesetzt werden. Zumindest für die freie Textergänzung durch den Benutzer ist ohnehin eine Benutzerschnittstelle als Teil der Basisfunktionalität bereitzustellen. Er hat auch die Rolle eines *weiterverarbeitenden Dienstes*, wenn Messergebnisse gespeichert werden sollen. Aus dem schon mehrfach genannten Grund der konzeptuellen Trennung zwischen Buchinhalten und dem Buchbetrachtungssystem einerseits und den integrierten Diensten andererseits bleibt aber die Modellierung eines Speicherungsdienstes als externem Dienst bestehen, da für ihn zwar eine Benutzerschnittstelle als leere Hülle vorgesehen ist, seine konkrete Realisierung aber erst durch die Spezifikation des Speicherungsverfahrens durch den Benutzer erfolgen kann.

In geringerem Ausmaß ist die vollständige Realisierung der Integration einer Benutzerschnittstelle eines Dienstes für die Erschließung denkbar. Für das Referenzprojekt wird dies am Beispiel der lexikalischen Ressourcen des Projekts *Deutscher Wortschatz* gezeigt: Dessen Ressourcen und Softwarekomponenten können für die Integration in das elektronische Buch angepaßt werden und lassen sich daher nahtlos in das dynamische elektronische Buch bzw. dessen Informationserschließungskomponente einfügen.

4.4.6.2 Kapselung von Diensten

Es stellt sich die Frage, ob zwischen der *vollständigen* Integration der Benutzerschnittstelle und der Aktivierung eines externen Dienstes eine *mittlere Ebene* der Diensteintegration zu finden ist, bei der wenigstens eine partielle Integration der Benutzerschnittstelle und mittelbar der Dienstenutzung selbst in das elektronische Buch möglich ist. Diese Frage ist deshalb von Bedeutung, weil bei einer reinen Aktivierung eines externen Dienstes – vergleichbar der Zuordnung von Medientypen und externen Anwendungen in

einem Webbrowser oder der Zuordnung von Dateitypen zu Softwarekomponenten in einem Betriebssystem – das Buchbetrachtungssystem nach Start des externen Dienstes keinerlei Kontrolle über die Nutzung des Dienstes mehr hat, daher keine Folgedienste gezielt zur Verfügung gestellt werden können.

Unter einer *Kapselung* der Benutzerschnittstelle eines externen Dienstes ist in diesem Sinn eine eingebettete Präsentation des Dienstes durch das Buchbetrachtungssystem zu verstehen, bei der dem Benutzer zwar die nicht modifizierte Benutzerschnittstelle des Dienstes zur Verfügung steht, sie aber in eine vom Buchbetrachtungssystem generierte Umgebung eingebettet bleibt. Die technische Voraussetzung ist, dass der externe Dienst über Kommunikationsprotokolle abgewickelt wird, die zum Buchbetrachtungssystem kompatibel sind. Konkret bedeutet die Realisierung dieses Konzepts für WWW-basierte Informationsdienste, deren Schnittstelle auf der Seite des Clients eingebettet werden kann (z. B. durch HTML-Frames).[48] Eine solche Form der Kapselung findet sich in WWW-basierten Informationssystemen bei der Realisierung von Metasuchmaschinen, die die Suchergebnisse verschiedener Suchmaschinen nicht zu einer einzigen Ergebnismenge unifizieren, sondern jeweils die Benutzerschnittstelle in mehreren Rahmen einer HTML-Seite parallel präsentieren. Der Anwendungsbereich Informationserschließung eignet sich, um das Konzept der Kapselung von Benutzerschnittstellen im Kontext dynamischer elektronischer Bücher zu illustrieren – die Kapselung und Nutzung eines Informationserschließungsdienstes erfolgt nach folgender Vorgehensweise:

1. Im Buchbetrachtungssystem ist ein Präsentationsformat für die integrierte Darstellung der Benutzerschnittstelle eines externen Dienstes spezifiziert (*style sheet* zur Transformation auf der Seite des Buchverwaltungssystem, präsentationsorientierte Routinen zur Fenstergenerierung auf der Clientseite, vorgefertigte HTML-Seitenschemata).
2. Der Diensteverwalter des Buchbetrachtungssystems enthält in der Liste verfügbarer Dienste einen Dienst (hier: eine Suchmaschine), für den dieses Präsentationsverfahren angewandt werden kann.
3. Der Benutzer aktiviert den Dienst, z. B. als allgemeine Grundfunktion des elektronischen Buchs (Informationserschließung) oder als Dienst, der einem konkreten Inhaltselement zugeordnet ist.
4. Das Buchbetrachtungssystem generiert die gekapselte Benutzerschnittstelle des Dienstes.
5. Der Benutzer interagiert mit dem Dienst.
6. Dabei generierte Daten können über „Nachfolgedienste“, die im gekapselten Diensteinterface vom Buchbetrachtungssystem angeboten werden, weiterverarbeitet werden (vor allem Speicherung).

Der letzte Schritt ist entscheidend: Ohne die Möglichkeit der Weiternutzung der Daten des Dienstes ist der Aufwand für seine Kapselung in der Benutzerschnittstelle des elektronischen Buch kaum zu rechtfertigen; damit die Daten genutzt werden können, ist neben der *Gestaltung* der gekapselten Benutzerschnittstelle eine weitergehende Integration des Kommunikationsprozesses zwischen Benutzer und externem Dienst erforderlich. Im Falle der Kommunikation mit einem Dienst im World Wide Web über HTTP als Protokoll bietet sich eine durch das Buchsystem vermittelte Aktivierung des Dienstes an, bei der

[48] Ein ähnlicher Ansatz findet sich etwa bei Informationsagenten im World Wide Web, die sich als Proxy zwischen den Benutzer (Webbrowser) und das World Wide Web als Informationsquelle schalten, die Interaktion des Benutzers auswerten und ggf. Vorschläge für die Navigation im World Wide Web errechnen, vgl. JOACHIMS & MLADENIC 1998 und unten Kap. 14.2.

die tatsächliche Datenanforderung nicht durch den Client (d. h. einen als Benutzerschnittstelle eines elektronischen Buchs konfigurierten Webbrowser), sondern durch das Buchverwaltungssystem erfolgt: In diesem Fall hat das Buchverwaltungssystem vollen Zugriff auf die übermittelten Daten und kann diese – einschließlich ihrer Binnenstruktur – dem Benutzer zur Speicherung über einen Speicherungsdienst anbieten.

Prinzipiell ist diese Vorgehensweise der Integration von Diensten durch Kapselung in der Benutzerschnittstelle auch für andere Kommunikationsprotokolle bzw. Dienstetypen im Internet möglich, wie Beispiele der Integration von telnet-basierten Diensten durch eingebettete Java-Applets zeigen. Hierfür ist aber der Entwicklungsaufwand ungleich höher. Daher beschränkt sich die Kapselung von Diensteinterfaces auf WWW-basierte Dienste.

4.4.6.3 Unabhängige Präsentation von Diensten

Die dritte und einfachste Variante der Integration eines externen Dienstes ist die reine Aktivierung des Dienstes aus dem elektronischen Buch heraus ohne Einflussnahme auf dessen Benutzerschnittstelle. Sie ist dann zu wählen, wenn die Integration technisch nicht oder nur mit großen Aufwand möglich ist bzw. wenn kein besonderes Erfordernis für die Integration vorliegt. Es ist zu prüfen, inwieweit eine solche Aktivierung über die Medien-Software-Zuordnungsverfahren auf der Basis der MIME-Typen, wie sie in gängigen Webbrowsern vorhanden sind, hinausgehen muss. Dabei spielen folgende Überlegungen eine Rolle:

- Die Zuordnung von MIME-Typen zu externen Applikationen und Viewern aus einem Webbrowser heraus stellt lediglich die Möglichkeit dar, bei der Traversierung einer Hypertextverknüpfung die mit dem Link verbundene Ressource nicht im Webbrowser, sondern in einem externen Programm anzuzeigen (z. B. Ghostview für Postsciпt-Dateien, ein Textverarbeitungsprogramm für DOC-Dateien oder eine Kompressionssoftware für Archivdateien). Im Fall der Bindung von Diensten an Inhaltsbestandteile eines elektronischen Buchs ist aber nicht die über einen Link verbundene Ressource, sondern der Inhalt selbst an den externen Dienst zu übergeben. Dafür ist ein anderes Aktivierungsmodell erforderlich.
- Der Rückgriff auf benutzerseitige Einstellungen des Viewing-Systems sollte mit Sicht auf das Architekturprinzip der Abstraktion von den Nutzungsmöglichkeiten eines (proprietären) Viewers vermieden werden, d. h. die konzeptuell bessere Lösung ist es, die notwendige Aktivierungsinformation zusammen mit den ohnehin vom Dienstverwaltungssystem gespeicherten Daten über verfügbare Dienste abzulegen.
- Da unterschiedliche Aktivierungsformen auf dieser allgemeinsten Ebene der Dienste-Integration denkbar sind (Starten lokaler Software, Verknüpfung zu einem Dienst im World Wide Web, Dienst auf Basis eines anderen Internet-Protokolls), ist die Verwendung eines vorgegebenen Aktivierungsverfahrens wie der MIME-Typen-Zuordnung für externe Applikationen kein hinreichend mächtiges Verfahren.

Der Start eines Dienstes als externe Anwendung bedeutet im Rahmen des Architekturmodells eine *unidirektionale* Kommunikationsform: Der Benutzer startet einen Dienst für einen im elektronischen Buch enthaltenen Inhalt, der Dienst wird mit diesem Inhalt instantiiert und der Benutzer kann mit dem Dienst interagieren. Damit verlässt der Benutzer aber den unmittelbaren Interaktionskontext des elektronischen Buchs. Eine Rückführung von Ergebnissen bzw. Daten, die der Benutzer mit Hilfe des Dienstes realisiert, ist anschließend nur über allgemeine und in der Regel betriebssystemabhängige Schnittstellen wie die Zwischenablage (Kopieren von Daten des externen Dienstes in die Zwischenab-

lage und Einfügen in die Schnittstelle des Speicherungsdienstes (Editorfenster)) oder das Dateisystem möglich (soweit etwa der Speicherungsdienst einen Zugriff auf das lokale Dateisystem zulässt).

Im Prototyp ist diese Form der Integration externer Dienste am Beispiel der Symbolmanipulation prototypisch realisiert, d. h. der Benutzer hat die Möglichkeit, eine Gleichung als Inhaltsbestandteil des elektronischen Buchs in einem Computeralgebrasystem als externem Dienst zu instantiieren und zu bearbeiten.

4.5 Ein Architekturmodell für dynamische elektronische Bücher

Nachdem voranstehend Einzelkonzepte dynamischer elektronischer Bücher eingeführt worden sind, soll nun ein allgemeines Modell der Architektur elektronischer Bücher vorgestellt werden. Es ist ohne Bezugnahme auf Realisierungstechnologien formuliert; seine Umsetzung in Weiterführung der im Referenzprojekt erreichten Ergebnisse wird in Teil III diskutiert.

Das Modell abstrahiert von den Details des Einzelfalls – gerade für den Multimediabereich sind bisher vor allem intuitionistische Ansätze üblich, die nicht zuletzt durch das hohe Maß an Interdisziplinarität bedingt sind: Da eine Vielfalt unterschiedlicher Techniken bei der Erstellung multimedialer Bücher und Anwendungen zusammenwirken, ist die Erarbeitung eines einheitlichen Modells bzw. einer konsistenten Entwicklungsmethode aufwendig. Zwei Formen der Modellbildung sind zu unterscheiden:

- Struktur-, Inhalts-, und Verhaltensmodelle, die Aufbau und Inhalt elektronischer Bücher zu beschreiben, aber von ihrer Entstehung abstrahieren (vgl. CATENAZZI & SOMMARUGA 1994), und
- Entwicklungsmodelle, die vor allem beschreiben, welche Verfahrensschritte bei der Erstellung elektronischer Bücher zu modellieren sind, vgl. oben Kap. 2.2.1.6.

Nachfolgend soll der erste Typus, die struktur- und inhaltsorientierten Modelle, betrachtet werden. Die unterschiedlichen Ebenen der Hypermediamodellierung (vgl. oben Kap. 2.2.1.6) lassen sich i. d. R. auf die Konzepte der theoretischen Informatik zurückführen (Automatentheorie, Graphen, formale Sprachen). Es existieren bisher kaum formale Modelle, die *elektronische Bücher* als Gegenstandsbereich haben. Ein erwähnenswerter Ansatz ist der Versuch von CATENAZZI & SOMMARUGA 1994, Struktur und Funktionsweise elektronischer Bücher mit mengentheoretischen Mitteln und unter Verwendung von Produktionsregeln zur Beschreibung der Buchstruktur zu modellieren: Das Modell orientiert sich am traditionellen Aufbau von Büchern, greift Elemente aus der Hypertexttheorie auf und versucht für Struktur und Nutzung elektronischer Bücher eine einfache formale Grammatik zu entwerfen. Aufgrund seiner Nähe zum traditionellen Buch ist es dem hier entwickelten Konzept dynamischer elektronischer Bücher verwandt und soll im Folgenden diskutiert werden: Der Ausgangspunkt der Modellierung elektronischer Bücher ist die Umsetzung der Buchstruktur durch Produktionsregeln, wie nachfolgend gezeigt wird:

Buch	::== $\text{Cover}_{\text{vorne}}$ Doppelseite* $\text{Cover}_{\text{hinten}}$
Doppelseite	::== Seite Seite
Seite	::== Seitenkomponenten
Seitenkomponenten	::== Text | Tabelle | Abbildung
Text	::== aktiver_Text | einfacher_Text
aktiver_Text	::== $\text{Verknüpfung}_{\text{hierarchisch}}$ | $\text{Verknüpfung}_{\text{Fenster}}$ | $\text{Verknüpfung}_{\text{Sprung}}$

Codebeispiel 4: Produktionsregeln für den Aufbau elektronischer Bücher (CATENAZZI & SOMMARUGA 1994: 321)

Ausgehend von dieser Grammatik lässt sich die *Nutzung* eines elektronischen Buchs mengentheoretisch beschrieben; gegeben sind die Mengen S aller Seiten eines Buchs sowie A als Menge aller Anfragen bezüglich des Buchs. Es lassen sich weiter darstellen:

- die aktuelle Position des Lesers im Buch als Element der Menge möglicher beschreibbarer Positionen im Informationsbestand,
- Zugangspfade und Makrostrukturen im Buch als Folgen von Links zwischen Informationseinheiten und
- Recherchen (einfache Modellierung einer Retrievalfunktion) als Relation $(S \times A) \rightarrow S$ über den Mengen der Seiten bzw. der möglichen Anfragen.

Das Modell berücksichtigt allerdings Aspekte der Dynamisierung elektronischer Bücher nicht. Zudem ist seine Modellierung informationeller Einheiten relativ grobkörnig und durch die Orientierung an Buchseiten rein präsentationsbezogen. Die oben in Kap. 4.2 diskutierten unterschiedlichen Ebenen der Informationsauszeichnung (und ihre Formalisierung durch Dokumentgrammatiken) bilden zusammen mit dem nachstehend beschriebenen Architekturmodell eine differenziertere Beschreibung elektronischer Bücher, in der die inhaltliche Strukturierung und die Präsentation in einem Buchbetrachtungssystem konzeptuell getrennt sind.

4.5.1 Allgemeine Anforderungen

Die Funktion eines Architekturmodells ist es, anhand allgemeiner Vorgaben eine Struktur für dynamische elektronische Bücher zu entwickeln, die als Grundlage einer Realisierung dienen kann. Zu den allgemeinen Vorgaben gehören die bereits mehrfach erwähnten Randbedingungen,

- elektronische Bücher unter Anwendung von Internet-Standards im World Wide Web und unter
- weitgehender Anwendung deklarativer Standards für die Informationsauszeichnung zu realisieren.

Hinzu kommen Anforderungsmerkmale, wie sie sich dem Software-Engineering bzw. den besonderen Merkmalen an das elektronische Publizieren ergeben: Dazu gehören die

- *Generalisierbarkeit* des Modells hinsichtlich seiner Strukturen und Inhalte,
- sein *modularer Aufbau*, der sich in der praktischen Umsetzung widerspiegeln soll
- und seine *Skalierbarkeit*.

Die Generalisierbarkeit des Modells betrifft seine Übertragbarkeit auf eine Vielfalt elektronischer Bücher mit unterschiedlichen Charakteristika (Art und Umfang multimedialer Elemente, Publikationshäufigkeit, Trägermedium, Distributionsmodell, vgl. oben Kap. 2.3 und 3.2).

Der *modulare Systemaufbau* greift eine Modellbildung aus dem Bereich der Programmiersprachen bzw. des Software-Engineering auf. Für die geforderte Allgemeinheit des Modells ist ein modulares Konzept sinnvoll, das aus klar abgegrenzten Grundkomponenten aufgebaut ist und zu dem bei Bedarf auf der Basis des Kernsystems zusätzliche Module mit dedizierter Funktionalität hinzugenommen werden können, wie interaktive Komponenten oder Informationsdienste. Die Modularität spiegelt sich nicht nur in der softwaretechnischen Realisierung (konkrete Komponenten: Präsentationssystem, Datenverwaltung, Diensteverwaltung etc.), sondern – und dabei handelt es sich um einen für elektronische Bücher typischen Aspekt – auch auf der Ebene der *Kodierungsgrundlage*,

d. h. der verwendeten Standards für die Informationsaufbereitung wieder: Die Verwendung der SGML-Standardfamilie garantiert modulare Erweiterbarkeit, da z. B. durch geeignete Kodierung von Metadaten zusätzliche Funktionalität integrierbar ist.

Skalierbarkeit als Ziel des Architekturmodells hängt mit der Modularität eng zusammen und meint die Erweiterbarkeit und Ausbaufähigkeit elektronischer Publikationen. Skalierbarkeit hat sowohl einen quantitativen (Umfang des präsentierten Materials) als auch einen qualitativen Aspekt (Einbindung zusätzlicher Präsentationsmodi für denselben Informationsbestand; unterschiedliche Granularität der Inhaltsaufbereitung).

Die Verwendung von Standards versucht eine Brücke von den rein konzeptuellen Aspekten („was ist wünschenswert für dynamische elektronische Bücher“) hin zu den Fragen der technischen Realisierung und der Generalisierbarkeit zu schlagen: Selbst wenn man einräumt, dass mit Hilfe einfacher Realisierungswerkzeuge wie höheren Programmiersprachen und graphischen Benutzerschnittstellen als Nutzungsumgebungen jedes konzeptuelle Modell umsetzbar ist, spielt die Verwendung allgemein verfügbarer Standards und insbesondere der SGML-Standards sowie der Netzwerkprotokolle eine wichtige Rolle für die Plausibilität des Ansatzes, da damit seine Praktikabilität nachgewiesen werden kann, ohne auf eine Insellösung beschränkt zu sein. Dieses Argument ist insofern zu relativieren, als eine Reihe der dazu herangezogenen Standards und Werkzeuge zwar allgemein verfügbar und öffentlich einsehbar sind, gleichzeitig aber noch nicht abschließend definiert sind (z. B. XML Schemata oder XPath, vgl. unten Kap. 5.2) oder noch keine weit verbreitete Nutzungsinfrastruktur (Autorenwerkzeuge/Viewer) existiert (z. B. XML).

Eine unmittelbare Konsequenz aus dem Globalziel *Generalisierbarkeit* sowie der Anwendung von Standards sowohl für die programmtechnische Realisierung als auch für die Informationsauszeichnung ist die weitgehende *Abstraktion* von

- einem bestimmten Trägermedium,
- der Ausführungsumgebung (Betriebssystem) und von den
- Dokumentformaten.

Die Konzeption eines elektronischen Buchs als netzbasierte Anwendung abstrahiert von einem bestimmten Trägermedium und stellt eine Weiterentwicklung elektronischer Publikationen dar, die an ein physisches Distributionsformat gebunden sind. Im Referenzprojekt hat diese Zielstellung wesentlich die Auswahl der konkreten Realisierungswerkzeuge beeinflusst; dort wurde zwar in einem ersten, durch verlegerische Überlegungen motivierten Schritt, eine Fassung auf der Basis des Datenträgers CD-ROM erstellt; diese konnte ohne wesentliche Modifikation zu einer netzbasierten Anwendung weiterentwikkelt werden. Beachtet man, dass bestimmte Funktionskomponenten von vornherein nur durch eine Netzintegration realisierbar sind (z. B. die Integration externer Dienste, soweit diese über ein Datennetz angefordert werden), ist das Ziel der Abstraktion vom physischen Trägermedium ein konstitutives Merkmal eines dynamischen elektronischen Buchs.

Die Abstraktion von der Ausführungsumgebung (Plattformneutralität) soll sicherstellen, dass das elektronische Buch unabhängig von der *ex ante* nicht bekannten technischen Infrastruktur des Benutzers eingesetzt werden kann; dieses Ziel hat entscheidenden Anteil am Erfolg des World Wide Web als plattformunabhängiger Informationsinfrastruktur einerseits, an der Durchsetzung der Programmiersprache Java und der mit ihr verbundenen Technologien andererseits. Mit wenigen Einschränkungen kann das im Referenzpro-

jekt entwickelte dynamische elektronische Buch als plattformneutrale Anwendung eingesetzt werden.[49]

Der dritte Aspekt der Abstraktion, die Dokumentenformate, lässt sich auf zwei Unteraspekte aufgliedern: Zum einen die Abstraktion von einem bestimmten Ausgangsformat (d. h. einer feststehenden Auszeichnungssprache für die Primärdaten des elektronischen Buchs) sowie die Abstraktion von einem Präsentationsformat. Beide Aspekte lassen sich durch die Verwendung einer SGML-basierten Metasprache wie XML in Verbindung mit geeigneten Transformations- und Präsentationssprachen realisieren und gewährleisten, dass prinzipiell beliebige Ausgangsformate in beliebige Zielformate transformiert werden können. Dies stellt das für das elektronische Publizieren wichtige Merkmal der medienneutralen Datenhaltung sicher, das die Wiederverwendbarkeit von Substanzen erleichtert. Auf der Nutzerseite ist für die Anpassung des Präsentationsformats an andere Ausführungsumgebungen lediglich eine Neudefinition der Transformations- und Präsentationsregeln, aber nicht notwendigerweise eine erneute Kodierung der Inhalte erforderlich.

4.5.2 Aufbau

Um das Architekturmodell einordnen und abgrenzen zu können, ist es sinnvoll, noch einmal Aufbauvarianten elektronischer Bücher zu diskutieren. Zudem hat die Entwicklung des im Referenzprojekt entwickelten elektronischen Buchs mehrere Aufbauvarianten durchlaufen, die stellvertretend für verschiedene Architekturmodelle stehen. In Analogie zum Dexter-Referenzmodell (vgl. oben Kap. 2.2.1.5) sind folgende funktionale Schichten eines elektronischen Buchs hinsichtlich ihrer Integration in das Modell zu unterscheiden:

- Datenspeicherung
- Präsentation der Inhalte
- Steuerung des Buchs

Die strukturell einfachste Realisierungsform, die in der Regel mit der Realisierung über einen physikalischen Datenträger zusammenfällt (CD-ROM-Publishing), ist die integrierte Entwicklung eines elektronischen Buchs mit Hilfe eines Multimedia-Autorensystems. Speicherungs-, Steuerungs- und Präsentationsebene bilden softwaretechnisch eine Einheit: Im Referenzprojekt entspricht dies der zunächst realisierten Variante, bei der Steuerung und Präsentation in einem zu diesem Zweck angepassten Webbrowser integriert sind und die Speicherung der primären Buchinhalte sowie der ergänzenden Komponenten unmittelbar über das lokale Dateisystem erfolgt. Vergleichbar mit dieser Lösung sind autorensystembasierte Lösungen, vor allem mit Macromedia Director realisierte Multimediaanwendungen, wie sie z. B. für die Mehrzahl der im Projektverbund Multimediabuch realisierten Vorhaben kennzeichnend sind (vgl. oben Kap. 3.1.2). Eine solche monolithische Lösung hat den grundsätzlichen Nachteil, dass sie sich nur bedingt in die durch das World Wide Web zur Verfügung stehende Infrastruktur integrieren lässt und – bei Distribution über einen Datenträger – Modifikationen und Updates einen hohen Aufwand nach sich ziehen.

---

[49] Lediglich diejenigen Multimediakomponenten, die in diesem Buch mit proprietären Autorensystemen entwickelt wurden (Macromedia Director), können auf Unix-Systemen nicht verwendet werden; ansonsten lässt sich das elektronische Buch prinzipiell auf MS-Windows-, Macintosh-, UNIX- und Linux-Systemen einsetzen.

Ein erster Differenzierungsschritt, der über dieses einfache Modell hinausgeht, ist die Realisierung des elektronischen Buchs als Client-Server-Anwendung, bei der der Server zunächst die Funktion des Speicherungsdienstes übernimmt: Im Referenzprojekt Physikalisches Praktikum war diese Lösung als Anwendung im World Wide Web problemlos zu realisieren, da durch die Verwendung WWW-kompatibler Kodierungsstandards und Werkzeuge die Übertragung ohne größere Modifikationen möglich war. Durch die Aufteilung der Ebenen Interaktionssteuerung und Präsentation auf den Client sowie Verwaltung, Transformation und Speicherung auf den Server entfällt – in technischer Hinsicht – die Problematik der Distribution,[50] wenn man Online-Zugang und WWW-Browser als technische Infrastruktur beim Benutzer voraussetzt. Zugleich ist die Versionierung der Inhalte (Ergänzung, Update, Korrektur) einfacher durchführbar und kann vor allem unmittelbar distribuiert werden. Dies ist dann von Bedeutung, wenn man den Entwicklungsprozess elektronischer Bücher als einen langfristigen inkrementellen Vorgang versteht. Bei einer solchen – einfachen – Client-Server-Lösung ist die Verwendung der Protokolle und Werkzeuge des World Wide Web nur ein Beispiel für die Umsetzung des elektronischen Buchs als Client-Server-Anwendung; grundsätzlich könnte eine andere technische Grundlage verwendet werden, was allerdings den Nachteil hätte, dass in weit größerem Ausmaß Softwarekomponenten neu zu realisieren wären. In praktischer Hinsicht war im Referenzprojekt die Entwicklung einer netzbasierten Lösung mit dem Vorteil verbunden, dass so zusätzliche Werkzeuge z. B. zur Informationserschließung eingebunden werden konnten, die einen WWW-Server voraussetzen.

Hinsichtlich der eingangs genannten Merkmale dynamischer elektronischer Bücher ist zu fragen, um welche zusätzlichen Funktionen ein einfaches Client-Server-Modell erweitert werden muss. Nachfolgend sei der Server als *Buchverwaltungssystem*, der Client als *Buchpräsentationssystem* bezeichnet. Die wesentliche Erweiterung des einfachen Client-Server-Konzepts betrifft das Buchverwaltungssystem: Es muss in der Lage sein,

- deklarativ ausgezeichnete Information zu analysieren und in ein Präsentationsformat umzuwandeln,
- die so aufbereiteten Daten an das Buchbetrachtungssystem zu liefern,
- den primären Inhaltsbestandteilen des Buchs Komponenten und Dienste zuzuordnen,
- Informationen über den Zustand der Buchnutzung zu verwalten,
- Informationen über den Benutzer auszuwerten und
- das elektronische Buch in einen konkreten Nutzungskontext einzuordnen.

Als ein zusätzlicher Aspekt kommt die konzeptuelle Trennung der Bereitstellung der Buchinhalte und der ihnen zugeordneten Komponenten und Dienste von der individuellen Nutzungssituation des Lesers und der Einbettung in einen Nutzungskontext hinzu. Unter einem Nutzungskontext ist der Einsatz eines elektronischen Buchs in einem Kontext zu verstehen, der hinausgehend über die Distribution des Buches durch Verlag oder digitale Bibliothek Dienste und zusätzliche Komponenten anbietet. Ein Szenario hierfür ist die Verwendung des elektronischen Buchs im Kontext des Lernprozesses, wenn nicht nur auf die individuelle Nutzung durch den einzelnen Leser abgestellt wird. Das Architekturmodell ist nicht nur so zu strukturieren, dass die Integration und Nutzung von Diensten

[50] Aus der Perspektive des Verlags ergeben sich aber bei einer netzbasierten Lösung eine Reihe zusätzlicher Probleme, da die Nutzung traditioneller Distributionskanäle (insb. Buchhandel) ausfällt und neue Vermarktungswege und Abrechnungsverfahren zu realisieren sind, z. B. über die Einbindung in digitale Bibliotheken.

möglich ist, sondern soll auch eine Differenzierung des Dienste- und Komponentenangebots nach Nutzungskontexten ermöglichen. Diese beiden Ebenen entsprechen konzeptuell

a) dem Ziel der Individualisierbarkeit durch den einzelnen Benutzer und
b) der Verfügbarkeit gruppenbezogener Dienste (und Inhalte).

Auf der Seite des Buchpräsentationssystems sind weniger deutliche Änderungen erforderlich; sie betreffen gegenüber dem einfachen Client-Server-Modell vor allem die Darstellung von Diensten in der Benutzerschnittstelle des Buchs (generische Dienste) bzw. seiner Inhaltselemente und Komponenten. Als Grobstruktur ergibt sich eine Client-Server-Architektur analog zu dem *three-tier-model* (vgl. BALZERT 1996: 634 ff., TURAU 1999: 10, sowie das oben in Kap. 2.2.1.5 diskutierte Dexter-Referenzmodell), bei dem der Client im wesentlichen die Aufgabe der Realisierung der Präsentationslogik übernimmt, der Server allgemeine Verwaltungsaufgaben und die Anwendungslogik umfasst und der Datenzugriff und die Speicherung der Daten auf einer dritten Ebene erfolgen:

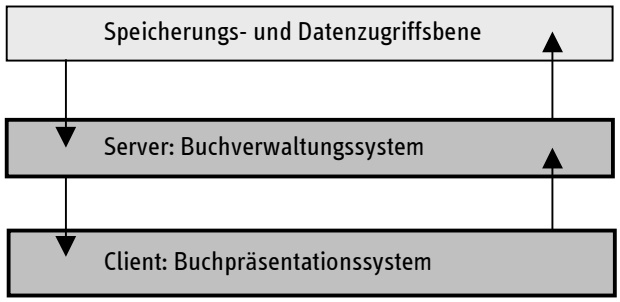

Abbildung 16: Drei-Ebenen-Systemarchitektur

Der oben dargestellte Entwicklungsgang lässt sich in Analogie zur Entwicklung der Architektur von Hypermediasystemen sehen, die sich konzeptuell in Weiterführung und Umsetzung des Dexter-Modells von monolithischen und i. d. R. mit proprietärer, nichtstandardisierter Technologie realisierten Systemen zu komponentenorientierten verteilten Anwendungen entwickelt haben, wie die folgende Abbildung deutlich machen soll:

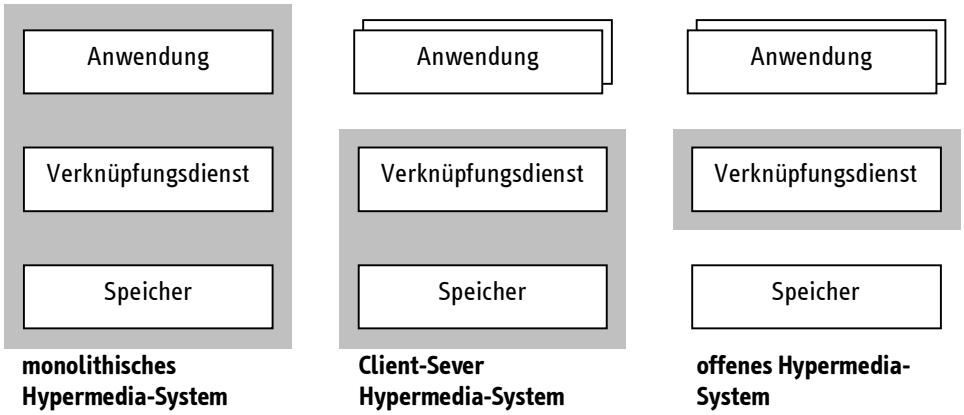

Abbildung 17: Entwicklung der Architekturen von Hypermedia-Systemen (WIIL & NÜRNBERG 1999: 428: Abb. 2)

Nachfolgend sollen zunächst das Navigations- und Strukturierungsmodell und anschließend daran der strukturelle Aufbau von Buchverwaltungs- und Buchbetrachtungssystem dargestellt werden.

4.5.2.1 Navigations- und Strukturierungsmodell

Aus der Aufgliederung der Inhalte des Buchs in den Basistext sowie in die ergänzenden Komponenten und Dienste und aus der Annahme einer lokal, d. h. innerhalb einer Struktureinheit wie einer Versuchsbeschreibung, sequentiellen Darstellungsart ergeben sich für das elektronische Buch vier aufeinander aufbauende Ebenen der Strukturierung und Nutzung:

- Sequentielle Abfolge von Informationseinheiten,
- Zuordnung von Komponenten und Diensten,
- hierarchische Zugriffsstrukturen und
- zusätzliche Hypertextverknüpfungen.

Auf der untersten Ebene ist der Ausgangspunkt in Analogie zum gedruckten Buch und unter der Annahme, dass die Informationsdarstellung und -aufnahme zunächst sequentiell erfolgt, die *sequentielle Abfolge* der Inhaltseinheiten des Basistexts, ggf. vom Präsentationssystem weiter in einzelne Präsentationseinheiten aufgegliedert:

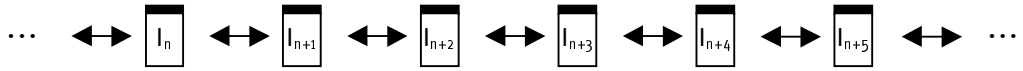

Abbildung 18: Inhaltseinheiten als sequentielle Abfolge

Ergänzend dazu erfolgt auf einer zweiten Ebene die *Zuordnung der Komponenten* und *Dienste* zu den Inhaltselementen, wobei die Zuordnung zwischen Komponente und Inhalts- bzw. Präsentationseinheit unterschiedliche Kardinalitäten aufweisen kann, d. h. eine Komponente kann an verschiedene Inhaltseinheiten gebunden sein. Umgekehrt können zu einer Inhaltseinheit mehrere Dienste oder Komponenten verfügbar sein:

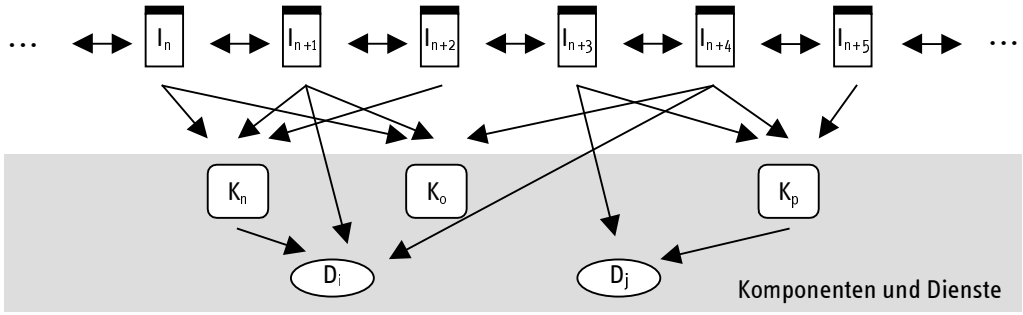

Abbildung 19: Zuordnung von Komponenten und Diensten

Auf einer dritten Ebene wird vorausgesetzt, dass die *Makrostruktur* der Inhalte eine Hierarchie ergibt und so eine Baumstruktur für den Zugriff auf die Buchinhalte und die Ihnen zugeordneten Komponenten und Dienste möglich ist (z. B. durch ein Inhaltsverzeichnis):

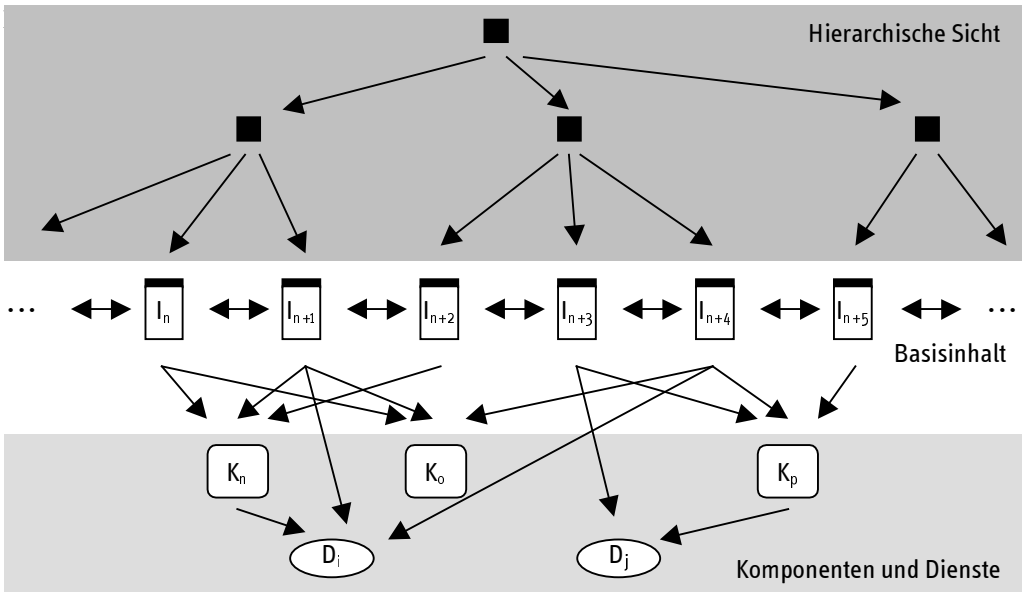

Abbildung 20: Hierarchischer Zugriff

Schließlich wird die Abfolge der Inhaltseinheiten durch ein Hypertextnetzwerk ergänzt, das aus automatisch generierten und manuell erstellten typisierten Verknüpfungen zwischen den beliebigen Inhaltseinheiten und Komponenten des Buchs bestehen kann:

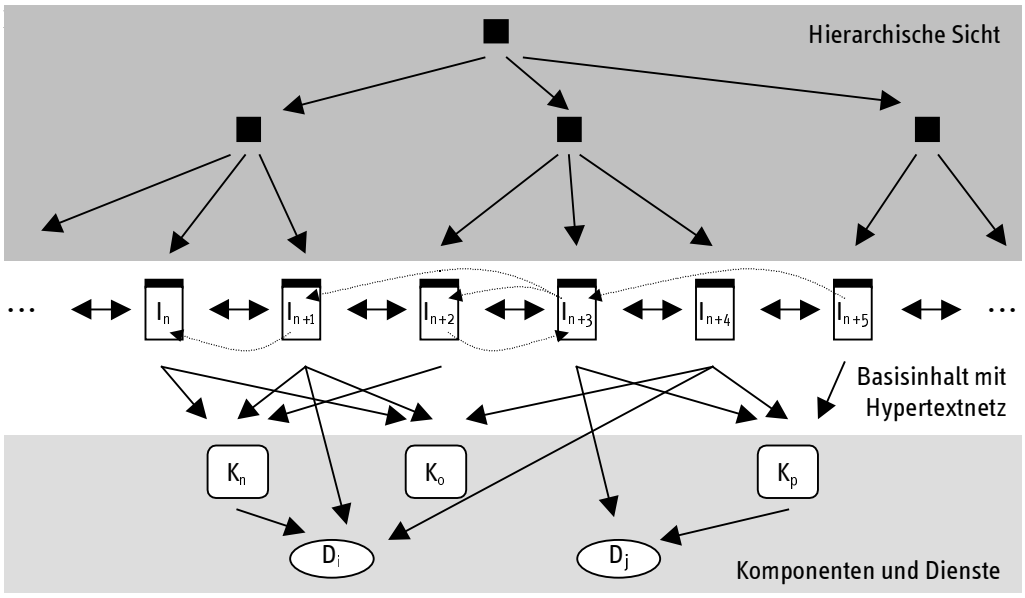

Abbildung 21: Zusätzliches Hypertextnetz

4.5.2.2 Buchbetrachtungssystem

Die Realisierung des voranstehend gezeigten Strukturierungsmodells ist Aufgabe der Informationsmodellierung und ihrer konkreten Umsetzung durch Speicherungs- Adressierungsverfahren (vgl. dazu unten Kap. 12). Auf der Seite des Benutzers muss das Buchbetrachtungssystem die notwendigen Funktionen für die Präsentation der Inhalte und Dienste bereitstellen. Das Buchpräsentationssystem verfügt über folgende Komponenten:

- ein Modul für die Darstellung von Inhalten (Mediendarstellung und Fensterverwaltung),
- Funktionen für die Navigation und die globale Steuerung der Buchdarstellung sowie
- eine Kommunikationsschnittstelle zum Server, über die Inhalte und Dienste angefordert und die entsprechenden Daten vom Server geladen werden.

Schematisch lässt sich das wie in der folgenden Abbildung gezeigt darstellen:[51]

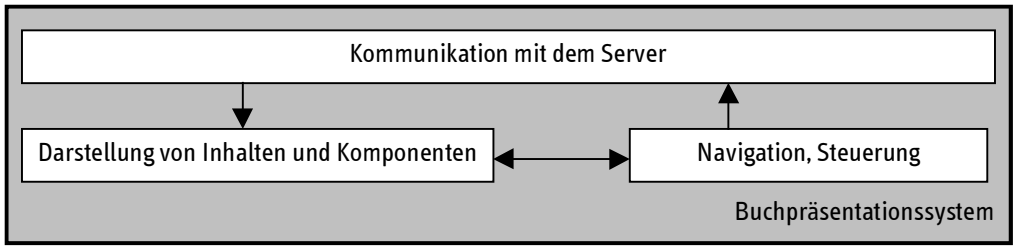

Abbildung 22: Schematischer Aufbau des Buchbetrachtungssystems

4.5.2.3 Nutzungskontext des elektronischen Buchs

Bei der Unterscheidung der vier Ebenen des Strukturierungs- und Navigationsmodells wurde nicht berücksichtigt, dass Inhalte, Komponenten und Dienste unterschiedlichen *Nutzungskontexten* zugeordnet sein können und nicht generell für alle Benutzer, sondern aus dem individuellen Kontext ermittelt werden (können). Aus dem oben dargestellten Dienstemodell ergibt sich eine Unterscheidung dreier Ebenen, in denen Inhalte gespeichert werden bzw. Dienste zur Verfügung stehen:

- Die Ebene der Primärdaten und Komponenten des Buchs mit generisch verfügbaren Diensten,
- die Ebene eines Nutzungskontexts für eine Mehrzahl von Nutzern, die konzeptuell und technisch von der Basisebene abgetrennt ist und
- die Ebene der individuellen Nutzung, über die Benutzerdaten ermittelt und benutzerbezogene Daten abgelegt werden können.

Diese Aufteilung lässt sich schematisch wie folgt darstellen:

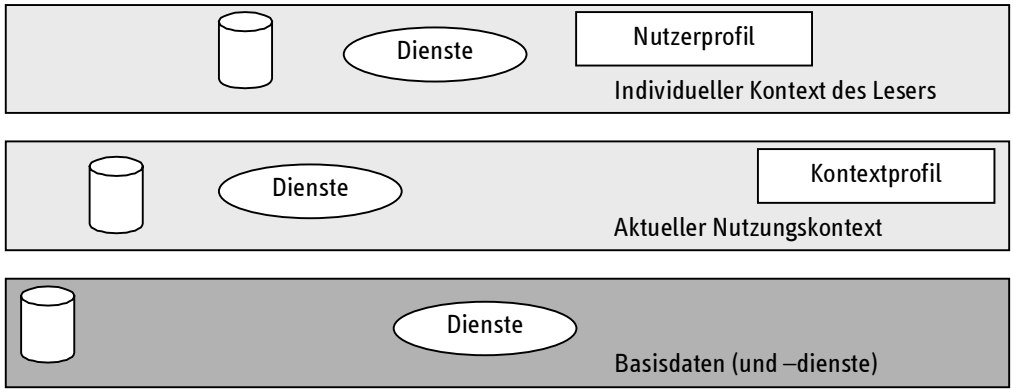

Abbildung 23: Unterscheidung von Zugriffsebenen für Inhalte und Dienste

[51] Es gilt grundsätzlich die Strategie, den Client im Rahmen der *three-tier*-Architektur möglichst auf wesentliche Funktionen zu beschränken (*thin client*) und diese mit standardisierten Technologien (WWW-Browser, SGML-/XML-Standrds) zu realisieren.

Neben den eigentlichen Daten und Diensten müssen auf der Ebene des Nutzungskontexts sowie für die individuelle Situation des Lesers Zusatzinformationen in einem Benutzer- bzw. Kontextprofil verfügbar sein, da sonst die Abtrennung von den Basisdaten und die Verwendung des Konzepts für unterschiedliche Benutzer und Kontexte nicht möglich ist bzw. ihre Verwaltung zentral über den Buchserver erfolgen müsste. Dies kann aber bei einer Vielzahl möglicher Nutzungskontexte bzw. individueller Benutzer nicht zu den Aufgaben des Buchverwaltungssystems gehören.[52]

4.5.2.4 Buchverwaltungssystem

Die zentrale Komponente innerhalb des Architekturmodels, das Buchverwaltungssystem, hat für jedes als Client-Server-Anwendung realisierte elektronische Buch die Aufgaben,

- den Zugriff auf die Daten (Speicherungsebene) zu gewährleisten,
- notwendige Transformationen der Daten in ein Darstellungsformat durchzuführen,
- Anfragen des Client auszuwerten (z. B. Zugriff auf die nächste Informationseinheit, Anfordern einer Komponente oder eines Dienstes) und
- die Kommunikation mit dem Client abzuwickeln.

Zusätzlich zu diesen Basisfunktionen muss das Buchverwaltungssystem über Module verfügen, die das oben dargestellte mehrschichtige Nutzungskonzept und den Zugriff auf Dienste innerhalb dieser verschiedenen Ebenen gewährleistet. Dazu gehören

- die Diensteverwaltung,
- die Kontextverwaltung und
- die Verwaltung benutzerbezogener Information.

Um die Interaktion der verschiedenen Komponenten zu gewährleisten, ist eine globale Steuerungskomponente erforderlich. Sie verwaltet Informationen über

- den aktuellen Buchzustand,
- den Benutzer (Schnittstelle zur Benutzerverwaltung) und
- die Einbettung des Buchs in einen Nutzungskontext (Schnittstelle zur Kontextverwaltung).

Unter dem *Zustand* des Buchs sind folgende Informationen zusammengefasst:

- Die aktuelle Position des Lesers im Basisdatenbestand,
- die verfügbaren Komponenten und Dienste, die sich aus den drei Konfigurationsebenen (Basisbestand, Nutzungskontext, individuelle Nutzungssituation) ermitteln lassen und
- die aktuelle Zuordnung der Dienste zu den Inhaltselementen und Komponenten der aktuellen Buchposition.

Aus den genannten Komponenten ergibt sich die in der folgenden Abbildung gezeigte Architektur des Buchverwaltungssystems:

[52] Das Konzept ist in diesem Sinn auf die Betrachtung des einzelnen elektronischen Buchs eingeschränkt; bei Eingliederung eines elektronischen Buchs in die technische Infrastruktur einer digitalen Bibliothek können ggf. manche Dienste auf die Bibliotheksebene verschoben werden, vgl. dazu unten Kap. 15.2.

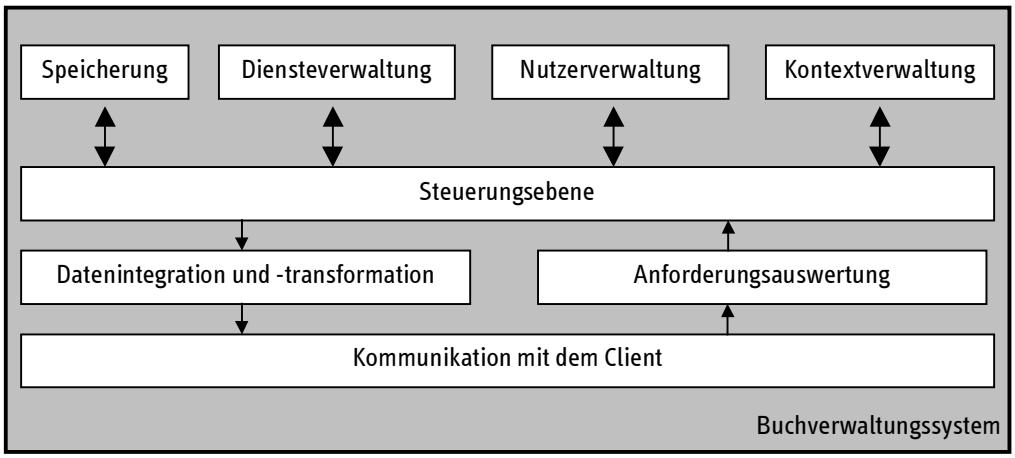

Abbildung 24: Aufbau des Buchverwaltungssystems

4.5.2.5 Gesamtstruktur

Aus den voranstehend dargestellten Komponenten und Schichten des dynamischen elektronischen Buchs resultiert die in Abbildung 25 gezeigte Gesamtarchitektur. Sie versucht die konzeptuelle Trennung zwischen Buchverwaltungssystem und seinem Basisdatenbestand sowie dem Nutzungskontext des Buchs und der individuellen Nutzungssituation zu veranschaulichen. Die drei Ebenen, aus denen Daten und Dienste verfügbar sind bzw. in denen Daten gespeichert werden können, sind durch die entsprechenden Komponenten des Buchverwaltungssystems miteinander verbunden.

Die Informationen über den einzelnen Benutzer bzw. den aktuellen Nutzungskontext sind jeweils zu einem Profil zusammengefasst, das das Buchverwaltungssystem zu Beginn einer Sitzung einliest und aus den darin enthaltenen Daten den Startzustand des Buchs ermittelt. Bei Beendigung der Interaktion wird das individuelle Nutzerprofil aktualisiert.

4.6 Fazit

Das voranstehende Kapitel hat auf einer allgemeinen Ebene Merkmale dynamischer elektronischer Bücher eingeführt und darauf aufbauend ein Architekturmodell für ihre Realisierung entwickelt. Dabei hat sich hinsichtlich der *Inhalte* eines elektronischen Buchs die Einbettung multimedialer interaktiver Komponenten, hinsichtlich der *Erweiterung seiner Nutzungsmöglichkeiten* die Integration des elektronischen Buchs in einen diensteorientierten Nutzungskontext als zentral herausgestellt. Das Architekturmodell verzichtet darauf, bereits an dieser Stelle eine detaillierte Konkretisierung hinsichtlich der technischen Randbedingungen (Standards, Werkzeuge) bzw. der Inhalte (Anwendungskontext des Referenzprojekts) vorzunehmen.

Das oben skizzierte Schema ist bewusst abstrakt gehalten, d. h. es geht daraus nicht im Einzelnen hervor, welche Standards und Werkzeuge zu seiner Realisierung bzw. zur Kodierung der Informationen und Dienste verwendet werden. In Teil II dieser Arbeit werden zur Präzisierung des Modells relevante Standards der deklarativen Informationsaufbereitung diskutiert, die nicht nur für die Kodierung der Primärdaten, sondern auch für die Aufbereitung von Metadaten eingesetzt werden können. Im Anschluss daran gibt Teil III einen Überblick zur verfügbaren technologischen Infrastruktur für die Realisierung dy-

namischer elektronischer Bücher und stellt den auf der Basis des hier eingeführten Architekturmodells realisierten Prototyp vor.

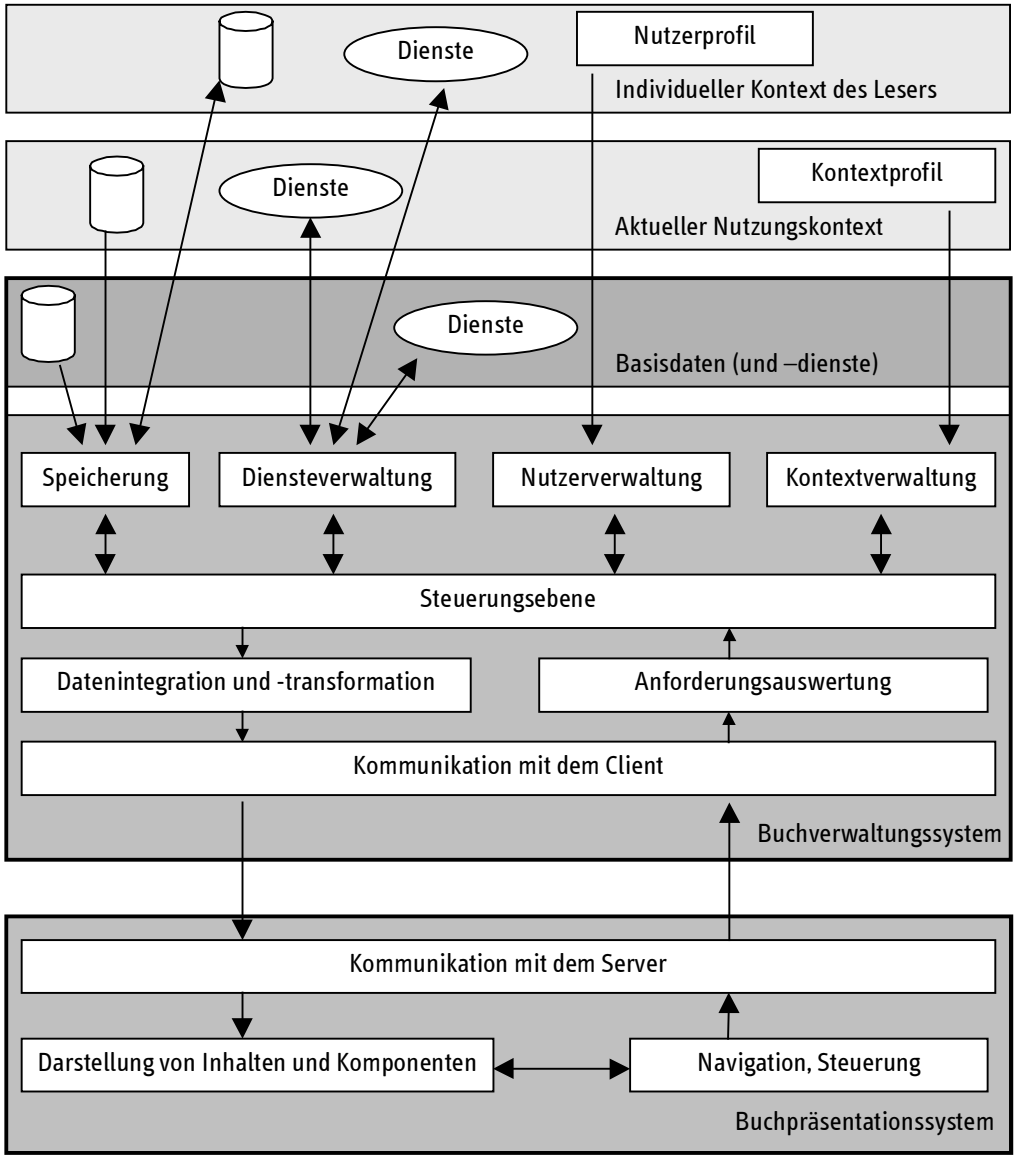

Abbildung 25: Schematischer Aufbau dynamischer elektronischer Bücher

Teil II: Strukturierung und Darstellung von Information

Der zweite Teil dieser Arbeit befasst sich mit der Frage, wie die in elektronischen Büchern enthaltenen Materialien – Text, Bild, Multimediakomponenten etc. – strukturiert und dargestellt werden können. Die Untersuchung orientiert sich primär an den funktionalen Aspekten der Aufbereitung; ein sekundärer Leitfaden sind die Standards im Umfeld der *Standard Generalized Markup Language* (SGML), die jeweils kurz vorgestellt und auf ihre Eignung für den Einsatz bei elektronischen Büchern hin untersucht werden. Eine Eingrenzung auf die Familie SGML-basierter Standards soll aus folgenden Gründen erfolgen:

- SGML ist ein seit langem eingeführter Standard für Dokumentenstrukturbeschreibungen, der als *Metasprache* eine hohe Ausdrucksfähigkeit besitzt und das Prinzip der deklarativen Informationsauszeichnung umsetzt.
- Die wichtigsten Standardisierungsbemühungen im Umfeld des *Electronic Publishing* wie Multimediakodierung (HyTime, SMIL), Beschreibung von Layout- und Präsentationsaspekten (DSSSL, CSS, XSL) der Spezifikation von Hypertextverknüpfungen (HyTime, XPointer, XLink) oder der Metadatenkodierung (RDF) basieren auf SGML.
- Durch die Verbreitung SGML-basierter Standards wie HTML und XML im Internet bzw. World Wide Web einerseits, im Verlagswesen andererseits deckt die SGML-Familie zwei wesentliche Bezugsfelder dieser Arbeit ab.

Bei der Auswahl der diskutierten SGML-Standards spielen sowohl das Innovationspotential eines Standards als auch seine praktische Operationalisierbarkeit eine Rolle: Für komplexe Standards wie DSSSL (vgl. unten Kap. 7.1) oder HyTime (vgl. unten Kap. 8.1) existiert zwar keine umfangreiche Softwareinfrastruktur – sie können daher im Rahmen dieser Arbeit nicht oder nur bedingt für die prototypische Realisierung herangezogen werden – in ihnen werden aber fundamentale Konzepte der Dokumentenstrukturierung und ihrer Auszeichnung eingeführt, die u. a. die Transformation unterschiedlicher Doku-

mentenstrukturierungen, die Kodierung zeitabhängiger Information und die Darstellungskodierung betreffen.

Im Zentrum steht das Konzept der Auszeichnung von Dokumenten bzw. multimedialer Informationseinheiten im Allgemeinen mit Hilfe geeigneter Markupsprachen. Unter *deklarativem Markup* ist in diesem Sinn die Verwendung standardisierter und wohldefinierter Kodes zu verstehen, die nicht an ein bestimmtes Softwarewerkzeug gebunden sind (wie bei Steuerzeichen in Textverarbeitungsprogrammen), aber gleichwohl automatisch ausgewertet werden können.

Eine im Umfeld von SGML und dem World Wide Web weit verbreitete Unterscheidung ist die Trennung zwischen der Kodierung von *Struktur*information und der eigentlichen *Darstellung* oder *Präsentation* von Dokumenten. Diese Unterscheidung muss man präzisieren bzw. verfeinern: Neben die Frage, wie Dokumente durch deklaratives Markup angereichert werden können und wie ein solches Strukturmarkup für die Funktionalität elektronischer Bücher zu nutzen ist (z. B. bei der Navigation, der Informationserschließung oder bei der Integration von Multimediakomponenten), treten weitere Anwendungsbereiche für deklaratives Markup:

- die Kodierung der Binnenstruktur komplexer Multimediainhalte unter besonderer Berücksichtigung zeitlicher und räumlicher Aspekte und
- die Kodierung von Metadaten unterschiedlicher Art, die einen Beitrag zur Funktionalität elektronischer Bücher leisten können.

Damit ergeben sich mit *Strukturierung*, *Multimediaaufbereitung*, *Darstellung* und *Metadatenkodierung* vier Anwendungsbereiche für deklaratives Markup, die mit Blick auf entsprechende SGML-Standards erörtert und im Kontext des Referenzprojekts *Multimediales Physikalisches Praktikum* diskutiert werden sollen. Die folgende Tabelle zeigt die wichtigsten Standards mit Bezug zu den genannten Ebenen im Überblick. Dabei ist allerdings zu beachten, dass SGML und XML *Meta*sprachen für die Informationskodierung sind, während es sich bei HTML um eine *Anwendung* von SGML bzw. XML handelt.

Struktureller Aspekt	SGML	HTML	XML
Darstellung und Transformation	DSSSL	CSS (1-3)	XSL/T
Multimediaintegration	HyTime	einzelne Tags innerhalb von HTML (<EMBED>, <APPLET>, <OBJECT>)	SMIL/SMIL BOSTON
Programmzugriff		SAX/DOM	

Tabelle 16: Übersicht ausgewählter Standards auf der Basis von SGML

5 SGML und XML: Metasprachen für die Informationsstrukturierung

Im Folgenden wird zwischen *Metasprachen* und ihren *Anwendungen* unterschieden. Bei SGML und XML handelt es sich um Metasprachen, die Konstruktionsregeln für konkrete Strukturierungs- oder Präsentationsformate festlegen. Erst durch die Definition eines konkreten Formats für eine bestimmte Anwendung (z. B. HTML) können sie in der Praxis des elektronischen Publizierens angewandt werden.

Auf die Darstellung der Grundlagen von SGML und XML folgt die Erläuterung von für das elektronische Publizieren relevanten Strukturierungsformaten sowie die Diskussion von Standards für Präsentationsgestaltung, die Kodierung von Multimediainformation und die Metadatenkodierung.

5.1 Standard Generalized Markup Language

Die *Standard Generalized Markup Language* (SGML) ist ein internationaler Standard (ISO 8879 / EN 28879 / DIN EN 28873) für die Definition von Informationsstrukturierungsformaten, der 1986 verabschiedet wurde. Seine Vorgeschichte reicht jedoch bis in die 60er Jahre zurück, als man versuchte, eine deklarative Auszeichnungssprache für die Aufbereitung von Dokumenten zu entwickeln.[53] Im Umfeld der Textverarbeitung in einer von Großrechnern mit proprietären Datenformaten geprägten DV-Landschaft sollte das Konzept eines *integrated text processing* unterstützt werden, das nicht nur die Bearbeitung von Text (*editing*), sondern auch die Speicherung (*storage*), Indexierung (*indexing*), Erschließung/Suche (*retrieval*) und die Ausgabeaufbereitung bzw. Präsentation berücksichtigt (vgl. GOLDFARB 1997: 659 f.). Aus diesen frühen Ansätzen, die bereits eine umfassende Konzeption für die Erstellung, Verwaltung und Nutzung von Dokumenten darstellen, hat sich SGML entwickelt. Es handelt sich um die formale Definition einer Syntax für den Aufbau von Informationsstrukturbeschreibungen.[54] SGML-basierte Strukturbeschreibungen treffen lediglich Aussagen über die *logische Struktur* von Dokumenten (bzw. informationellen Einheiten), nicht aber über Bedeutung, Verarbeitung oder Präsentation (Layout).

SGML hat zunächst im Verlagswesen und in der technischen Dokumentation Verbreitung gefunden;[55] durch die Ausweitung zu einer „Familie verwandter Standards", die nicht nur die Strukturierung, sondern auch die *Verarbeitung* und *Präsentation* von Information kodieren und vor allem durch die Entscheidung, HTML als Auszeichnungssprache für das World Wide Web als SGML-Anwendung zu definieren, ist SGML zu einem zentralen Standard im elektronischen Publizieren und darüber hinaus (z. B. E-Commerce) geworden. Der SGML-Standard definiert Regeln für die Erstellung von Informations-

---

[53] Die von Charles Goldfarb entwickelte *Generalized Markup Language* ist ein unmittelbarer Vorläufer von SGML, vgl. GOLDFARB 1997: 654 ff.

[54] Auch wenn SGML für die Strukturierung von textbasierten Dokumenten konzipiert worden ist, rechtfertigt seine Anwendung für fast alle Bereiche der Verarbeitung von Information den allgemeineren Begriff „Informationsstrukturierung".

[55] Für SGML im engeren Sinn dürfte dies noch heute gelten, vgl. etwa für den deutschsprachigen Raum die in MÖHR & SCHMIDT 1999 zusammengestellten Anwendungsbeispiele.

strukturbeschreibungen, die als *document type definitions* (DTD) auf die Kodierung konkreter informationeller Einheiten (Dokumente) angewandt werden können.

Die Übersicht zum Aufbau von SGML als Voraussetzung für die Weiterentwicklung der XML-basierten Aufbereitungsstandards erläutert

- den Aufbau von Dokumententypdefinitionen,
- die Definition von Elementen, Attributen und Entitäten sowie
- Konzepte für die Modularisierung und die sekundäre Strukturierung.

5.1.1 Aufbau einer document type definition

Eine *document type definition* legt in der DOCTYPE-Vereinbarung einen Namen für den Dokumenttyp und einen Status für die Dokumententypdefinition fest. Über die DOCTYPE-Angabe kann ein nach dieser DTD kodiertes Dokument die Dokumenttypdefinition referenzieren. Dies ist bei der Verarbeitung von SGML-Dokumenten wichtig, da z. B. ein Parser nur bei vorliegender DTD überprüfen kann, ob das Dokument im Sinn dieser DTD korrekt formatiert ist. Um die Definitionen in der DTD von den in spitzen Klammern gesetzten Anwendungen von SGML-Markup in einer Dokumentausprägung zu unterscheiden, beginnen sie mit der Zeichenfolge <! und enden mit einer schließenden spitzen Klammer (>). Die DOCTYPE-Vereinbarung hat folgende Struktur:

```
<!DOCTYPE Bezeichner Suchverweise/Vereinbarungen>
<!DOCTYPE Bezeichner PUBLIC "ISO//Besitzer//Klasse Spezifikation//Sprache" “Speicherort”>
```

Codebeispiel 5: Syntax der DOCTYPE-Vereinbarung

Es ist zu unterscheiden, ob eine DTD öffentlich (PUBLIC) oder systemgebunden (SYSTEM) sein soll; zusätzlich kann ihre Version, ihr Status (standardisiert/nicht standardisiert) sowie ihre Position im Dateisystem angegeben werden, wie die Beispiele einiger DOCTYPE-Vereinbarungen zeigen:

```
<!DOCTYPE bericht PUBLIC "- //Wolff//DTD Brief//DE" “http://test.org/bericht.dtd”>
<!DOCTYPE bericht SYSTEM "bericht.dtd">
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html40/strict.dtd">
```

Codebeispiel 6: Beispiele für DOCTYPE-Vereinbarungen

Es ist zwischen der Festlegung des Dokumenttyps in der DTD selbst und der Referenzierung einer bestimmten DTD über eine DOCTYPE-Anweisung in einem SGML-Dokument zu unterscheiden. In der DTD muss der Name des Wurzelelements mit dem Namen der DTD übereinstimmen; die DOCTYPE-Anweisung, die diesen Namen festlegt, umschließt das Wurzelelement und alle übrigen Definitionen:

```
<!DOCTYPE konzept [
                        <!element konzept - - unterkonzept+>
                        <!—alles Weitere … -->
                    ]
>
```

Codebeispiel 7: DOCTYPE-Anweisung in einer DTD

5.1.1.1 Elementdefinitionen

Der wichtigste Bestandteil einer DTD sind die Elementdefinitionen. In ihnen ist festgelegt, aus welchen Konstrukten ein einer bestimmten DTD entsprechendes Dokument aufgebaut sein kann und in welcher Beziehung die verschiedenen Elemente zueinander ste-

hen. Über die Inhaltsmodelle (*content models*) der Elemente, in denen festgelegt ist, welche anderen Elemente ein Element beinhalten kann, entsteht eine hierarchische Struktur, die man sich als Baum vorstellen kann. Die Mächtigkeit dieser Strukturierungssprache ist durch die Operatoren für den Aufbau der Inhaltsmodelle bestimmt. Die Anwendung eines Elements bei der Auszeichnung eines SGML-Dokuments erfolgt (*default*) durch Kodierung des Elementnamens in spitzen Klammern (Startmarke, *start tag*) vor dem Inhalt des Elements und dem Elementnamen mit vorangestelltem </ und schließender spitzer Klammer als Endmarke (*end tag*), z. B. <Element> der Inhalt des Elements </Element>.

Die Elementdefinition hat folgende Syntax: <!ELEMENT Bezeichner Verkürzung Inhaltsmodell>, wobei mehr als ein Elementbezeichner in einer Elementdefinition eingeführt werden kann und die Verkürzung angibt, ob Anfangs- und/oder Endmarke bei der Kodierung entfallen können:

```
- - Anfangs muss Endmarke kodiert werden
- o Endmarke  kann entfallen
o - Anfangsmarke kann entfallen
o o Anfangs und Endmarke können entfallen
```

Codebeispiel 8: Verkürzungsoperatoren in der Elementdefinition[56]

Die Grundannahme ist hier, dass Anfangs- *und* Endmarke kodiert werden müssen, z. B. beim <TABLE>-Element in HTML – der *Inhalt* eines Tabellenelements muss von <TABLE> … </TABLE> umschlossen sein. Umgekehrt kann beim Abschnittselement in HTML (*paragraph*, <p>) die Endmarke entfallen, d. h. man kann sie wie folgt kodieren:

```
<p> Inhalt des ersten Paragraphen.
<p> Inhalt des zweiten Paragraphen.
<p> Inhalt des dritten Paragraphen.
```

Codebeispiel 9: Anwendung von Verkürzungen durch Auslassen der Endmarken

Die Verkürzungen sollen die Kodierung vereinfachen, da Umfang und Kodierungsaufwand verringert wird. Sie machen aber die Verarbeitung von SGML-Dokumenten unnötig aufwendig; in XML (s. u. Kap. 5.2) ist die Elementverkürzung nicht zulässig. Die Beschreibung in der Elementdefinition dient der Definition seines Inhaltsmodells, d. h.

- der Angabe, welche anderen Elemente eine Element beinhalten kann bzw.
- bei Elementen, die die Blätter des Elementbaums bilden, der Angabe eines *Datentyps* für den Elementinhalt.

Zu den elementaren Datentypen für nicht weiter strukturierte Inhalte in einer Elementdefinition gehören

- PCDATA (*parsed character data*) für vom SGML-Verarbeitungssystem geprüfte Textdaten und
- CDATA für reine Textdaten (ohne weitere Verarbeitung, z. B. Auflösung von Entitätsreferenzen).

Das Typsystem von SGML ist schwach ausgeprägt, was auf den Ursprung von SGML im Dokumentations- bzw. Verlagsbereich zurückzuführen ist: Für Texte lässt sich kein formales Datenschema angeben; diesem Nachteil versuchen Datenschemata für XML abzuhelfen, um die Realisierung von Schnittstellen zwischen XML und Datenbanken zu ver-

[56] Das Symbol für die Verkürzungsmöglichkeit ist der Buchstabe ‚o', nicht die Ziffer ‚0' (*tag **o**mission*).

einfachen und einen vom DTD-Entwurf unabhängigen Mechanismus der Formatbeschreibung zu erhalten, vgl. unten Kap. 5.2.6.

Ein Element, das als Inhalt keine anderen Elemente, sondern nur Text enthalten soll, könnte wie folgt definiert sein: <!ELEMENT ABSATZ - - #PCDATA)>. Enthält ein Element andere Elemente, so ist deren Anordnung und ihr Auftreten ggf. durch Operatoren und Konnektoren zu bestimmen. Die Operatoren legen fest, ob ein Element optional ist und wie oft es ggf. auftreten kann:

+ Element kann 1-n-mal auftreten
? Element ist optional
* Element kann 0-n-mal auftreten

Codebeispiel 10: Operatoren in SGML

Konnektoren definieren die Reihenfolge der Elemente und definieren Alternativen:

, SEQ-Konnektor, gibt die Reihenfolge der Elemente an
& AND-Konnektor, erlaubt eine beliebige Reihenfolge (Permutation) der Elemente
| OR-Konnektor, trennt Alternativen des Inhaltsmodells

Codebeispiel 11: Konnektoren in SGML

Die Elemente im Inhaltsmodell können zusätzlich durch den Klammeroperator (()) gruppiert werden. Auf die so entstehenden *Modellgruppen* können die Operatoren und Konnektoren wiederum Anwendung finden. Einige Beispiele für Elementdefinitionen sollen die strukturellen Möglichkeiten für die Einführung von Markupelementen verdeutlichen:

```
<!ELEMENT TITEL – o #PCDATA)>                    <!-- Endmarke kann entfallen, Element
                                                 enthält nur charater data -->
<!ELEMENT ABSATZ - - (SATZ)+)>                   <!-- Ein Absatz enthält einen oder mehrere
                                                 Sätze-->
<!ELEMENT SATZ - o #PCDATA)>                     <!-- Ein Satz enthält nur charater data-->
<!ELEMENT KAPITEL - - (TITEL, ABSATZ+)>          <!-- Ein Kapitel enthält einen Titel und einen
                                                 oder mehrere Absätze (in dieser Reihenfolge)
                                                 -->
<!ELEMENT BERICHT - - (TITEL, VORWORT?, KAPITEL+, <!-- Ein Bericht hat einen Titel, optional ein
INDEX?) >                                        Vorwort, ein oder mehrere Kapitel und
                                                 optional einen Index -->
<!ELEMENT BILDUNTERSCHRIFT - - EMPTY>            <!-- Eine Bildunterschrift enthält keine
                                                 weiteren Daten -->
<!ELEMENT P - O (%INLINE;)*>                     <!-- Paragraph-Element aus HTML V.4; hat als
                                                 Inhalt beliebig viele Elemente, die über die
                                                 Parameter-Entität %inline aufzulösen sind --
                                                 >
<!ELEMENT TABLE - - (CAPTION?, (COL*|COLGROUP*), <!-- Tabellen-Element aus HTML V.4; einzig
THEAD?, TFOOT?, TBODY+)>                         zwingener Inhalt ist TBODY; TBODY selbst
                                                 muss wenigstens eine Tabellenzeile (<TR>)
                                                 enthalten, die TBODY-Marken können
                                                 entfallen (Verkürzung o o) -->
```

Codebeispiel 12: Beispiele für Elementdefinitionen

5.1.1.2 Attribute

Für die Elemente können Attribute definiert werden, die als Attribut-Wert-Paare (Syntax: Attributname=Attributwert) in den Startmarken der Elemente angegeben werden können.

Die Definition eines Attributs bzw. einer Attributliste erfolgt nach folgender Syntax: <!ATTRIBUTE element(gruppe) bezeichner wertebereich voreinstellung auftreten>. In einer Attributdefinition

- gibt der bezeichner den Elementnamen bzw. die Elementliste an, auf welche Elemente das Attribut sich beziehen soll,
- legt der wertebereich einen Datentyp für das Attribut fest bzw. definiert die möglichen Attributwerte,
  definiert die voreinstellung einen Defaultwert für das Attribut und
- legt das auftreten fest, ob das Attribut spezifiziert werden muss oder optional ist.

Die Attributdefinition <!ATTRIBUTE (bericht, kapitel) sprache (DE|EN|FR) DE> führt demnach ein Attribut sprache ein, das den Elementen bericht und kapitel zugeordnet ist, dessen Wertebereich die Wertausprägungen DE, FR und EN sind und das als Defaultwert DE hat. Für Attribute existieren eine Reihe von Datentypen mit z. T. vordefinierter Bedeutung:

CDATA	beliebige Zeichenkette
NUMBER	In der Regel eine Ziffernfolge (muss wenigstens eine Ziffer enthalten)
ID	Bezeichner, der als Verweisziel verwendet wird
IDREF/IDREFS	Verweis(e), die auf ID-Attribute anderer Elemente referieren

Codebeispiel 13: Attributtypen in SGML

Um die Kodierung einer Mehrzahl von Attributen zu vereinfachen, können Attribute in einer Attributliste (<!ATTLIST ...>) zusammengefasst werden, wie das folgende Beispiel zeigt:

```
<!ATTLIST (BERICHT|KAPITEL|SATZ)
    sprache  (DE|EN|FR)   DE
    bezeichner ID #REQUIRED      -- ein bezeichner (Datentyp ID) muss vorhanden sein (#REQUIRED) -
-
    verweis  IDREFS  #IMPLIED    -- eine Liste von Verweisen ist optional -->
```

Codebeispiel 14: Beispiel einer Attributliste

5.1.1.3 Entitäten

Ein weiteres Element der SGML-Auszeichnung sind *Entitäten*, Platzhalter, die bei der Verarbeitung eines SGML-Dokuments durch ein durch sie referenziertes Objekt ersetzt werden. Entitäten werden u. a. verwendet, um

- einzelne (Sonder-)Zeichen zu kodieren (*character entities*),
- Texte zu ersetzen,
- in SGML geschützte Zeichen zu kodieren (z. B. <, >, &) oder
- auf externe Ressourcen zu verweisen

Zusätzlich bilden sie als sog. Parameterentitäten ein zusätzliches Strukturierungsmittel bei der Definition einer DTD. Im einfachsten Fall dient die Entität als einfache Ersetzungsregel nach folgendem Muster: <!ENTITY bezeichner referenzierteZeichen(kette)>, z. B.

```
<!ENTITY hyphen"-" >
<!ENTITY minus "-" >
<!ENTITY deg "Grad" >
```

Codebeispiel 15: Beispiele für Entitätsdefinitionen

Sonderzeichen, die nicht im ASCII-Zeichensatz enthalten sind, wie er für die Kodierung einer DTD i. d. R. verwendet wird, werden durch Entitäten eingeführt, wie dies z. B. für die Sonderzeichen des im Deutschen gilt:[57]

Definition einer Entität	*Verwendung*	*darzustellendes Zeichen*
<!ENTITY Auml CDATA "Ä" -- latin capital letter A with diaeresis, U+00C4 ISOlat1 -->	Ä	Ä
<!ENTITY auml CDATA "ä" -- latin small letter a with diaeresis, U+00E4 ISOlat1 -->	ä	ä
<!ENTITY Ouml CDATA "Ö" -- latin capital letter O with diaeresis, U+00D6 ISOlat1 -->	Ö	Ö
<!ENTITY ouml CDATA "ö" -- latin small letter o with diaeresis, U+00F6 ISOlat1 -->	ö	ö
<!ENTITY Uuml CDATA "Ü" -- latin capital letter U with diaeresis, U+00DC ISOlat1 -->	Ü	Ü
<!ENTITY uuml CDATA "ü" -- latin small letter u with diaeresis, U+00FC ISOlat1 -->	ü	ü
<!ENTITY szlig CDATA "ß" -- latin small letter sharp s = ess-zed, U+00DF ISOlat1 -->	ß	ß

Tabelle 17: Zeichenentitäten für die wichtigsten „Sonderzeichen“ des Deutschen

Die Referenzierung einer Entität in einem SGML-Dokument erfolgt durch Verwendung eines vorangestellten &-Operators, dem Bezeichner für die Entität und einer Abschlusskennzeichnung (;): &Entitätsname;, z. B.

```
Universit&auml;t oder Universit&auml t <!-- Beenden durch Blank; möglich, aber ungebräuchlich -->
```

Codebeispiel 16: Beispiele für die Verwendung von Entitäten

Die Definition von Entitäten für geschützte SGML-Zeichen ist notwendig, wenn diese Zeichen selbst dargestellt werden müssen:

```
<!ENTITY lt    CDATA “<“ > <!—“less than” -->
<!ENTITY gt    CDATA “>“ > <!—“greater than” -->
<!ENTITY amp  CDATA “&  > <!—“ampersand ” -->
```

Codebeispiel 17: Geschützte SGML-Zeichen als character entities

Die Anwendung von Entitäten für Textersetzungen kann bis zur Ersetzung des Entitätsbezeichners durch ganze Dokumente führen, d. h. man kann eine externe SGML-Datei über eine Entität referenzieren und ins Dokument einbinden. Im folgenden Ausschnitt aus einer DTD wird eine Entität definiert, die auf eine externe Datei verweist; diese Datei wird im Hauptdokument durch Verwendung der Entität eingebunden.

```
<!DOCTYPE bericht SYSTEM
  [
    <!-- ... -->
    <!ENTITY kap31 SYSTEM “/einVerzeichnis/Entitaeten.doc“>
    <!-- ... -->
  ]
<!-- Anwendung im SGML-Dokument: -->
<bericht>
&kap31; <!-- Datei wird eingebunden -->
```

Codebeispiel 18: Einbindung externer Ressourcen durch Entitäten

Ist die Definition einer Entität zusätzlich durch das Schlüsselwort SUBDOC gekennzeichnet, so können die externen Dateien jeweils eine eigene DTD verwenden; damit ist eine einfache Möglichkeit für die Modularisierung von Markup gegeben:[58]

[57] Entitätsdefinitionen aus dem *character entity set* für den ISO Latin 1-Zeichensatz, der Bestandteil des SGML-Standards ist und in dieser Form auch für die HTML-DTD gilt.

```
<!DOCTYPE bericht SYSTEM
  [
      <!ENTITY kap31 SYSTEM “C:\kurse\EPV\test.doc“ SUBDOC>
    <!-- ... -->
  ]
>
```

Codebeispiel 19: Externe Entitäten als Subdokumente mit eigener DTD

Ein zweiter Anwendungsbereich für Entitäten sind Ersetzungsmechanismen innerhalb der DTD. Elemente lassen sich über sog. Parameterentitäten zusammenfassen und gruppieren; auf diese Weise kann man sog. *floating elements* definieren, d. h. Strukturbestandteile, die frei innerhalb des Dokuments auftreten können (z. B. Fußnoten, Bilder, Anker etc.), aber nicht in allen betroffenen Inhaltsmodellen (Elementdefinitionen) neu spezifiziert werden sollen. Syntaktisch unterscheidet sich die Definition einer Parameterentität von einfachen dokumentbezogenen durch die Verwendung eines %-Symbols: <!ENTITY % bezeichner bezeichner_der_referenzierten_Elemente>. Es wird nicht der Entitätsname durch das referenzierte Objekt (Zeichen, Datei etc.), sondern der Entitätsname durch ein SGML-Element ersetzt. Die Anwendung einer Parameterentität erfolgt ebenfalls unter Voranstellung eines %-Symbols: %Entitätsname; Beispiele aus der HTML-4-DTD sollen dies verdeutlichen: Ein Dokument ist aus Kopf (<head>) und Körper (<body>)aufgebaut; diese Struktur wird über die Parameterentität html.content definiert. Der Körper eines HTML-Dokuments enthält Elemente auf der Ebene der Zeichenkodierung (z. B. Textmarkup wie Fettdruck oder Kursivsatz) und blockartige Bestandteile (Absätze, Tabellen, Formulare etc.); die vielfältigen Ausprägungen dieser beiden Inhaltstypen werden über zwei Parameterentitäten unterschieden (%inline und %block), die weitere Parameterentitäten enthalten. %inline wird in den einzelnen Elementdefinitionen für die Inhaltsdefinition eingesetzt (s. o. Codebeispiel 12).

```
<!ENTITY % html.content "HEAD, BODY">
<!ELEMENT  HTML O O (%html.content;)>
<!ENTITY % block  "P | %heading; | %list; | %preformatted; | DL | DIV | NOSCRIPT |
                   BLOCKQUOTE | FORM | HR | TABLE | FIELDSET | ADDRESS">
<!ENTITY % inline "#PCDATA | %fontstyle; | %phrase; | %special; | %formctrl;">
<!ELEMENT  BODY O O (%block;|SCRIPT)+ +(INS|DEL) -- document body -->
```

Codebeispiel 20: Unterscheidung elementarer Inhaltsmodelle in HTML 4

5.1.1.4 Dokumentenauszeichnung

Ein abschließendes Beispiel soll die Bestandteile einer SGML-DTD und ihre Anwendung für die Dokumentaufbereitung im Zusammenhang zeigen: Zunächst definiert eine einfache DTD die Bestandteile eines Berichts. Die Dokumentausprägung (Dokumentinstanz) zeigt die Anwendung des SGML-Markup:

```
<!-- ein SGML-Beispiel -->
<!DOCTYPE bericht
  [
    <!ELEMENT bericht - -%bericht.inhalt>
    <!ENTITY bericht.inhalt (titel, vorwort?, kapitel+,register?)>
    <!ELEMENT (titel|satz|registereintrag) – O #PCDATA)>
```

[58] Dies setzt allerdings voraus, dass in der FEATURES-Sektion der *SGML declaration* der Eintrag SUBDOC YES vorhanden ist.

```
    <!ELEMENT kapitel - - (titel, absatz+)>
    <!ELEMENT absatz - - (satz)+)>
    <!ELEMENT register - - (titel, registereintrag+)>
    <!ATTLIST (bericht|kapitel)
                sprache  (DE|EN|FR)   DE
                bezeichner ID         #REQUIRED
                verweis  IDREFS       #IMPLIED>
  ]
>
```

Codebeispiel 21: Beispiel-DTD für einen Bericht

```
<bericht sprache=DE ID=bericht2459>
  <titel>Beispiel einer SGML-Dokumentauspr&auml;gung</titel>
  <kapitel sprache=DE ID=kap1>
    <titel>Einf&uumlrung<titel>
    <absatz>
      <satz>Erster Satz
      <satz>Zweiter Satz ...
    </absatz>
  </kapitel>
  <kapitel sprache=EN ID=kap2>
    <titel>Kapitel 2
    <absatz>
      <satz>Erster Satz in kap. 2.
    </absatz>
  </kapitel>
  <!-- ... -->
</bericht>
```

Codebeispiel 22: Beispiel einer Dokumentinstanz

5.1.2 Modularisierung durch Kataloge

Eine Möglichkeit der Modularisierung ist die Verwendung von SGML-Katalogen (*catalogs*), die eine Abbildung zwischen SGML-Bezeichnern (*public identifier*) und DTD-Modulen herstellen (vgl. MALER & EL ANDALOUSSI 1996: 285). Sie werden z. B. in der DocBook-DTD verwendet, um den modularen Aufbau dieser umfangreichen Spezifikation zu bewerkstelligen (vgl. MALER & ALLEN 1997A:7 ff. und unten Kap. 6.5). Der nachfolgende Ausschnitt aus dem DocBook *catalog file* zeigt die Abbildung von Bezeichner auf Dateinamen, in denen sich dann die SGML-Definitionen befinden (Elemente, Entitäten). Durch den Katalog können in den einzelnen DTD-Modulen logische Bezeichner verwendet werden, deren Abbildung auf das Dateisystem in der Katalogdatei leicht angepasst werden kann.

```
<!- Katalogdaten für DocBook V3.0 -->
<!-- DocBook driver file -->
PUBLIC "-//Davenport//DTD DocBook V3.0//EN" "dbook30.dtd"
<!-- Module von DocBook -->
PUBLIC "-//USA-DOD//DTD Table Model 951010//EN" "cals-tbl.dtd"
PUBLIC "-//Davenport//ELEMENTS DocBook Information Pool V3.0//EN" "dbpool.mod"
PUBLIC "-//Davenport//ELEMENTS DocBook Document Hierarchy V3.0//EN" "dbhier.mod"
PUBLIC "-//Davenport//ENTITIES DocBook Additional General Entities V3.0//EN" "dbgenent.mod"
<!-- Entitätsdefinitionen nach ISO (Zeichensätze etc.) -->
PUBLIC "ISO 8879:1986//ENTITIES Publishing//EN" ".\entities\isopub.ent"
PUBLIC "ISO 8879:1986//ENTITIES General Technical//EN" ".\entities\isotech.ent"
```

```
PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN" ".\entities\isolat1.ent"
PUBLIC "ISO 8879:1986//ENTITIES Added Latin 2//EN" ".\entities\isolat2.ent"
```

Codebeispiel 23: Ausschnitt aus dem SGML-Katalog für DocBook 3

Ein analoger Vorschlag für die Modularisierung von XML, *XML Catalog*, liegt bisher nur in der Form eines nicht-standardisierten Entwurfs vor, wird aber bereits von einigen XML-Werkzeugen unterstützt (vgl. BRADLEY 2000: 131 ff. und *http://www.ccil.org/ ~cowan/ XML/XCatalog.html*).

5.1.3 Sekundäre Strukturierung durch Architekturmuster

Seit der Festlegung von SGML als internationalem Standard hat es einige Weiterentwicklungen gegeben, die vor allem die Modularisierung des Entwurfs von DTDs sowie deren dynamische Anpassung betreffen: Dazu gehören *architectural forms* (vgl. MALER & EL ANDALOUSSI 1996: 293 f., Lobin 2000: 85 ff.; siehe unten Kap. 8.1 zu HyTime), die Regeln für Aufbau und Inhalt von Elementdeklarationen definieren. Als Teil der *SGML Extended Facilities*, die im HyTime-Standard definiert sind (*Architectural Form Definition Requirements* in Annex 3 des Standards, vgl. ISO/IEC 10744:1997), sind sie offizieller Teil des SGML-Standards. Mit ihrer Hilfe besteht die Möglichkeit einer sekundären Kontrolle für SGML-Dokumentgrammatiken, d. h. man hat durch die Definition einer sekundären DTD eine strukturelle Kontrolle über die primäre DTD (vgl. KIMBER 1997 und LOBIN 2000: 85 ff.).

Eine solche Vorgehensweise ist folgendermaßen begründet: Eines der Grundanliegen eines deklarativen Markupstandards wie SGML ist es, für unterschiedliche Anwendungen Strukturmuster zu definieren. Die *Spezialisierung* von Dokumenttypen und Dokumenttypdefinitionen kann dem Ziel der Wiederverwendbarkeit durch einheitliche Auszeichnungsstandards zuwiderlaufen. Es ist wünschenswert, sekundäre Kontrollmechanismen zur Verfügung zu haben, die zwar nach wie vor die Spezialisierung von Dokumenttypen zulassen, aber gleichzeitig die Kontrolle unterschiedlicher „Anwendungs-DTDs“ durch eine Architektur-DTD, d. h. durch *architectural forms*, ermöglichen.[59]

Der Entwurf einer Architektur-DTD erfolgt nach denselben syntaktischen Regeln wieder einer einfachen DTD. Zur Kontrolle von Dokumentinstanzen ist allerdings eine zusätzliche Referenz auf das *architectural form* erforderlich, damit die Architektur-DTD zur Dokumentvalidierung herangezogen werden kann. Zu den Anwendungen der *architectural forms* zählen

- die kontrollierte Migration von SGML-kodierten Daten in XML-Formate,
- die Verwendung von Untermengen umfangreicher DTDs (wie HTML, TEI oder CALS, vgl. unten Kap. 6) für spezielle Anwendungen und
- die Spezialisierung von Dokumententypbeschreibungen, wobei die *architectural forms* die Aufgabe übernehmen, die Konformität mit einer Ausgangs-DTD zu gewährleisten (vgl. SIMONS 1998: Kap. 4).

Die Verwendung von *architectural forms* erfolgt durch folgende Schritte:

---

[59] Wie KIMBER 1997: Kap. 1 ausführt, handelt es sich bei Architekturmuster nicht um eine *Meta*-DTD, die andere DTDs strukturiert, sondern um „a set of rules that govern a class or superclass of documents“, der Bezug liegt also bei den Dokumenten, nicht einer DTD, vgl. aber LOBIN 2000: 86 ff., der am Begriff der Meta-DTD festhält.

- Eine DTD, die zur sekundären Strukturierung verwendet werden soll, muss vorliegen oder definiert werden.
- In der DTD, die durch das *architectural form* kontrolliert werden soll, muss dieses durch entsprechende Vereinbarungen bekannt gegeben werden (durch eine NOTATION-Vereinbarung in SGML, durch eine Verarbeitungsanweisung in XML).
- Einzelnen Elementen bzw. Attributen können nun kontrollierende Entsprechungen aus dem *architectural form* zugewiesen werden. Beispiele für diesen Prozess finden sich bei KIMBER 1997, SIMONS 1998 und LOBIN 2000: 86 ff.

5.1.4 Verarbeitung von SGML

Nach der Einführung der Grundkonzepte von SGML, ist zu fragen, auf welche Weise SGML-kodierte Dokumente verarbeitet werden können. Es ist zu unterscheiden, ob ein SGML-Werkzeug

- beliebige SGML-DTDs verarbeiten können soll oder
- an die spezifische Struktur einer einzelnen DTD angepasst ist.

Letzeres verringert die Komplexität der Verarbeitung erheblich. Schon zu Beginn der Entwicklung von SGML hat sich eine Softwareinfrastruktur herausgebildet,[60] die eher an einzelne DTDs angepasst ist als an die generische Verarbeitung. Die Webbrowser, die gewissermaßen ein Analyse- und Präsentationssystem für die HTML-DTD darstellen, sind das bekannteste Beispiel für diese Entwicklung. Daneben existieren aber eine Reihe von generischen Werkzeugen für SGML, u. a.

- graphische Editoren für den DTD-Entwurf,
- Editoren, die den Benutzer beim Auszeichnen von SGML-Dokumenten unterstützen,
- Parser für SGML-kodierte Dokumente, die die syntaktische Korrektheit eines Dokuments durch Abgleich mit der verwendeten DTD prüfen und die Voraussetzung für die Interpretation der in einem SGML-Dokument enthaltenen Inhaltsstrukturen sind,
- Konvertierungswerkzeuge, die sowohl den Übergang von proprietären Formaten zu SGML als auch die Abbildung unterschiedlicher DTDs aufeinander unterstützen,
- SGML-orientierte Speicherungs- und Verwaltungssysteme (Redaktionssysteme, Datenbanken),
- SGML-Betrachtungssysteme, die die Darstellung von SGML-Dokumenten am Bildschirm erlauben und
- SGML-Formatierungswerkzeuge, die im Verlagswesen die Schnittstelle zwischen strukturierter Information (SGML) und der Druckausgabe bilden (als Teil der Druckvorstufe).[61]

In der Regel steht am Ende der Verarbeitung eines SGML-Dokuments ein proprietäres Format bzw. Softwarewerkzeug; das World Wide Web ist eine bemerkenswerte Ausnah-

---

[60] Vgl. MARCOUX & SÉVIGNY 1997: 591 und KRÜGER 1999, der einen Überblick zu bedeutenden SGML-Pilotprojekten und SGML-Werkzeugen gibt, sowie die Informationssammlung zu SGML/XML, die seit vielen Jahren von Robin COVER betreut wird und jetzt bei der *Organization for the Advancement of Structured Information Standards* (OASIS) angesiedelt ist (COVER 1999). Sie stellt die wohl umfangreichste Sammlung an SGML- und XML-Informationen dar.

[61] Dies erfolgt meist über proprietäre Formate, da sich für die entsprechenden Standards wie DSSSL (vgl. unten Kap. 7.1) noch keine breite Softwareinfrastruktur herausgebildet hat, vgl. SPENGLER 1999: 154.

me – allerdings mit der Einschränkung, dass bisher nur eine einzige DTD und vereinfachte Formatierungs- bzw. Präsentationsstandards verwendet werden.[62] Den Verarbeitungsablauf macht Abbildung 26 deutlich:

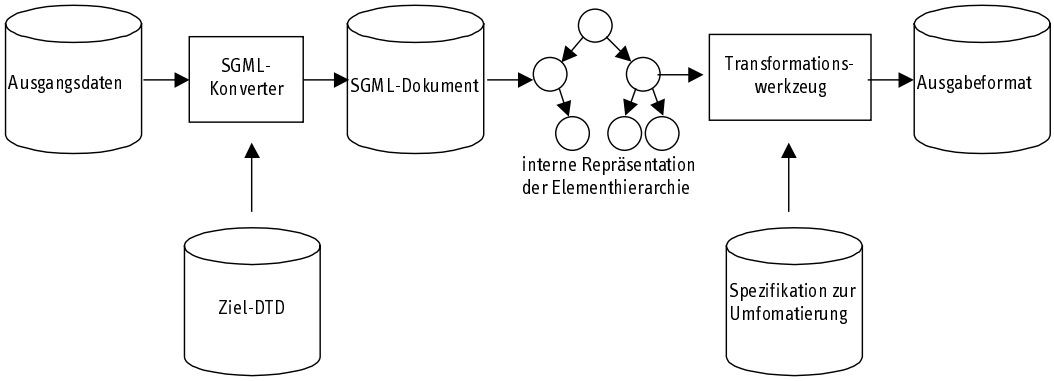

Abbildung 26: Prozesskette bei der Verarbeitung von SGML-Dokumenten

Mit Blick auf das Referenzprojekts kann man diese Prozesskette wie folgt instantiieren:

- Das proprietäre Ausgangsformats wird mit Hilfe von Konvertierungswerkzeugen auf die Ziel-DTD abgebildet (HTML/XHTML/abgeleitete XML-DTD),
- die so entstandenen SGML-/XML-Dokumente werden weiterverarbeitet (Anreicherung durch Hypertextverknüpfungen, Einbettung von Multimediakomponenten, Korrekturen etc.) und
- unter Berücksichtigung von Formatierungsvorschriften wird schließlich die Präsentation erstellt (Webbrowser).

5.1.5 Einordnung und Alternativen

Die Bewertung von SGML in Bezug auf seine Eignung für die Erstellung und Aufbereitung dynamischer elektronischer Bücher muss unter Beachtung folgender Aspekte erfolgen:

- Der prinzipiellen Mächtigkeit von SGML für die deklarative Strukturierung von Information,
- der Rolle von SGML als Ausgangspunkt einer Familie von Kodierungsstandards,
- der praktischen Anwendbarkeit im Vergleich mit seinen Alternativen und
- der Bewertung im Vergleich mit alternativen Lösungen.

Die Möglichkeiten der Definition von Strukturierungsformaten, wie sie in SGML gegeben ist, eignet sich grundsätzlich dafür, komplexe Strukturen, wie sie in dynamischen elektronischen Büchern vorliegen, aufzubereiten. Die Herkunft von SGML aus der technischen Dokumentation bzw. dem Verlagswesen mit Orientierung an textueller Information schränkt dies zwar ein, da nicht-textuelle Daten nicht direkt mit SGML ausgezeichnet werden können und lediglich als externe Ressourcen wie black-box-Komponenten referenziert werden können; dieser Nachteil kann aber bei Verwendung von Standards, die von SGML abgeleitet sind und auf den gleichen Grundprinzipien aufbauen, ausgeglichen werden (HyTime; SMIL, s. u. Kap. 8). Gleiches gilt für die Frage der Präsentation bzw.

[62] Vgl. FAUSEY & SHAFER 1997: 637 ff., die verschiedene Verarbeitungsmodelle für SGML vorstellen und mit der standardisierten Verarbeitung mit Hilfe von DSSSL vergleichen.

des Layouts – SGML soll die *logische Struktur* aufbereiten und keine Aussage über Gestaltung, Präsentation oder Bedeutung der kodierten Information machen. Diese zusätzlichen Markupebenen können allerdings durch die SGML-basierten Transformations- und Präsentationsstandards (DSSSL, CSS, XSL, s. u. Kap. 7) übernommen werden. Folgendes wird damit deutlich: Die Bedeutung von SGML für die Kodierung dynamischer elektronischer Bücher liegt weniger auf der unmittelbaren Anwendbarkeit von SGML selbst als vielmehr in der Definition grundlegender Prinzipien und Verfahren für die Strukturierung von Information. Die in SGML definierten Mechanismen werden in den zahlreichen, von SGML abgeleiteten Standards wiederverwendet.

Die praktische Anwendbarkeit ist vor allem durch die wenig weit verbreiteten und kostenintensiven Softwareprodukte für die Verarbeitung von SGML-Dokumenten bestimmt. Dies gilt weniger für Konvertierungs- und Analysewerkzeuge – eine Reihe von SGML-Parsing-Systemen ist frei verfügbar – als für die notwendigen Präsentationssysteme. Keines der weit verbreiteten Multimedia-Autorensysteme unterstützt SGML als *Meta*sprache. Da aber die elektronische Präsentationsmöglichkeit eine *conditio sine qua non* für das elektronische Buch ist, scheidet SGML für die unmittelbare Anwendung aus. Soweit entsprechende Transformationsprozesse z. B. von einer SGML-DTD in HTML oder eine XML-DTD mit beigeordnetem *style sheet* für die Layoutgestaltung definiert sind, müsste dieses Argument nicht für die Aufbereitungs-, Verwaltungs- und Speicherungsphase eines elektronischen Buchs gelten; wie die Diskussion von XML aber noch zeigen wird, existieren gute Argumente, XML den Vorzug zu geben.

Der letzte Gesichtspunkt, die Diskussion denkbarer Alternativen, betrifft sowohl die in den folgenden Kapiteln zu prüfenden Standards, die von SGML abgeleitet sind sowie konkrete Anwendungen von SGML, als auch standardisierte oder proprietäre Strukturierungsformate, die in keiner direkten Beziehung zu SGML stehen.

Soweit man Standards für die Strukturierung von Information betrachtet, bietet sich die *Open Document Architecture* (ODA, vgl. STEINMETZ 1999: 652, BORMANN & BORMANN 1991: 249 ff. und MARCOUX & SÉVIGNY 1997: 588 ff.) in Verbindung mit dem *Open Document Interchange Format* an, ein internationaler Standard (ISO 8613), der eine standardisierte Semantik zur Dokumentengestaltung und deren Bearbeitung definiert und stärker als SGML auf den Bürobereich ausgerichtet ist, da sich ODA aus Forschungsprojekten im Bereich der Bürokommunikation entwickelt hat, vgl. SEIDT 1994: 185f.

ODA deckt im Gegensatz zu SGML nicht nur den *Struktur*aspekt ab, sondern berücksichtigt auch das *Layout* der zu kodierenden Dokumente und spezifiziert einen Formatierungsprozess. ODA verfolgt einen deskriptiven und objekt-orientierten bzw. hierarchischen Ansatz und definiert generische und spezifische Strukturen sowie verschiedene Dokumentenklassen für die Informationsstrukturierung, ohne aber wie SGML das Markup über einfache Zeichenketten zu realisieren; ODA-Dokumente sind anders als SGML-Dokumente nicht unmittelbar lesbar. ODA definiert ein Prozessmodell für das Editieren, Formatieren sowie die Präsentation von Dokumenten und wird mit ODIF um ein *Austauschformat* zwischen verschiedenen (auch proprietären) Dokumentensystemen ergänzt.

Anders als ein SGML-Dokument, das nur in der Zusammenschau mit seiner DTD korrekt interpretiert werden kann, sind ODA-Dokumente selbstbeschreibend – die logische Struktur einer ODA-Dokumentklasse, die durch ein konkretes Dokument instantiiert wird, entspricht der Rolle der DTD für ein SGML-Dokument. Der wesentliche Unterschied zwischen SGML und ODA liegt in der unterschiedlichen Sichtweise: Bei ODA steht der Verarbeitungsprozess im Mittelpunkt, während SGML den Strukturaspekt betont. Grundsätzlich ist aber eine Umwandlung von SGML- in ODA-Dokumentformate

und umgekehrt möglich; dafür existiert mit der *Office Document Language* (*ODL*) ein standardisiertes Format zur Darstellung von ODA-Dokumenten in SGML (vgl. NICHOLAS & WELSCH 1992: 115 ff.).

Weitere Alternativen zu SGML als Metasprache für die Informationsstrukturierung kann man für einzelne Prozessstufen des elektronischen Publizierens aufzeigen, aber nicht für die Gesamtheit der notwendigen Verarbeitungsschritte bei der Erstellung einer elektronischen Publikation:

- In den meisten kommerziellen Textverarbeitungssystemen lassen sich durch Formatvorlagen, *style sheets* etc. generische Strukturierungsmerkmale einführen, die in etwa dieselbe Funktion wie das deklarative Markup in SGML übernehmen können und im Vorfeld der SGML-Aufbereitung weit verbreitet sind.[63]
- Druckorientierte Präsentationsformate wie Postscript oder das *Portable Document Format* eigen sich zwar zur Aufbereitung unterschiedlichster Dokumentstrukturen und umfassen (insb. PDF) Erweiterungen für die elektronische Nutzung (Hypertextverknüpfungen, Schnittstellen zu externen Programmen), sind aber durch die Ausrichtung am Druckformat und das fehlende deklarative Dokumentmarkup nur bedingt geeignet für die Verwendung als Kodierungsbasis elektronischer Bücher (vgl. MERZ 1997A, 1997B: 8, 207 ff.).
- Gerade für den naturwissenschaftlichen Fachtext weit verbreitete Satzsysteme wie TEX (vgl. KNUTH 1986 und unten Kap. 6.6) lassen zwar die Definition neuer Markupelemente zu und verfügen über plattformneutrale Ausgabeformate; es fehlt aber an wesentlichen Funktionen für die Nutzung im elektronischen Medium wie Hypertextverknüpfungen. Sie sind daher eher als Ausgangspunkt mit anschließender Überführung in SGML-basierte Kodierungsformate denkbar.

Diese kurze Diskussion des Umfelds von SGML soll nun durch die Einführung von XML weitergeführt werden; die grundlegenden Eigenschaften von SGML wie oben dargelegt werden vorausgesetzt.

5.2 Extensible Markup Language

Die Weiterentwicklung von SGML zur *Extensible Markup Language* (vgl. BRAY, PAOLI & SPERBERG-MCQUEEN 1998, BRADLEY 2000) unter Koordination des *World Wide Web Consortium* hat das Ziel, die Mächtigkeit von SGML als Metasyntax für Dokumentenstrukturbeschreibungen für das World Wide Web zu operationalisieren. XML bietet dem Autor eines elektronischen Buchs den aus SGML bekannten Vorteil, unterschiedliche Dokumententypbeschreibungen nach Bedarf erstellen und die Auszeichnungsmarken der Struktur seiner Inhalte anpassen zu können. XML ist somit ebenfalls eine Metasprache und weist gegenüber SGML vor allem syntaktische Vereinfachungen auf, um die Entwicklung von Parsern und Formatierern zu erleichtern. XML ist auf der gleichen logischen Ebene wie SGML verankert – als *Meta*syntax für die Entwicklung unterschiedlicher Strukturbeschreibungen; es ist keine „Anwendung“ eines Standards in einem exakt

[63] Es gibt z. B. für *Microsoft WinWord* oder *Adobe FrameMaker* Konverter, die eine Umwandlung in SGML auf der Basis der in den Dokumenten verwendeten Formatvorlagen vornehmen; ähnlich arbeiten HTML-Konverter in Textverarbeitungsprogrammen. Für den Autor, der entweder an die Verwendung eines bestimmten Systems gewöhnt ist oder keine SGML-Editoren zur Hand hat, dürfte dies bei der Textproduktion für dynamische elektronischer Bücher der einfachste Weg sein.

definierten Format, wie dies für HTML gilt. XML wird durch einige Substandards ergänzt, die seine Flexibilität auf die Aspekte

- Verknüpfung (*Extensible Link Language*, XLink),
- Adressierung (XPointer, XPath),
- Gestaltung (*Extensible Style Language*, XSL) und
- Datenschemata und Strukturierungsmechanismen als Alternative zum DTD-Entwurf (XML-Data, *Document Content Description*, XML-Schemata)

anwenden. BRAY, PAOLI & SPERBERG-MCQUEEN 1998: Kap. 1.1 – *Origin and Goals* definieren als Ziele für XML:

- Nutzbarkeit im Internet bzw. World Wide Web,
- Kompatibilität zu SGML,
- einfache Programmierung (im Vergleich zu SGML),
- keine optionalen Features wie in SGML[64],
- klare Strukturierung von Dokumenten durch "lesbares" bzw. vom Benutzer interpretierbares Markup,
- präzise und formale Definition der Metasprache und
- eine Vielzahl von Anwendungen (*electronic publishing*, *electronic commerce*, WWW-Informationssysteme etc.).

Zu den grundsätzlichen Modifikationen von XML gegenüber SGML zählt die Unterscheidung zwischen *validen* und *wohlgeformten* Dokumenten (*valid* vs. *well-formed documents*). Das Konzept wohlgeformter Dokumente verlangt für eine XML-Dokumentinstanz lediglich die *syntaktische Korrektheit* im Sinne der allgemeinen XML-Syntax sowie die vollständige Deklaration aller Entitäten. Für die Überprüfung der Wohlgeformtheit ist daher keine DTD erforderlich. Ein *valides* Dokument liegt erst vor, wenn für das Dokument eine zugehörige DTD existiert und das Dokument unter Anwendung dieser DTD korrekt geparst werden kann. Die Einführung wohlgeformter Dokumente soll die Verarbeitung erleichtern, da bei vielen Anwendungen ein Parsen anhand einer DTD nicht erforderlich ist und somit zusätzlicher Verarbeitungsaufwand vermeiden werden kann. Für die Regeln der Syntax sind demgemäss *validity constraints* bzw. *wellformedness constraints* definiert, die festlegen, was die jeweiligen syntaktischen Anforderungen an wohlgeformte und valide Dokumente sind. Eine Übersicht der Unterschiede zwischen SGML und XML geben BRAY, PAOLI & SPERBERG-MCQUEEN 1997 und CLARK 1997.

5.2.1 Syntax und Aufbau von DTDs

XML sieht wie SGML die Unterscheidung zwischen logischer und physischer Struktur vor und arbeitet weitgehend mit denselben syntaktischen Mitteln. Dies bedeutet, dass wie in SGML für unterschiedliche Dokumententypen DTDs definiert werden können, die sich aus Deklarationen von Elementen, Attributen, Attributlisten, Entitäten, Kommentaren und Verarbeitungsvorschriften (*processing instructions*, PI) zusammensetzen. Die Physikalische Struktur besteht aus der in einer konkreten Dokumenteninstanz vorhandenen Mischung textueller und binärer Daten. Das Schema des Dokumentaufbaus in XML sieht wie folgt aus:

---

[64] Dies bezieht sich vor allem auf weniger häufig verwendete Sondermerkmale von SGML, die hier nicht dargestellt werden, deren Mächtigkeit aber die Verarbeitung von SGML-Dokumenten erschwert.

```
Dokument ::= Prolog Element MixedContent*
```

Codebeispiel 24: Schema des Dokumentaufbaus in XML

Jedes XML-Dokument besteht aus einem Prolog, einem Element und beliebig vielen Bestandteilen, die Elemente sein können oder direkt Daten enthalten. Der Prolog eines XML-Elements besteht aus einer optionalen XML-Deklaration, einer optionalen DOCTYPE-Deklaration, und beliebig vielen Kommentaren und Verarbeitungsvorschriften:

```
[XML-Deklaration] [DOCTYPE-Deklaration] (Kommentare | Processing Instructions | Whitespace)*
```

Codebeispiel 25: Aufbau des Prologs eines XML-Dokuments

Anders als in SGML kann wenigstens bei lediglich wohlgeformten Dokumenten die DOCTYPE-Angabe entfallen, da für die Prüfung auf Wohlgeformtheit die DTD nicht bekannt sein muss, der Parser nur die Konformität mit der generischen XML-Syntax prüft. Die nachfolgenden Beispiele zeigen den Unterschied zwischen validen und wohlgeformten Dokumenten:

```
<?XML version="1.0" --XML-Deklaration-->
<BEGRUESSUNG>Hallo, Welt!</BEGRUESSUNG>
```

Codebeispiel 26: Ein einfaches wohlgeformtes XML-Dokument ohne DTD

Diese Dokumentausprägung ist zwar nach der XML-Syntax wohlgeformt, nicht aber valide, da keine DTD im Dokument enthalten ist oder vom Dokument referenziert wird. Für ein *valides* Dokument gelten (u. a.) folgende einfache Gültigkeitsbeschränkungen:

- Der Name der DTD muss dem Wurzelelement entsprechen (vgl. SGML) und
- die interne oder externe DTD darf nicht leer sein.

Um obiges Dokument gültig werden zu lassen, muss eine DTD spezifiziert werden, die entweder – wie bei SGML üblich – in einer externen Ressource vorliegt (externe DTD) und über die DOCTYPE-Angabe im Dokument direkt oder indirekt über einen *public identifier* referenziert wird oder als interne DTD im XML-Dokument selbst enthalten ist:

```
<?XML version="1.0" ?>
<!DOCTYPE BEGRUESSUNG SYSTEM "hallo.dtd">
<BEGRUESSUNG>Hallo, Welt!</BEGRUESSUNG>
```

Codebeispiel 27: Valides XML-Dokument mit externer DTD

```
<?XML version="1.0" encoding="UTF-8" ?>
<!DOCTYPE BEGRUESSUNG
  [
    <!ELEMENT BEGRUESSUNG (#PCDATA)>
  ]
>
<BEGRUESSUNG>Hallo, Welt!</BEGRUESSUNG>
```

Codebeispiel 28: Valides XML-Dokument mit interner DTD

Die XML-Deklaration ist ein Beispiel der Verwendung von Verarbeitungsanweisungen (*processing instructions*, Syntax: <? PI-Ziel Anweisung ?>). Im Prinzip kann unter Angabe eines geeigneten Ziels und einer passenden Anweisung jede beliebige Funktion des verarbeitenden Programms ausgelöst werden (Beispiel: *Server Side Includes* auf einem Webserver). Grundsätzlich sollte man aber von PIs zurückhaltend Gebrauch machen, da sie die konzeptuelle Trennung von Inhaltsaufbereitung und Verarbeitung aufheben. Vie-

les, was in einer Verarbeitungsanweisung kodiert werden kann, lässt sich durch einen DSSSL-Prozessor bzw. die Verarbeitung von *style sheets* (CSS2/XSL) regeln.

5.2.2 Verwendung von Elementen

Bei der Verwendung der Elemente weist XML einen wesentlichen Unterschied zu SGML auf: Die Regeln zur Optionalität von Start- und/oder Endmarken sind eingeschränkt, zu jedem nicht-leeren Element (d. h. für alle Elemente, die zwischen Start- und Endmarke Inhalt kodieren) muss sowohl eine Start- als auch eine Endmarke vorhanden sein, d. h. *tag omission* ist nicht zulässig.

Elemente, die zwischen Start- und Endmarke keinen weiteren Inhalt aufweisen, nennt man leere Elemente. Für sie muss in XML eine einzelne *empty element*-Marke verwendet werden; sie ist neben Start- und Endmarken ein dritter Typus von Elementkennzeichnungen (gewissermaßen eine Zusammenziehung von Start- und Endmarke zu einem Element):

```
<NICHTLEERESELEMENT> Inhalt …</NICHTLEERESELEMENT>
<-- empty element-Marke: -->
<LEERESELEMENT att1="abc" att2="def"/>
```

Codebeispiel 29: Verwendung von Elementmarken in XML

Diese syntaktische Einschränkung vereinfacht das Parsen von Dokumenten und erhöht die Lesbarkeit der Dokumente. Zusätzlich verlangt die Syntax die vollständige Inklusion von Elementen, d. h. ein Element, dessen Endmarke innerhalb des Inhalts eines anderen Elements steht, muss auch seine Endmarke vor der Endmarke des umgebenden Elements aufweisen. Dies ist eine weitere Vereinfachung gegenüber der freieren Syntax von SGML. Im folgenden Beispiel ist die erste Kodierungsvariante, die in SGML möglich wäre, als XML-Markup ungültig:

```
<!-- ungültig:
            <A>… Inhalt in A… <B> … Inhalt in A, B … </A> … Inhalt in B … </B>
   gültig: -->
            <A>… Inhalt in A … <B> … Inhalt in A, B …</B> </A> <B>… Inhalt in B … </B>
```

Codebeispiel 30: In XML ungültige Überkreuzung von Elementmarken

Diese Eingrenzung der formalen Mächtigkeit von XML entspricht der Beschränkung auf kontextfreie Grammatiken und hat zur Folge, dass sich die Elementstruktur von XML-Dokumenten als Baum darstellen lässt.[65] Hinsichtlich der praktischen Anwendung von XML und der Entwicklung XML-basierter Analysewerkzeuge ist damit eine entscheidende Vereinfachung verbunden.[66]

Auf der Ebene der Attribute verlangt XML anders als SGML die Einfassung von Attributwerten durch Trennzeichen (‘ oder “). Von diesen einfachen, aber für die Komplexitätsreduktion entscheidenden Vereinfachungen abgesehen, ist die XML-Syntax für den Aufbau einer DTD weitgehend mit der SGML identisch (Kompatibilitätsaspekt), XML ist

---

[65] Vgl. BRAY, PAOLI & SPERBERG-MCQUEEN 1998: Appendix E, WALSH 1997: 106 f. und die Arbeiten von BRÜGGEMANN-KLEIN (BRÜGGEMANN-KLEIN 1993A, 1993B)

[66] Dies hat dazu geführt, dass sich XML-Werkzeuge deutlich schneller und in größerer Vielfalt etablieren konnten, als dies für SGML der Fall war, vgl. dazu die Übersicht bei COVER 1999 und unten Kap. 5.2.7.

insgesamt eine vereinfachte Version von SGML. Dies gilt auch für die Gliederung von DTDs mit Hilfe von Parameterentitäten, Beispiel:

```
<!ENTITY % block "absatz | listing | rahmen">
<!ELEMENT  bericht          (kapitel+)>
<!ELEMENT  kapitel          (ueberschrift, (%block;)*, abschnitt+)>
<!ELEMENT  abschnitt        (ueberschrift,(%block;)*,unterabschnitt*)>
<!ELEMENT  unterabschnitt   (ueberschrift, (%block;)+)>
```

Codebeispiel 31: Elementdefinitionen und Parameterentitäten in XML

Wie in SGML lassen sich über externe Parameterentitäten zusätzliche DTDs in ein Dokument einbinden. Mit Hilfe von Parameterentitäten können bedingte Abschnitte einer DTD verwendet oder ignoriert werden (sog. *marked sections*). Ein Anwendungsfall ist das Einbinden von Kommentaren in der Entwurfsphase eines Dokuments (Korrekturen, Hinweise etc.), die für die abschließende Produktion z. B. einer Druckvorlage in der Endphase vom XML-Parser ignoriert werden sollen.

Abschließend soll ein kleines Beispiel einer XML-DTD Unterschiede zu SGML verdeutlichen. Es handelt sich um eine DTD, die im Rahmen eines elektronischen Wahlverfahrens (vgl. WOLFF 1998, WOLFF 1999B) entwickelt wurde und die der Kodierung von Wahlentscheidungen dient. In den folgenden beiden Codebeispielen werden erst die DTD und dann ihre Anwendung gezeigt.

```
<?xml encoding="UTF-8"?>
<!ELEMENT VOTE (DECISION)+>
<!ELEMENT DECISION EMPTY>
<!ATTLIST DECISION  type       (selection|specification)  #REQUIRED
                    name       CDATA                      #REQUIRED
                    role       CDATA                      #REQUIRED
                    selection  CDATA                      #REQUIRED>
```

Codebeispiel 32: XML-Beispiel-DTD

```
<!DOCTYPE VOTE SYSTEM "vote.dtd">
<VOTE>
  <DECISION type="selection" name="Fritz M&uuml;ller" role="Vorsitzender" selection="JA"/>
  <DECISION type="selection" name="Helga Meier" role="Beigeordnte" selection="JA"/>
  <DECISION type="selection" name="Maria Schmidt" role="Schatzmeister" selection="Enthaltung"/>
  <DECISION type="selection" name="Egon Schulz" role="Schriftführer" selection="NEIN"/>
</VOTE>
```

Codebeispiel 33: Anwendung der XML-Beispiel-DTD

Obwohl XML ein noch junger Standard ist – die ersten Entwürfe stammen aus dem Jahr 1997, die Anerkennung als Empfehlung des *World Wide Web Consortium* 1998 – hat sich rasch nicht nur eine breite Palette an Softwarewerkzeugen, sondern auch eine Vielzahl an abgeleiteten Standards herausgebildet. Diese kann man in zwei Kategorien einteilen:

- Standards, die die generische Funktionalität von XML erweitern wie die in den folgenden Abschnitten diskutierten Konzepte für die Einführung von Namensräumen für XML-Elemente, die Hypertext-Relationierung und die Adressierung von XML-Dokumenten oder die Layoutgestaltung und
- anwendungsorientierte Standards, die XML für ein bestimmtes Strukturierungsproblem wie den mathematischen Formelsatz (MathML) oder die Kodierung elektronischer Bücher nutzen (OEB, XHTML).

Bevor die Verknüpfung von Information mit Hilfe von XLink und die Adressierung von Elementen in XML näher betrachtet wird, sollen XML-Namensräume eingeführt werden.

5.2.3 Namensräume in XML

Da XML dazu dienen soll, einen gegebenen Informationsbestand in einer Reihe von Schritten und durch heterogene Softwaremodule mit Information anzureichern (wobei unter „Information" im weiteren Sinn nicht nur die Attributwerte und Elementinhalte der Bestandteile einer DTD zu verstehen sind, sondern auch die An- oder Abwesenheit dieser Elemente selbst), ergibt sich das Problem der eindeutigen Resolution bzw. Zuordnung von Attributnamen und Elementen zu unterschiedlichen DTDs. Da nicht auszuschließen ist, dass unterschiedliche Entwickler für unterschiedliche Probleme denselben Attribut- oder Elementnamen verwenden, ist ein geeigneter Resolutionsmechanismus erforderlich. Ein geläufiges Beispiel ist die Verwendung bibliographischer Angaben wie z. B. *Autor* oder *Titel*, die sowohl auf (globaler) Dokumentebene als auch bei nachträglichen Ergänzungen für einzelne Dokumentteile auftreten können oder jeweils der Meta- bzw. „Standardinformationsebene" zuzuordnen sind, wenn sowohl die Autoren einer bibliographischen Liste als auch der Autor der Liste selbst kodiert werden sollen.

Zur Lösung dieses Problems ist vom *World Wide Web Consortium* eine Standardisierungsempfehlung entwickelt worden (vgl. BRAY, HOLLANDER & LAYMAN 1999). Sie regelt die Zuordnung von in einem Dokument mit „gemischtem Markup" verwendeten XML-Elementen zu unterschiedlichen Namensräumen. Ein XML-Namensraum ist wie folgt definiert:

> An **XML namespace** is a collection of names, identified by a URI, which are used in XML documents as element types and attribute names. XML namespaces differ from the „namespaces" conventionally used in computing disciplines in that the XML version has internal structure and is not, mathematically speaking, a set. (BRAY, HOLLANDER & LAYMAN 1999: Kap. 1).

Um in einem XML-Dokument einen Namensraum zu deklarieren und einem Element zuzuordnen, wird der Name des Namensraums in Verbindung mit dem Bezeichner xmlns als Attribut der Bezugsmarke verwendet; das Attribut ist die URI, unter der die vollständige DTD des Namensraums zu finden ist:

```
<einElement xmlns:einNamensraum='http://www.test.org/einNamensraumSchema.dtd'>
```

Codebeispiel 34: Referenzierung von XML-DTDs über Namensraumattribute

Eine solche Zuweisung bewirkt, dass das Element einElement und der darin enthaltene Inhalt den Namensraum einNamensraum zugewiesen bekommt, unabhängig davon, welche DTD für das Dokument, in dem sich das mit dieser Namensraumzuweisung versehene Element befindet, spezifiziert ist.

Liegen in einem Dokument eine oder mehrere Namensraumzuweisungen vor, so lassen sich qualifizierte Bezeichner verwenden, bei denen dem eigentlichen Element- oder Attributnamen jeweils der Bezeichner für den ihm zugeordneten Namensraum vorangestellt ist. Im nachfolgenden Beispiel wird der HTML-Namensraum eingeführt und explizit zur Auszeichnung der Elementmarken verwendet:

```
<?xml version="1.0">
<html xmlns:html='http://www.w3.org/TR/REC-html40'>
  <html:head><html:title>Dokumentitel</html:title>
  </html:head>
```

```
  <html:body>
    <html:p> Hier beginnt der erste Paragraph. </html:p>
  </html:body>
</html:html>
```

Codebeispiel 35: Kodierung eines Verweises auf einen XML-Namespace

Der Namensraum wird in der Wurzelmarke – HTML – deklariert und steht für alle Kindelemente zur Verfügung. Da die Semantik der Zuordnung vorsieht, dass alle Kindelemente den deklarierten Namensraum per Default erben, kann das Namensraumpräfix in obigem Beispiel entfallen, da per Default der an der Wurzel deklarierte Namensraum gültig ist; dies zeigt das folgende Beispiel:

```
<?xml version="1.0">
<html xmlns:html='http://www.w3.org/TR/REC-html40'>
  <head>
    < title>Dokumentitel</title>
  </head>
  <body>
    <p> Hier beginnt der erste Paragraph...</p>
  </body>
</html>
```

Codebeispiel 36: Implizite Verwendung von Elementen eines XML-Namensraums

Will man Elemente aus verschiedenen Namensräumen in einem Dokument mischen, so deklariert man über das xmlns-Attribute alle benötigten Namensräume und referenziert sie über das jeweilige Namensraumpräfix:

```
<?xml version="1.0">
<html xmlns:html='http://www.w3.org/TR/REC-html40
      xmlns:biblio='http://www.biblio.org/dublinschema.dtd'>
  <head>
    <biblio:title>Dokumentitel</biblio:title>
    <biblio:author>Hans Schmidt</biblio:title>
    <biblio:description>Ein Beispieldokument </biblio:description>
  </head>
  <body>
    <p> Hier beginnt der erste Paragraph...</p>
  </body>
</html>
```

Codebeispiel 37: Mischung mehrerer Namensräume in einem XML-Dokument

Bei Verwendung mehrerer Namensräume kann ein Namensraum als Default gekennzeichnet sein, so dass die Namensraumkennzeichnung nur für alle zusätzlich zum Default verwendeten Namensräume erforderlich ist. Die besondere Bedeutung der XML-Namensräume ist darin zu sehen, dass sie über die Modularisierungsmöglichkeiten von SGML hinaus (z. B. Subdokumente) ein einfaches Verfahren bereitstellen, um Konflikte bei der Interpretation von Elementen zu vermeiden. Dies ist vor allem dann von Bedeutung, wenn Dokumente aus unterschiedlichen Quellen miteinander kombiniert werden sollen. Abschließend sei auf die Problematik hingewiesen, dass Namensräume erst nach Abschluss der XML 1.0-Empfehlung eingeführt wurden; daher stehen bisher keine Mechanismen zur Verfügung, um die Verwendung von Elementen aus anderen Namensräumen in einer XML-DTD zu spezifizieren, was vielfach wünschenswert wäre (vgl. CLARK 1999: Appendix C) Im Referenzprojekt betrifft dies auf der Ebene der Inhaltskodierung

z. B. die Verwendung von Formelmarkup (MathML, vgl. unten Kap. 6.6) innerhalb der durch eine andere XML-DTD kodierten Teile des Buchs und auf der Ebene der Verwaltung des Buchs die Dienstekoordination.[67] Ungeachtet dessen lassen die hier verwendeten XML-Werkzeuge die Verwendung von Namensräumen zu (Parser, Transformationswerkzeuge) und sie sind in XML-Substandards wie XSL/T explizit berücksichtigt bzw. vorausgesetzt (vgl. unten Kap. 7.1.2).

5.2.4 Hyperlinks in XML

Die Hypertexttheorie hat verschiedene Modelle entwickelt, wie Verknüpfungen in Hypermedia-Systemen modelliert werden können (vgl. DAVIS 1995); SGML sieht nur einfache Referenzierungsmechanismen (ID, IDREF) vor und definiert keine allgemeine Verknüpfungssprache.[68] In der *Hypertext Markup Language* sind Links als einfache unidirektionale und nicht typisierte Verknüpfungen vorgesehen, für die ein eigenes Element (die Ankermarke, `<A>`) verwendet wird. Ergänzend zu XML liegt eine Sprachspezifikation vor, die wesentlich mächtigere Verknüpfungstypen zulässt: Die *XML Linking Language* (XLink; vgl. DEROSE et al. 2000) beschreibt die Kodierung von Hypertextverknüpfungen in XML-Dokumenten. Anders als in HTML, wo Hypertextverknüpfungen auf Elementebene zu kodieren sind, können XML-Links als *Attribute* in *alle* Elemente einer XML-DTD integriert werden und unterschiedlichen *Typs* sein, d. h. dem Benutzer steht es frei, für seine Anwendung die benötigte Menge an Verknüpfungstypen zu definieren und das Verhalten eines Präsentationssystems an unterschiedliche Linktypen anzupassen. DEROSE et al. 2000: Kap 1. beschreiben die Eigenschaften von XLink wie folgt:

> XLink provides a framework for creating both basic unidirectional links and more complex linking structures. It allows XML documents to:
> - Assert linking relationships among more than two resources
> - Associate metadata with a link
> - Create link databases that reside in a location separate from the linked resources.

XLink setzt zur Beschreibung der Eigenschaften von Verknüpfungen folgende Begriffe voraus (vgl. DEROSE et al. 2000: Kap. 1.3):

XML-Begriff	*Erläuterung*
arc	Gibt Eigenschaften und Regeln für die Traversierung einer Verknüpfung an (z. B. Direktionalität der Verknüpfung)
resource	Adressierbare Informationseinheit
linking element	Gibt Linkexistenz und Linkcharakteristika an
locator	Bezeichnet innerhalb des *link elements* die adressierte Ressource
traversal	Benutzen („Verfolgen“) des Links
multi-directional link	Link, der von mehr als einer Ressource aus traversiert werden kann
inline link	Eine Verknüpfung, die selbst Teil einer Ressource ist (d. h. der Normalfall einer Verknüpfung: ausgehend von Ressource A wird auf Ressource B verwiesen)
out-of-line-link	Eine Verknüpfung, die nicht selbst in einer Ressource enthalten ist, sondern z. B,. in einer externen Datei gespeichert ist

Tabelle 18: Konzepte der Extensible Link Language (XLink)

[67] Um dennoch eine DTD angeben zu können, wurde ein allgemeines Inhaltsmodell (`ANY`) für Elemente verwendet, die Kindelemente aus anderen DTDs enthalten.

[68] Die Definition beliebiger Verknüpfungsmechanismen ist natürlich möglich – sonst könnte es keine SGML-kompatible XLink-Spezifikation geben; es hat sich nur kein einheitlicher Standard für die Hypertext-Relationierung in SGML herausgebildet.

Mit Hilfe dieser Grundbegriffe lassen sich die Verknüpfungen in XLink nach folgenden Kriterien unterscheiden:

Eigenschaft	*Erläuterung*
Linkbeziehung	Typ des Links, d. h. Angabe einer semantischen Rolle für die Verknüpfung
Topologie	in-line, out-of-line, Links mit variabler Ressourcenanzahl
Lokator	Zieladresse einer Verknüpfung
Lokator-Sprache	Syntax, die für die Festlegung der Ressourcenadressierung verwendet wird (URL, SQL-Query, XPointer, Dateinamen, *digital object identifier* etc.)
Formatierung	Art der Präsentation des Links (Text, Graphik, etc.)
Verhalten	Wie wird die adressierte Ressource präsentiert? Wann wird die Traversierung ausgelöst?

Tabelle 19: Eigenschaften von Verknüpfungen in XLink

Grundsätzlich unterscheidet XLink zwischen einfachen vier Typen von Verknüpfungskonzepten:

- Einfache Links (*simple links*), die in etwa den aus HTML bekannten Verknüpfungen entsprechen,
- erweiterte Verknüpfungen (*extended* links), die externe (*out-of-line*) und 1:n-Verknüpfungen sowie die Erstellung von Verknüpfungsgruppen ermöglichen,
- *arcs* (Kanten), d. h. Spezifikationen für die Eigenschaften der Linktraversierung und
- Lokatoren, d. h. derjenige Teil der Verknüpfung, der die verbundene Ressource identifiziert (Adressierung).

Das nachfolgende Beispiel zeigt einen HTML-Anker, ein eigenständiges XLink, das nicht in ein Element eingebettet ist und XLink-Attribute als Teil eines Überschriftelements. Semantische Attribute wie role oder title sind nicht spezifiziert.

```
<!-- HTML-Anker:-->
<A href="http://www.informatik.uni-leipzig.de">Ein Beispiel-Verweis</A>
<!—eigenständiges XLink-Element-->
<xlink:simple href="students.xml">Ein Beispiel-Verweis</xlink:simple>
<!-- simple XML-Link in H1-Element:-->
<H1 xmlns:xlink="simple" xlink:href="http://www.informatik.uni-leipzig.de">Ein Beispiel-Verweis</H1>
```

Codebeispiel 38: Beispiele für HTML- und XLink-Verknüpfungen

Die Einführung der Verknüpfungen greift auf die Möglichkeit der Verwendung von Namensräumen zurück. Neben dem Element XLink bzw. dem Attribut zusätzlichen Attribut xmlns:xlink ist die Syntax der *Lokatoren* anders. Es sind nicht nur URIs zulässig, sondern auch XPointer (s. u.), d. h. ein erweitertes Adressierungsschema relativ zum aktuellen Dokument.

Im Unterschied zu einfachen Links können erweiterte Links auf mehr als eine Ressource verweisen (1:n-Verknüpfung), d. h., dass von einem Ausgangspunkt mehrere Zielpunkte durch Links mit unterschiedlicher Typisierung eingeführt werden können, eine Forderung, die in der Hypertexttheorie seit langem vorgesehen ist, für die in den meisten Hypertextanwendungen bisher aber keine adäquate Entsprechung vorgesehen ist bzw. die durch Hilfslösungen bereitgestellt werden muss.[69] Die erweiterten Möglichkeiten von XLinks – Typisierung, Benennung, Spezifikation des Traversierungsverhaltens etc. – werden durch zusätzliche Attribute in den Linkelementen realisiert:

[69] Z. B. durch Einführung unterschiedlicher graphischer Symbole, die für je einen Linktypus stehen und die zusätzlich zur eigentlichen Ausgangsressource als zugehörige Links präsentiert werden.

Attribut	*Erläuterung*
href	referenzierte Adresse, verwendet URIs oder XPointer als Lokatoren
(content-)role	bezeichnet die semantische Rolle (den Typ) des Links
(content-)title	gibt dem Link einen Namen
show	gibt an, wie die referenzierte Adresse angezeigt werden soll (new|parsed|replace))
actuate	gibt an, wie der Link ausgelöst werden soll (durch den Benutzer oder automatisch (wie z. B. bei <IMG SRC="..."> in HTML
behavior	nähere Spezifikation des Linkverhaltens; ohne inhaltliche Vorgabe, daher anwendungsspezifisch
from	Gibt ggf. die Ausgangsressource an
to	Gibt ggf. die Zielressource an

Tabelle 20: Attribute erweiterter Verknüpfungen in XLink

Diese Attribute können zusammen mit XLinks verwendet werden; inwieweit das Viewing-System sie versteht, ist implementierungsabhängig: Anders als für HTML-Anker gibt es bisher keine „fest verdrahtete" Standardimplementierung; da aber die Implementierung entsprechender Parsingmechanismen durch die Vielzahl entsprechender XML-Werkzeuge gut möglich ist, wird dieser Nachteil schnell behoben werden können.

Ein wesentlicher Aspekt der Verwendung von Links ist die Tatsache, dass über eine geeignete Festlegung in der DTD jedes Element als Link verwendet werden kann: Man muss lediglich die erforderlichen Linkattribute in der DTD für die gewünschten Elemente deklarieren. Es entfällt die zusätzliche Verwendung eines Ankerelements für jede Ressource (jedes Element), für das eine Verknüpfung vorgesehen ist. Die Standardfunktionalität der Ankermarke in HTML erreicht man für ein Element einer XMKL-DTD mit folgender Attributdeklaration:[70]

```
<!ELEMENT ElementMitEinfachemLink ANY>
<!ATTLIST ElementMitEinfachemLink
          xmlns:xlink    CDATA                        #FIXED "http://www.w3.org/XML/XLink/0.9"
          xlink:type     (simple|extended|locator|arc) #FIXED "simple"
          xlink:href     CDATA                        #REQUIRED
          xlink:role     (Detail|Beispiel|Literatur)  "Detail"
          xlink:title    CDATA                        #IMPLIED
          xlink:show     (new|parsed|replace)         "replace"
          xlink:actuate  (user|auto)                  "user"
>
```

Codebeispiel 39: Deklaration einer erweiterten Verknüpfung in XLink

Das folgende Beispiel zeigt die Anwendung eines solchen typisierten Links; es bleibt allerdings offen, wie die semantische Auszeichnung des Links (Rolle, Titel) für die Optimierung der Präsentation, d. h. in der Benutzerschnittstelle, erfolgt:

```
Dieses Argument lässt sich durch folgendes
<ElementMitEinfachemLink  xmlns:xlink
                          xlink:href="http://www.test.org"
                          xlink:role="Beispiel"
                          xlink:title="Referenz zu http://www.test.org"
                          xlink:show="new">
Beispiel</ElementMitEinfachemLink> noch weiter verdeutlichen ...
```

Codebeispiel 40: Anwendung einer erweiterten Verknüpfung in XLink

[70] Man beachte, dass die aktuelle Fassung von XLink in DeRose et al. 2000 noch nicht abschließend spezifiziert ist; insbesondere wird keine DTD für XLink angegeben; die Beispiele können daher von einer endgültigen Fassung von XLink abweichen.

Die Attribute xmlns:link, xmlns:type und xmlns:inline haben in der Attributdeklaration einen festen Wert (#FIXED) bzw. eine Defaultvorgabe und müssen nicht gesondert spezifiziert werden.

Im Gegensatz zu einfachen Verknüpfungen erlauben erweiterte Links auch 1:n-Beziehungen sowie die Möglichkeit durch externe Kodierung von Verknüpfungen auf (*read only*) Ressourcen zuzugreifen, in denen die Deklaration von Linkelementen nicht möglich ist:

> An extended link is a link that associates an arbitrary number of resources. All of its participating resources may be remote, in which case the link is out of line, or some of the link's own content can serve optionally as local resources, in which case the link is inline. [DEROSE et al. 2000: Kap. 3.1]

Eine derartige separate Verwaltung von Verknüpfungselementen ist kein prinzipiell neuer Gedanke; es gibt bereits eine Reihe von Hypermedia-Systemen, die diese für die Verwaltung komplexer Informationsbestände hilfreiche Funktionalität aufweisen (z. B. Hyper-G/HyperWave, vgl. LENNON 1997: 116f.). Mit Hilfe der einfachen Ankermarken ist diese Funktionalität allerdings nicht bzw. nur auf Umwegen möglich. Um Mehrfachverweise kodieren zu können, muss das Linkelement eine Binnenstruktur erhalten, damit mehr als eine Zieladresse angegeben werden kann. Im nachfolgenden Beispiel wird der erweiterte Verweis als Container (erweiterterlink) modelliert, in dem dann die einzelnen Verweise (einzelverweis), die von einem Startpunkt ausgehen sollen, kodiert werden können:

```
<!ELEMENT  erweiterterlink einzelverweis+>
<ATTLIST     erweiterterlink
       xmlns:xlink          CDATA                          #FIXED  "http://www.w3.org/XML/XLink/0.9"
       xlink:type           (simple|extended|locator|arc)  #FIXED  "extended"
>

<!ELEMENT  einzelverweis ANY>
<ATTLIST  einzelverweis
           xmlns:xlink     CDATA      #FIXED  "http://www.w3.org/XML/XLink/0.9"
           xlink:type      (locator)  #FIXED  "locator"
           xlink:href      CDATA      #REQUIRED
           xlink:role      (Detail|Beispiel|Literatur)  "Detail">
```

Codebeispiel 41: Definition von Mehrfachverweisen (1:n-Links) in XLink

Die Anwendung des erweiterten Link könnte wie folgt aussehen:

```
<erweiterterlink xlink:type ="extended" >
  <einzelverweis xlink:href="http://www.test.org/a.xml"   type="locator" role="Beispiel">
  <einzelverweis xlink:href="http://www.test.org/b.xml"   type="locator" role="Gegenbeispiel">
  <einzelverweis xlink:href="http://www.test.org/c.xml"   type="locator" role="Detail">
</erweiterterlink>
```

Codebeispiel 42: Anwendung erweiterter Links in XLink

Wie ein Browser dieses Verhalten tatsächlich realisiert, ist eine Implementierungsfrage; ähnlich wie bei der Typisierung von Links ist ein einheitlicher Standard kaum vorstellbar. Eine Lösung, die eine Vielzahl von Anwendungen abdecken kann, ist die Verwendung von Kontextmenüs, die sich bei Aktivierung durch den Benutzers öffnen (oder sich bereits bei einem MOUSEOVER-Ereignis aktivieren).

Die erweiterten Links lassen sich zu *Gruppen* zusammenfassen, die dann in einer externen Datei als *out-of-line*-Verknüpfungen abgelegt werden. Auf diese Weise können Verknüpfungen für Ressourcen definiert werden, in denen die Abspeicherung von Links

direkt nicht möglich ist. Damit ist eine wesentliche Voraussetzung für die Hypertextualisierung multimedialer Inhalte erfüllt: Die Verknüpfungsmöglichkeiten sind nicht mehr auf die *textuellen* Teile eines multimedialen Buchs beschränkt; Referenzen auf Bilddateien können über externe Linkgruppen als *out-of-line*-Links spezifiziert werden. Im Fall des physikalischen Praktikums betrifft dies z. B. die Typisierung der Multimediaelemente, die Unterscheidung referentieller von indexikalischen Links und die Kodierung einfacher Navigationsverknüpfungen. Anzumerken ist allerdings, dass die freie Definierbarkeit von Linktypen gegenüber den untypisierten syntaktischen Links in HTML Nachteile mit sich bringt, da sie die Wiederverwendbarkeit ggf. einschränkt, wenn Dokumente mit unterschiedlicher Linktypologie zusammengeführt werden sollen.

Neben der *Typisierung* und der Gruppierung zu erweiterter Links sieht XLink die Spezifikation des Verhaltens bei der Linkverfolgung (*Traversierung*) vor; dabei sind die aus der Hypertexttheorie bekannten Linkverhaltensweisen berücksichtigt (vgl. oben Kap. 2.2.1.4):

- Automatische Aktivierung von Verknüpfungen (xlink:show="auto"), wie dies bei Verknüpfungen auf einzubettende Bilder oder Multimediakomponenten sinnvoll sein kann,
- benutzergesteuerte Aktivierung (xlink:show="user"), der Normalfall der Linktraversierung – eine Verknüpfung wird erste bei explizitem Auslösen durch den Benutzer verfolgt,
- ersetzende Traversierung, bei der die Zielressource die bisher dargestellte informationelle Einheit verdrängt (xlink:actuate="replace"),
- Traversierung mit Öffnen eines zusätzlichen Präsentationsfensters (xlink:actuate = "new") und
- Traversierung durch Einbettung des Elementbaums der Zielressource in den bisher dargestellten Kontext (xlink:actuate="parsed"). Diese sog. *Inklusion* oder *Transklusion* der Zielressource als Einbettung in die Ausgangsressource war bereits wesentlicher konzeptueller Bestandteil des Hypertextsystems von Ted NELSON, Xanadu, vgl. oben Kap. 2.2.1.1 und 2.2.1.7.

Die Entwicklung von XLink ist derzeit noch nicht abgeschossen; es liegt bisher noch keine normative DTD vor (vgl. DEROSE et al. 2000; BRADLEY 2000: 173 ff., GOLDFARB & PRESCOD 1998: 500 ff.). XLink-fähige Werkzeuge sind ebenfalls noch im experimentellen Stadium. Daher wird man sich bei der Realisierung der in XLink vorgesehenen Konzepte auf Behelfslösungen stützen müssen oder durch eine geeignete Transformation als XLinks kodierte erweiterte Verknüpfungen in einfachere Konzepte, z. B. HTML-Anker, übersetzen müssen. Dies geschieht im Referenzprojekt dadurch, dass als *extended links* im Sinne von XLink eingebundene Komponenten (vgl. unten Anhang 17.1.1) bei einem Transformationsschritt (XSL/T, s. u. Kap. 7.1.2.1) in Skripten umgewandelt werden, die bei Anforderung einer Präsentationseinheit durch den Benutzer Verknüpfungssymbole und HTML-Anker erzeugen.

5.2.5 Adressierung in XML

Die Erweiterung der Verknüpfungsmöglichkeiten ist ein wesentlicher Aspekt für die Hypertextualisierung von multimedialen Dokumenten; eine weitere, nicht minder bedeutsame Frage ist die des Adressierungs*format*s und der strukturellen Möglichkeiten der Adressierung in Hypermedia-Dokumenten – in den bisherigen Beispiel wurde implizit davon ausgegangen, dass in einer Verknüpfung eine Zieladresse angegeben wird, die nach

dem URI-Standard eine absolute oder relative Adresse mit Bezug auf das Dateisystem eines Web-Servers ist; ein dynamischer Link liegt in diesem Sinn allenfalls dann vor, wenn nicht auf eine feste Ressource (Text, Bild etc.) verwiesen wird, sondern durch die Verknüpfung eine Routine oder ein Programm (z. B. eine Datenbankschnittstelle) aufgerufen wird, das dann dynamisch eine strukturierte informationelle Einhit generiert und an den Client zurückliefert. Die Binnenstruktur von SGML- oder XML-basierten Dokumenten ist mit diesem einfachen Adressierungsmechanismus nicht erschlossen; man kann so z. B. nicht auf ein vorangegangenes Element eines bestimmten Typs verweisen. Allenfalls über die Spezifikation von im Dokument enthaltenen Spungzielmarken (in HTML: `name`-Attribut der Ankermarke, das als Präzisierung des Sprungziels an die `href`-URL angehängt werden kann: `<A href="#Sprungzielname">`).

Um die Adressierung des Elementbaums eines XML-Dokuments zu ermöglichen, sind beim W3C zwei Standardisierungsvorschläge erarbeitet worden, die in engem Zusammenhang mit der *Extensible Style Language* als Verarbeitungs- und Formatierungsstandard zu sehen sind:

- Die *XML Path Language* (vgl. CLARK & DEROSE 1999) und
- die *XML Pointer Language* (XPOINTER, vgl. DEROSE, DANIEL & MALER 1999).

Ausgangspunkt von XPath ist eine Syntax für Ausdrücke, deren Auswertung bezüglich eines gegebenen XML-Elementbaums folgendes ergeben kann:

- Eine Knotenmenge,
- einen Wahrheitswert,
- einen numerischen Wert oder
- eine Zeichenkette.

Die Auswertung erfolgt dazu relativ zum Kontext des Ausdrucks, d. h. eines Kontextknotens, seiner Position und Größe, einer Menge von Variablenbindungen, der verfügbaren Funktionen und der im Skopus des Ausdrucks gültigen Vereinbarungen über XML-Namensräume. Der Auswertung eines XPath-Ausdrucks ist ein Datenmodell zugrunde gelegt, das ein XML-Dokument als Baum interpretiert. Dieser ist aus folgenden Knotentypen aufgebaut ist (vgl. CLARK & DEROSE 1999: Kap. 5):

- Ein Wurzelknoten (*root node*),
- Elementknoten (*element nodes*),
- Textknoten (*text nodes*),
- Attributknoten (*attribute nodes*),
- Namensraumknoten (*namespace nodes*)
- Knoten mit Verarbeitungsanweisungen (*processing instruction nodes*) und
- Kommentarknoten (*comment nodes*).

Zur programmtechnischen Umsetzung in XML-APIs wie dem *document object model* (DOM) siehe unten Kap. 5.2.7.1.2. Die Knotentypen spiegeln die generischen Strukturbestandteile von XML wider und erlauben es, alle Bestandteile eines Dokuments zu adressieren.

Die wichtigste Unterform eines XPath-Ausdrucks sind sog. *location paths*, die aus dem Kontext des Ausdrucks Elemente oder Teilbäume adressieren. Für den Aufbau solcher *location paths* führt XPath das Konzept einer *axis* ein, worunter eine in Abhängigkeit des aktuellen Kontext zu selektierende Datenmenge eines Dokuments zu verstehen ist: „**[Definition:] axis** – A reserved name that defines a sequence of data portions. Typically, an XPointer utilizes axes in combination with predicates to select particular data

portions. [...]“ (DEROSE, DANIEL & MALER 1999: Appendix A) Folgende *axis*-Typen sind in XPath vorgesehen:

Relation	*Erläuterung*
child	Selektiert die unmittelbaren Kindknoten des aktuellen Kontextknotens, also seine direkten Nachfolger im Elementbaum
descendant	Selektiert die Nachfahren des Kontextknotens, d. h. seine Kinder und alle mittelbaren Nachfolgerknoten
parent	Wählt den Elternknoten des Kontextknotens aus
ancestor	Selektiert alle Vorfahren des Kontextknotens im Baum aus
following-sibling	Selektiert die Nachfolger auf der gleichen Ebene (rechte Nachbarknoten) des Kontextknotens
preceding-sibling	Wählt die Vorgängerknoten auf der gleichen Ebene des Kontextknotens aus (linke Nachbarn)
following	Selektiert alle dem Kontextknoten im Dokument *nachfolgenden* Knoten mit Ausnahme seiner descendants
preceding	Selektiert alle im Dokument dem Kontextknoten *vorangehenden* Knoten mit Ausnahme seiner descendants
attribute	Selektiert die Menge der Attribute des Kontextknotens
namespace	Selektiert die Namensraumknoten des aktuellen Kontextknotens
self	Wählt den Kontextknoten selbst aus
descendant-or-self	Wählt den Kontextknoten und alle seine Nachfolger im Baum aus
ancestor-or-self	Wählt den Kontextknoten und alle seine Vorgänger im Baum aus

Tabelle 21: axis-Relationen für den Aufbau von location paths in XPath

Die *axis*-Relationen ermöglichen die dynamische Referenzierung der Strukturbestandteile eines XML-Dokuments und nutzen so die deklarative Auszeichnung für die Traversierung des Dokuments. XPath ist sowohl Ausgangspunkt für die in XSL enthaltene Transformationssprache (XSLT) als auch für das Verweisformat der XPointer.

Ein *location path* kann mit Hilfe der *axis*-Relationen aufgebaut werden; man unterscheidet *absolute* und *relative location paths*; sie sind aus einzelnen *location steps* aufgebaut. Jeder *location step* besteht aus

- einer axis-Relation, die die Beziehung zwischen dem Kontextknoten, dem Elementbaum und den selektierten Knoten angibt,
- einem *node test*, der einen *Knoten*typ (Element, Attribut, Namensraum etc.) angibt und
- null oder mehr Prädikaten, die über den durch die axis-Relation und den *node test* selektierten Knoten operieren.

Unter einem Prädikat ist ein XPath-Ausdruck zu verstehen, der die aus der axis-Relation und dem *node test* resultierende Knotenmenge filtert und eine neue Knotenmenge produziert. Ein Prädikat kann selbst einen *location path* enthalten. Die nachfolgenden Produktionen sollen die wesentlichen Elemente des Aufbaus eines *location path* verdeutlichen:

```
location path            ::==  relative_location_path     |  absolute_location_path
absolute_location_path   ::==  '/' relative_location_path? |  absolute_location_path_kurzform
relative_location_path   ::==  location_step  |  relative_location_path  '/' location_step |
                               relative_location_path_kurzform

location_step            ::==  axis-Spezifikation node_test prädikat*
axis-Spezifikation       ::==  axis-Relation '::' | axis-Spezifikation_Kurzform
prädikat                 ::==  '[' Ausdruck  ']'
```

Codebeispiel 43: Aufbau von location steps in XPath

Mit Hilfe dieses rekursiven Aufbaus von *location paths* lassen sich differenzierte Selektionsmechanismen definieren. Einige Beispiele sollen die Anwendung dieses Formats verdeutlichen:

location path	*Interpretation*
following-sibling::kapitel[position()=1]	axis-Spezifikation: following-sibling *node test*: kapitel (ein Elementknoten) Prädikat: position()=1 Ein relativer *location path*, der das nächste Element vom Typ kapitel auf der Ebene des Kontextknotens selektiert.
/descendant::olist/child::item	*step* 1: /descendant::olist axis-Spezifikation: descendant *node test*: olist (ein Elementknoten) Prädikat: — *step* 2: child::item axis-Spezifikation: child *node test*: item (ein Elementknoten) Prädikat: — Ein absoluter *location path*, der alle Elemente vom Typ geordnete Liste unter den direkten und mittelbaren Nachfolgern des Kontextknotens auswählt und im zweiten Schritt aus diesen Listen alle unmittelbaren Nachfolger vom Typ item selektiert.
child::p[attribute::type='aufgabenstellung']	axis-Spezifikation: child *node test*: p (ein Elementknoten) Prädikat: attribute::type='aufgabenstellung' Selektiert alle direkten Nachfolger des Kontextknotens vom Typ p, die ein Attribut vom Typ aufgabenstellung enthalten.
self::text()	axis-Spezifikation: self *node test*: text() (Textknoten) Prädikat: — Selektiert den Text des Kontextknotens.

Codebeispiel 44: Beispiele für location paths in XPath

Wie die Beispiele andeuten, definiert XPath für den Aufbau von Ausdrücken

- eine Menge logischer Operatoren und Vergleichsoperatoren,
- arithmetische Operatoren für die Manipulation numerischer Werte,
- eine Syntax für Funktionsaufrufe sowie
- eine Basismenge an Funktionen (*core function library*), die über Knotenmengen (u. a. last(), position(), count(), name()), Zeichenketten (u. a. concat(), contains(), start-with()) oder Zahlen (u. a. sum(), round()) operieren bzw. einen Wahrheitswert liefern (not(), true(), false(), lang()). Die Funktionsmenge ist für spätere Ergänzungen offen.

Auf der Basis der durch XPath definierten Syntax für die dynamische und flexible Adressierung des Elementbaums eines XML-Dokuments führt XPointer ein Adressierungsformat ein, das die bekannte URI-Syntax erweitert und die Möglichkeit bietet, an die Stelle eines *fragment identifiers* eines URI einen XPath-Ausdruck zu setzen. XPointer ist eine Weiterentwicklung von Referenzierungsformaten, die im Rahmen der *Text Encoding Initiative* (TEI, vgl. GIORDANO 1994) bzw. durch den HyTime-Standard entwickelt wurden. XPointer (genauer: *XPointer fragment identifier*) operieren durch Einführung der Semantik der XPath *location paths* auf dem Elementbaum eines XML-Dokuments. Drei Verwendungsarten lassen sich unterscheiden:

- Die direkte Adressierung eines ID-Attributs eines Dokuments durch Verwendung von dessen Attributwert als Bezeichner, z. B. *http://www.test.org/seite34.xml#Position1*.

Position1 ist der Bezeichner eines Elements im Dokument seite34.xml; die Verwendung im URI ist eine Kurzform für die vollständige XPointer-Angabe xptr(id(„intro“)). Diese Kurzform wird v. a. aus Kompatibilitätsgründen zu HTML eingeführt („bare name adressing“), vgl. DEROSE, DANIEL & MALER 1999: Kap. 3.

- Eine Kurzform, die ausgehend vom Wurzelelement eines Dokuments durch ‚/‘ getrennte Zahlenwerte kodiert und jeweils *n*te Kindelement des Vorgängers auswählt (sog. *tumbler*, z. B. wählt /5/3/2 ausgehend von der Dokumentwurzel deren fünftes Kindelement aus (z. B. das fünfte Kapitel), davon wiederum das dritte Kindelement (z. B. der dritte Absatz) und schließlich innerhalb dieses Absatzes das zweite Kind (z. B. der zweite Satz).
- Die Vollform der XPointer-Syntax (stark vereinfacht nach DEROSE, DANIEL & MALER 1999: Kap. 3, Kap. 5.2.1):

```
XPointerAdresse             ::==  (  'xptr (' allgemeiner_location_path ')'     )+
allgemeiner_location_path   ::==  bereich  |  location_path
bereich                     ::==  'range' '::' location_path ',' location_path
```

Neben der Verwendung der XPath *location paths* führt XPointer einige zusätzliche Konzepte bzw. Präzisierungen ein (DEROSE, DANIEL & MALER 1999: Kap. 5.):

- Zwei zusätzliche axis-Relationen, range:: und string::, die keine Knotenmenge als Teilbaum im Sinne von XPath liefern, sondern einen kontinuierlichen Bereich eines Dokuments, wie er z. B. vom Benutzer selektiert wird und der sich nicht durch axis-Relationen von XPath beschreiben lässt, bzw. einen Bereich, der durch Zeichenkettenvergleich im Dokumentinhalt ermittelt wird,
- zusätzliche relative *location paths*, here() und origin(), die zur relativen Adressierung in Abhängigkeit von der Position des XPointer selbst bzw. vom Ausgangspunkt einer Hypertexttraversierung verwendet werden,
- eine neues Prädikat, unique(), das einen Wahrheitswert liefert, der angibt, ob das Ergebnis eines XPointer-Ausdrucks eine Knotenmenge oder ein einzelner Knoten ist und
- schließlich ein Verfahren, das definiert, wie der Kontextknoten im Sinne von XPath initialisiert werden muss. Der Kontextknoten ist dabei der Wurzelknoten des Dokuments, auf das der XPointer gerichtet ist.

Für die Anwendung von XPath und XPointer ergeben sich vielfältige Möglichkeiten: Die Adressierungsmöglichkeit bezüglich des Elementbaums statt fester Ressourcen und Ressourcenbezeichner kann u. a. dazu dienen,

- ein generisches Navigationsmodell zu implementieren, in dem man statt „fester“ Verweise jeweils auf das nächste Element im Elementbaum vom gewünschten Typ verweist; ein solches Modell kann dann unabhängig von einem festen Datenbestand arbeiten, was die dynamische Integration von Inhalten vereinfacht, oder
- Hypertextlinks zu definieren, die ihren Bezugspunkt über einen *Bereich* eines Dokuments erstrecken (entweder ein Teilbaum im Sinne der axis-Relationen von XPath oder ein Bereich, der durch einen XPointer mit range-axis ausgewählt wird.). Ein Anwendungsbeispiel sind die multimedialen Ergänzungen im physikalischen Praktikum, die jeweils nicht auf eine einzelne Ressource bezogen, sondern für einen ganzen Dokumentbereich (oder mehrere Dokumentbereiche) gültig sind. Ein Kombination aus erweiterten XLink-Verknüpfungen und XPointern kann den Benutzer dadurch unterstützen, dass er bei einer multimedialen Ergänzung eine Liste aller Dokumentbereiche erhält, die für diese Ergänzung von Bedeutung ist.

Wie für XLink gilt allerdings für XPath und XPointer, dass die Spezifikationen noch nicht abgeschlossen sind und kaum Implementierungen der entsprechenden Funktionalität vorliegen. Eine Erweiterung der DOM-basierten XML-Werkzeuge um XLink bzw. XPointer ist aber in absehbarer Zeit zu erwarten. Insofern kann zwar für das Referenzprojekt spezifiziert werden, in welcher Form diese Technologien zum Einsatz kommen können, ihre praktische Umsetzung hat aber wenigstens partiell mit anderen Mitteln zu erfolgen.

5.2.6 Schemata für die Strukturbeschreibung in XML

Die Entwicklung von SGML zu XML, die als Vereinfachung von SGML auf der Basis gemeinsamer Strukturierungsmechanismen wie der Beschreibung informationeller Strukturen durch DTDs beschrieben wurde, wird durch eine Reihe von Standardisierungsvorschlägen des *World Wide Web Consortium* weiter differenziert und soll gewährleisten, dass XML den weitergehenden Anforderungen als Datenbeschreibungssprache für verschiedene Anwendungen gerecht werden kann.

Dabei sollen Defizite von SGML, die sich darauf zurückführen lassen, dass SGML ursprünglich aus dem Verlags- bzw. Dokumentationsbereich entstammt, beseitigt werden. Zu ihnen gehören:

- Die Typisierung und Beschreibung der in einem SGML-Dokument enthaltenen Daten: Für die in einem SGML-Element enthaltenen Daten stehen – abgesehen von durch ein *content model* des Elements spezifizierte Elemente – im wesentlichen nur *Zeichendatentypen* zur Verfügung.[71] Da sich XML zu einer Datenaustauschsprache mit vielfältigen Anwendungen weiterentwickelt, ist eine präzisere Beschreibung der Semantik (Datentyp, Wertebereich) von Elementinhalten wünschenswert: Im Sinn einer inhaltsorientierten Auszeichnung ist es erforderlich, Informationen über Wertebereiche, die Kardinalität oder das Darstellungsformat eines Elementinhalts (z. B. ein Datensatz eines Versuchs, der durch eine in einem dynamischen elektronischen Buch enthaltene Simulation generiert wurde) explizit zu kodieren.
- Weitergehende Definitionsmöglichkeiten für Strukturelemente, die Konzepte aus der objekt-orientierten Modellierung wie Vererbung (Spezialisierung, Generalisierung aufgreifen und es erlauben, Elemente mit Bezug zu bereits vorliegenden Elementdefinitionen vorzunehmen.
- Ein Mechanismus, mit dem man die Struktur eines XML-Dokuments durch ein XML-Dokument, also ohne Verwendung einer DTD, beschreiben kann. Eine solche Beschreibung kann anders als eine DTD, für die nicht dieselben syntaktischen Konventionen gelten wie für Dokumentinstanzen, von XML-verarbeitenden Werkzeugen analysiert werden.

Zur Lösung dieser Probleme gibt es bereits verschiedene Standardisierungsvorschläge, die von der *XML Schema Working Group* des *World Wide Web Consortium* erarbeitet wurden:

- *Document Content Description for XML* (DCD, BRAY, FRANKESTON & MALHOTRA 1998), ein Versuch, XML-Strukturen unter Verwendung der Konventionen des *Resource Description Framework* (RDF, vgl. unten Kap. 9.1) zu beschreiben,

[71] `#PCDATA`, `RCDATA`, `CDATA`, vgl. LOBIN 2000: 160f.; RIEGER 1995: 171 ff. Die Weiterentwicklung von SGML zu WebSGML (Annex K und L des ISO-Standards 8879, 1997) erlaubt die Einführung von Datentypen für SGML-Attribute, vgl. GOLDFARB 1999: 19 ff.

- *Schema for Object-Oriented XML* (vgl. DAVIDSON et al. 1999), ein Vorschlag, der die Beschreibung XML-basierter Datenstrukturen mit Hilfe von Konzepten aus der objekt-orientierten Modellierung ermöglicht
- eine *Document Definition Markup Language* (DDML, vgl. BOURRET et al. 1999) und
- *XML Schema* (*Structures* & *Datatypes*, vgl. THOMPSON et al. 1999 und BIRON & MALHOTRA 1999), ein umfassender Ansatz, der aufbauend auf den voranstehenden genannten und weiteren Vorschlägen (XML-Data, vgl. LAYMAN et al. 1998) die Typisierung und Beschreibung von XML-Strukturen durch Schemata zu standardisieren versucht.

Im Folgenden wird nur *XML Schema* diskutiert, da seine Standardisierung am weitesten fortgeschritten ist und dieser Ansatz Konzepte der anderen Standardisierungsbemühungen aufgreift. XML Schema ist in einen Struktur- (*XML Schema: Structures*) und in einen Datentypteil (*XML Schema: Datatypes*) gegliedert. THOMPSON et al. 1999: Kap. 1.2 beschreiben die Zielsetzung für den Entwurf von Schemata wie folgt:

> The purpose of *XML Schema: Structures* is to provide an inventory of XML markup constructs with which to write schemas.
>
> The purpose of an *XML Schemas: Structures* schema is to define and to describe a class of XML documents by using these constructs to constrain and document the meaning, usage, and relationships of their constituent parts: datatypes, elements and their content, attributes and their values. [...] Schemas are intended to document their own meaning, usage, and function through a common documentation vocabulary.

Wie eine DTD dienen Schemata der Definition zulässiger Markupkonstrukte, die in XML-Dokumentinstanzen zulässig sind; die Ausdrucksmöglichkeiten einer DTD unterscheiden sich von denen eines XML-Schemas allerdings erheblich. Ein Schema kann die in der folgenden Tabelle zusammengefassten Bestandteile aufweisen (nach THOMPSON et al. 1999: Kap. 2.4). Dabei wird deutlich, wie *XML Schema* die Strukturmerkmale einer DTD „rekonstruiert“:

Merkmal	*Zweck*	*Benannt*	*Verwendung in Dokumentinstanz*
Schema	Kapselt alle Vereinbarungen des Schemas	Ja	Nein
Einfache Typdefinition	Definiert atomare Datentypen als Einschränkungen für Elementinhalte und Attributwerte	Ja	Nein
Komplexe Typdefinition	Eine Menge von Einschränkungen für Elementinhalte und Attributwerte	Ja	Nein
Elementdeklaration	Weist einem Elementnamen einen Datentyp zu, entspricht einer `<!ELEMENT…>`-Vereinbarung in einer SGML- bzw. XML-DTD.	Ja	Ja
Attributdeklaration	Weist einem Attributnamen einen einfachen Datentyp zu, wobei die Zuordnung für den umgebenden Typ lokal gültig ist.	Ja	Ja
Inhaltstyp	Einfacher Typ oder Inhaltsmodell (*content model*); der Inhaltstyp restringiert die möglichen Elementinhalte in einer Dokumentinstanz.	Nein	Nein
Inhaltsmodell (*element content model*)	Restringiert die Inhalte von Elementen in Dokumentinstanzen	Nein	Nein
Definition von Attributgruppen	Weist einem Namen eine Sammlung von Attributdeklarationen zu	Ja	Nein
Ableitende Typdefinition	Definition eines Typs auf der Basis eines anderen Typs, wobei Inhaltstyp und/oder Attribute von diesem Ausgangstyp übernommen werden	Ja	Nein

Merkmal	*Zweck*	*Benannt*	*Verwendung in Dokumentinstanz*
Querverweise zu Schemakomponenten in verschiedenen Namensräumen	Integriert Definitionen und Deklarationen aus anderen Schemata	Nein	Nein
Referentielle Constraints	Zusätzliche referentielle Mechanismen innerhalb eines Dokuments (UNIQUE, KEY, KEY REFERENCE)	Ja	Nein

Tabelle 22: Strukturelemente eines XML-Schemas

Die Syntax für den Aufbau eines Schemas ist wie folgt definiert:

```
Schema ::==   Präambel ( Annotation | Datentypdefinition | Typdefinition | Elementdeklaration |
                  Attributgruppendefinition | Notationsdeklaration | Inklusion | Import |
                  allgemeineEinschränkung )*
```

Codebeispiel 45: Syntax des Schemaaufbaus

Das nachfolgende Beispiel zeigt einige der in Tabelle 22 aufgelisteten Strukturelemente eines XML-Schemas:

```
<!-- Schema-Element-->
<schema   zielnamensraum="http://purl.org/metadata/dublin_core"
          version="1.0"
          xmlns="http://www.w3.org/1999/XMLSchema">

  <!-- abgeleitete (einfache) Datentypdefinition mit Wertebereichsconstraint -->
  <datatype name="PositiveGanzzahl" source="Integer">
    <minExclusive value="0"/>
  </datatype>
  <!-- ... -->
</schema>

<!-- Definition eines komplexen Typs -->
<type name="groessenangabe" type="decimal">
  <restrictions>
    <minInclusive value="0">
  </restrictions>
  <attribute name="einheit" type="NMTOKEN"/>
</type>

<!-- Elementdefinition auf der Basis eines vorab definierten Typs -->
<element name="laenge" type="groessenangabe"/>

<!-- Verwendung des Elements: -->
<laenge einheit="cm">30.15</laenge>

<!-- Attributdefinition auf der Basis eines (durch XML Schema: Datatypes) vordefinierten Datentyps -->
<attribute name="attr_1" type="integer"/>

<!-- Attributdefinition auf der Basis eines abgeleiteten Datentyps -->
<attribute name="attr_2" type="PositiveGanzzahl"/>

<!-- Definition eines (benannten) Inhaltsmodells als Elementgruppe-->
<group name="eineModellgruppe">
  <element ref="einElement"/>
</group>
```

```
<element name="einElement">
  <type>
    <group ref="eineModellgruppe"/>
    <attribute ...> ... </attribute>
  </type>
</element>

<!-- Typdefinition mit mehreren Bestandteilen -->
<type name="personenName">
  <elemen name="vorname" minOccurs="0" maxOccurs="*"/>
  <element name="nachname"/>
</type>

<!-- Definition eines abgeleiteten Typs -->
<type name="nameMitTitel" source="personenName" derivedBy="extension">
  <element name="Titel" minOccurs="0"/>
</type>

<!-- Elementdefintion auf der Basis des abgeleiteten Typs -->
<element name="adressat" type="nameMitTitel"/>

<!-- Verwendung des abgeleiteteten Typs -->
<adressat>
  <titel>Dr.</titel>
  <vorname>Gudrun</vorname>
  <nachname>Maier</nachname>
</adressat>
```

Codebeispiel 46: Beispiele für Strukturbestandteile eines XML Schemas

An den Beispielen lassen sich bereits einige Eigenschaften der Konstrukte in einer XML-Schemadefinition zeigen. Nachfolgend sollen Unterschiede zum Mechanismus der DTD-Definition von SGML und XML hervorgehoben werden:

- Die Schemadefinition erfolgt mit den syntaktischen Mitteln von XML; die Strukturelemente des Standards lassen sich durch sich selbst (vgl. THOMPSON et al. 1999: Kap. Appendix A) bzw. durch eine XML-DTD beschreiben (THOMPSON et al. 1999: Appendix B).
- Durch den Mechanismus der Typdefinition ist es möglich, nicht nur aus einer differenzierten vordefinierten Typenmenge auszuwählen und Einschränkungen für Wertebereich und Kardinalität anzugeben (s. u.), sondern es können auch abgeleitete Typen gebildet werden (Prinzipien der Unterklassenbildung und der Vererbung).
- Bei der Ableitung neuer Typen werden Erweiterung (*extension*) und Restriktion (*restriction*) unterschieden.
- Die aus der objektorientierten Programmierung bekannten Merkmale, Klassen als *abstrakt* oder *final* zu kennzeichnen, wenn von ihnen entweder keine Instantiierung erfolgen soll oder von ihnen keine weiteren Klassen abgeleitet werden sollen, finden für die Typdefinition Anwendung.
- Ebenfalls in Analogie zu objektorientierten Konzepten wird ein impliziter Basistyp für alle komplexen Datentypen angenommen (sog. *ur-type*, Thompson et al. 1999: Kap. 3.6.6), den man sich folgendermaßen vorstellen kann:

  ```
  <type name="ur-type" content="mixed"><any/><anyAttribute/></type>
  ```

Der zweite Teil des Standardisierungsvorschlags – *XML Schemas: Datatypes* – führt einen Mechanismus der Datentypisierung ein, der folgenden Anforderungen gerecht werden soll (vgl. BIRON & MALHOTRA 1999: Kap. 1.2)

- Eine ausreichende Menge primitiver Datentypen soll fest vordefiniert zur Verfügung stehen und sich nicht nur an SGML, sondern auch an Programmier- und Datendefinitionssprachen wie Java oder SQL orientieren,
- das Typsystem soll sich für den Datenaustausch mit unterschiedlichen Typen von Datenbanksystemen eignen,
- Anforderungen der lexikalischen Repräsentation von Daten sollen von denen des zugrundeliegenden Datenmodells unterschieden werden und
- die Definition neuer Typen durch Ableitung von bestehenden (primitiven) Typen unter Angabe zusätzlicher Einschränkungen (Wertebereich, Genauigkeit, Format etc.) soll möglich sein.

Die für ein XML Schema definierten Datentypen lassen sich nach folgenden Kriterien klassifizieren (BIRON & MALHOTRA 1999: Kap. 2.4):

- *atomare* und *zusammengesetzte* Datentypen,
- *primitive* und *generierte* (d. h. durch Bezug auf andere Datentypen definierte) Datentypen und
- *vordefinierte* und *benutzerdefinierte* Datentypen.

Zu jedem Datentyp können Einschränkungsmöglichkeiten (*facets*) definiert werden, die Eigenschaften bestimmen wie den Wertebereich, eine Ordnungsrelation, die Kardinalität, die Länge (der lexikalischen Repräsentation) oder zusätzliche Formatbestimmungen (z. B. bei Zeitangaben und Datumsformaten) bestimmen). Die Basis des Typensystems bilden die atomaren, primitiven und vordefinierten Datentypen:

Datentyp	*Bedeutung*	*Darstellung/Beispiel*
string	Basistyp für Zeichenketten	*Universal Character Set/Unicode*-Zeichenfolge (ISO 10646)
boolean	Datentyp für Wahrheitswerte	true, false
float, double	Gleitkommazahl (nach IEEE 754) mit Darstellungsbreite von 32/64 Bit	0, NAN (*not a number*), 345.532E3, -5E4
decimal	Kommazahl beliebiger Genauigkeit	8924.478219
timeInstant	Zeitangabe im Format CCYY-MM-DDThh:mm:ss.sss (Jahrhundert, Jahr, Monat, Tag, Datum-Zeit-Trenner (T), Stunde, Minute, Sekunde (mit beliebiger Genauigkeit der Angabe von Sekundenbruchteilen, nach ISO 8061)	2000-02-14T20:54:21+01:00 (mit Angabe der Zeitverschiebung +1 gegenüber der *coordinated universal time*)
timeDuration	Angabe einer Zeitdauer im Format PnYnMnDTnHnMnnS (P Vorzeichen, n Anzahl, T Datum-Zeit-Trenner)	1Y4M12DT3H15M
recurringInstant	Wiederkehrende Zeitpunkte im Format eines timeInstant, wobei je zwei Zeichen einer Einheit durch – ersetzt werden	--02-14T20:54:00+01:00 (jedes Jahr am 14. 2. um 20 Uhr 54 MEZ)
binary	Binärdaten	endliche Bitfolge
uri	*Uniform Resource Identifier* (vgl. BERNERS-LEE, FIELDING & MASINTER 1998 [RFC 2396])	http://buch.org/seite1.xml

Tabelle 23: Atomare primitive Datentypen in XML Schemas

Über die in Tabelle 23 hinaus genannten atomaren primitiven Datentypen werden zusätzlich *abgeleitete* Datentypen eingeführt wie:

- integer für Ganzzahlen (Basis: decimal),
- Language für die Angabe von Sprachcodes (Basis: string),
- Name für XML-Bezeichner, die im Sprachstandard von XML vorgesehenen Datentypen (NMTOKEN, NMTOKENS, ID, IDREF, IDREFS) und
- metasprachliche Elemente einer DTD (ENTITY, ENTITIES, NOTATION)

Wie für die XML-Strukturen wird für die Datentypen sowohl eine Definition mit den Strukturen der XML Schemata selbst als auch durch eine XML-DTD angegeben (BIRON & MALHOTRA 1999: Appendizes A & B).

Dieser Standardisierungsvorschlag befindet sich noch in der Überarbeitung. Als Tendenz ist jedoch für XML die Ablösung von den Einschränkungen und Mechanismen des Mutterstandards SGML deutlich zu erkennen: An die Stelle einer DTD als Definition zulässiger Dokumentstrukturen tritt ein direkt mit den Mitteln von XML kodiertes Datenschema, das nur noch auf einer Metaebene durch eine DTD beschrieben ist. Dies gilt auch für die am weitesten verbreitete DTD, HTML, für deren modulare Weiterentwicklung XHTML eine Beschreibung durch XML Schemata vorgesehen ist (vgl. unten Kap. 6.2). Gleichzeitig finden die Konzepte aus Programmiersprachen bzw. der Datenmodellierung Eingang in die Definition von Dokumentstrukturen. Dies ist im Kontext des Modells für dynamische elektronische Bücher überall dort relevant, wo für eingebettete Komponenten oder mit dem elektronischen Buch verbundene Dienste über Datenschnittstellen verfügen, die mit den Mitteln von XML beschrieben werden sollen bzw. die XML als Datenaustauschformat verwenden. XML-Werkzeuge (Parser, Transformatoren) sind allerdings noch nicht an den neuen Standard angepasst sind oder implementieren ihn noch nicht vollständig, d. h. sie greifen für die Validierung von Dokumenten nicht auf Schemata, sondern nach wie vor auf DTDs zurück. Hinweise auf Ansatzpunkte für eine präzisere Definition von XML-Dokumenten durch XML Schemata finden sich in den Kommentaren zu den für den Prototyp eines dynamischen elektronischen Buchs entwickelten Dokumententypgrammatiken (vgl. unten Anhang 17.1).

5.2.7 Verarbeitung von XML

Für die Nutzung der XML-basierten Standards sind geeignete Werkzeuge erforderlich. Seit der Einführung von XML hat sich schnell eine breite Palette von Werkzeugen herausgebildet. Nachfolgend sollen

- grundlegende Programmierschnittstellen für XML, die Ausgangspunkt der Realisierung einer Vielzahl XML-basierter Werkzeuge sind, und
- die wichtigsten Typen von XML-Werkzeugen eingeführt werden.

5.2.7.1 APIs für die Manipulation von XML-Dokumenten

Für die Manipulation und Verarbeitung in XML kodierter Daten sind zwei *application programming interfaces* (APIs) entwickelt worden, die große Verbreitung gefunden haben:

- Das von David MEGGINSON entwickelte *Simple API for XML* (vgl. *http://www.megginson.com/SAX/index.html*) und
- das *Document Object Model* (DOM), ein vom *World Wide Web Consortium* vorgelegter Standard für die Implementierung von XML-Werkzeugen.

Da im Rahmen dieser Arbeit vor allem Werkzeuge eingesetzt werden, die das *Document Object Model* implementieren, wird es nachfolgend ausführlicher beschrieben; SAX wird lediglich knapp eingeführt.

5.2.7.1.1 Simple API for XML

Das *Simple API for XML* (SAX) fasst in seiner ursprünglichen Fassung[72] Schnittstellen und Klassen für die Manipulation von XML-Dokumenten zusammen,

- die von einem Parser zu implementieren sind (u. a. die Schnittstelle org.xml.sax.Parser) und
- die von XML verarbeitenden Anwendungen eingesetzt werden (insbesondere die Schnittstelle org.xml.sax.DocumentHandler, die mit Hilfe eines Parsers die Dokumentverarbeitung übernimmt). Zusätzlich beinhaltet SAX
- Hilfsklassen für die Verarbeitung von Eingabequellen und zur Fehlerbehandlung (Ausnahmen).

SAX-basierte Parser repräsentieren das Eingabedokument nicht als Baum, dessen Knoten die Elemente und Elementinhalte eines XML-Dokuments sind, sondern der Parsingalgorithmus arbeitet *ereignisgesteuert*: Das Auffinden von XML-Merkmalen (Starttags, Endtags etc.) wird direkt an das verarbeitende System gemeldet, ohne dass der Parser hierzu einen vollständigen Baum des Dokuments generieren müsste. Mit dieser einfachen Vorgehensweise kann die Speicherlast des XML-Verarbeitungssystems reduziert werden (vgl. dazu ausführlich *http://www.megginson.com/SAX/event.html*, mit Beispielen). SAX wird von den meisten XML-Parsern implementiert und hat, nicht zuletzt, da es das erste spezifizierte XML-API war, weite Verbreitung gefunden (vgl. JOHNSON 2000).

5.2.7.1.2 Document Object Model (DOM)

Das *Document Object Model* (vgl. APPARAO et al. 1998; WOOD et al. 1999) definiert einen Standard für den programmtechnischen Zugriff auf die Elemente HTML- und XML-kodierter Information. Dazu gehören unter anderem

- die Festlegung einer Typenhierarchie in Analogie zu den Strukturelementen einer DTD (*Document, Element, Attribute, Text, Processing Instruction, Comment* etc.) und
- die Definition einer *interface definition Language* (IDL) als Schnittstelle zwischen Dokumentmodell und Entwicklungsumgebung (z. B. einer Programmiersprache).

Auf dieser Basis können *Application Programming Interfaces* (APIs) für HTML- oder XML-kodierte Elemente in einheitlicher Weise aufgebaut werden, d. h. die programmtechnische Manipulation strukturierter Dokumente wird auf der Basis einer einheitlichen Zugriffssprache ermöglicht. Die im DOM definierten Methoden erlauben

- den Aufbau von XML-/HTML-Dokumenten,
- die Manipulation von Elementbäumen solcher Dokumente,
- die Navigation in der Dokumentrepräsentation und
- das Ändern und Löschen von Dokumenten und Dokumentteilen.

---

[72] Version 2 von SAX, die auch Namensräume unterstützt, ist derzeit in einer Betaversion verfügbar, vgl. *http://www.megginson.com/SAX/SAX2/*.

Als generisches Interface ist das DOM nicht auf eine einzelne Anwendung wie z. B. ein Viewing-System (Webbrowser) hin ausgelegt, sondern soll als Basis jeder Software dienen, die XML-/HTML-Dokumente generiert, darstellt oder manipuliert.

Das *Document Object Model* ist als Objektmodell im Sinne der objektorientierten Modellierung aufgebaut, da die durch eine DOM-basierte Anwendung verarbeiteten Dokumente als Objekte repräsentiert werden und das DOM für diese Objekte (bzw. die zugrundeliegenden Klassen) geeignete Zugriffs- und Manipulationsmethoden und deren Semantik sowie die Zusammenhänge zwischen den verschiedenen Klassen definiert.

Die Objektrepräsentation eines XML-Dokuments[73] stellt ein Dokument als Baum (*tree*) bzw. Wald (*forest*; *grove*, vgl. oben Kap. 7.1.1 zu DSSSL) von Elementknoten bzw. Elementinhalten dar, wie das nachfolgende Beispiel illustrieren soll:

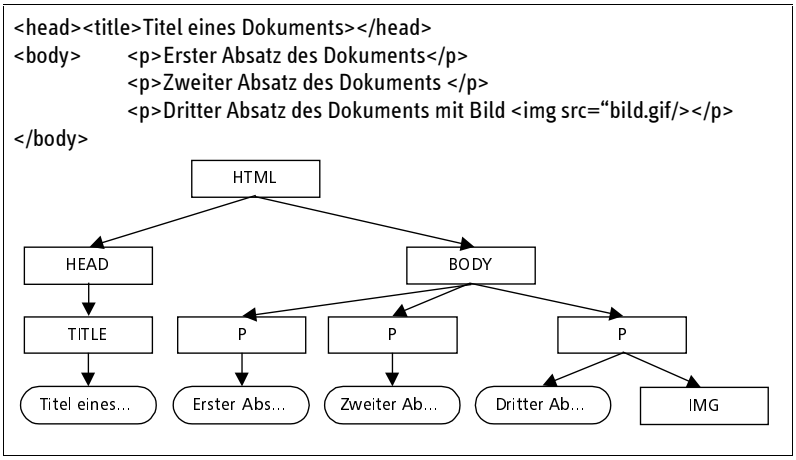

Abbildung 27: Beispieldokument und Repräsentation in DOM

Eine Spezifikation einer DOM-IDL zwischen HTML/XML und Java ist in APPARAO 1998 enthalten. Sie gibt die abstrakte Schnittstellendefinition für die Strukturbestandteile von HTML und XML und ihre Umsetzung durch ein Java-API an. Die DOM-Spezifikationen unterscheiden verschiedene Niveaus: Level 1 (APPARAO et al. 1998) definiert die elementaren Strukturbestandteile definiert, während Level 2 (WOOD et al. 1999) weitergehend ein API für die Verwendung von Namensräumen (XML Namespaces, vgl. Kap. 5.2.3), *style sheets* bzw. CSS (vgl. Kap. 7.1.2f.) und die Ereignisverarbeitung beinhaltet. Für DOM Level 1 liegen bereits viele Softwarewerkzeuge vor, während Level 2 noch im Entwurfsstatus ist.

DOM Level 1 definiert in Analogie zu den Strukturbestandteilen von XML die in Tabelle 24 gezeigten Schnittstellen, wobei grundsätzlich ein `Node`-Objekt für den Aufbau des Dokumentbaums verwendet wird. Knotenobjekte können Kinder haben und sind dann innere Knoten des Dokumentbaums oder sie haben keine Kinder und sind äußere Knoten bzw. Blätter des Baums. Die nachfolgende Tabelle zeigt die einzelnen Bestandteile im DOM und deren mögliche Kindknoten:

[73] APPARAO et al. 1998: 12 weisen darauf hin, dass DOM als Objektrepräsentation für XML- und HTML-Dokumente dient, nicht umgekehrt für den Einsatz von XML zur Repräsentation von Objekten gedacht ist. Allerdings ist diese Modellierungsrichtung ebenfalls denkbar, wie die Versuche zeigen, XML als Modellierungsinstrument für den Aufbau von Software-Komponenten (insb. Java Beans) einzusetzen, vgl. JOHNSON 1999, 2000.

DOM-Strukturelement	*mögliche Kinder*
Document	Element (max. ein Element), ProcessingInstruction, Comment, DocumentType
DocumentFragment	Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
DocumentType	– keine Kindknoten –
EntityReference	Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
Element	Element, Text, Comment, ProcessingInstruction, CDATASection, EntityReference
Attr	Text, EntityReference
ProcessingInstruction	– keine Kindknoten –
Comment	– keine Kindknoten –
Text	– keine Kindknoten –
CDATASection	– keine Kindknoten –
Entity	Element, ProcessingInstruction, Comment, Text, CDATASection, EntityReference
Notation	– keine Kindknoten –

Tabelle 24: Strukturelemente im Document Object Model

Für die Manipulation von Knotenmengen stehen eine Reihe von Datenstrukturen (NodeList, NamedNodeMap) zur Verfügung, die verwendet werden, um die Attributwerte eine Elements eines Dokuments zu verarbeiten. Level 1 des *Document Object Model* gliedert sich inhaltlich in zwei Teile: Das Core Model, das wie oben beschrieben die wesentlichen Strukturen für die Verarbeitung von XML-Dokumenten enthält, sowie das *Document Model Level 1* für HTML, das als Spezialisierung des Core DOM spezifische Klassen und Methoden für die Verarbeitung von HTML definiert, z. B. HTMLFormElement, HTMLFontElement oder HTMLMetaElement als IDL-Schnittstellen (*interface definition Language*, vgl. unten Kap. 12.1.3 zu CORBA), die von der Basis-Schnittstelle HTML-Element abgeleitet sind.[74]

Um Implementierungen in unterschiedlichen Sprachen zu ermöglichen, ist das DOM als Sammlung abstrakter Schnittstellen nach dem CORBA-Standard bzw. der CORBA-*Interface Definition Language* spezifiziert (CORBA 2.2, vgl. APPARAO et al. 1998: 10); für die Programmiersprache Java und JavaScript enthält die Spezifikation unmittelbar die Definition des *CORBA Language Bindings* für DOM. Das nachfolgende Codebeispiel zeigt das Java-Binding für die Schnittstelle Element des DOM Core Level 1 – die dort aufgeführten Methoden müssen z. B. von einem DOM-basierten XML-Parser implementiert werden, um den DOM-konformen Zugriff auf XML-Elemente zu gewährleisten.

```
public interface Element extends Node
{  public String getTagName();
   public String getAttribute(String einAttributName);
   public void setAttribute(String einAttributName, String einAttributWert) throws DOMException;
   public void removeAttribute(String einAttributName) throws DOMException;
   public Attr getAttributeNode(String einAttributName);
   public Attr setAttributeNode(Attr einAttribut) throws DOMException;
   public Attr removeAttributeNode(Attr einAttribut) throws DOMException;
   public NodeList getElementsByTagName(String einElementName);
   public void normalize();                                                    }
```

Codebeispiel 47: Java-Binding für die Schnittstelle Element in DOM Level 1

[74] Diese Sonderstellung ist durch die große Bedeutung von HTML bedingt; an sich wäre es denkbar, eine Spezialisierung des DOM für beliebige DTDs zu definieren; es erscheint aber einfacher, jeweils auf ein gemeinsames Modell, das *core model*, zurückzugreifen, vgl. WOOD et al. 1999: 18 ff. [Kap. 2].

Wie die Methodennamen bereits andeuten, kann man mit diesen Methoden die Attribute eines Elements auslesen, modifizieren oder löschen oder sich auf der Basis von Elementnamen eine Liste der zu einem Element gehörenden (eingebetteten) Kindknoten ausgeben lassen und so den Dokumentbaum erstellen.

Aufbauend auf dem DOM Level 1 Core definiert DOM Level 2

- sinnvolle *Erweiterungen* des Level 1 Core wie die notwendige Funktionalität für die Erzeugung von Dokumenten, den Import von Dokumenten aus anderen Dokumenten und die Ermittlung des zu einem Attribut gehörenden Elementes im Dokumentenbaum und
- vollständige Neudefinitionen zusätzlicher Leistungsbereiche des DOM.

Zu letzteren gehören

- der Zugriff auf *XML Namespaces*, entweder durch *Ergänzung* des DOM Level 1 Core oder durch Neudefinition der im DOM Level 1 enthaltenen Schnittstellen,[75]
- eine generische Schnittstellenspezifikation für die Einbindung von *style sheets*, die nicht an ein bestimmtes *style sheet*-Modell wie CSS oder DSSSL gebunden ist (WOOD et al. 1999: 33 ff.: Kap. 3),
- Schnittstellen für die Verwendung von *Cascading Style Sheets* zusammen mit den DOM Core- bzw. DOM HTML-Schnittstellen (vgl. WOOD et al. 1999: 39 ff.),
- Ein Nachrichtenmodel für DOM, das nicht nur spezifiziert, wie Nachrichten versandt und im Dokumentbaum verarbeitet werden sollen, sondern auch Standardnachrichtentypen für die Interaktion mit Dokumenten definiert. Ein besonderes und naheliegendes Ziel ist die Modellierung der in den gängigen Browsern (Netscape, Microsoft Internet Explorer) verwendeten Nachrichtentypen, vgl. Wood et al. 1999: 89 ff. wie sie bisher bereits z. B. durch eingebettete JavaScript-Event Handler verarbeitet werden können.
- Schnittstellen für den einfachen Zugriff und die Iterierung von Dokumentenbäumen mit Hilfe von Iteratoren (*Iterator*), Dokumentfiltern, die bestimmte Knoten aus einem Dokument herausfiltern können (*NodeFilter*) und Baumtraversierungsklassen (*Treewalker*) und schließlich
- Schnittstellen für die Repräsentation von zusammenhängenden Dokumentbereichen (*document ranges*), wobei ein Bereich als der Inhalt zwischen zwei Endpunkten (Knoten im Dokumentbaum) definiert ist, der durch den Bereich ausgewählt wird.

Auch wenn bisher noch keine vollständige Implementierung des DOM Level 2 vorliegt und die Implementierung von DOM Level 1 in den gängigen WWW-Browsern noch Defizite aufweist (vgl. KROCK 1999), so spielt das *Document Object Model* für die Manipulation elektronischer Bücher bereits jetzt an verschiedenen Stellen eine wichtige Rolle:

- Generell bei der Aufbereitung und Generierung XML-basierter Texte, z. B. zur Validierung von Dokumenten mit Hilfe einer XML-DTD[76], der Überführung von Dokumenten aus einer DTD in eine andere DTD, d. h. die programmtechnische Umsetzung der in XSL definierten Transformationsprozesse für die Dokumentstruktur,

[75] Da DOM Level 2 derzeit noch in der Entwurfsphase ist, kann nicht sicher beurteilt werden, welche der beiden Realisierungsalternativen schließlich den Vorzug erhalten wird; kurzfristig ist vermutlich die erste Lösung besser geeignet, um bestehende DOM-Werkzeuge wie XML-Parser um DOM Level 2-Funktionalität zu erweitern.

[76] Im Rahmen des Referenzprojekts wurde z. B. ein DOM-basierter Parser eingesetzt, um die Konvertierung der Dokumente nach XHTML zu prüfen und optimieren.

- bei der Dokumentengenerierung auf der Serverseite bzw. beim Aufbau einer Dokumentrepräsentation aus einem XML-basierten Speicherformat, z. B. in einer objektorientierten Datenbank),
- bei der Implementierung von Buchfunktionalität unter Nutzung eines WWW-Browsers (Nutzung der DOM-analogen Dokumentmodelle des Browsers, auf die über JavaScript zugegriffen kann, z. B. Navigationssteuerung, Hyperlinks etc.)

Mittlerweile implementieren alle weit verbreiteten XML-Parser neben SAX auch das *Document Object Model*, zumindest Level 1. Eine Übersicht dieser Werkzeuge findet sich auf den XML-Seiten von Robin COVER bei OASIS (COVER 1999)

5.2.7.2 Werkzeuge

Wie bereits oben in Kap. 5.2.7.2 auch für SGML sollen auch für XML typische Klassen XML-verarbeitender Werkzeuge aufgezeigt werden. Ihnen ist gemeinsamen, dass sie in der Regel auf der Basis der beiden voranstehend beschriebenen APIs (SAX und DOM) und mit Hilfe objektorientierter Programmiersprachen (Java, C++) realisiert sind. Im Gegensatz zu SGML hat sich das Angebot an XML-Werkzeugen sehr schnell und dynamisch entwickelt und es liegen mittlerweile vielfältige Werkzeuge für alle relevanten Aufgaben vor, die zum Teil auch als *open source*-Projekte frei verfügbar sind. Dabei werden allerdings noch nicht alle neueren Spezifikationen (z. B. XLink, XML Schemata) auch von allen Werkzeugen unterstützt.[77]

Die nachfolgende Tabelle fasst die wichtigsten Typen von Werkzeugen unter Angabe von Beispielen zusammen und zeigt auf, inwieweit für diese Arbeit von Bedeutung sind:

Typ	*Funktion*	*Beispiel/Beschreibung*	*Relevanz*
Entwurfs-werkzeuge	Entwurf von XML-DTDs	*IBM Visual DTDTool*, formularbasierter Entwurf von Elementhierarchien, Darstellung der DTD als *tree view*.	Teilweise bei Entwurf der DTDs
Parser	Analyse: Verarbeiten von XML-Dokumenten; Grundlegendes Werkzeug für alle anderen Typen	*Apache Xerxes*-Parser mit Unterstützung für Namensräume, DOM 1/2, SAX 2 und XML Schema (partiell), vgl. h*ttp://xml.apache.org/*.	Einsatz verschiedener Parser bei unterschiedlichen Verarbeitungsschritten (Offline-Dokumenttransformation; Verarbeitung und Koordination von Inhalten, Diensten und Komponenten).
Editoren	Bearbeitung: Kodierung von XML-Dokumenten	*XMetaL*, graphischer XML-Editor als Weiterentwicklung des HTML-Editors HoTMetaL, *http://www.softquad.com/products*	Kein Einsatz von Editoren, da die XML-Bearbeitung programmgestützt und durch Skripte erfolgte.
Viewer	Präsentation: Betrachten von XML-Dokumenten unter Zuhilfenahme von Präsentationsinformation (*style sheets*)	*Microsoft Internet Explorer* V. 5, vgl. *http://www.microsoft.com.*	XML-Transformation wird durch den Server durchgeführt, daher kein eigener Viewer erforderlich.

[77] Aktuelle Übersichten zu XML-Werkzeugen finden sich bei Cover 2000 sowie auf XML- Websites wie *http://www.xml.org*, *http://www.xml.com* oder *http://www.garshol.priv.no/download/xmltools/cat_ix.html*.

Typ	*Funktion*	*Beispiel/Beschreibung*	*Relevanz*
Datenbanken	Speicherung: Datenbanksysteme zur Speicherung XML-kodierter Daten und Unterstützung des Zugriffs durch entsprechende *XML query languages* wie XQL[78]	*Tamino*, ein XML-basiertes Datenbanksystem, vgl. *http://www.softwareag.com/tamino.*	Im Rahmen dieser Arbeit keine datenbankbasierte Speicherung der Daten.
Transformationswerkzeuge	Werkzeuge für die Transformation von XML-Dokumenten durch XSL-*style sheets*.	*Saxon*, XSL/T-basierter Transformationsprozessor für XML-Dokumente, vgl. *http://users.iclway.co.uk/mhkay/saxon/.*	Einsatz von SAXON für die Offline-Vorverarbeitung der Inhalte (Partitionierung).
Integrierte Publikationssysteme für das World Wide Web	*XML Web Publishing*: Serverseitige Werkzeuge für die Transformation und Präsentation XML-basierter Dokumente im World Wide Web (Viewerfunktion)	*Cocoon*, ein generisches System für die Verarbeitung von XML-Dokumenten im World Wide Web, vgl. *http://xml.apache.org.*	Cocoon wurde im Rahmen dieser Arbeit als Ausgangspunkt für die Realisierung des Buchservers verwendet.

Tabelle 25: Übersicht zu verschiedenen Typen XML-verarbeitender Werkzeuge

[78] Die *XML Query Language* XQL ist ein bisher nicht standardisierter Vorschlag für eine Abfragesprache für XML-basierte Daten, vgl. ROBIE 1998 und ABITEBOUL, BUNEMAN & SUCIU 1999. Das *World Wide Web Consortium* hat erst kürzlich die Anforderungen an XML-basierte Abfragesprache spezifiziert, vgl. FANKHAUSER, MARCHIORI & ROBIE 2000.

6 Standardisierte Anwendungen von SGML und XML

Nachdem die Grundprinzipien von SGML und XML als Metasprachen für die Informationsstrukturierung eingeführt wurden, ist nun zu untersuchen, welche vordefinierten und standardisierten SGML- bzw. XML-Anwendungen bzw. *document type definitions* sich für die Kodierung dynamischer elektronischer Bücher eignen, bzw. wie auf dieser Basis eigene Formate entwickelt werden können. Folgende Formate kommen aufgrund ihrer weiten Verbreitung bzw. ihres Detailliertheitsgrads für das elektronische Publizieren in Betracht:

- Die *HyperText Markup Language* (HTML) bzw. deren XML-basierte Fassung *Extensible HyperText Markup Language* (XHTML),
- die Dokumentformate der *Text Encoding Initiative* (TEI),
- *DocBook* als differenziertes Format für die elektronische Kodierung von Softwaredokumentation,
- der *open eBook*-Standard für die Kodierung elektronischer Bücher insbesondere im Kontext der sich entwickelnden Hardwarelösungen für elektronische Buchbetrachter (vgl. oben Kap. 3.3) sowie
- Spezialstandards auf der Basis von XML wie die *Mathematical Markup Language* (MathML).

6.1 Hypertext Markup Language

Die *Hypertext Markup Language* ist als Kodierungsformat der Dokumente im World Wide Web die mit Abstand am weitesten verbreitete Sprache für die Strukturierung von Hypermedia-Dokumenten. Sie wurde ursprünglich von Tim BERNERS-LEE konzipiert (vgl. BERNERS-LEE 1999). Als Anwendung von SGML wird sie vom W3C fortgeschrieben; derzeit liegt die HTML-Spezifikation in der Version 4.0.1 vor (vgl. RAGGETT, LE HORS & JACOBS 1999). Die nachfolgende Darstellung der wichtigsten Merkmale von HTML verfolgt mehrere Ziele:

- Die Strukturierungsmöglichkeiten von HTML sollen veranschaulicht werden; es ist zu klären, in welchem Umfang und für welche Funktionen die gegebenen Möglichkeiten für dynamische elektronische Bücher ergänzt werden müssen.
- Es soll deutlich gemacht werden, welche Vorteile die Verwendung einer Metasprache und die mit ihr verbundene Freiheit zur Neudefinition von Dokumentenformaten mit sich bringen.
- HTML selbst ist Ausgangspunkt einiger Weiterentwicklungen (XHTML, OEB); zu deren Verständnis ist die Kenntnis der Grundkonzepte von HTML notwendig.
- Schließlich sollen Ausschnitte aus den für das Referenzprojekte kodierten HTML-Dokumenten die praktische Anwendung von HTML und ihre Grenzen aufzeigen.

HTML-Dokumente bestehen in der Regel aus drei Teilen:

- einer `DOCTYPE`-Anweisung, die die verwendete HTML-DTD spezifiziert,
- einem Dokumentkopf (`<HEAD>…</HEAD>`), in dem der Dokumenttitel, die Einbindung von *style sheets* und/oder zusätzliche Verweise, Metadaten als Attribut-Wert-Listen in

META-Elementen sowie ggf. Skriptcode oder Verweise auf Skriptdateien enthalten sind und
- dem Dokumentkörper, der den Inhalt des HTML-Dokuments enthält (<BODY> … </BODY>).

6.1.1 Text- und Zeichenmarkup

Zur Aufbereitung des Dokumentinhalts unterscheidet HTML ein absatzbezogenes und ein zeichenorientiertes Inhaltsmodell. Das absatzbezogene Modell umfasst eine Reihe von Elementen für die logische Strukturierung von Texten; dazu gehören

- Gliederungsebenen für Überschriften (<H1>... <Hn>)
- allgemeine Absatzformate (<P>, <PRE>)
- ungeordnete und geordnete Listen mit Listeneinträgen und Definitionslisten (<OL>, <UL>, <LI>, <DL>, <DD>)
- Absatzformate für ausgewählte spezifische Anwendungen (<ADDRESS>, <BLOCKQUOTE>)

Auf der Ebene der Textformatierung innerhalb von Absätzen sind zu unterscheiden:

- Logische Auszeichnungsmarken, die die Funktion eines Ausdrucks oder einer Textpassage kennzeichnen und die fast alle in der Parameterentität %phrase zusammengefasst sind, wie die nachfolgende Tabelle zeigt:

Element	*Interpretation*
<ABBR>	Abkürzung, Kurzform
<ACRONYM>	Akronym
<CITE>	Zitat oder Quellenverweis
<CODE>	Quellcode
<DFN>	Definition
<EM>	*emphasis*, logische Hervorhebung
<KBD>	*keyboard*, Text, der vom Benutzer eingegeben werden soll
<Q>	Kurzes Zitat
<SAMP>	*sample*, Beispiel
<STRONG>	Hervorhebung (starke Emphase)
<VAR>	Variableninstanz (insb. in Quellcode)

Tabelle 26: Logische Textauszeichnung in HTML

- An der Präsentation orientierte Marken für die Textauszeichnung:

Element	*Interpretation*
<B>	*bold*, Fettdruck
<BIG>	Satz in größerer Schrift
<I>	*italics*, Kursivsatz
<SMALL>	Satz in kleinerer Schrift
<STRIKE>	durchgestrichene Schrift
<TT>	*teletype*, Satz in Nicht-Proportionalschrift
<U>	*underlined*, Textunterstreichung

Tabelle 27: Schriftformatierung in HTML

Die Elemente für die Schriftformatierung sind zwar noch Bestandteil der HTML-DTD; da sie aber der Trennung von logischer Auszeichnung und Kodierung der Präsentationsei-

genschaften zuwiderlaufen, müssen sie zukünftig durch entsprechende Layoutangaben in *style sheets* ersetzt werden (vgl. RAGGETT, LE HORS & JACOBS 1999: Kap. 15.2.1).

6.1.2 Weiterführende Funktionalität in HTML

Zu den Textauszeichnungen treten weitere Elemente für besondere Präsentationsformate, die Einbettung von Hypertextverknüpfungen bzw. Multimediaelementen, den Aufbau von Formularen und die Fenstersteuerung:

- Über die <TABLE>-Marke lassen sich Tabellen erstellen, deren Inhalt durch <TR> (*table row*) und <TD>-Marken (*table data*) gefüllt werden kann.
- Das <FORM>-Element und die zugehörigen Auszeichnungsmarken für Steuerelemente (<BUTTON>, <INPUT>, <SELECT>, <TEXTAREA> etc.) erlauben den Aufbau von Formularen, deren Inhalt lokal über ein Skript oder durch Programme auf dem Server ausgewertet werden können.
- Mit den Elementen <FRAME> und <FRAMESET> lassen sich Steuerdateien erzeugen, die den Darstellungsbereich des Präsentationssystems flexibel in Zeilen und Spalten aufgliedern und so die gleichzeitige Darstellung mehrerer HTML-Dateien zulassen.
- Bilder, Programme (z. B. Java-Applets, proprietäre plug-in-Anwendungen) lassen sich mit den Elementen <IMG> (Bilder) bzw. <OBJECT> (generische Marke für einzubettende Inhalte) in ein HTML-Dokument integrieren.[79] Um für zukünftige Medientypen offen zu sein, schlägt die aktuelle HTML-4.0.1-Spezifikation die ausschließliche Verwendung der <OBJECT>-Marke für alle einzubettenden Inhalte vor. Abgesehen von der Möglichkeit, über <PARAM>-Elemente Parameterwerte an das eingebettete Objekt zu übergeben, werden die eingebetteten Inhalte aber als *black-box*-Container behandelt.[80] Sog. *image maps* ermöglichen die Zuordnung von sensitiven Flächen zu Bildbereichen, bei denen der Benutzer durch Mausklick z. B. einen Hyperlink traversieren kann (Elemente <MAP>, <AREA>).
- Schließlich erlaubt das <A>-Element (*anchor*, Anker) die Kodierung von Hypertextverknüpfungen; sie sind unidirektionale 1:1-Verknüpfungen. Die <A>-Marke kann auch zur Identifikation eines Sprungziels (Attribute name, id) verwendet werden.

Die noch näher zu behandelnde Möglichkeit der Verwendung von *style sheets* hat mittlerweile zu einer Reduktion des Umfangs der HTML-DTD geführt, da die Empfehlung gilt, dass alle präsentationsbezogenen Markupelemente (z. B. Farb- oder Hintergrundattribute in Textauszeichnungsmarken) nicht im HTML-Dokument selbst, sondern über entsprechende Formatanweisungen in einem beigeordneten *style sheet* erfolgen sollen.

6.1.3 Diskussion

Die *Hypertext Markup Language* hat weite Verbreitung gefunden, da sie einfache Mittel für die Text- und Dokumentenstrukturierung bereitstellt, die für viele Anwendungen ge-

---

[79] Die vielfach verwendeten Marken <APPLET> für Java-Applets bzw. <EMBED> für *plug-in*-Anwendungen sind entweder nicht mehr Teil der HTML-Spezifikation oder als *deprecated* gekennzeichnet, vgl. RAGGETT, LE HORS & JACOBS 1999: Kap. 13.1.

[80] Schnittstellen zwischen der HTML-Datei und dem eingebetteten Objekt ergeben sich erst durch die Nutzung von Skriptsprachen (z. B. *JavaScript*) und speziellen APIs für den Datenaustausch bzw. Funktionsaufrufe zwischen HTML-Datei und Einbettungen (z. B. *LiveConnect*).

eignet ist und die Möglichkeit des Aufbaus von Hypertexten zulässt. Seit ihrer Einführung haben zahlreiche Erweiterungen (Formulare, Programmierbarkeit durch Skripte, die Einbettungsmöglichkeiten für multimediale Inhalte und die Fenstersteuerung durch Frames) die Anwendungsmöglichkeiten deutlich erweitert. Die Reduktion der Leistungsfähigkeit von SGML auf ein einzelnes Anwendungsformat hat dem deklarativen Markup von Inhalten für viele Anwendungen im World Wide Web zum Durchbruch verholfen. Dieser Einfachheit stehen als Nachteile die fehlende Erweiterbarkeit und Anpaßbarkeit an differenziertere Formate gegenüber. Sie hat zu einer Vielzahl von Sonderlösungen geführt, bei denen mit Hilfe von Skripten etc. versucht wird, fehlende Funktionalität oder Aufbereitungsmöglichkeiten zu ergänzen (vgl. FLYNN 1997: 618 ff.).

Im Referenzprojekt *Multimediales Physikalisches Praktikum* wurde ungeachtet dieser Nachteile zunächst HTML als Aufbereitungsformat verwendet, da für SGML keine geeigneten Präsentationswerkzeuge zur Verfügung standen und davon auszugehen war, dass eine Anpassung an XML-basierte Standards problemlos zu bewältigen sein dürfte. Die entscheidenden Nachteile von HTML bei der Definition von Dokumentenformaten und der nur rudimentären deklarativen Einbindung multimedialer Inhalte konnten aber nur durch die angedeuteten Auswege (Skripte) beseitigt werden. Bevor die verschiedenen Möglichkeiten der Entwicklung von Auszeichnungsstandards diskutiert wird, die den Anforderungen eines komplexen Multimediaprojekts Rechnung tragen, sollen einige Alternativen und Weiterentwicklungen von HTML vorgestellt werden.

6.2 Extensible HyperText Markup Language

Die große Verbreitung von HTML und seine gute Eignung für eine Vielzahl von Publikationsvorhaben einerseits, die Vereinfachungen von XML gegenüber SGML und seine erweiterten Möglichkeiten für die Einführung von Hyperlinks und die Adressierung von Dokumentensubstrukturen andererseits legen es nahe, HTML nicht als SGML-, sondern als XML-Anwendung zu modellieren. Unter dem Namen *Extensible Hypertext Markup Language* hat eine Arbeitsgruppe des W3C die Spezifikation einer XML-basierten *document type definition* für HTML vorgelegt (vgl. PEMBERTON et al. 1999). Sie hat folgende wesentliche Ziele:

- Die Definition einer Reformulierung von HTML als XML-Anwendung, um HTML-Dokumente in XML-basierter Software betrachten und bearbeiten zu können. Dies erfolgt unter Bewahrung der Kompatibilität zu existierenden HTML-basierten Softwaresystemen (Medientyp text/html).
- Die Anwendbarkeit des ursprünglich für HTML definierten *Document Object Model* (DOM, vgl. Kap. 5.2.7.1.2) und der hierfür vorhandenen Softwareinfrastruktur (Parser, Filter, Generatoren etc.) für XHTML-Dokumente.
- Die Definition von XHTML als modulare Menge von Substandards unter dem konzeptuellen Dach von XML, die jeweils untereinander interoperable Dokumente kodieren; zu ihnen gehört XHTML Basic, der Entwurf einer standardisierten Untermenge von XHTML, die besonders der Darstellung in Viewingsystemen mit eingeschränkten Betrachtungsmöglichkeiten dienen soll (Mobiltelephonie, persönliche Assistenten etc., vgl. ISHIKAWA et al. 1999).

Der letzte Punkt, die Modularisierung von XHTML in der Version 1.1 des Standards (W3C-Entwurf, Januar 2000), ist durch die Notwendigkeit einer feinkörnigeren Aufgliederung der Funktionalität von (X)HTML für unterschiedliche Anwendungszwecke motiviert:

> With the introduction of the XHTML family of modules and document types the W3C has helped the internet content-development community from the days of malformed, non-standard markup into the well formed, valid world of XML. [...] Content developers who base their content upon the functionality expressed in this specification can be confident that it will be consistently portable across XHTML family conforming user agents. [ALTHEIM & MCCARRON 2000A: Kap. 1]

ALTHEIM & MCCARRON 2000: Kap. 1 und ALTHEIM et al. 2000B: Kap. 4 beschreiben die Aufteilung der Markupelemente von HTML/XHTML auf einzelne Module, die nach Bedarf verwendet werden können und deren Zusammenspiel durch einen SGML catalog geregelt wird (vgl. ALTHEIM & MCCARRON 2000A: Appendix C.1 und oben Kap. 5.1.2). Die Module gliedern sich dabei in

- Basismodule (Struktur, Textauszeichnung, Hypertext, Listen),
- ein Applet-Modul,
- Module für die Texterweiterung (Präsentation, Editionsmerkmale),
- Formularmodule,
- Tabellenmodule,
- Module für *image maps*,
- ein Modul für eingebettete Objekte,
- Module für Frames,
- ein Ereignis- und ein Skriptmodul,
- je ein Modul für Metainformation, *style sheets*, das base- und das link-Element von (X)HTML sowie
- ein sog. *legacy*-Modul, das mittlerweile verworfene Elemente von HTML enthält.

Zusätzlich zur Modularisierung ist die Beschreibung von XHTML durch XML-Schemata vorgesehen, wird aber erst nach der Verabschiedung des XML-Schema-Standards erfolgen (vgl. oben Kap. 5.2.6 und ALTHEIM et al. 2000: Appendix A). Bei Betrachtung der für die Präsentation der im Referenzprojekt enthaltenen Daten erforderlichen Module von XHTML zeigt sich allerdings, dass die Modularisierung für diesen Anwendungsfall kaum Vorteile ergibt, da Elemente aus fast allen Modulen erforderlich sind; dies kann allerdings nicht verwundern, da ein dynamisches elektronisches Buch mit multimedialen Inhalten ein breites Spektrum an Funktionalität umfasst.

Für die Erstellung konformer XHTML-Dokumente müssen einige Konformitätsregeln erfüllt sein. Dazu gehören die erfolgreiche Validierung durch die vom W3C definierten XHTML-DTDs (XHTML-1.0-Strict-, XHTML-1.0-Transitional- und XHTML-1.0-Frameset-DTD), ein HTML-Wurzelelement mit Angabe des korrekten XML-Namensraums und die Spezifikation eines DOCTYPE-Elements. Das nachfolgende Beispiel zeigt ein konformes XHTML-Dokument:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/strict.dtd">
<html xmlns="http://www.w3.org/TR/xhtml1/strict" xml:lang="en" lang="en">
  <head>
    <title>Titel eines minimalen XHTML-Dokuments </title>
  </head>
  <body>
    <p>Textteil eines minimalen XHTML-Dokuments.</p>
  </body>
</html>
```

Codebeispiel 48: Beispiel eines minimalen konformen XHTML-Dokuments

Die Vereinfachungen und Änderungen von XML gegenüber SGML spiegeln sich in den Wohlgeformtheitsregeln für XHTML wieder, d. h., dass einige unter SGML zulässige Kodierungsformen in XHTML nicht erlaubt sind, wie

- sich überlappende Elemente (z. B. ein wichtiger Begriff),
- das Auslassen (SGML *tag omission*) von Endmarken nicht leerer Elemente (z. B. <p>erster Absatz<p>zweiter Absatz<p> ...)
- die Angabe von Attributwerten ohne Begrenzer bzw. die Attributminimierung (z. B. <applet code=x.class> oder <dl compact>),
- leere Elemente ohne entsprechende Markierung (z. B. <img src="x.gif">) und
- die Verwendung von Elementausschlüssen (*SGML element exclusions*), die in XML-basierten DTDs nicht zulässig ist.

Eine Mischung von verschiedenen DTDs bzw. unterschiedlichen XML-Namensräumen innerhalb eines XHTML-Dokuments ist möglich und an vielen Stellen sinnvoll, wie z. B. die Einbettung von MathML-Formelsatz in einem XHTML-Dokument (vgl. dazu den folgenden Abschnitt). XHTML führt nur minimale substantielle Änderungen gegenüber HTML V. 4.0 ein, wie die Definition des Euro-Symbols als Sonderzeichen in den für XHTML zulässigen Zeichensätzen. Seine hauptsächliche Bedeutung liegt in der (syntaktischen) Anpassung von HTML an XML; sie spielt im Rahmen dieser Arbeit eine wichtige Rolle, da sich die ursprünglich für HTML V 3.2 bzw. V 4.0 formatierten Dokumente nur nach einer Anpassung an XML (XHTML) durch XML-Softwarewerkzeuge verarbeiten lassen. Diese Anpassungen bestehen im Wesentlichen aus einer Reihe von Ersetzungsroutinen, die analog zu den oben genannten Beispielen nicht XML-kompatible SGML-HTML-Konstrukte konvertieren und in entsprechende XHTML-Elemente umwandeln.[81] Die Rückwärtskompatibilität zur weitgehend SGML-/HTML-basierten WWW-Software ist gleichzeitig ein großer Vorteil.

6.3 Open eBook

Der *open eBook*-Standard (OEB), der sich derzeit noch im Entwurfsstadium befindet (vgl. McCrary 1999 und *http://www.openebook.org*), definiert auf der Basis von XML bzw. HTML ein Kodierungsformat für elektronische Bücher, das sich vor allem für die Aufbereitung von Texten zur Nutzung in elektronischen Buchbetrachtungssystemen eignet (vgl. oben Kap. 3.3):

> The purpose of the Open eBook Publication Structure is to provide a specification for representing the content of electronic books. Specifically:
> - The specification is intended to give content providers (e. g. publishers, and others who have content to be displayed) and tool providers minimal and common guidelines which ensure fidelity, accuracy, accessibility, and presentation of electronic content over various electronic book platforms.
> - The specification seeks to reflect established content format standards.

[81] Dazu gehören die Einführung von *empty element*-Marken, die Einführung von Trennzeichen für Attributwerte, die Entfernung sich überkreuzender HTML-Marken etc. Die Umwandlung erfolgt mit einem XML-Parser, der die vorliegenden Dokumente mit der XHTML-DTD abgleicht; auftretende Fehler wurden klassifiziert und durch eine Sammlung von Ersetzungsroutinen beseitigt.

- The goal of this specification is to provide the purveyors of electronic-book content (publishers, agents, authors et al.) a format for use in providing content to multiple reading systems. [MCCRARY et al. 1999: 1].

Ausgehend von diesen Zielstellungen gilt für die Bestandteile der OEB-DTD:

- Die Syntax von OEB basiert auf XML und ist grundsätzlich erweiterbar; es gelten die oben eingeführten Einschränkungen von XML gegenüber SGML. Lesesysteme für elektronische Bücher, die OEB verwenden wollen, müssen XML verarbeiten können.
- Die konkrete Ausprägung der Markupelemente von OEB ist von HTML abgeleitet; es können bestehende, in HTML kodierte Inhalte problemlos in OEB überführt werden.
- Abgleitet von *Cascading Style Sheets* (level 1) definiert OEB eine eigene Layoutsprache als Medientyp (text/x-oeb1-css); wie in HTML können Layoutanweisungen in einzelnen Elementen, dem Dokumentkopf oder über *style sheets* eingebunden werden.
- OEB integriert die Beschreibungskategorien des Dublin Core Sets für die Kodierung v. a. bibliographischer Metainformation (s. u. Kap. 9.2.1).

Dokumente elektronischer Bücher sind entweder sog. *basic OEB documents*, falls ausschließlich die in der OEB-DTD enthaltenen Elemente verwendet werden, oder *extended OEB documents*, falls zusätzliche Elemente eingeführt werden, für die dann jeweils eine gültige Layoutanweisung nach dem OEB-CSS-Format vorliegen muss.

In Abweichung von HTML führt OEB zusätzlich das Konzept eines Pakets (*package*) ein, das für eine Publikation die zugehörigen Informationen (Dateinamen, Metadaten etc.) bündelt. Die im zugehörigen *package file* enthaltenen Informationen sind

- Die package identity, die das OEB package eindeutig bezeichnet,
- metadata, d. h. Angabe der wesentlichen bibliographischen Daten,
- manifest, d. h. eine Liste der zum elektronischen Buch gehörenden Dateien,
- spine, d. h. in Analogie zum physischen Buchrücken eine Anordnung der Dokumente für eine lineare Lesereihenfolge der Seiten (Dokumente; „an arrangement of documents providing a linear reading order“, MCCRARY et al. 1999: 12). Dieser „logische Buchrücken“ muss nicht unbedingt alle Dokumente des Pakets enthalten, da Dokumente wie in HTML z. B. durch eingebettete Hypertextverknüpfungen erreichbar sein können und nicht in der Default-Lesereihenfolge enthalten sind.
- tours, d. h. eine Sammlung zusätzlicher „Leserouten“ durch das Buch und
- guide, d. h. eine Sammlung von Verknüpfungen zu wesentlichen Strukturbestandteilen des elektronischen Buchs (wie Inhaltsverzeichnis, Register, Glossar etc.).

Mit der Einführung von Makrokonzepten aus der Hypermedia-Theorie[82] wie tours und guide versucht OEB die Funktionalität elektronischer Bücher zu erweitern und zu vereinheitlichen. Es ist allerdings keine starre Systematik möglicher guide-Elemente vorgegeben, d. h. die Realisierung von ausgewählten Pfaden durch den Hypermedia-Datenbestand bleibt dem einzelnen Entwickler überlassen.

Die nachfolgende Tabelle vergleicht in Anlehnung an MCCRARY 1999: Appendix A OEB und HTML.

[82] Vgl. ISAKOWITZ, STOHR & BALASUBRAMIAN 1995 und TOCHTERMANN 1995, die in ihren Hypermedia-Modellen bzw. Methodologien ebenfalls solche Makrokonzepte definieren; vgl. oben Kap. 2.2.1.

Element	*open eBook*	*HTML 4.0*
<a>	+	+
<area>	+	+
<b>	+	+
<base>	+	+
<big>	+	+
<blockquote>	+	+
<body>	+	+
 	+	+
<caption>	+	+
<cite>	+	+
<code>	+	+
<dd>	+	+
<dfn>	+	+
	+	+
<dl>	+	+
<dt>	+	+
	+	+
<h1>…<h6>	+	+
<head>	+	+
<hr>	+	+
<html>	+	+
<i>	+	+
<img>	+	+
<kbd>	+	+
	+	+
<link>	+	+
<map>	+	+
<meta>	+	+
<object>	+	+
	+	+
<p>	+	+
<param>	+	+
<pre>	+	+
<q>	+	+
<samp>	+	+
<script>	+	+
<small>	+	+
	+	+
	+	+
<style>	+	+
<sub>	+	+
<sup>	+	+
<table>	+	+

Element	*open eBook*	*HTML 4.0*
<td>	+	+
<th>	+	+
<title>	+	+
<tr>	+	+
<tt>	+	+
	+	+
<var>	+	+
<abbr>	-	+
<acronym>	-	+
<address>	-	+
	deprecated	*deprecated*
<font>	*deprecated*	*deprecated*
<s>	*deprecated*	*deprecated*
<strike>	*deprecated*	*deprecated*
<u>	*deprecated*	*deprecated*
<applet>	-	*deprecated*
<basefont>	-	*deprecated*
<bdo>	-	+
<button>	-	+
<col>	-	+
<colgroup>	-	+
<del>	-	+
<dir>	-	*deprecated*
<fieldset>	-	+
<form>	-	+
<frame>	-	+
<frameset>	-	+
<iframe>	-	+
<input>	-	+
<ins>	-	+
<isindex>	-	*deprecated*
<label>	-	+
<legend>	-	+
<menu>	-	*deprecated*
<noframes>	-	+
<noscript>	-	+
<optgroup>	-	+
<option>	-	+
<select>	-	+
<tbody>	-	+
<textarea>	-	+
<tfoot>	-	+
<thead>	-	+

Tabelle 28: Vergleich von open eBook und HTML

Abgesehen von der mit *open eBook* eng verbundenen Frage, ob und wie schnell sich hardware-basierte Buchviewer durchsetzen werden, hat der Vergleich mit HTML ein klares Ergebnis: Wesentliche funktionale Bestandteile von HTML wie die Kodierung von Formularen und die Definition von Fensterbereichen (*frames*) sind in OEB nicht enthalten, andere deutlich vereinfacht (so ist die <TABLE>-Marke in OEB enthalten, die Elemente <COL>, <COLGROUP>, <TFOOT>, <THEAD> fehlen aber). Umgekehrt enthält OEB mit Ausnahme der voranstehend genannten Elemente auf *package*-Ebene aber keine

Marken, die nicht auch in den HTML-DTDs enthalten sind. OEB ist insofern ein XML-basiertes Subset von HTML.

Abgesehen von den noch näher zu untersuchenden Makrokonzepten für die Buchnavigation ergibt sich mit OEB kein wesentlicher Beitrag für die Kodierung dynamischer elektronischer Bücher; betrachtet man die fehlenden Möglichkeiten für den Formularaufbau und die Fenstersteuerung, so wird man zu der Ansicht kommen, dass einige der Möglichkeiten, die einen genuinen Mehrwert dynamischer Bücher ausmachen, wie die dynamische Konfiguration der Benutzerschnittstelle eines elektronischen Buchs mit Hilfe von Frames oder die Einführung zusätzlicher interaktiver Elemente durch formularbasierte Verfahren, in OEB nicht umgesetzt werden können, was auch auf die eingeschränkten Möglichkeiten der eBook-Plattformen zurückzuführen sein dürfte.

6.4 Dokumentformate der Text Encoding Initiative (TEI)

Eine der umfangreichsten Sammlungen von Dokumententypbeschreibungen, vornehmlich für literarische oder historische Texte, ist die von der *Text Encoding Initiative* (TEI) zusammengestellte Sammlung von *Text Encoding Guidelines* (SPERBERG-MCQUEEN & LOU BURNARD 1994). Dabei handelt es sich um eine in internationaler Kooperation erstellte und erweiterbare Sammlung von SGML-basierten *document type definitions* für die Kodierung unterschiedlichster Textsorten. Das Ziel der *Text Encoding Initiative* ist die einheitliche Aufbereitung und der Austausch wissenschaftlicher Texte in verschiedenen Sprachen durch Kodierung mit SGML-konformen DTDs (TEI-DTDs). Sie umfasst:

- Allgemeine *TEI header elements*, die vornehmlich der Kodierung von Metadaten dienen und vier wesentliche Teile enthalten:

TEI Header Element	*Interpretation*
<teiHeader>	
<fileDesc> … </fileDesc>	bibliographische Beschreibung
<encodingDesc> … </encodingDesc>	editorische Notizen
<profileDesc> … </profileDesc>	Zusatzinformationen zu den kodierten Texten, die über die eigentlichen bibliographischen oder editorischen Angaben hinausgehen (z. B. zum Sprachgebrauch bei der Kodierung gesprochener Sprache)
<revisionDesc> … </revisionDesc>	Änderungsinformationen („Logfile“)
</teiHeader>	

Tabelle 29: Aufbau eines TEI-Header

- Eine Basismenge an Strukturierungsmerkmalen, die in allen Textsorten Verwendung finden (Gliederung in Absätze, Hervorhebungen, Datums- und Zahlenangaben etc.).
- Eine Reihe von DTDs für unterschiedliche Textsorten; das Spektrum reicht von literarischen Textsorten (Prosa, Drama, Lyrik, Sonderformen) bis hin zu Lexika, terminologischen Datenbanken und kritischen Editionen.
- Sammlungen von Entitäten und Zeichensätzen.
- Eine ausführliche Spezifikation für Hypertextverweise; dieses System ist Grundlage der in XPath, XLink und XPointer definierten XPointer.

Die Richtlinien der TEI orientieren sich primär an der editorischen Aufbereitung literarischer und historischer Texte, sind aber offen sowohl für die Aufbereitung im Vorfeld der Erstellung von Druckwerken als auch für die Erstellung elektronischer Corpora bzw. dynamisch generierter Sichten auf TEI-kodierte Dokumente (vgl. BARNARD & IDE 1997: 625f.). Ebenso wie bei naturwissenschaftlichen Lehr- und Lerntexten, die hier im

Mittelpunkt stehen, handelt es sich bei den in den TEI-Richtlinien kodierten Textsorten um komplexe Dokument- bzw. Informationsstrukturen, für die einfache Formate wie HTML keine ausreichende Beschreibungsmächtigkeit aufweisen, insbesondere, wenn man an inhaltsorientiertes bzw. funktionsorientiertes Markup denkt. Für die Strukturierung elektronischer Bücher sind die TEI-DTDs von Bedeutung, da

- erstmalig das erweitere Referenzsystem der XPointer eingeführt wird,
- viele der in TEI definierten Elemente auch in elektronischen Büchern Verwendung finden können, soweit sie über das Strukturierungsinventar von HTML hinausgehen und es sinnvoll erscheint, sie für ein generisches elektronisches Buchformat zu übernehmen,
- das Aufgreifen der Struktur der TEI-Header für die Spezifikation von Metadaten für Dokumente sinnvoll sein kann und
- eine Umwandlung der in SGML definierten Elemente der TEI-DTDs problemlos möglich erscheint.

Eine ganze Reihe großer Forschungsvorhaben wie z. B. das des *British National Corpus*, wenden die TEI-Richtlinien an, um Texte zu kodieren und zu erschließen (vgl. GIORDANO 1994).

6.5 DocBook

Die DocBook-Initiative, die seit einigen Jahren von der sog. Davenport Group entwickelt wurde und derzeit durch ein *Technical Committee* der *Organization for the Advancement of Structured Information Standards* (OASIS) weiterentwickelt wird, hat sich zum Ziel gesetzt, eine umfassende SGML-/XML-DTD für die elektronische Softwaredokumentation zu definieren, die als Alternative zu HTML bei der Kodierung umfangreicher elektronischer Publikationen herangezogen werden kann:

> The DocBook DTD defines structural and content-based SGML markup for computer documentation, with a primary emphasis on software documentation and related classes of technical documents. Its main high-level hierarchical structures are for books, reference entries (for example, „man pages“), and articles. [MALER & ALLEN 1997A:1]

DocBook definiert eine modular aufgebaute, umfangreiche DTD, deren Referenzversion in SGML kodiert ist, für die aber eine inhaltliche weitgehend äquivalente XML-Fassung erarbeitet wird. Die Konzepte, die in DocBook verwendet werden, orientieren sich, wie die nachfolgende Übersicht zeigen wird, weitgehend an der traditionellen Buchterminologie. Der Aufbau der DocBook-DTD hat folgende wesentliche Bestandteile:

- Auf der obersten Ebene dient das Konzept einer Menge (`<SET>`) dazu, mehrere einzelne Publikationseinheiten (Bücher, `<Book>`) zusammenzufassen und zu erschließen (Titel, Inhaltsverzeichnis, Register, allgemeine Metadaten).
- Unterhalb dieser Ebene gibt eine DTD die Strukturbestandteile eines Buches an (Titel (`<TITLE>`), Inhaltsverzeichnis (`<TOC>`), Vorwort (`<PREFACE>`), Glossar (`<GLOSSARY>`), Appendizes (`<APPENDIX>`); Gliederung in Teile (`<PART>`), Kapitel (`<CHAPTER>`), Abschnitte (`<PARA>`) etc.).
- Auf der Ebene der Inhaltskodierung der einzelnen Abschnitte eines *DocBook*-Buchs sind vielfältige Strukturkomponenten für die Textauszeichnung vorgesehen:
  - Verschiedene Paragraphentypen (Paragraphen, die Text und Objekte enthalten können; einfache Paragraphen, die nur Text und *inline*-Elemente enthalten; formale Paragraphen, die einen Titel tragen müssen),

– Sonderformen von Absätzen (*Abstracts* (<ABSTRACT>), Hervorhebungen (<HIGHLIGHTS>), Marginalien und Hinweise (<CAUTION>, <TIP>, <NOTE> etc.),
– verschiedene Listentypen (ungeordnet, geordnet, Variablen- und Glossarlisten),
– Beschreibungselemente für Programmlistings, Quellcodebeispiele, Bildschirmdialoge und *screen shots*,
– Elemente zur Kodierung von Abbildungen, Diagrammen und Tabellen und
– Elemente für die Kodierung mathematischer Formeln.

Trotz der Einschränkung auf die Anwendungsdomäne Softwaredokumentation enthält DocBook eine umfangreiche Menge an Strukturierungsmerkmalen, die viele wesentliche Elemente traditioneller Bücher umfasst und für elektronische Bücher anwendbar erscheint. Sie geht über einfachere Formate wie HTML oder OEB in dieser Hinsicht hinaus. An den Nutzungsmöglichkeiten des elektronischen Mediums orientierte Strukturbestandteile fehlen aber weitgehend (Interaktion; Skripteinbettung; Formulare; mächtige Hypertextverknüpfungen). Für die Anwendung im Kontext der Aufbereitung elektronischer Bücher kommt ähnlich wie bei den *TEI Guidelines* allenfalls die Übernahme einer Untermenge der Elemente von DocBook in Betracht, für die DTDs wie die HTML- oder XHTML-DTD keine oder nicht hinreichend differenzierte Elemente enthalten. Eine solche Wiederverwendung hat grundsätzlich den Vorteil, dass ein standardisiertes oder semistandardisiertes Format wie DocBook weiter verbreitet ist und sich leichter in bestehende Softwaresysteme integrieren lässt. Zudem kann man sich DocBook als ein geeignetes Werkzeug der sekundären Strukturierung durch *architectural forms* vorstellen.

6.6 Mathematical Markup Language: XML-basierter Formelsatz

In diesem Abschnitt wird nicht ein weiteres Strukturierungsformat für elektronische Dokumente, sondern ein Spezialfall für die Kodierung bestimmter Informationsformate diskutiert: Der mathematische Formelsatz. In der Systematik der Informationsstrukturierung handelt es sich um ein Teilproblem, das in ähnlicher Weise für viele andere Wissenschaftsdomänen existiert: Innerhalb der generellen Problematik der Aufbereitung multimedialer Inhalte existieren Sonderformate, die in der Regel eine hohe Binnenkomplexität aufweisen und die von einem generischen Standard wie den voranstehend diskutierten Strukturierungsverfahren nicht erfasst werden. Weitere Beispiele dieser Art sind:

- die Kodierung chemischer Strukturen mit deklarativem Markup, das sowohl für eine Textrepräsentation als auch für die Visualisierung ausgewertet werden kann[83]
- die Kodierung spezialisierter Informationsformate wie z. B. Wirtschaftsdaten, die ebenfalls textuell (z. B. tabellarisch) ausgegeben oder durch Informationsgraphiken visualisiert werden können[84] oder
- die Kodierung linguistischer Strukturinformation, die ebenfalls formalisiert dargestellt oder durch Sprachtechnologieprodukte (z. B. maschinelle Übersetzung) ausgewertet und verarbeitet werden.[85]

---

[83] Vgl. dazu den Entwurf einer *Chemical Markup Language* (MURRAY-RUST 1998).

[84] Gerade der E-Commerce wird als ein wichtiger Anwendungsbereich von XML angesehen, da sich auf einfache Weise neue Dokumentformate für spezialisierte und proprietäre Informationen entwerfen lassen; man vergleiche dazu etwa die vielfältigen Fallstudien in GOLDFARB & PRESCOD 1998, insb. *part three*, Kap. 14-20.

[85] Vgl. etwa WOLFF & QUASTHOFF 1999 mit einem XML-basierten Format für multilinguale Lexika und Corpora oder REHM 1999 zur Textannotation durch SGML/DSSSL. LOBIN

Die nachfolgende Diskussion von MathML ist insofern einerseits durch die Erfordernisse des Referenzprojekts als eines naturwissenschaftlichen Lehrbuch mit hohem Anteil mathematischer Formeln entstanden, steht aber andererseits stellvertretend für die Problem*klasse* der anwendungsbezogenen Informationsstrukturierungsformate.

Nicht nur in naturwissenschaftlichen Publikationen spielt die Darstellung mathematischer Symbole, d. h. die Präsentation einer zweidimensionalen Symbolsprache, eine wesentliche Rolle. Bei der Erstellung von Publikationen mit elektronischen Mitteln existieren eine Reihe von Alternativen für den Formelsatz:

- Die weiteste Verbreitung in der Mathematik und in den Naturwissenschaften dürfte das von Donald E. KNUTH entwickelte Satzsystem TEX gefunden haben (vgl. KNUTH 1984). TEX ist – in Kombination mit einer Schriftgenerierungssoftware (*MetaFont*) – ein mächtiges System für den elektronischen Satz und hat seine Stärke im Bereich der Formelpräsentation. Es dient darüber hinaus als Austausch- und Konvertierungsformat für an sich nicht TEX-basierte Software für den Formelsatz.
- Textverarbeitungssysteme bieten i. d. R. eigene Komponenten (z. B. *Microsoft Equation Editor* als Teil des MS-Office-Pakets) für den Formelsatz an, die bei der Texterstellung genutzt werden können.
- Es existieren viele dedizierte Softwarepakete für den mathematischen Formelsatz; MathType (vgl. DESIGN SCIENCE 1999 bzw. *http://www.mathtype.com*, derzeit MathType V. 4) ist ein weit verbreitetes Beispiel, das im Rahmen dieser Arbeit verwendet wird.

Im Bereich der SGML-basierten Markupsprachen liegen bereits seit langem *document type definitions* für den Formelsatz vor (z. B. die Standard-DTD ISOMATH, vgl. ISO/IEC JTC1/WG4 N1990 (1990)). In einer früheren, aber nicht offiziell standardisierten Fassung 3.0 der *HyperText Markup Language* (vgl. ION & MINER 1999 Kap. 1.2.1) waren Elemente („*HTML Math*“) für den Formelsatz vorgesehen. Die praktische Anwendung des Formelsatzes für elektronische Publikationen auf SGML- bzw. HTML-Basis ist allerdings bisher am Mangel geeigneter Viewing-Systeme für deklaratives Formelmarkup gescheitert: Die gängigen WWW-Browser (Netscape, Microsoft Internet Explorer) unterstützen bisher keinerlei SGML-basiertes Formelmarkup, was in der Praxis zu einer Reihe wenig befriedigender Lösungen geführt hat:

- Generieren von Formeln als GIF-Bilder aus dem Ausgangsformat, wohl die derzeit gebräuchlichste Lösung für den Formelsatz in HTML-basierten elektronischen Büchern[86],
- Einbindung von Plug-ins in den Browser, die deklaratives Formelmarkup unterstützen (TEX, MathML, s. u.) und
- hilfsweise Darstellung einfacher Formeln durch Verwendung der gewöhnlichen Satzmöglichkeiten (Hoch- und Tiefstellen von Zeichen, Einbinden einzelner Zeichen sonst nicht verfügbar Symbolschriften als GIF-Bilder).

Diese Erfahrungen haben das Referenzprojekt *Multimediales Physikalisches Praktikum* maßgeblich geprägt, wobei für die Produktionsfassung nach Evaluierung einer Vielzahl

---

2000: 149 ff. zeigt, wie die in einer Phrasenstrukturgrammatik enthaltene Information (Produktionen der Grammatik, Auszeichnung lexikalischer Einheiten) durch eine SGML-/XML-DTD ausgedrückt und zur Informationsauszeichnung verwendet werden kann.

86 Dies belegen etwa die in MeDoc/InterDoc verfügbaren elektronischen Lehrwerke zur Mathematik und Informatik, die in der Regel als GIF-Bilder gespeicherte Formeln enthalten, vgl. oben Kap. 3.1.1.

von Verarbeitungsalternativen letztlich die Standardvariante der Generierung von Formeln als GIF-Bilder mit all ihren Nachteilen gewählt wurde (vgl. HOFMANN, WALTHER & WOLFF 1998: 3 ff.).

Das Desiderat deklarativen Formelmarkups für elektronische Publikationen im World Wide Web ist seit langem bekannt; eine Arbeitsgruppe des W3C hat mit MathML die Spezifikation einer Markupsprache als XML-Anwendung erarbeitet (vgl. ION & MINER 1999, POPPELIER, MINER & ION 2000). MathML vermeidet die oben dargestellten Nachteile für eine Formelpräsentation in SGML/XML/HTML. MathML bringt folgende Vorteile:

- Bei deklarativem Formelmarkup ist die weitere Editierbarkeit gewährleistet, d. h. anders als eine Bitmap-Darstellung kann der Inhalt der Formel bearbeitet werden, die Formeldarstellung endet nicht in einer „Format-Sackgasse“ (binäre „Datenkonserve“, deren Inhalt nicht mehr analysiert werden kann).
- Es ist gewährleistet, dass der Inhalt der Formel geparst und z. B. für die Indexierung herangezogen werden kann.
- Mit Hilfe von deklarativem Markup ist die Möglichkeit der Skalierung und einer unterschiedlichen Darstellung (*style sheets*) gegeben. Lösungen über skalierbare Vektorgraphiken sind entweder noch nicht hinreichend standardisiert (z. B. *Scalable Vector Graphics* (SVG), vgl. (FERRAIOLO et al. 1999) oder WWW-fähige Vektorgraphik-Software unterstützt den Formelsatz nicht hinreichend (z. B. Macromedia Flash).

Die Definition von MathML erfolgte unter Berücksichtigung verwandter Aktivitäten wie *OpenMath* (vgl. CAPROTTI, CARLISLE & COHEN 1999) und ist mit TEX und den im SGML-Standard enthaltenen Formelsatz-DTDs (ISO 12083) kompatibel (vgl. ION & MINER 1999: Kap. 1.3.2). MathML kodiert sowohl die für die *Präsentation* einer Formel benötigte Information (*presentation markup*) als auch den *Inhalt* der Formel, z. B. eine bestimmte mathematische Idee oder eine physikalische Gesetzmäßigkeit (*content markup*). Dazu baut MathML auf folgenden Konzepten auf:

- *Presentation elements* beschreiben die logische Struktur einer bestimmten Notation; dazu gehören z. B. die MathML-Elemente <mrow> oder <mtable> für die Beschreibung einer Zeile bzw. einer Tabelle innerhalb einer Formel. Sie gliedern sich in *layout schemata* (wie die gerade angeführten Elemente) und einzelne *token elements*, die einzelne darstellbare Symbole oder Zeichen kodieren, z. B. <mi> für die Kodierung eines Bezeichners, <mo> für die Kodierung eines Operators oder <mn> für die Repräsentation von Zahlen.
- *Content elements* repräsentieren direkt ein bestimmtes mathematisches Konzept, d. h. nicht eine Notation, die für ein Konzept steht. Sie ermöglichen eine präsentationsunabhängige Kodierung mathematischer Ideen. Beispiele hierfür sind das +-Zeichen für den Additionsoperator, das <apply>-Element für die Anwendung eines Operators auf seine Argumente oder das <fn>-Element für die Definition von Funktionen.
- *Interface elements* bilden die Schnittstelle zwischen MathML-Code und einem Formelprozessor, der die in MathML kodierte Formel verarbeitet. Dazu gehört als eine Formel kapselndes „Top-Level-Element“ die Marke <math>, die eine MathML-Formel umschließt und in ihren Attributen generelle Information zur Verarbeitung der Formel enthalten kann.

Der Aufbau komplexer Formeln setzt eine rekursive Verwendbarkeit der verschiedenen *presentation* und *content elements* voraus; dies ist in MathML möglich, da die Elemente in der MathML-DTD so definiert sind, dass sie in sich geschachtelt werden können. Bei

der Analyse eines Math-ML-Ausdrucks kann man *expression trees* aufbauen, deren Blätter entweder leer sind oder *token elements* enthalten.

Durch die Unterscheidung zwischen *presentation markup* und *content markup* bieten sich unterschiedliche Möglichkeiten der deklarativen Kodierung eines bestimmten mathematischen Ausdrucks; je nach Anwendungsgebiet ist eine der beiden Alternativen oder eine Mischform von *presentation* und *content markup* zu wählen. Für künftige Versionen von MathML ist die Verwendung von *style sheets* im Sinn der *Cascading Style Sheets*-Spezifikation vorgesehen.

Das folgende Beispiel zeigt für den Ausdruck (a + b) * c die Aufbereitung in MathML *presentation* bzw. *content markup*, das MathML *presentation markup* wurde dabei mit Hilfe von MathType generiert:

Presentation Markup	*Content Markup*
<MATH DISPLAYSTYLE='TRUE'> <MROW> <MO STRETCHY='FALSE'>(</MO> <MI>A</MI> <MO>+</MO> <MI>B</MI> <MO STRETCHY='FALSE')</MO> <MO>*</MO><MI>C</MI> </MROW> </MATH>	<MATH> <APPLY><TIMES/> <APPLY><PLUS/> <CI> a </CI> <CI> b </CI> </APPLY> <CI> c </CI> </APPLY> </MATH>

Abbildung 28: Beispiel für presentation und content markup in MathML

Das Beispiel verdeutlicht den Unterschied zwischen präsentationsorientierter und inhaltsorientierter Kodierung: Während sich im linken Beispiel alle Elemente auf die *Darstellung* der Formel beziehen, ist im rechten Beispiel der logische Zusammenhang (Anwendung von Operatoren auf Argumente) kodiert. Um das rechte Beispiel darstellen zu können, wird vom MathML-Interpreter zusätzlich eine Übersetzung in ein entsprechendes Präsentationsäquivalent erwartet.

Sowohl für das Präsentationsmarkup als auch für das Inhaltsmarkup sind in MathML für alle gebräuchliche Bestandteile mathematischer Notation Elemente sowie eine Defaultinterpretation der Elemente vordefiniert (vgl. ION & MINER 1999 Kap. 3 u. 4 und Anhang C u. F). Ein wesentlicher Vorteil eines SGML- bzw. XML-basierten Ansatzes ist seine Offenheit, d. h. MathML ist grundsätzlich erweiterbar und kann an andere Notationen bzw. neu eingeführte Symbole angepasst werden. Ein Abgleich mit dem im Referenzwerk *Multimediales Physikalisches Praktikum* verwendeten Symbolinventar zeigt, dass MathML ausreichende Ausdrucksstärke besitzt, wie Abbildung 29 (Gleichung für die Viskosität von Flüssigkeiten, GESCHKE 1998: 100, Gl. 17) zeigen soll.

Wie das Beispiel deutlich machen soll, eignet sich MathML aufgrund seiner Eigenschaft als XML-Anwendung kaum für die Quellcode-Kodierung von Formeln, da dies einen zu hohen Editierungsaufwand mit sich bringen würde.[87]

Besondere Elemente in MathML wie <MACTION> können benutzt werden, um innerhalb eines Ausdrucks bestimmte Aktionen auszulösen (z. B. dynamische Hervorhebung). Die Konformität von MathML mit XML hat zur Folge, dass grundsätzlich die in XML

[87] Ein Vergleich mit TEX macht deutlich, dass bei MathML ein deutlich höherer Eingabeaufwand erforderlich ist: Das obige Beispiel könnte in TEX durch den folgenden, deutlich kürzeren Ausdruck kodiert werden: $$\eta = \frac{2(\rho_K - \rho_Fl)g}{9l}r^2t\lbrace1-\frac{r}{R}\rbrace^n$$. Der TEX-Ausdruck enthält weniger deklarative Information zur gewünschten Darstellung – diese kann nur in Kenntnis des TEX-Satzalgorithmus beurteilt werden.

(bzw. XLink) vorgesehenen syntaktischen Mechanismen für die Einführung von Hyperlinks in MathML zur Verfügung stehen, d. h., dass die einzelnen Elemente von MathML Anker- bzw. Zielattribute tragen können und für die Referenzierung innerhalb eines Hypertexts zur Verfügung stehen.

Formeldarstellung	MathML Presentation Markup
$\eta = \frac{2(\rho_K - \rho_{Fl})g}{9l} r^2 t \left(1 - \frac{r}{R}\right)^n$	<math displaystyle='true'> <mrow><mi>η</mi><mo>=</mo><mfrac> <mrow><mn>2</mn><mo stretchy='false'>(</mo> <msub><mi>ρ</mi> <mi>K</mi></msub> <mo>-</mo> <msub><mi>ρ</mi><mrow><mi>F</mi> <mi>l</mi></mrow></msub> <mo stretchy='false'>)</mo><mi>g</mi> </mrow> <mrow><mn>9</mn><mi>l</mi></mrow> </mfrac> <msup><mi>r</mi><mn>2</mn></msup><mi>t</mi> <msup><mrow><mrow><mo>(</mo> <mrow><mn>1</mn><mo>-</mo> <mfrac><mi>r</mi><mi>R</mi></mfrac> </mrow><mo>)</mo></mrow> </mrow><mi>n</mi> </msup> </mrow> </math>

Abbildung 29: Formelbeispiel in MathML (GESCHKE 1998: 100, Gl. 17)

Die für die Anwendung von MathML in elektronischen Publikationen benötigte Softwareinfrastruktur ist noch nicht vollständig vorhanden – MathML liegt als Spezifikation erst seit dem Frühjahr 1998 vor; die gängigen WWW-Browser unterstützen derzeit MathML noch nicht direkt. Verschiedene Softwarepakete machen den Einsatz von MathML zur *Präsentation* von Formeln in elektronischen Publikationen möglich:

- Experimentelle Browser-Implementierungen unterstützen bereits den Einsatz von MathML. Hierzu zählt vor allem der vom W3C entwickelte Browser Amaya (vgl. Abbildung 30 bzw. QUINT & VATTON 1997 und *http://www.w3c/Amaya)*.
- Plug-ins bzw. Java-Applets für WWW-Browser verfügen mittlerweile über MathML-Interpreter, so dass in einer HTML-Seite enthaltener MathML-Code im Browser dargestellt werden kann (z. B. das WebEQ-Paket, vgl. *http://WebEQ.com)* oder das IBM TexExplorer-plug-in).

Im Rahmen des Referenzprojekts *Multimediales Physikalisches Praktikum* konnte MathML noch nicht eingesetzt werden, da die Spezifikation noch nicht abgeschlossen war, keine geeignete Software zur Verfügung stand bzw. die Evaluierung früherer Fassungen von WebEQ und TEXExplorer ein negatives Ergebnis brachten (vgl. HOFMANN, WALTHER & WOLFF 1998); zudem wäre der Aufwand für die in jedem Fall erforderliche *Neuerfassung* der Formeln zu groß gewesen. Dies ändert jedoch nichts an der grundsätzlichen Eignung von MathML.

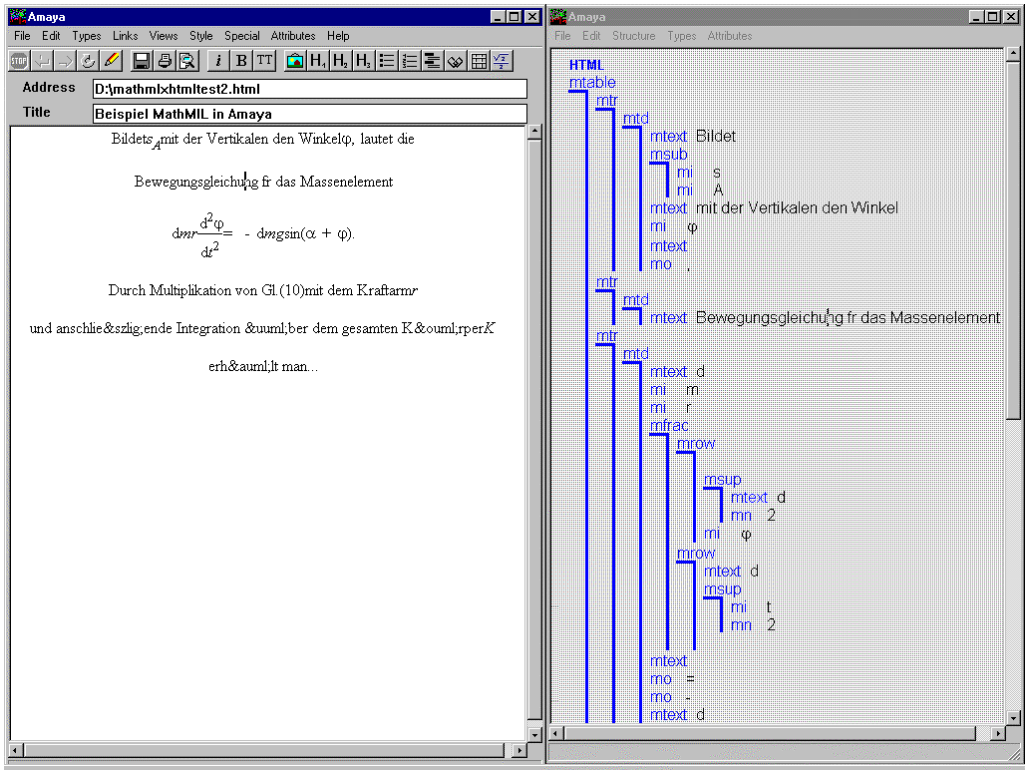

Abbildung 30: MathML-Beispiel in Amaya mit Strukturansicht

Neben dem Einsatz von MathML für die *Darstellung* von Formelsatz ergibt sich ein zweites Anwendungsfeld: Die Verwendung als Austauschformat zwischen elektronischen Publikationen und Anwendungen für die Symbolmanipulation mit Hilfe von Computeralgebrasystemen.[88] Wie in Kap. 4.4 ausführlich dargelegt wurde, sind generische Dienste ein wesentliches Konzept für dynamische elektronische Publikationen. Ein gutes Beispiel für einen solchen generischen Dienst ist eine Schnittstelle zwischen Publikation und Formelmanipulator. Eine prototypische Realisierung auf der Basis der Architektur für dynamische Publikation wird in Kap. 14.2.3 erörtert. MathML dient dort als ein Kodierungsformat, das dem Benutzer den Formelsatz nicht nur in geeigneter Weise präsentiert, sondern ihm den unmittelbaren Übergang in ein Computeralgebrasystem ermöglicht.

[88] Die Hersteller solcher Systeme (z. B. Maple, Mathematica, MuPad) waren am Entwicklungsprozess von MathML beteiligt, vgl. ION & MINER, Anhang D und unten Kap. 14.2.4.

7 Transformation und Präsentation von Dokumenten

SGML und XML dienen als Metasprachen nur der *logischen* Strukturdefinition, anders als TEX oder Postscript aber nicht der Festlegung eines bestimmten Layouts, d. h. einer konkreten physikalischen Präsentationsform auf Papier, Bildschirm etc. Daher sind Ergänzungen auf der Basis von SGML notwendig. Dies gilt in ähnlicher Weise für XML und auf der Anwendungsebene von SGML und XML (HTML/XHTML, XML-DTDs).

Vereinfachend lassen sich Paare von Standards für die Kodierung der logischen Struktur bzw. des Layouts von Dokumenten nach folgendem Schema bilden:

Strukturmarkup	*Präsentationsmarkup und Transformationen*
SGML	DSSSL *Document Style Semantics Specification Language*
XML	XSL *Extensible Style Language*
HTML	CSS *Cascading Style Sheets* – nur Präsentation

Tabelle 30: SGML- und XML-basierte Standards für Struktur- und Präsentationsmarkup

Dieser Vergleich ist nur bedingt korrekt, da DSSSL, XSL und CSS zwar einen gemeinsamen Ausgangspunkt haben (DSSSL), aber eine unterschiedliche Mächtigkeit aufweisen und ihre Anwendbarkeit jeweils nicht nur auf das jeweilige Gegenüber beschränkt ist: DSSSL als generischer SGML-Partnerstandard kann prinzipiell bei der Verarbeitung beliebiger SGML-Dokumente, also auch für XML- oder HTML-Dokumente Verwendung finden, während umgekehrt XSL als Layout- und Transformationssprache die syntaktischen Einschränkungen von XML voraussetzt und nicht für beliebige SGML-Dokumente eingesetzt werden kann. Die Layoutstandards unterscheiden sich hinsichtlich Entwicklungsgrad und Status: Während DSSSL als internationaler Standard formal eine allgemein akzeptierten Status erreicht hat, ist XSL bisher lediglich ein *working draft* des W3C und die *Cascading Styles Sheets* (Level 1 und 2) sind eine Empfehlung (*recommendation*) des W3C.

Umgekehrt sagt der Status wenig über die *praktische Anwendbarkeit* aus: Aufgrund seiner Mächtigkeit gibt es für DSSSL bisher praktisch keine vollständige Implementierung (vgl. BEHME & MINTERT 1998: 172). Die Transformationssprache von DSSSL erfordert einen leistungsfähigen Interpreter für eine Lisp-/Scheme-ähnliche Sprache.

Für XSL liegen wenigstens partielle Implementierungen vor; es ist zu erwarten, dass XSL mittelfristig in die Formatierungsinterpreter der Webbrowser Eingang finden wird (z. B. Microsoft Internet Explorer V 5).[89] CSS ist bereits seit einiger Zeit in den Webbrowsern verfügbar und kann in ihnen für die Spezifikation von Formatvorlagen (*style sheets*) verwendet werden.

[89] Weitere Implementierungen sind etwa Cocoon, ein serverseitiger XSL-/XSLT-basierter Dokumentprozessor als Erweiterung des Apache-Webservers, vgl. *http://xml.apache.org/cocoon/* oder XT, eine XSLT-Implementierung von James Clark, vgl. *http://www.jclark.xom/XT*. Im Rahmen dieser Arbeit findet Cocoon als *publishing engine* Verwendung, die die Transformation von XML-Dokumenten durch XSL/T-Skripte leistet und somit Voraussetzung der Implementierung des Buchverwaltungssystems ist.

Bei CSS beschränkt sich die Verwandtschaft mit DSSSL im Wesentlichen auf eine gemeinsame Syntax für Stilformatierungen. Während für die Darstellung von SGML- oder XML-Dokumenten die Spezifikation von Layoutinformation unerlässlich ist, kann bei HTML im Prinzip darauf verzichtet werden, da die HTML-Browser jeweils eine Standardinterpretation für die Layoutdarstellung aufweisen. In diesem Fall dient CSS als zusätzliche Möglichkeit, ein Layout einheitlich zu definieren und die Gestaltung besser zu kontrollieren, nicht aber als notwendige Voraussetzung der Dokumentdarstellung.

Bevor die Standards näher vorgestellt werden, sei auf die Tatsache verwiesen, dass für SGML- bzw. XML-Dokumente eine Präsentationsspezifikation *zwingend erforderlich* ist, da ansonsten benutzerdefinierte Dokumentenstrukturbeschreibungen nicht zu interpretieren wären, während die Webbrowser für die Darstellung von HTML eine Defaultinterpretation liefern können. Die Präsentationsspezifikation für HTML-Dateien durch CSS hat in diesem Fall den Status einer zusätzlichen, optionalen Präsentationssteuerung.

7.1 Transformations- und Layoutsprachen

Nachfolgend soll zwischen Sprachen unterschieden werden,

- die sowohl Transformationen der logischen Struktur von Dokumenten als auch die Spezifikation ihres Ausgabeformats (Layout) ermöglichen (DSSSL und XSL) und
- Sprachen, die sich ausschließlich auf den Präsentationsaspekt beschränken und nicht für die Dokumententransformation unabhängig von der Präsentation geeignet sind (*Cascading Style Sheets* (CSS).

7.1.1 Document Style Semantics and Specification Language

Die *Document Style Semantics Specification Language* ist ein internationaler Standard für die Manipulation und Darstellung von Dokumenten (ISO/IEC 10179). Die folgende, der Einleitung zu ISO/IEC 10179 entnommene Definition verdeutlicht die beiden Hauptanliegen von DSSSL: Die Definition einer Transformationssprache für SGML und die Festlegung eines Kodierungsformat für die layoutbezogene Verarbeitung von SGML-Dokumenten:

> This International Standard is designed to specify the processing of valid SGML documents.
>
> DSSSL defines the semantics, syntax, and processing model of two Languages for the specification of document processing:
>
> a) The transformation Language for transforming SGML documents marked up in accordance with one or more DTDs into other SGML documents marked up in accordance with other DTDs. The specification of this transformation process is fully defined by this International Standard.
>
> b) The style Language, where the result is achieved by applying a set of formatting characteristics to portions of the data, and the specification is, therefore, as precise as the application requires, leaving some formatting decisions, such as line-end and column-end decisions, to the composition and layout process.

DSSSL greift zwei wesentliche Aspekte der Verarbeitung von SGML-Dokumenten auf, die in SGML selbst nicht berücksichtigt sind:

- die Notwendigkeit, Dokumente, die in verschiedenen DTDs kodiert sind, zusammenzuführen bzw. durch Transformationsprozesse auf andere DTDs abzubilden und
- die Ergänzung der logischen und physischen Struktur eines SGML-Dokuments durch präsentationsorientiertes Markup (Layoutinformation).

Diese beiden Prozesstypen werden in DSSSL durch

- einen *SGML Tree Transformation Process (STTP)*, in dem ein Baum (*grove*) aus Wurzel, Elementknoten und Daten- bzw. Zeichenknoten aufgebaut wird und
- den *SGML Tree Formatting Process (STFP)*, der aus SGML-Dokumenten unter Anwendung von Stilspezifikationen beliebige Ausgabeformate generieren kann.[90]

Die nachfolgende Abbildung zeigt die Verarbeitung von Dokumenten mit Hilfe von DSSSL im Überblick:

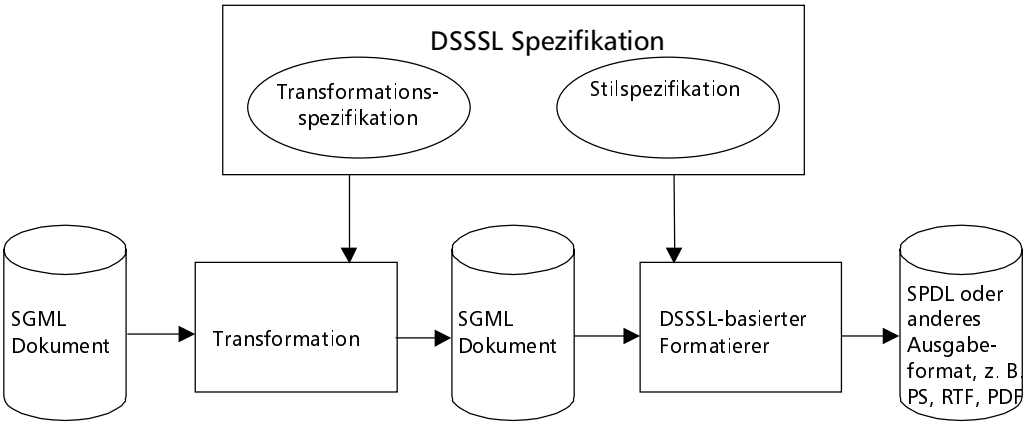

Abbildung 31: Schematischer Überblick der Verarbeitungsschritte in DSSSL [nach ISO/IEC ISO/IEC 10179 1996(E): 11 Abb. 1]

Die beiden Ziele – formale Spezifikation von Formatierungs- und Verarbeitungsinformation für elektronische Dokumente – erreicht DSSSL durch eine deklarative (Programmier-)Sprache, die von dem Lisp-Dialekt Scheme abgeleitet ist und die durch Schnittstellen zu traditionellen Programmiersprachen ergänzt werden kann, wobei die Format- und Verarbeitungsinformation von der Kodierung der logischen Struktur (in SGML) getrennt verwaltet wird. Die *Detail*spezifikation des Layouts für ein bestimmtes Ausgabegerät ist nicht Aufgabe von DSSSL, sondern kann von anderen Sprachen (z. B. Graphikbeschreibungssprachen, proprietäre Formate, die an bestimmte Geräte z. B. in der Druckvorstufe gebunden sind etc.) übernommen werden.

Sowohl die Transformationsvorgänge als auch die Layoutprozesse von DSSSL setzen auf einem gemeinsamen Algorithmus für Transformation und Formatierung von baumähnlichen Datenstrukturen auf (*grove building*, vgl. ADLER 1997: 599f.). Für die jeweiligen Eingangsdaten, d. h. ein SGML-Dokument sowie die zugehörigen DTDs, wird der Elementbaum als interne Repräsentation aufgebaut; dieser Elementbaum wird dann von den Transformations- und Formatierungssroutinen von DSSSL auf der Basis der vom Benutzer spezifizierten Angaben transformiert. Dies zeigen die beiden folgenden Abbildungen für den Transformations- bzw. für den Layoutprozess:

[90] Die Ausgabeformate sind im Standard selbst nicht festgelegt; ihre Erzeugung muss mit DSSSL definiert werden (z. B. PDF, Postscript, HTML).

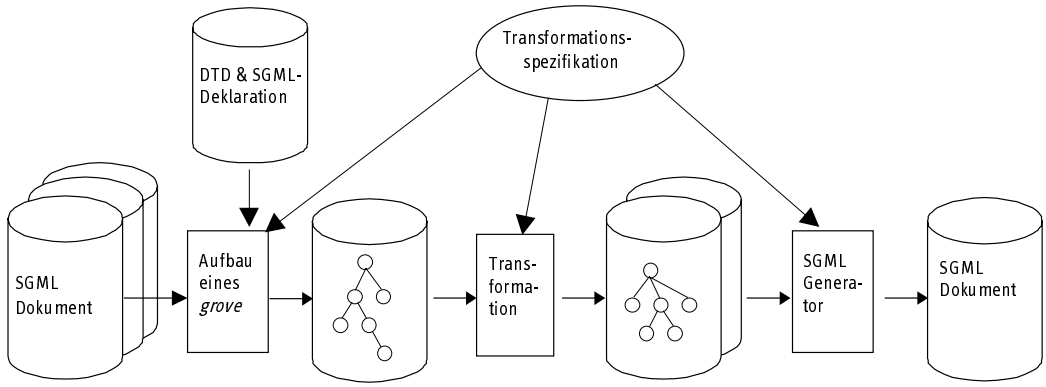

Abbildung 32: Transformationsprozesse in DSSSL [nach ISO/IEC 1996: 13 Abb. 2]

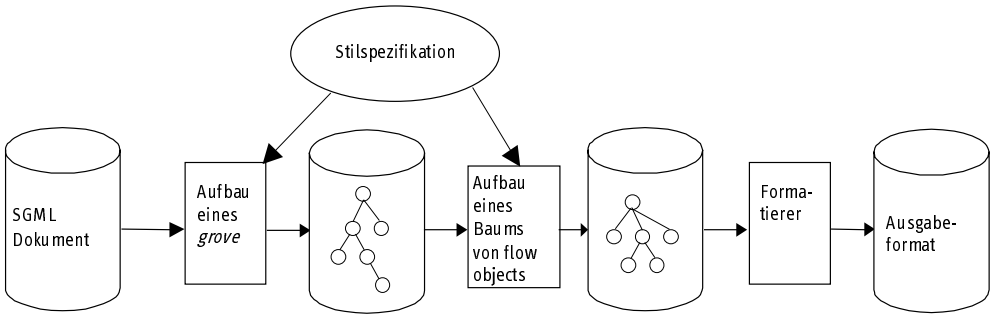

Abbildung 33: Formatierungsprozesse in DSSSL [nach ISO/IEC 1996: 19 Abb. 3]

Wie die beiden voranstehenden Abbildungen zeigen, ähneln sich die beiden Prozesstypen: In beiden Fällen wird aus dem deklarativen Markup eines oder mehrerer SGML-Dokumente ein Elementbaum (*grove*) konstruiert, der dann entweder nach vorgegebenen Regeln umformatiert und in ein anderes SGML-Dokument oder ein bestimmtes Ausgabeformat umgewandelt wird.

Bei der Formatierung entsteht ein *flow object tree*, dessen Blätter Objekte vom Typ vordefinierter *flow objects* sind, die für das endgültige Layout angeordnet werden müssen. Die verschiedenen *flow objects* sind als Klassen im DSSSL-Standard vordefiniert (z. B. *sequence*, *anchor*, *rule*, *score*, *glyph-annotation* etc.). Mit jedem *flow object* sind *formatting characteristics* assoziiert, die den Formatierungsprozess steuern. Neben den Transformations- und Präsentationsprozessen definiert DSSSL eine

- *Standard Document Query Language* (SDQL), die der Identifikation und Adressierung von Teilen eines SGML-Dokuments dient, sowie eine
- *Expression Language*, die der Erzeugung und Manipulation von Objekten in den durch DSSSL spezifizierten Prozessen dient. Die DSSSL-*Expression Language* ist als eine Untermenge des Scheme-Standards eine funktionale Programmiersprache (vgl. ISO/IEC 1996: 10.).

Das syntaktische Schema für Prozeduren sieht in DSSSL – nach der Vorgabe von Lisp bzw. Scheme – wie folgt aus: (Bezugsobjekt (Verarbeitungsvorschriften)). Das folgende schematische Beispiel zeigt eine Prozedur, in der ein Dokument in DSSSL geparst wird und dessen Elemente vom Typ KAPITEL in Abhängigkeit vom Wert ihres Attributs typ unterschiedlich verarbeitet werden (vgl. BEHME & MINTERT 1998: 117 ff.):

```
(element KAPITEL
    (let ( (kapitel-typ (attribute-string „typ“)))
    (if (and   kapitel-typ
            (string=? kapitel-typ „vorwort“)
            (Anweisungen für die Verarbeitung des Vorworts)
            (Anweisungen für die Verarbeitung anderer Kapiteltypen))))
```

Codebeispiel 49: Beispiel einer Verarbeitungsprozedur in DSSSL

DSSSL ist ein sehr mächtiger Standard, für den bisher nur wenige Softwarewerkzeuge existieren, z. B. die DSSSL-Engine *Jade* von James CLARK, vgl. *http://www.jclark.com/Jade*. Für die Anwendung im Referenzprojekt *Multimediales Physikalisches Praktikum* kam er aus diesem Grund nicht in Betracht. DSSSL hat aber eine große konzeptuelle Bedeutung als *Ausgangspunkt* für die Entwicklung von Transformations- und Layoutsprachen, die im Zusammenhang mit HTML und XML unmittelbar anwendbar sind.

7.1.2 Extensible Style Language

Die Ergänzung der logischen Struktur durch Formatierungsanweisungen erreicht man in XML durch die *Extensible Style Language* (XSL; vgl. ADLER et al. 1997) – im wesentlichen eine Weiterentwicklung der für HTML gebräuchlichen *Cascading Style Sheets* und damit ein Standard, der es erlaubt, für jedes Element einer XML-DTD Formatierungsangaben zu definieren (Schrift; Farbe; Ränder; Einbettung der Funktionalität von Skriptsprachen etc.).

Die *Extensible Style Language* dient der Definition von *style sheets* für die Verarbeitung von XML-Dokumenten und hat wie DSSSL eine Doppelfunktion, wie die Definition von XSL zeigt:

> XSL is a Language for expressing stylesheets. It consists of two parts:
> a Language for transforming XML documents, and
> an XML vocabulary for specifying formatting semantics.
> An XSL stylesheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into an XML document that uses the formatting vocabulary. [DEACH 1999: Abstract].

Um sowohl Dokumenttransformationen (*tree transformations*) als auch die Dokumentausgabe (*formatting*) zu realisieren, verwendet XSL in Anlehnung an DSSSL ein Verarbeitungsmodell, in dem ein Ausgangsbaum (Repräsentation eines XML-Dokuments) unter Anwendung eines XSL-*style sheet* in einen *result tree* umgewandelt wird; der *result tree* wird anschließend formatiert ausgegeben. Die Steuerung des Transformationsprozesses erfordert wie bei DSSSL eine Dokumentmanipulationssprache, die für XSL als *XSL Transformations* (XSL/T, CLARK 1999) definiert ist.

7.1.2.1 Transformationen in XSL

XSLT definiert die Generierung des Ausgabebaums als Grundlage der formatierten Ausgabe. Neben Anweisungen für den Import und die Inklusion von *style sheets* (<xsl:import href="…"/>, <xsl:include="…"/>) ist für die Festlegung der Dokumentverarbeitung die Definition von *template*s das wesentliche Element: Ein *template* enthält eine Musterbeschreibung, die bestimmte Teile des Eingabebaums selektiert und gibt für sie Verarbeitungsvorschriften an. Für den aktuell zu verarbeitenden Knoten (*current node*) wählt der XSL-Prozessor die passenden *templates* aus und verarbeitet sie. Für den Abgleich von *tem-*

plates und Dokumentinhalten werden Muster definiert, die sich der Syntax und der *expression language* von XPath bedienen (vgl. oben Kap. 5.2.5). Dies zeigen die folgenden Beispiele:

Muster	*Interpretation*
title	Element vom Typ title
/	Wurzelelement
*[position()=1 and self::absatz]	Jedes Element, das den Typ absatz hat und das erste Kindelement seines Elternelements ist (der erste absatz innerhalb einer Folge von Absätzen, deren Elternelement z. B. kapitel ist)

Codebeispiel 50: Beispiele für Selektionsmuster in XSL-Templates

Die so definierten Muster lassen sich mit Pfadangaben bzw. LocationPaths kombinieren. Der Aufbau eines *template* hat folgende Struktur:

```
<xsl:template match="...">
  <! --Verarbeitungsvorschriften,
       z. B. Formatierungsanweisungen unter Verwendung von XSL-formatting objects (s. u.) oder
       explizite Definition der Ausgabe; Anweisungen zur Verarbeitung von Kindelementen
  -->
</xsl:template>
```

Codebeispiel 51: Aufbau eines XSL-Templates

Das nachfolgende Beispiel zeigt die Anwendung eines solchen *template*. Vorausgesetzt ist ein XML-Text, in dem Zitate durch ein Element <zitat> hervorgehoben sind, z. B. ... führt dazu aus: <zitat> In dieser Hinsicht ... zu bemerken.</zitat>. Wie aus dem ... Um eine geeignete Schriftformatierung für alle Zitate zu erreichen, könnte man folgendes *template* verwenden:

```
<xsl:template match="zitat">
    <fo:inline-sequence font-style="italic">
      <xsl:Apply-templates/>
    </fo:inline-sequence>
  </xsl:template>
```

Codebeispiel 52: Beispiel eines XSL-Template für die Schriftformatierung

Das *template* kennzeichnet alle Elemente des Typs Zitat als ein *formatting object* des Typs inline-sequence mit kursiver Schrift und verarbeitet rekursiv alle Kinder dieser Elemente (d. h. alle Knoten, die unterhalb eines zitat-Elements im Eingabebaum hängen). Die Weiterverarbeitung der Kindelemente durch die XSL-Anweisung apply-templates kann mit Auswahlanweisungen (select) präzisiert werden. Im folgenden Beispiel werden von einem *template* alle Elemente vom Typ kapitel verarbeitet, wobei die Weiterverarbeitung der Kindelemente durch ein select-Attribut auf Elemente vom Typ absatz eingeschränkt ist:

```
<xsl:template match="kapitel">
  <fo:block>
    <Xsl:Apply-templates select=".//absatz">
  </fo:block>
</xsl:template>
```

Codebeispiel 53: Selektive Verarbeitung von Kindelementen in XSL-Templates

Ein drittes Beispiel soll nicht die Anwendung von Formatierungsanweisungen, sondern die Umwandlung von Dokumentstrukturen zeigen; ein Anwendungsfall hierfür ist die

Abbildung einer beliebigen XML-DTD auf XHTML, da ein XHTML-Dokument leichter in einem Webbrowser angezeigt werden kann. Das Beispiel (vereinfacht nach CLARK 1999: Appendix C.1) besteht aus vier Teilen

- Eine DTD für einen einfach strukturierten Bericht,
- ein XSL-*style sheet* zur Transformation dieser DTD,
- das Beispieldokument für die Bericht-DTD und
- XHTML-Output nach den Vorgaben des *style sheet*.

```
<!ELEMENT bericht (titel, kapitel*)>
<!ELEMENT titel #PCDATA>
<!ELEMENT kapitel (titel, absatz*)>
<!ELEMENT absatz #PCDATA>
```

Codebeispiel 54: Beispiel-DTD für die Anwendung eines XSL-style sheet

Die Umwandlung in XHTML soll mit dem nachfolgenden *style sheet* generiert werden. Durch *namespace*-Anweisungen ist festgelegt, dass der *default namespace* der Namensraum von XHTML ist; der XSL-Namensraum ist zusätzlich (xmlns:xsl=...) inkludiert:

```
<xsl:stylesheet  xmlns:xsl="http://www.w3.org/XSL/Transform/1.0"
                 xmlns="http://www.w3.org/TR/xhtml1">
<xsl:strip-space method="xml" elements="bericht kapitel absatz ">
<xsl:output method="xml" indent="yes">

<xsl:template match="bericht">
  <html>
    <head>
      <title>
        <xsl:value-of select="titel"/>
      </title>
    </head>
    <body>
      <xsl:Apply-templates/>
    </body>
  <html>
</xsl:template>

<xsl:template match="bericht/titel">
  <h1>
    <xsl:Apply-templates/>
  </h1>
</xsl:template>

<xsl:template match="kapitel/titel">
  <h2>
    <xsl:Apply-templates/>
  </h2>
</xsl:template>

<xsl:template match="absatz">
  <p>
    <xsl:Apply-templates/>
  </p>
</xsl:template>
```

Codebeispiel 55: XSL-style sheet für die Übersetzung einer XML-DTD nach XHTML

Das *style sheet* gibt für die verschiedenen Elemente der DTD an, welche korrelierenden Elemente der Ziel-DTD zu erzeugen sind; die Inhalte (Kindelemente) der Elemente werden durch die apply-templates-Anweisung verarbeitet. Bei Überschriften (Ausgangs-DTD: titel) wird durch je ein *template* unterschieden, ob es sich um Überschriften erster oder zweiter Ordnung handelt; die entsprechende XHTML-Marke wird generiert. Das *style sheet* soll auf das nachfolgende Beispiel angewandt werden:

```
<!DOCTYPE bericht SYSTEM "bericht.dtd">
<bericht>
    <titel>Huapt&uuml;berschrift</titel>
    <kapitel>
      <titel>Erstes Kapitel</titel>
      <absatz>Erster Absatz</absatz>
      <absatz>Zweiter Absatz</absatz>
    </kapitel>
    <kapitel>
      <titel>Zweites Kapitel</titel>
      <absatz>Erster Absatz</absatz>
      <absatz>Zweiter Absatz</absatz>
    </kapitel>
</bericht>
```

Codebeispiel 56: Ausgangsdokument für die Übersetzung nach XHTML

Als Ausgabe erzeugt der XSL-Prozessor folgendes XHTML-Dokument:

```
<html xmlns="http://www.w3.org/TR/xhtml1">
  <head>
    <title>Haupt&uuml;berschrift</title>
  </head>
  <body>
    <h1> Haupt&uuml;berschrift </h1>
    <h2> Erstes Kapitel </h2>
      <p>Erster Absatz</p>
      <p>Zweiter Absatz</p>
    <h2> Zweites Kapitel </h2>
      <p>Erster Absatz</p>
      <p>Zweiter Absatz</p>
  </body>
</html>
```

Codebeispiel 57: Ausgabedokument in XHTML

Im Beispiel wird dasjenige titel-Element, das unmittelbarer Nachfolger des Wurzelelements bericht ist, zweifach verarbeitet: Im ersten *template*, um als title-Element des Dokumentkopfs ausgegeben zu werden, im zweiten *template*, wo es zur Generierung einer Überschrift erster Ordnung im Dokumentrumpf verarbeitet wird.

Das Beispiel zeigt die Semantik der Auswertung von *templates*: Ist in einem *template* ein Element enthalten, das nicht dem XSL-Namensraum angehört, so wird aus ihm ein gleichnamiges Element erzeugt, d. h. für die body-Marke des ersten *template* in Codebeispiel 55 wird in der Ausgabe ebenfalls eine body-Marke erzeugt (*literal result element*).[91] In den *templates* können zusätzlich Variablen und Parameter eingesetzt werden: Variablenvereinbarungen lassen sich wie folgt treffen:

[91] Strukturbestandteile eines XML-Dokuments (Elemente, Attribute etc.) können in einem *template* durch xsl-Anweisungen *errechnet* werden, vgl. CLARK 1999: Kap. 7.1.2 ff.

```
<xsl:variable name="name_der_variable" select=" Wert_der_Variable ">
```

Codebeispiel 58: Variablenvereinbarung in XSL

Den Einsatz von Variablen zeigt das folgende Beispiel, in dem Attributwerte für die Ausgabe aus XSL-Variablen ermittelt werden. Die XSL-Variablen werden aus den Daten des Eingabedokuments errechnet:

```
<!-- Eingabedokument -->
<abbildung>
  <href>diagramm_fadenpendel.gif</href>
  <groesse hoehe="400" breite="200"/>
</abbildung>

<!-- style sheet -->
<xsl:variable name="abildungen">/mumed/bilder</xsl:variable>
<xsl:template match="abbildung">
  <img src="{$abbildungen}/{href}" width="{groesse/@breite}" heigth="{groesse/@hoehe}"/>
</xsl:template >

<!-- Ausgabedokument -->
<img src="/mumed/bilder/diagramm_fadenpendel.gif" width="200" heigth="400"/>
```

Codebeispiel 59: Verwendung von Variablen und Zugriff auf Attribute und Attributwerte in XSL

Die Ausdrücke für den Zugriff auf Attribute sind in geschweifte Klammern gefasst, Attributnamen werden direkt verwendet, Attributwerte durch ein vorangestelltes @ angesprochen, Variablen durch vorangestelltes $-Zeichen. Durch die Konstrukte <xsl:param .../> bzw. <xsl:with-param .../> lassen sich Parameter definieren und an *templates* übergeben (z. B. um ein Aufzählungsformat festzulegen).

7.1.2.2 Formatierung in XSL

Hinsichtlich der Dokumentformatierung sind in XSL in Weiterführung von CSS Vorgaben für alle wesentlichen Aspekte der Präsentation vorgesehen (vgl. die Übersicht der *XSL Requirements* bei WALSH 1999):

- Allgemeine Formatierung (Positionierung, Ausrichtung, typographische Gestaltung, Seitenaufteilung, Paginierung, Animation/eingebettete Objekte, Tabellensatz, Seitenumbruch und Spaltensatz etc.),
- Verwendung von Farben und Schriften,
- Verwendung unterschiedlicher Schrift- und Schreibsysteme,
- Einbettung ausführbarer Skripte und Interaktivität von Dokumenten (Ereignis- und Formularverarbeitung) und
- Erweiterbarkeit, Modularität und Einbettung von Metainformation zum Formatierungsprozess.

Die Semantik der Formatierung wird in einem *style sheet* durch die Verwendung vordefinierter *formatting objects* (FO) kodiert. Der umfangreiche Katalog von XSL-FOs ist weitgehend mit den in CSS definierten Formatierungskonstrukten identisch. Im zweiten Subprozess von XSL werden die Bestandteile des zunächst erstellten *result trees* als Instanzen dieser *formatting objects* für die Ausgabe interpretiert. Die Repräsentation eines *formatting object* erfolgt als XML-Element mit assoziierter Attribut-Wert-Liste. Das Formatierungsmodell geht von rechteckigen Flächen zw. Abständen (*spaces*) aus, aus denen sich durch area-container, block-area, line-area und inline-area vier unterschiedliche

Subtypen von Ausgabeflächen beschreiben lassen. XSL definiert auf der Basis dieses Modells, wie ein bestimmtes *formatting object* dargestellt werden soll. Dazu ist für jedes *formatting object* festgelegt, wie es verwendet werden soll, welche Attribute für das *formatting object* angewandt werden können (z. B. Farbe, Schrift) und welche Ausgabe ggf. generiert wird.[92] XSL unterscheidet folgende Typen von *formatting objects*:

- Layout und Paginierung: *formatting objects* für die Definition des Seitenlayouts (fo:page-master) bzw. von Seitensequenzen[93] (fo:page-sequence),
- FOs für die Definition von Regionen innerhalb einer Ausgabeeinheit (Seite), z. B. fo:region-start, fo:region:before und fo:region:body,
- FOs für die Bindung von Folgen formatierter Inhalte auf Blockebene an Ausgaberegionen (fo:static-content, fo:flow),
- FOs für die Formatierung auf Absatzebene (*block*), z. B. fo:display-graphic,
- FOs für *inline*-Elemente wie Graphiken im Fließtext, (Einzel-)Zeichen oder eingebettete Container (fo:character, fo:inline-graphic, fo:inline-included-container).
- FOs für Listen, Tabellen und Elemente ohne feste Positionierung (Fußnoten, *floating graphics* etc.) wie fo:table, fo:table-row, fo:list-block, fo:list-item, fo:footnote, fo:float und
- FOs für die Formatierung von Hypertextverknüpfungen und interaktiven Auswahlmöglichkeiten wie fo:link, fo:multi-case und fo:multi-toggle.

Für jedes Formatierungsobjekt ist festgelegt, welche Attribute es tragen kann, z. B. kann man einem fo:block u. a. folgende Attribute zuweisen:

- Allgemeine Attribute zur Positionierung wie position, top, right, bottom, left,
- Eigenschaften bezüglich Rand- und Hintergrundformatierung wie background, border und padding oder
- Schriftformatierung (font, font-family, font-weight etc.).

7.1.2.3 Anwendung und Einordnung von XSL

Im Rahmen einer XML-basierten Prozesskette für die Erstellung und Präsentation elektronischer Bücher spielt die Anwendung von XSL auf mehreren Ebenen eine Rolle:

- XML-Dokumente können nur präsentiert werden, wenn Präsentationsanweisungen – XSL-*style sheets* – definiert sind, die die XML-DTDs interpretieren können; abgesehen von Anwendungen, die für genau eine XML-DTD entwickelt werden, ist XSL somit eine notwendige Voraussetzung für den Einsatz von XML.
- Der *tree transformation process* von XSL eignet sich für die Zusammenführung und Präsentation von Dokumenten, die auf der Basis unterschiedlicher DTDs aufgebaut sind oder aus heterogenen Quellen zusammengeführt werden müssen.
- Eine Überführung in standardisierte Formate wie HTML oder XHTML ist mit XSL ohne weiteres möglich. Dies gilt nicht nur für textorientierte Datenformate: XML-kodierte Datensätze, die aus einer Datenbank selektiert oder durch eine Simulation er-

---

[92] Bei „logischen“ *formatting objects* wie z. B. fo:region-start wird lediglich der Beginn eines fo:region-body-Bereichs definiert; daher sind mit einem region-start keine spezifischen Inhalte assoziiert.

[93] Die Einführung seitenorientierter Layoutelemente ist ein wesentlicher Unterschied zwischen CSS und XSL, der für den im Referenzprojekt definierten Buchviewer anwendbar ist und „traditionelle“ Konzepte des Buchsatzes verfügbar macht: Die vom Inhalt weitgehend unabhängige Aufteilung von Dokumenten in Präsentationseinheiten.

zeugt werden, können durch eine XSL-Transformation in deklarative Graphikformate umgewandelt werden, die dann ein entsprechender Viewer formatiert und anzeigt.[94]

Obwohl die Spezifikationen für XSL und XSLT jüngeren Datums sind,[95] existiert anders als für DSSSL bereits eine recht vielfältige Auswahl von XSL-Prozessoren:

- die Lotus XSL-Engine (*http://www.alphaworks.ibm.com/tech/lotusxsl*),
- der XSLT-Prozessor XT von James Clark (*http://www.jclark.xom/XT*),
- der XSL:P-Prozessor, der im Rahmen der Apache-Serverwerweiterung Cocoon eingesetzt wird (vgl. *http://www.clc-marketing.com/xslp/*).

Dabei ist es für den Benutzer unwichtig, ob ein XSL-Prozessor auf der Serverseite integriert ist und einen XML-Datenstrom vor der Anzeige in einem Webbrowser in HTML/XHTML umwandelt, oder ob der XSL-Prozessor Bestandteil des Browsers ist.

7.2 Cascading Style Sheets

Ein dritter Standard, der die Präsentationsspezifikation in HTML regelt, sind *Cascading Style Sheets* (CSS). Anders als XSL und DSSSL sind sie an eine bestimmte DTD gebunden und enthalten keine Spezifikation eines Transformationsprozesses. Es handelt sich um einfache Stilspezifikationen, in denen für die verschiedenen Elemente der HTML-DTD Layoutattribute festgelegt werden können. Die *style sheets* sind *kaskadierend*, da mehr als nur ein *style sheet* für ein Dokument gültig sein kann und sich *style sheets* gegenseitig überlagern können. Damit wird es dem Leser ermöglicht, durch eigene Layoutvorgaben *style sheets* der Autorenseite zu überschreiben. Gewichtungs- und Auswertungsregeln legen die endgültige Auswahl einer Stilvorgabe fest.

CSS definiert ein Formatierungsmodell, das von DSSSL abgeleitet ist und Grundlage für XSL ist; die Layoutattribute bilden eine Untermenge der in XSL verfügbaren *formatting objects*. Der Kern der CSS-Funktionalität ist in der Spezifikation für *CSS level 1* (LIE & BOS 1999) definiert; *CSS level 2* (BOS et al. 1998) erweitert und generalisiert die Möglichkeiten von CSS1 für zusätzliche Ausgabegeräte, z. B. die Sprachausgabe für in HTML und CSS kodierte Dokumente. Eine Erweiterung zu *CSS level 3* ist in Vorbereitung und wird CSS v. a. um die Spezifikation verschiedener Typen von Benutzerschnittstellen bzw. die Integration interaktiven Verhaltens (Ereignisverarbeitung; Einbettung von Skripten) in HTML-Dokumente durch CSS-Formatierungsanweisungen ergänzen (vgl. APPARAO, GLAZMAN & WILSON 1999; ÇELIK, LINSS & KUWAMOTO 1999).

CSS unterscheidet sich von XSL nicht nur hinsichtlich des eingeschränkten Anwendungsbereichs, sondern auch in seiner mehr an DSSSL orientierten Syntax; die Zuweisungen von Stilspezifikationen zu Elementen der HTML-DTD erfolgt in einem *cascading style sheet* nach folgendem Muster:

```
Elementname(n) {   Formatattribut1 : Attributwert1;
                   Formatattribut2 : Attributwert2;
                   ....
                   FormatattributN : AttributwertN;
               }
```

Codebeispiel 60: Schematischer Aufbau eines cascading style sheet

[94] Dabei ist vor allem an die XML-basierte Spezifikation der *scalable vector graphics* (SVG) zu denken, vgl. Clark 1999: Appendix 3.2 und FERRAIOLO 1999.

[95] Der erste Entwurf für XSL stammt aus dem Jahr 1997; die Aufteilung bzw. Ergänzung durch XSL/T von Mitte 1999, vgl. ADLER 1997, DEACH 1999, CLARK 1999.

Das nachfolgende Beispiel zeigt ein einfaches *style sheet*, das im Rahmen des Referenzprojekts für die Layoutkodierung eingesetzt wird:

```
body  { font-family:Arial;
        font-style: normal;
        color: black;
        font-size: 14pt
      }
p     { font-size: 12pt }
h1    { color: black;
        font-size: 36pt
      }
h2    { color: red;
        font-size: 24pt
      }
h3    { font-size: 18pt }
h4    { font-style: bold;
        color: black;
        font-size: 12pt }
h5    { color: red;
        font-size: 12pt
      }
a     { text-decoration:none;
        color: green; font-size: 12pt
      }
em    { color: #444444 }
code  { color : blue;
        font-style:italic
      }
```

Codebeispiel 61: Beispiel eines Cascading Style Sheet

Bei der Auswertung des *style sheet* gelten Vererbungsregeln, die bestimmen, dass Kindelemente Eigenschaften ihrer Elternelemente übernehmen, soweit sie das jeweilige Attribut nicht explizit überschreiben. Die Integration der Formatanweisungen bei der Darstellung eines HTML-Dokuments erfolgt auf eine der folgenden Arten:

- durch eine separate *style sheet-Datei*, die über ein link-Element im Kopf der HTML-Datei referenziert wird,
- über explizite style-Elemente im Dokumentkopf,
- durch eine CSS-@import-Anweisung, die ebenfalls eine externe *style sheet*-Datei lädt oder
- elementweise als style-Attribut einzelner HTML-Marken.

Die Anwendung zeigt exemplarisch der nachfolgende Codeausschnitt aus einem HTML-Dokument:

```
<html>
  <head>
    <!-- Einbindung eines style sheet durch ein link-Element: -->
    <link rel="stylesheet" type="text/css" href="basic1.css" title="style1">
    <!-- Explizite style-Elemente und Einbindung durch @import-Anweisung -->
    <style type="text/css" title="style2">
      EM {color : blue }
      @import url(basic2.css);
    </style>
  </head>
```

```
  <body>
    <h1>CSS-Einbindung</h1>
    <p> Der <em>hervorgehobene Text ist blau gef&auml;rbt</em>, wenn nicht explizit
        <em style="color : pink">pink als Stil im em-Element angegeben wird.</em>
    </p>
  </body>
</html>
```

Codebeispiel 62: Einbindung von CSS-Stilangaben in HTML-Dokumenten

Die Vorteile von CSS – einfache Syntax, zahlreiche Formatierungsmerkmale und weit verbreitete Implementierungen in HTML-Browsern – lassen diesen Standard in der Kombination mit XSL als einen sinnvollen Bestandteil einer Prozesskette für die Produktion dynamischer elektronischer Bücher erscheinen: Ausgehend von XML-Dokumenten kann XSL genutzt werden, um diese Dokumente nach HTML/CSS zu konvertieren und anschließend zu präsentieren. Im einfachsten Fall erfordert dies, anders als eine rein XSL-basierte Lösung, keine Erweiterung der Funktionalität des WWW-Clients, da der Datenstrom, der vom Server geladen wird, bereits nach HTML/CSS umgewandelt ist. Andere Lösungen (s. o. Kap. 7.1.2.3 zu XSL-Werkzeugen) wie die Kombination von XML-Dokumenten mit CSS benötigen auf dem Client zusätzliche Module (einen XSL-Prozessor, vgl. LIE 1998).

8 Kodierung von Multimediainformation

Die voranstehenden Kapitel haben sich mit der Strukturierung und Darstellung von Information befasst. Die Problematik multimedialer Daten wurde dabei ausgespart. Unter einer Multimediakomponente sei ein Informationsbestand verstanden, der kontinuierliche, zeitabhängige Medien beinhaltet und interaktiv nutzbar ist (vgl. STEINMETZ 1999: 7 ff., 815 ff., SCHULMEISTER 1997: 39 ff.). In der Regel wird die Verwendung zeitabhängiger Medien als das konstituierende Merkmal multimedialer Anwendungen verstanden; für dynamische elektronische Bücher ist aber auch der Aspekt der Interaktivität wesentlich. Unter die allgemeine Definition eines *multimedia document* nach HyTime (ISO 10744), „a collection of information that is identified as a unit and that is intended for human perception" fallen grundsätzlich sowohl *diskrete/zeitunabhängige* als auch *kontinuierliche* bzw. *zeitabhängige* Medien. In den oben diskutierten (Meta-)Sprachen XML und SGML bzw. HTML und ihren Präsentationsspezifikationen lassen sich multimediale Elemente (Filme, Animationen, Simulationen, Audiodateien) allenfalls als in ihrer Struktur nicht transparente Komponenten einbinden, d. h. durch Verwendung geeigneter plug-ins können in gängigen Webbrowsern Multimediadateien integriert und abgespielt werden; ihre Binnenstruktur ist aber nicht oder nur bedingt deklarativ kodiert. Dies gilt auch für die Autoren- bzw. Produktionsseite: Autorensysteme für Multimediaanwendungen wie Macromedia Director oder Asymetrix ToolBook verwenden proprietäre Datenformate ohne deklarative Informationsstrukturierung. Für die Verwendung deklarativer Informationsstrukturierung in Multimediaanwendungen sprechen aber eine Reihe von Gründen:

- Ein deklaratives Auszeichnungsformat kann als Austauschformat zwischen verschiedenen Multimediaanwendungen dienen und die Wiederverwendbarkeit der in der Regel mit beträchtlichem Kosten- und Ressourcenaufwand entwickelten Multimediakomponenten unterstützen.
- Durch ein deklaratives Format ergeben sich neue Anwendungsmöglichkeiten im Bereich des dynamischen Verhaltens solcher Komponenten, da ihre innere Struktur zur Laufzeit analysiert und ggf. beeinflusst werden kann.
- Die Integration über die einfache Einbindung als Komponenten kann verbessert werden, z. B. hinsichtlich der Einbindung von Hypertextverknüpfungen oder der standardisierten Anbindung zusätzlicher Anwendungen.

Für die einheitliche Kodierung und Darstellung von Multimediaanwendungen haben sich eine Reihe von Standards entwickelt; es geht nachfolgend aber nicht um die Kodierung von Einzelformaten, d. h. um Standards für die Kodierung von Videoanwendungen (z. B. MPEG, AVI, QuickTime) oder Audiodateien (MP3, RealAudio), sondern nur um den Aspekt der Integration generischer Medienelemente in interaktiven Multimediaanwendungen.[96] Zu diesen Standards gehören u. a.

- MHEG (*Multimedia and Hypermedia information coding Expert Group*, ISO/IEC 13522, vgl. MEYER-BOUDNIK & EFFELSBERG 1995)

[96] Eine umfassende Übersicht zu den verschiedenen Standards und Formaten für einzelne Medientypen gibt STEINMETZ 1999.

- die *Hypermedia/Time-Based Structuring Language* (HyTime, vgl. NEWCOMB & NEWCOMB 1992, NEWCOMB 1995A, 1995B) sowie
- die *Synchronized Multimedia Integration Language* (SMIL, vgl. BOUTHILLIER 1998, HOSCHKA et al. 1998, AYARS et al. 2000).[97]

Die beiden letzteren Standards basieren auf SGML und lassen sich somit in die in den vorangegangenen Kapiteln erörterte Infrastruktur für die deklarative Informationsstrukturierung integrieren. Die nachfolgende Darstellung beschränkt sich auf HyTime und SMIL. Es ist eine Entwicklung zu beobachten, die analog zur Ausbildung von Präsentationsstandards im SGML-Bereich verläuft: HyTime ist ein seit längerem verfügbarer internationaler Standard, für den aber nur wenige Anwendungen und Softwaresysteme existieren,[98] während sich SMIL aufbauend auf HyTime-Konzepten als Standardisierungsvorschlag des W3C in kurzer Zeit zu einem anwendungsnahen Standard entwickelt hat.

Das zentrale Problem beider Standards ist die Verarbeitung temporaler Constraints in Multimediasystemen. Ein Multimediadokument wird allgemein als eine Informationssammlung (*collection of information*) verstanden, die sich logisch in einzelne *media items* gliedert. Im Gegensatz dazu verwendet der klassische Dokumentbegriff keine Zeitabhängigkeit und sieht keine Präsentationsdauer vor, da nur diskrete Medien (Text, Bild etc.) im Dokument enthalten sind. Bei der deklarativen Strukturierung der Medienelemente stellen sich folgende Fragen:

- Wie lassen sich die *media items* durch den Einsatz temporaler Beschränkungen (*temporal constraints*) ausrichten bzw. anordnen (*alignment*) ?
- Wie können zeitabhängige Medien adressiert werden, um z. B. interaktive Hypertextverknüpfungen in zeitabhängigen Medien verwenden zu können ?

Sowohl in HyTime als auch in SMIL werden Zeitachsen eingeführt (Zeitstrahl, *timeline*), entlang derer sich Medienelemente ausrichten und adressieren lassen. Ihre zeitliche Erstreckung wird durch

- die Startzeit des Abspielens,
- ihre Abspieldauer und
- die Endzeit des Abspielens

kodiert. Die Abspieldauer (*duration*) ergibt sich entweder aus

- der Festlegung durch den Autor oder
- auf Präsentationsebene durch direkte Interaktion des Benutzers bzw. durch Abhängigkeit von anderen *media items*.

Prinzipiell können zwischen verschiedenen *media items* unterschiedliche Relationen bestehen:

- Bezug der Startpunkte zweier *media items*,
- Beziehungen zwischen Start und Ende,
- Beziehungen zwischen zwei Endpositionen,
- Beziehung zwischen den Abspiellängen.

---

[97] Zu weiteren Multimediastandards vgl. ERFLE 1994: 451 ff., BOLL, KLAS & WESTERMANN 1999A, 1999B. ERFLE 1993: 399 ff. gibt einen ausführlichen Vergleich verschiedener Standards hinsichtlich der Kodierung zeitbezogener Information.

[98] Vgl. NEWCOMB 1995B und die Liste verfügbarer Werkzeuge unter *http://www.hytime.org/tools*.

Bei Kombination der verschiedenen Relationen ergeben sich 13 unterschiedliche temporale Bezüge zwischen zwei *media items*. Die Platzierung auf einem absoluten Zeitstrahl ist ungeeignet zur Erfassung der Benutzerinteraktion, da in diesem Fall die absoluten Ereigniszeitpunkte nicht *a priori* bekannt sind. Ein einfaches Beispiel eines durch HyTime bzw. SMIL deklarativ zu beschreibenden Kodierungsproblems könnte z. B. folgende Situation sein: *Starte Videosequenz A 10 sec. nach Ende der Audiosequenz B.*

8.1 Hypermedia/Time-Based Structuring Language

Ähnlich wie der Präsentations- und Transformationsstandard DSSSL hat HyTime eine *Doppelfunktion* als Sprache für die Kodierung multimedialer Information sowie als eine Art konzeptueller Überbau zu SGML. Die *Hypermedia/Time-based structuring Language* ist als ISO/IEC 10744 seit 1992 ein internationaler Standard; sie wurde seit 1984 durch Charles GOLDFARB als „Überbau" zu SGML und parallel zur *Standard Music Description Language (SDML)* entwickelt (vgl. NEWCOMB & NEWCOMB 1992: 459). NEWCOMB et al. 1991: 68 definieren HyTime wie folgt:

> HyTime is a standard neutral markup Language for representing hypertext, multimedia, hypermedia, and time- and space-based documents in terms of their logical structure. Its purpose is to make hyperdocuments interoperable and maintainable over the long term. HyTime can be used to represent documents containing any combination of digital notations. HyTime is parsable as Standard Generalized Markup Language (ISO 8879:1986). HyTime provides standardized means of expressing
>
> (1) intra- and extra-document locations, and arbitrary links between them,
> (2) the scheduling of multimedia objects in 'finite coordinate spaces', and
> (3) rendering instructions for arbitrarily projecting such objects onto other finite coordinate spaces, and other constructs.

Wie SGML definiert HyTime vor allem die logische Struktur unabhängig von der Kodierung einzelner Medienelemente und verwendet dazu die syntaktischen Konventionen von SGML. Es werden die bereits in Kap. 5.1.2 angesprochenen *architectural forms* als „Oberklassen" für SGML-DTDs als („Meta-DTDs") eingeführt. Der HyTime-Standard besteht aus fünf Modulen (vgl. GOLDFARB et al. 1997, ISO/IEC 10744:1997):

1. Basismodul: Enthält einige *architectural forms* und SGML.
2. Das *location address module*, das die Syntax für die Adressierung von Informationseinheiten definiert.
3. Das *hyperlinks module* instantiiert das *location address module* und definiert Typen von Verknüpfungen zwischen Informationseinheiten.
4. Das *scheduling module* beschreibt finite Koordinatenräume (*finite coordinate spaces*, FCS) beliebiger Dimensionalität, in denen die Medienelemente angeordnet werden können und Ereignisse (*events*) als Teile solcher Räume, die Medienelemente enthalten können.
5. Das *rendition module* beschreibt Verfahren zur Projektion und Modifikation von Ereignissen in unterschiedlichen Räumen.

HyTime bietet somit Lösungsansätze für

- *linking*: Aufbau von Verbindungen zwischen einzelnen Medienelementen,
- *scheduling/rendition/alignment*: Ereigniskonzept mit *bounding boxes*, darin jeweils Referenzen auf *media items*.

Für die Adressierung und Referenzierung von Medienelementen verwendet HyTime verschiedene Verknüpfungstypen und Adressierungsarten. HyTime unterscheidet zwischen

- *contextual links* (clink), bei denen sich der Anker der Verknüpfung innerhalb des Dokuments befindet, von dem die Verknüpfung ausgeht (inline-Links, vgl. oben Kap. 5.2.4) und
- *independent links* (ilink), die außerhalb der miteinander verknüpften Ressourcen kodiert und verwaltet werden können und somit die Möglichkeit von *out-of-line-links* definieren.

Die letztere Möglichkeit der externen Linkverwaltung ist bei multimedialer Information erforderlich, da man bei binär kodierten Multimediaformaten nicht die Möglichkeit hat, in den Medienelementen selbst deklarativ Verknüpfungen einzufügen und die Links *out-of-line* verwaltet werden müssen.

Neben diesen Linktypen definiert HyTime unterschiedliche Arten der Adressierung, sowohl eine *absolute* Adressierung über ID-Attribute von SGML-Elementen (nameloc) als auch eine *relative* Adressierung über Positionen im SGML-Elementbaum (treeloc). Eine dritte Möglichkeit ist die datenbezogene Adressierung (dataloc), bei der die Adressierung über einen Zähler primitiver Datenelemente (Wörter, Frames etc.) erfolgt. Zusätzlich definiert HyTime die Abfragesprache HyQ, ein Vorläufer der SDQL von DSSSL (vgl. oben Kap. 7.1.1) und der in XPath und XPointer definierten Adressierungsmöglichkeiten.

Die physische Strukturierung eines HyTime-Dokuments erschließt die einzelnen Medienelemente durch Referenzierung, d. h. sie sind nicht Teil des Dokuments selbst und werden vom HyTime-System zur Laufzeit durch die entsprechenden Referenzen identifiziert und eingebunden. Grundlage für das *alignment* bzw. die Präsentation zeitabhängiger Daten ist in HyTime das *Event*-Konzept, das für jedes *media item* eine *bounding box* kodiert, in der das Element logisch, d. h. durch eine Referenz auf die Binärdaten z. B. eines Films, enthalten ist und die zur Präsentationsplanung verwendet wird (*event scheduling*).

Die *bounding boxes* der Medienereignisse sind in finiten Koordinatenräumen (*finite coordinate space*) platziert und erhalten durch Angabe des *extent* eines *media items* bezogen auf die *dimension* jeder Achse eine adressierbare Struktur, wobei sich die logische Gliederung in kleinste Einheiten (*quantum*) aus der Definition des Koordinatenraums ergibt.

Bei relativer Adressierung bezogen auf andere *media items* müssen zwischen den Medienelementen *event references* eingeführt werden, die durch *dimension references* auf

- *first quantum,*
- *quantum count* oder
- *last quantum.*

eines Ereignisses verweisen. Mit dem Referenzmechanismus können alle genannten Relationstypen kodiert werden (Beispiele im Annex C der Norm, jeweils mit Kodierungsalternativen). In einem Dokument können absolute Anordnung und Referenzen gemischt kodiert werden. Zusätzlich bietet HyTime durch Funktionen und Operatoren (*HyFunk*, *HyOp*) die Möglichkeit, Medienreferenzen dynamisch zu spezifizieren bzw. zur Laufzeit zu errechnen. vgl. dazu Beispiele bei ERFLE 1993, 1994.

Besteht zwischen den Medienelementen und der angestrebten Darstellungsplattform ein Unterschied bezüglich der Granularität der Strukturierung (verschiedene Skalen und/oder Einheiten), so kann man durch Einführung von Projektionsmechanismen verschiedene Koordinatenräume aufeinander abbilden. Dazu führt HyTime das Konzept

eines *projector* ein, dem ein *projector scope* zugeordnet ist und der verwendet werden kann, um ein Medienelement von einem Ausgangs- in einen Zielkoordinatenraum abzubilden. Ein Beispiel für einen solchen Vorgang ist die Abbildung einer in einzelne (logische) *frames* aufgeteilten Multimediaanimation auf einen Zeitstrahl, bei dem eine physische Einheit und nicht eine logische Einteilung die Gliederungseinheit darstellt.

Aufgrund seiner großen Komplexität hat HyTime bisher vor allem im Bereich der technischen Dokumentation bzw. der öffentlichen Verwaltung Anwendung gefunden, so im Rahmen der *Interactive Technical Manuals*-Initiative *(IETM)* des US-Verteidigungsministeriums (vgl. NEWCOMB 1995B mit weiteren Beispielen). Obwohl viele HyTime-Anwendungen naheliegend erscheinen, vor allem für die Kodierung und Verwaltung komplexer Multimediaanwendungen, hat HyTime bisher keine Auswirkungen auf Multimedia-Autorensysteme gehabt und – trotz seines Status als SGML-Schwesterstandard – im Bereich des World Wide Web kaum Beachtung gefunden. Dies gilt allerdings nicht für die konzeptuelle Ebene der Weiterentwicklung von Spezifikationen für das World Wide Web, da in zahlreichen Standards Ideen aus HyTime aufgegriffen worden sind (XPath, XPointer). Am deutlichsten erscheint dies bei SMIL, einer für das World Wide Web spezifizierten Multimediasprache, die wie HyTime die Problematik der Kodierung zeitabhängiger Information lösen soll.

8.2 Synchronized Multimedia Integration Language

Wie HyTime ist die vom W3C als Standard vorgeschlagene *Synchronized Multimedia Integration Language* eine deklarative Sprache für die Kodierung von Multimediainformation. Sie ist formal als XML-DTD definiert und kann von XML-basierten Werkzeugen verarbeitet werden, wenigstens insofern die Manipulation der in einerm SMIL-Dokument definierten Strukturen im Vorfeld der eigentlichen Präsentation erforderlich ist. Für die Präsentation von bzw. die Interaktion mit SMIL-Dokumenten sind allerdings spezifisch für SMIL entwickelte Viewer oder plug-ins erforderlich.

Das Ziel von SMIL ist es, dem Entwickler einen Standard an die Hand zu geben, mit dem interaktive, zeitabhängige Multimediapräsentationen kodiert werden können; die Komplexität ist im Vergleich mit HyTime deutlich reduziert; SMIL beschränkt sich auf die Aspekte der Multimediakodierung im engeren Sinn und enthält anders als HyTime keine metasprachlichen Konstrukte. Es beschreibt folgende Aspekte einer Multimediapräsentation: Das Zeitverhalten einer Multimediapräsentation, ihr Layout am Bildschirm und die Verlinkung der Medienobjekte bzw. ihre interaktive Nutzung.

8.2.1 Aufbau von SMIL-Dokumenten

Analog zu HTML ist ein SMIL-Dokument in Kopf und Körper eingeteilt, d. h. ein einfaches SMIL-Dokument weist folgende Struktur auf:

```
<smil>  <head><!—z. B. Layoutinformation --></head>
        <body><!—z. B. Definition von Ablaufsequenzen --></body>
</smil>
```

Codebeispiel 63: Grundstruktur eines SMIL-Dokuments

Das <head>-Element kann folgende top-level-Elemente von SMIL enthalten:

- <layout> Angabe von Layoutinformation,
- <meta> Angabe von Metainformation,
- <switch> Auswahlsteuerung, z. B. für alternative Layouts.

Innerhalb des Dokumentkopfs wird das grundlegende Layout einer Präsentation angegeben. Das layout-Element hat selbst eine Reihe von Kindern, die die Spezifika des räumlichen und zeitlichen Layouts präzisieren (region-Element). Die Layoutkodierung arbeitet mit dem Konzept einer *abstrakten Darstellungsfläche* (*abstract rendering surface*), die sich dann je nach Art der Medienelemente als visuell oder auditiv (oder beides) konkretisiert. Dazu können unter Verwendung von switch-Konstrukten (Auswahlanweisungen) verschiedene Layoutvarianten kodiert werden (z. B. mit Hilfe alternativer Layoutsprachen wie CSS2). Eine solche Layoutalternative zeigt das nachfolgende Beispiel, in dem eine region r durch zwei verschiedene Kodierungsstandards definiert wird (CSS und SMIL). Im Rumpf des Dokuments wird in der Teilfläche ein Bild zehn Sekunden lang angezeigt.

```
<smil>
  <head>
    <switch>
      <layout  type="text/css"> [region="r"]  {  top: 20px;  left: 20px  }    </layout>
      <layout><region  id="r" top="20"left="20"   />                          </layout>
    </switch>
  </head>
  <body>
    <seq> <img region="r" src=”http://www.w3.org/test” dur="10s"   /></seq>
  </body>
</smil>
```

Codebeispiel 64: Layoutalternativen in SMIL

Layoutangaben können sowohl in der SMIL-Syntax als auch in dem in CSS2 vorhandenen Verfahren angegeben werden. Die nachfolgende Tabelle zeigt die wesentlichen Elemente von SMIL und gibt an, welchem Layouttypus sie entsprechen (absatz- oder blockbasiert bzw. ohne Darstellung bei Kontrollflusselementen).

SMIL-Element	*Erläuterung*	*Darstellungsanweisung (CSS2)*
A	Startpunkt einer Verknüpfung	{display:block}
ANCHOR	Anker einer Verknüpfung	{display:block}
ANIMATION	Element für die Einbindung einer Animation	{display: block; position: absolute}
BODY	Beinhaltet den Dokumentrumpf	{display: block}
HEAD	Beinhaltet den Dokumentkopf	{display: none}
IMG	Bindet ein Bild (Bitmap) ein	{display: block; position: absolute}
LAYOUT	Definiert die Positionierung von Flächen in der Darstellungsfläche	{display: none}
META	Kodiert Metainformation zu einem SMIL-Dokument	{display: none}
PAR	Kodiert die synchrone (parallele) Ausführung zweier Medienelemente	{display: block}
REGION	Definiert einen Darstellungsbereich	{display: none}
REF	Generisches Element bei nicht eindeutiger Kategorisierung eines Medienelements	{display: block; position: absolute}
ROOT-LAYOUT	Wurzelelement der Layoutspezifikation	{display: none}
SEQ	Steuerungselement für die sequentielle Wiedergabe von Medienelementen	{display: block}
SMIL	Wurzelelement von SMIL-Dokumenten	{display: block}
SWITCH	Steuerelement für Auswahlanweisungen	{display:block}
TEXT	Kodiert ein Medienelement vom Typ Text	{display: block; position: absolute}
TEXTSTREAM	Kodiert ein Medienelement vom Typ Textstrom	{display: block; position: absolute}
VIDEO	Kodiert ein Medienelement vom Typ Video	{display: block; position: absolute}

Tabelle 31: SMIL-Elemente und ihre Darstellung

Der Dokumentrumpf enthält die Spezifikation der eigentlichen Inhalte der Präsentation und ihres logischen Zusammenhangs, d. h. hier wird angegeben, welche Elemente (z. B. Videos, Text etc.) auftreten sollen, wie ihr zeitlicher Ablauf ist und welche Verknüpfungen vorhanden sein sollen.

Die in SMIL 1.0 vordefinierten Medientypen sind ANIMATION, AUDIO, IMG, REF, TEXT, TEXTSTREAM und VIDEO. Über die Interpretation der Elemente ist in der DTD nichts spezifiziert, d. h. die Semantik der Präsentation eines Animationselements bleibt der Implementierung des Viewers vorbehalten. Es gelten folgende syntaktische Konventionen:

- Alle Medienelemente werden durch einen Hyperlink auf der Basis des URI-Schemas in das Dokument eingebunden.
- Es gilt die Unterscheidung von diskreten Medienelementen (*without intrinsic duration*): TEXT, IMG und kontinuierliche Medienelemente (*with intrinsic duration*): ANIMATION, AUDIO, TEXTSTREM, REF, VIDEO.

Nur die kontinuierlichen Medienelemente können zusätzlich clip-Attribute tragen, die das Zeitverhalten des je abgespielten Clips steuern. Tabelle 32 zeigt einige der möglichen Attribute von SMIL-Elementen.

Attribut	*Interpretation*
begin	Startzeitpunkt eines Medienelements
clip-begin	Startzeitpunkt eines Unterbereichs (*clip*) eines Medienelements
clip-end	Endzeitpunkt eines Unterbereichs (*clip*) eines Medienelements
dur	Abspieldauer eines Medienelements
end	Endzeitpunkt eines Medienelements
src	Verweis auf die Datenquelle eines Medienelements
system-screen-size	Spezifikation der Bildschirmgröße

Tabelle 32: Ausgewählte Attributtypen in SMIL

8.2.2 Synchronisationssteuerung in SMIL

SMIL enthält zwei Grundtypen der Mediensynchronisation: Sequentieller und paralleler Ablauf. Das par-Element dient der zeitlichen Steuerung der Parallelführung und Überlappung von Multimediaelementen. Das erste Beispiel zeigt den zeitverzögerten Beginn des Abspielens eines Elements:

```
<par>
  <audio id="a" begin="6s" src="audio" />
</par>
|---------------par------------|
<------6s----->|------a-------|
```

Codebeispiel 65: Zeitverzögerter Beginn des Abspielens

Das zweite Beispiel zeigt die Synchronisation durch eine Referenzierung zwischen Medienelementen mit Hilfe von ID-Attributen:

- Während der ersten sechs Sekunden geschieht nichts,
- dann startet eine Audiodatei,
- nach weiteren vier Sekunden (also nach insgesamt 10 Sekunden) wird ein Bild eingeblendet:

```
<par> <audio id="a" begin="6s"   />
      <img begin="id(a)(4s)"     />
</par>
```

```
|---------------par---------------------|
<------6s------> |-------------a-------|
                   <---4s--->|---img--|
```

Codebeispiel 66: Synchronisation zweier Medienelemente

Neben der parallelen Darstellung können Elemente durch Verwendung von seq-Marken sequentiell dargestellt werden. Wie die beiden Beispiele zeigen, trennt SMIL Layout und Steuerung: Das Layout für eine Präsentation ist im Kopf, seine Zeitsteuerung, d. h. der eigentliche Ablauf, ist im Rumpf des SMIL-Dokuments enthalten.

Das tatsächliche Ergebnis der Synchronisation hängt von der Funktionalität des Viewers ab, z. B. kann bei der parallelen Darstellung mehrerer Clips jeder Clip nach einer eigenen Uhr getaktet sein (*hard synchronisation*) oder jedes Kind eines par-Elements erhält eine eigene Uhr (*soft synchronisation*). Jedes Medienelement eines SMIL-Dokuments verfügt über Startzeitpunkt (begin), Dauer (dur) und Ende (end). Sie werden nach den Eigenschaften *implizit*/*explizit* und gewünscht/tatsächlich (*effective*) unterschieden. Der Benutzer nimmt jeweils die *effektiven* Werte wahr. Sie werden nach einem im SMIL-Standard definierten Darstellungsalgorithmus aus den impliziten, expliziten und gewünschten Werten errechnet (vgl. HOSCHKA et al. 1998: Kap. 4.2.4).

Sowohl in den Layoutanweisungen im Dokumentkopf als auch in der Multimediakodierung im Dokumentrumpf können Alternativen kodiert werden, d. h. SMIL verfügt über Anweisungen zur Steuerung des Kontrollflusses. Unter mehreren Alternativen wählt der Viewer die erste akzeptable Alternative aus. Bei der Kodierung sollten alternative Darstellungen in absteigender Priorität kodiert werden. Um zu testen, ob eine Darstellungsalternative akzeptabel ist, wertet der Viewer eine Reihe von sog. Testattributen wie die Systembitrate, die Bildschirmauflösung und die Farbtiefe der Bildschirmdarstellung, wie das folgende Beispiel für die Auswahl zweier Audioelemente mit unterschiedlicher erforderlicher Systembitrate (*Quality-of-Service*-Parameter) zeigt:

```
<SWITCH>
  <AUDIO src="audiodatei_hohe_qualitaet" system-bitrate="16000" />
  <AUDIO src="audiodatei_normale_qualitaet" system-bitrate="8000" />
</SWITCH>
```

Codebeispiel 67: Medienauswahl auf der Basis der Bitrate:

8.2.3 Anwendung und Diskussion

Für die Präsentation einer SMIL-Datei ist ein Viewer erforderlich, der SMIL-Dateien parsen und die in ihnen enthaltenen multimedialen Inhalte darstellen kann; derzeit existieren eine Reihe solcher Viewing-Systeme, die unterschiedliche technische Lösungen bereitstellen, etwa

- die Realisierung als Java-Applet, das SMIL-Dateien darstellt und so die Möglichkeit bietet, sie in HTML-Seiten einzubetten. Diese Möglichkeit realisiert der SMIL-Viewer SOJA (*SMIL Output in Java Applets*), vgl. *http://www.helio.org/products/smil/*.
- eigenständige SMIL-Viewer, die unabhängig von einem WebBrowser sind, z. B. der GRINS-Player für SMIL (vgl. *http://www.cni.nl/GRiNS* oder *http://www.real.com* (Real G2-Player)).

SMIL stellt eine eigenständige Anwendung von XML dar; die Integration in HTML ist Bestandteil einer umfassenden Weiterentwicklung von SMIL (SMIL Boston), die neben

der inhaltlichen Erweiterung und Differenzierung des Standards (z. B. durch zusätzliche Synchronisationsmöglichkeiten) seine Modularisierung und Integration in die bestehende Infrastruktur an Standards und Werkzeugen anstrebt. SMIL Boston ist in die Module *animation, content control, event, layout, linking, media objects, metainformation, structure, timing and synchronization* sowie *transition effects* gegliedert, vgl. die Übersicht von TEN KATE, WUGOFSKI & SCHMITZ 2000 (Teil 2 der SMIL Boston-Spezifikation, AYARS et al. 2000).

Im Rahmen des Referenzprojekts wurden atomare zeitabhängige Medienelemente wie Videosequenzen nicht erstellt bzw. werden in noch andauernden Arbeiten für die Evaluierung von SMIL anhand eines Einzelbeispiels (Versuchsablauf *Dampfdichte nach Menzies*) exemplarisch untersucht. Da die im Referenzprojekt verfügbaren Autorensysteme SMIL bisher nicht unterstützen, konnte der Standard nicht berücksichtigt werden. Ohne geeignete Hilfe durch Autorensysteme erscheint die Umsetzung deklarativen Markups für komplexe interaktive Multimediaanwendungen aber als kaum realistisch.

Neben dieser technischen und pragmatischen Einordnung ist aber auch die inhaltsbezogene Fragestellung relevant, für welche Typen interaktiver Komponenten sich SMIL grundsätzlich eignet: Dabei scheiden komplexere Multimediakomponenten, die wie die Simulationen im Referenzprojekt die Flexibilität und Mächtigkeit einer höheren Programmiersprache erfordern, aus; auch interaktive Animationen lassen sich kaum durch diesen Standard beschreiben, da sie neben der Verknüpfung zeitabhängiger Medien und einer Benutzerschnittstelle noch über vielfältige Funktionen der Visualisierung und Medientransformation verfügen, die sich mit SMIL nur unter sehr hohem Kodierungsaufwand beschreiben ließen. Am ehesten in Frage kommen für die Anwendung von SMIL Anwendungen, die konzeptuell ein Hypertextnetz entstehen lassen, dessen Knoten auch Multimediaelemente sein können und deren Verknüpfung durch SMIL beschrieben werden; im Referenzprojekt fallen in diese Kategorie interaktive Versuchsdarstellungen, die eine den *image maps* verwandte Funktionalität aufweisen.

9 Metadatenkodierung für elektronische Publikationen

Die Unterscheidung zwischen Daten und Metadaten ist bei zahlreichen Anwendungen der Informationstechnologie von Bedeutung: Sie trennt die von einem System zu verarbeitenden Daten von den dazu benötigen Hilfsinformationen. Ein Beispiel sind Metadaten in der Datenbanktechnologie, die die Struktur von Tabellen einer Datenbank oder die Mächtigkeit der Datenmanipulationssprache beschreiben, und auf diese Weise konzeptuell von den in der Datenbank gespeicherten Informationen getrennt sind. Für das elektronische Publizieren sind bibliographische Datensätze, mit denen ein Werk beschrieben wird und über die es sich z. B. in einer digitalen Bibliothek erschließen lässt, ein typisches Beispiel für die Einbindung von Metadaten. Neben der eigentlichen Inhaltskodierung und der Spezifikation von Verknüpfungen und Multimediainformation spielt die Kodierung von Metadaten eine traditionell wichtige Rolle im Kontext des elektronischen Publizierens. Unter Metadaten als „Daten über Daten“ werden im Allgemeinen alle Informationen verstanden, die nicht zum eigentlich kodierten Informationsbestand gehören, aber zu dessen Speicherung, Erschließung und Darstellung benötigt werden. BERNERS-LEE fasst dies zu folgender Definition zusammen: „Machine readable information about web resources and other things“ (BERNERS-LEE 1997A). Metadatenstandards sind ein Schlüsselwerkzeug der Dynamisierung von Dokumenten, da mit ihnen im Rahmen einer modularen Architektur Zusatzinformation für unterschiedlichste Anwendungen deklarativ kodiert werden kann. LASSILA 1998: 31 nennt folgende Anwendungsgebiete für Metadaten im World Wide Web:

- *Cataloging*, d. h. Kodierung bibliographischer Information, Suchmaschinen und Ressourcenermittlung (Unterstützung der Informationserschließung durch Metadaten),
- *Electronic Commerce*, d. h. die Kodierung von Metadaten für den elektronischen Handel im Internet,
- *Content Rating*, d. h. die Bewertung von Information hinsichtlich ihrer Inhalte,[99]
- *Intellectual Property Rights*, also die Absicherung bzw. Publikation von Rechten an Publikationsinhalten durch Metadaten, die Kodierung digitaler Signaturen durch Metadaten und
- *Privacy*, d. h. die Kodierung von Information über den Grad an Privatheit, die ein Benutzer personenbezogenen Daten beimisst und die eine unberechtigte Auswertung verhindern helfen (soll).

Die zusätzliche Funktionalität dynamischer elektronischer Bücher bietet vielfältige Ansatzpunkte für die Einführung von Metainformation. Zu den Anwendungsbereichen für Metainformation gehören u. a.

- die kontextsensitive Auswertung von Verknüpfungen,
- die Integration zusätzlicher Softwaremodule,
- die Auswertung benutzerbezogener Information oder
- die Informationserschließung.

[99] Der Standardisierungsvorschlag für eine *Platform for Internet Content Selection* (PICS) ist einer der konzeptuellen Vorläufer für das generische Metadatenformat des *Resource Description Framework*, vgl. MILLER, RESNICK & SINGER 1996.

Bei Metadaten kann es sich um beliebige Information handeln, die über den primären Informationsbestand einer Publikation hinausgeht. Ihre Kodierung kann auf unterschiedliche Weise erfolgen: So ist die Unterscheidung zwischen Information, die auf der Ebene einfachen deklarativen Markups (HTML, XML) kodiert ist, und Metadaten nicht immer eindeutig zu treffen.[100] Verwendet man beim Aufbau einer elektronischen Publikation eine XML-DTD, die für jeden logisch unterscheidbaren Teil der Publikation eine eigene Marke bereithält und resultieren nicht alle Marken in einer unterschiedlichen Präsentation in der Bertrachtungssoftware, so liegt bereits Metainformation vor, ohne dass diese bereits ausgewertet werden würde. Unter Metainformation im engeren Sinn ist aber nur diejenige Information zu verstehen, die einem Dokument (Buch, Seite etc.) für einen bestimmten Zweck explizit hinzugefügt wird. Nachfolgend sollen einige Standardisierungsbemühungen für im Kontext dieser Arbeit relevante Metadatenbereiche vorgestellt werden. Bei der Analyse verfügbarer Metadatenstandards sind zwei Kriterien wesentlich:

- Die Syntax des Standards und seine Eignung für die Auszeichnung SGML-/XML-kodierter Dokumente und
- der Inhalts- oder Anwendungsbezug eines Standards.

Dieser Unterscheidung folgend, haben sich sowohl dedizierte Standards für die gebräuchlichsten Anwendungen von Metainformation (*electronic commerce*, bibliographische Daten in digitalen Bibliotheken und Online-Datenbanken etc.) einerseits, generische Metadatenstandards andererseits, herausgebildet.

Aufgrund der Vielfalt denkbarer Anwendungen der Metadatenkodierung ist es nicht sinnvoll, ein konzeptuelles Modell für *alle* Anwendungen von Metadaten zu definieren; der nachfolgend dargestellte Ansatz basiert auf der Annahme, dass ein hinreichend allgemeines syntaktisches Schema für die Kodierung von Metadaten zur Verfügung steht, das zunächst anwendungsunabhängig ist; dedizierte Metadatenmodelle für bestimmte Anwendungen sollen dann mit Hilfe des allgemeinen syntaktischen Schemas operationalisiert werden (z. B. Kodierung bibliographischer Daten mit Hilfe des *Dublin Core Model* und syntaktische Umsetzung in XML mit Hilfe eines RDF-Schemas). Es gilt die gleiche methodische Prämisse wie bei der Definition einer geeigneten DTD für dynamische elektronische Bücher: Soweit wie möglich sind bereits eingeführte Metadatenstandards zu verwenden, um die Wiederverwendbarkeit bzw. Interoperabilität der funktionalen Komponenten eines elektronischen Buchs zu erhöhen. Nur für Bereiche, für die kein geeigneter dedizierter Metadatenstandard vorliegt, können neue Schemata eingeführt werden. Nachfolgend soll das *Resource Description Framework* vorgestellt werden, ein generischer Metadatenstandard, für den dedizierte Anwendungen existieren.

9.1 Resource Description Framework

Das *Resource Description Framework* (vgl. MILLER 1998; BRICKLEY, LAYMAN & GUHA 1998; LASSILA & SWICK 1999) definiert einen Formalismus zur Kodierung von Metadaten, der syntaktisch auf der Basis der Konventionen von XML kodiert werden kann und der die deklarative Auszeichnung von XML-Dokumenten mit Metainformation beliebiger Natur ermöglicht. Grundlage von RDF ist eine einfache Attribut-Wert-Zuordnung, die

[100] Insbesondere konkurrieren die Standardisierungsbemühungen des W3C partiell, wenn z. B. XML Schemata eine semantische Datentypbeschreibung für XML einführt, die auch durch entsprechende RDF-Schemata erfolgen könnte; vgl. zu diesem Problem das sog. *Cambridge Communiqué* (SWICK & THOMPSON 1999 und BERNERS-LEE 1998A).

Aussagen über Ressourcen (i. d. R. XML-Dokumente oder Teile von XML-Dokumenten) kodiert. RDF definiert nur ein Modell für die *Kodierung* solcher Aussagen und macht keine Annahmen über Art und Mächtigkeit der Formalismen, die zur Auswertung der Metainformation benötigt werden (wie z. B. ein prädikatenlogisches Kalkül oder Inferenzregeln); insofern unterscheidet sich RDF von mächtigeren Wissensrepräsentationssprachen wie CYCL (vgl. LENAT et al. 1990) oder KIF (vgl. GENESERETH 1995). RDF baut auf den drei Grundbegriffen Ressource, Eigenschaft und Aussage auf, wie Tabelle 33 zeigt:

Resource	Ein beliebiges, in XML kodiertes Bezugsobjekt (Dokumente, Elemente etc.); die Ressource ist der Bezugspunkt einer RDF-Aussage, d. h. es wird *Information über die Ressource* kodiert.
Property	Eine beliebige Eigenschaft der Ressource, die geeignet ist, diese hinsichtlich einer bestimmten Anwendung (z. B. bibliographische Daten) zu beschreiben.
Statement	Die Zuordnung von Ressource, Eigenschaft und Eigenschaftswert ergibt eine RDF-Aussage (RDF-Statement).

Tabelle 33: Grundbegriffe des Resource Description Framework

Den Zusammenhang der Zuordnung von Metainformation zu Ressourcen kann man in Anlehnung an sprachliche Strukturen wie folgt veranschaulichen:

Subjekt	die Ressource	*http://www.test.org/testseite.html*
Prädikat	die Eigenschaft	Autor
Objekt	der Wert der Eigenschaft	Hans Müller

Tabelle 34: Interpretation von RDF-Tripeln als Subjekt-Prädikat-Objekt-Beziehung

Eine graphische Darstellung zeigt die Zuordnung von Metainformation als gerichteten Graphen, bei dem die Kante vom Subjekt zum Objekt (Wert der Eigenschaft) zeigen und der Name der Eigenschaft (das Prädikat) die Kante benennt:

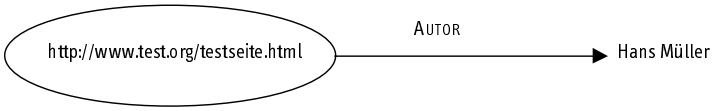

Abbildung 34: Graphische Darstellung einer Ressourcenbeschreibung in RDF

Für Kodierung strukturierter Information, bei der der Wert der Eigenschaft nicht wie im obigen Beispiel ein Literal ist, kann der Eigenschaftswert selbst wieder eine Ressource sein, die durch weitere Prädikate und deren Wertausprägungen beschrieben wird. Die Syntax von RDF verwendet die Konventionen von XML sowie XML-Namensräume (siehe oben Kap. 5.2.3). Zur Kennzeichnung der Metainformation wird eine RDF-Marke eingeführt, die innerhalb eines XML-Dokuments die Kodierung der Metainformation kenntlich macht. Innerhalb der RDF-Marke werden die RDF-Aussagen innerhalb einer description-Marke kodiert (Details und formale EBNF-Notation der Syntax von RDF finden sich in LASSILA & SWICK 1998: Kap. 2):

```
<rdf:RDF>
  <rdf:Description about="http://www.test.org/testseite.html>
    <s:Autor>Hans Müller</s:Autor>
  </rdf:Description>
</rdf:RDF>
```

Codebeispiel 68: Kodierung einer RDF-Relation in XML

Zweierlei ist dabei vorausgesetzt: Es existiert ein Schema, das den RDF-Namensraum definiert, sowie ein RDF-Schema s, das die Eigenschaft *Autor* definiert. Bei Einbettung in ein XML-Dokument kann die zusätzliche explizite Referenz auf den Namensraum wie folgt kodiert werden:

```
<?xml version="1.0" ?>
  <rdf:RDF   xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#">
             xmlns:s=http://www.schema.org/schema/>
  <rdf:Description about="http://www.test.org/testseite.html>
    <s:Autor>Hans M&uuml;ller</s:Autor>
  </rdf:Description>
</rdf:RDF>
```

Codebeispiel 69: Spezifikation eines RDF-Schemas in XML

Durch die Verwendung verschiedener Namensräume lassen sich Eigenschaften, die unterschiedlichen Datenkodierungsschemata oder Ontologien entstammen, nach einem einheitlichen Verfahren als Metadaten einbinden.

Neben der Einbettung der Eigenschaft als Element innerhalb der Beschreibung (*description*) und der Angabe des Objekts bzw. des Attributwerts als Inhalt des Eigenschaftselements ist für RDF eine verkürzte Syntax definiert, bei der die Eigenschaften selbst (im Sinn der XML-Syntax) Attribute des description-Elements sind. Das obige RDF-Statement erhält danach folgende Kodierung:

```
<rdf:RDF>
  <rdf:Description   about="http://www.test.org/testseite.html
                     s:Autor="Hans Müller"/>
  </rdf:Description>
</rdf:RDF>
```

Codebeispiel 70: RDF-Kodierung durch Attributwerte

Inhaltlich sind beide Darstellungen äquivalent; die Frage, ob man die Eigenschaften in XML als Attribute oder Elemente kodiert, kann u. a. von der Art der weiterverarbeitenden Programme abhängen.

Nicht in allen Fällen reicht eine einfache Attribut-Wert-Zuweisung als syntaktisches Modell für die Metadatenkodierung aus. Deshalb sind in der RDF-Syntax Container vorgesehen, die strukturierte Datentypen zulassen:

Bag	Eine ungeordnete Liste von Ressourcen oder Literalen; sie dienen zur Kodierung von Attributwerten als Mengen.
Sequence	Eine geordnete Liste von Literalen oder Ressourcen.
Alternative	Eine Liste alternativer Werte für eine Eigenschaft (z. B. in unterschiedlichen Formaten oder Sprachen).

Tabelle 35: Containerklassen in RDF

Bei allen Containern sind Werteduplikate zulässig, d. h. sie entsprechen nicht dem klassischen Mengenbegriff. Bei der Verwendung von Containern wird in der Beschreibung zusätzlich ein type angegeben, die einzelnen Werteausprägungen durchnumeriert, wie das folgende Beispiel zeigt:

Aussage (Metainformation):
Die Ressource http://www.test.org/testseite.html lässt sich durch die Schlagworte Clustering, Retrieval und Suchmaschine beschreiben.

Umsetzung in RDF:

```
<rdf:RDF>
  <rdf:Description about="http://www.test.org/testseite.html">
    <s:Schlagworte>  <rdf:Bag>    <rdf:li>Clustering</rdf:li>
                                  <rdf:li>Retrieval</rdf:li>
                                  <rdf:li>Document</rdf:li>
                     </rdf:Bag>
    </s:Schlagworte>
  </rdf:Description>
</rdf:RDF>
```

Codebeispiel 71: Verwendung von Containerklassen in RDF

Darstellung der RDF-Kodierung der Aussage als Graph:

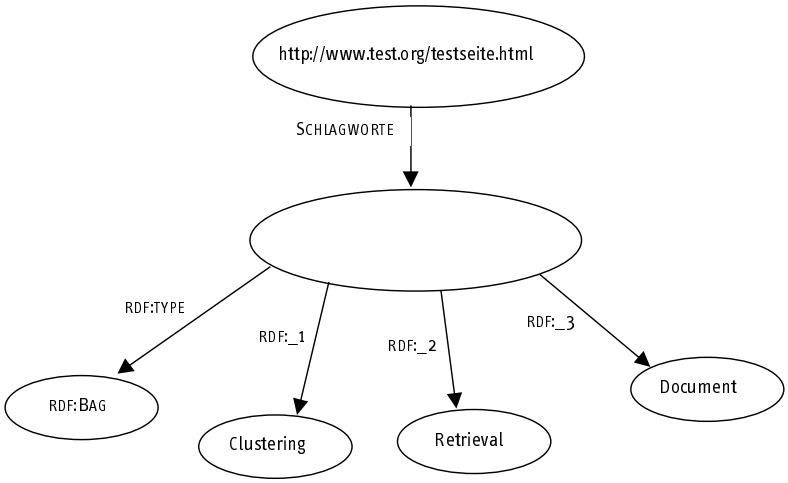

Abbildung 35: Graphische Darstellung von Mehrfachattrbutwerten in Containern

An die Stelle der direkten Wertangabe (die Schlagworte bezeichnenden Zeichenketten selbst) kann ein Verweis auf Ressourcen treten:

```
<!-- [...] -->
<rdf:li ressource="http://www.keywords.org/keywords/clustering.html"/>
<!-- [...] -->
```

Codebeispiel 72: RDF-Verweis auf externe Ressourcenbeschreibung

Neben der Verwendung der Attributcontainer für die Kodierung von Metainformation zu einer Ressource kann mit Hilfe geeigneter Referenzierungen Metainformation bezüglich der Container deklariert werden, wenn z. B. festgehalten werden soll, wer die Schlagworte als Metainformation zur Indexierung vergeben hat („Meta-Meta-Information“):

```
<rdf:Bag ID="Schlagworte">
  <rdf:li>Clustering</rdf:li>
  <rdf:li>Retrieval</rdf:li>
  <rdf:li>Suchmaschine</rdf:li>
</rdf:Bag>
<rdf:Description about="#Schlagworte">
    <s:Indexierer>Hans Müller</s:Indexierer>
</rdf:Description>
```

Codebeispiel 73: Metainformation zu RDF-Containern

Die Verwendung von Containern für Attributwerte ist zu unterscheiden von der mehrfachen Spezifikation desselben Attributs für eine Ressource, wenn z. B. festgehalten werden soll, dass es für eine Ressource eine Liste von Ressourcen mit ähnlichem Inhalt gibt:

> Aussage: *„Zur Datei* http://www.test.org/testseite.html *existieren die inhaltlich ähnlichen Dokumente* http://www.beispiel.net/beispiel.html, http://www.synonym.com/ synonym.html *und* http://www.thesaurus.de/oberbegriff.html.*“*

In diesem Fall liegt es näher, an Stelle eines Containers eine Liste mit RDF-Eigenschaften anzulegen:

```
<rdf:RDF>
  <rdf:Description about="http://www.test.org/testseite.html">
    <s:aehnlich=http://www.beispiel.net/beispiel.html/>
    <s:aehnlich="http://www.synonym.com/synonym.html"/>
    <s:aehnlich="http://www.thesaurus.de/oberbegriff.html"/>
  </rdf:Description>
</rdf:RDF>
```

Codebeispiel 74: Mehrfachverwendung von Eigenschaften in RDF

Die bisherigen Beispiele haben gezeigt, wie man RDF als Modell für die Kodierung von Metainformation für elektronische Ressourcen verwenden kann. Es ist naheliegend, dass nicht nur Metainformation erster Ordnung, die direkt auf eine Ressource bezogen ist, sondern auch Metainformation höherer Ordnung, d. h. Metainformation über Metainformation kodiert werden kann: Dazu ist das bisherige dreigliedrige Modell für die Kodierung von Metainformation um eine weitere Kategorie type zu erweitern, die es erlaubt, ein Modell für RDF-Aussagen aufzubauen (*reification*). Es werden Aussagen über Metaaussagen möglich, wie das folgende Beispiel zeigen soll:

> Aussage: *„Die Beschlagwortung der Ressource* http://www.test.org/testseite.html *mit dem Deskriptor Clustering ist gültig bis zum 1. 5. 2000.“*

Es wird eine Aussage über eine Metainformation (die Zuordnung eines Schlagworts zu einer Ressource) gemacht; betrachtet man diese Aussage selbst als Ressource und gibt ihren Typ als RDF-Statement an, so kann die *reification* wie folgt kodiert werden:

```
<rdf:RDF>
  <rdf:Description>
    <rdf:subject resource="http://www.test.org/testseite.html"/>
    <rdf:predicate>Schlagwort</rdf:predicate>
    <rdf:object>Clustering</rdf:object>
    <rdf:type resource=http://www.w3.org/TR/WD-rdf-syntax#Statement/>
    <s:GültigBis>1.5.2000</s:GültigBis>
  </rdf:Description>
</rdf:RDF>
```

Codebeispiel 75: Reifizierung von RDF-Aussagen

Die Definition der Syntax von RDF ist nur ein erster Schritt zur interoperablen Verwendung von Metainformation. Um Metainformation durch unterschiedliche Softwarewerkzeuge analysieren und auswerten zu können, muss die Kodierung der Metainformation selbst in einheitlicher Weise erfolgen, d. h. soweit wie möglich müssen einheitliche Kodierungsschemata zur Anwendung kommen.

Auf der Basis von RDF und der Syntax von XML (bzw. der XML-Namespaces) liegt parallel zur Standardisierung der Syntax von RDF ein W3C-Standard zur Kodierung von Metadatenschemata vor (BRICKLEY, GUHA & LAYMAN 1998). Mit seiner Hilfe kann die

Semantik der in den RDF-Aussagen verwendeten deskriptiven Elemente kodiert werden, d. h. es wird Information zur Interpretation der Metainformation kodiert. Der Vorschlag stellt keine verbindliche Ontologie für die Spezifikation von Metadaten auf, sondern beschreibt einen Mechanismus für die Definition solcher Schemata für Metainformation. Dies setzt Vorarbeiten aus der Metadatenforschung bzw. der Digital Library-Forschung fort, wo seit langem an der Ausarbeitung von Standards für Metadaten gearbeitet wird.

Zur Kodierung der Datenschemata für RDF-Aussagen wird auf die RDF-Syntax als Modellierung des Zusammenhangs zwischen Ressourcen, Prädikaten und Prädikatausprägungen zurückgegriffen, d. h. ein RDF-Schema ist eine Sammlung von RDF-Ressourcen, die ein RDF-Vokabular definieren. Um dies zu ermöglichen, wird in Analogie zu Konzepten der objektorientierten Modellierung ein Typsystem für die Aussagen über ein RDF-Vokabular definiert. Das RDF-Schemamodell enthält die folgenden Basisklassen, die die oben eingeführte formale Modellierung von RDF direkt widerspiegeln:

rdfs:Ressource	Alle Aussagen in RDF beziehen sich auf Ressourcen; dies ist die Wurzelklasse des Typsystems.
rdf:Property	Repräsentiert die Menge der Ressourcen, die Eigenschaften sind.
rdfs:Class	Allgemeines Typ- oder Kategorienkonzept ähnlich dem Klassenkonzept der objektorientierten Programmierung.

Tabelle 36: Basisklassen der RDF-Schema-Syntax

Zusätzlich zu den Basisklassen können in einem RDF-Schema folgende Basiseigenschaften (der Basisklassen) angegeben werden:

rdf:type	Gibt an, dass eine Ressource zu einer Klasse gehört.
rdfs:subClassOf	Stellt eine transitive Ober-/Unterberiffsbeziehung zwischen Klassen her.
rdfs:subPropertyOf	Ermöglicht die Spezialisierung von Eigenschaften.

Tabelle 37: Eigenschaften der RDF-Basisklassen

Zusätzlich zu diesem einfachen Klassensystem werden für RDF-Schemata eine Menge von Constraints eingeführt, die die inhaltliche Präzisierung der Metadatenzuordnung erlauben:

rdfs:ConstraintResource	Eine Unterklasse von rdfs:Resource, die Ressourcen kodiert, die für die Spezifikation von Constraints verwendet werden.
rdfs:ConstraintProperty	Eine Unterklasse von rdfs:Property, die Eigenschaften kodiert, die Constraints darstellen.
rdfs:Range	Eine Instanz der Klasse rdfs:ConstraintProperty, die Beschränkungen für Attributwerte kodiert. Als Werte kann eine (noch zu definierende) kanonische Menge von Datentypen verwendet werden.
rdfs:Domain	Eine Unterklasse von rdfs:ConstraintProperty, die Klassen angibt, für die eine Eigenschaft kodiert werden kann. Eine Eigenschaft kann beliebig vielen (auch keinen) Klassen zugeordnet werden.

Tabelle 38: Constraintklassen in RDF-Schemata

Zur vereinfachten Dokumentation und Lesbarkeit eines Schemas existieren zusätzlich die Eigenschaften rdfs:comment und rdfs:label, die einen natürlichsprachlichen Kommentar zu einer Ressource bzw. eine lesbare Umschreibung eines Ressourcennamens kodieren.

Das nachfolgende Anwendungsbeispiel, das in modifizierter Form aus BRICKLEY, GUHA & LAYMAN: Kap. 7 übernommen ist, zeigt die Definition eines RDF-Schemas für die Spezifikation von Suchanfragen: Es werden Klassen für Suchanfragen, Suchergebnisse und Suchdienste definiert, die jeweils über bestimmte Eigenschaften verfügen. Diese RDF-Schema-Elemente können in unterschiedlichen Anwendungen (XML-Dokumenten

mit entsprechender Softwareinfrastruktur, z. B. Anbindung an Suchdienste als externe Informationsdienste) verwendet werden und es stünde für unterschiedliche Suchdienste ein einheitliches Beschreibungsvokabular zur Verfügung.

```
<rdf:RDF   xmlns:rdf="http://www.w3.org/TR/WD-rdf-syntax#"
           xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#">
  <rdfs:Class rdf:ID="SuchAnfrage">
    <rdfs:subClassOf     rdf:resource="http://www.w3.org/TR/WD-rdf-schema#Resource"/>
  </rdfs:Class>
  <rdfs:Class rdf:ID="Suchergebnis">
    <rdfs:subClassOf     rdf:resource="http://www.w3.org/TR/WD-rdf-schema#Resource"/>
  </rdfs:Class>
  <rdfs:Class rdf:ID="Suchdienst">
    <rdfs:subClassOf     rdf:resource="http://www.classtypes.org/useful_classes#Service"/>
  </rdfs:Class>
  <rdf:Property ID="AnfrageString">
    <rdf:domain rdf:resource="#SuchAnfrage"/>
    <rdf:range           rdf:resource="http://www.w3.org/TR/WD-rdf-schema#Literal"/>
  </rdf:Property>
  <rdf:Property ID="Suchdienst">
    <rdf:domain rdf:resource="#SuchAnfrage"/>
    <rdfs:range rdf:resource="#SuchDienst"/>
  </rdf:Property>
  <rdf:Property ID="Ergebnis">
    <rdf:domain rdf:resource="#SuchAnfrage"/>
    <rdfs:range rdf:resource="#SuchErgebnis"/>
  </rdf:Property>
  <rdf:Property ID="ErgebnisSeite">
    <rdfs:range rdf:resource="http://www.datatypes.org/useful_types#WebPage"/>
  </rdf:Property>
  <rdf:Property ID="ErgebnisÜberschrift">
    <rdf:domain rdf:resource="#Suchergebnis"/>
    <rdfs:range rdf:resource="http://www.w3.org/TR/WD-rdf-schema#Literal"/>
  </rdf:Property>
  <rdf:Property ID="ErgebnisBewertung">
    <rdf:domain rdf:resource="#Suchergebnis"/>
    <rdfs:range rdf:resource="http://www.datatypes.org/useful_types#FloatZeroToOne"/>
</rdf:Property>
</rdf:RDF>
```

Codebeispiel 76: Anwendungsbeispiel für RDF-Schemata

In diesem Beispiel existiert eine RDF-Klasse SuchAnfrage, die direkte Subklasse des Typs Ressource ist und über die Eigenschaften Anfragestring, SuchDienst und Ergebnis verfügt. Die Eigenschaft Anfragestring hat als Wert ein RDF-Literal.

Das Beispiel zeigt den Wert einer einheitlichen Schemadefinition: Erst wenn bei der Metadatenkodierung auf ein einheitliches Schema zurückgegriffen wird, werden XML-Dokumente und ihre Softwareinfrastruktur interoperabel. Wenn dagegen die Schnittstellen zu Rechercheanwendungen in unterschiedlichen Schemata („Dialekten“) kodiert sind, ist ein Austausch oder die dynamische Anbindung zusätzlicher Dienste kaum möglich.

9.2 Anwendungsspezifische Metadatenschemata

Neben der Spezifikation der Syntax von Metadatenstandards ist die Betrachtung ihres konkreten *inhaltlichen* Bezugs eine wesentliche Voraussetzung für den Einsatz und die Operationalisierung von Metadatenstandards. Zu den Anwendungsgebieten von Metainformation zählen

- die Speicherung bibliographischer Information zur Erschließung informationeller Einheiten (s. u.),[101]
- die Beschreibung und Indexierung multimedialer Binärformate, die einer direkten Erschließung nicht oder nur bedingt (Mustererkennung) zugänglich sind (Bitmaps, Sprachdaten, Videodaten etc., vgl. WILKIE & MILLS 1997),
- die Beschreibung technischer Systemparameter als Metainformation (z. B. in Datenbanksystemen oder Multimediainformationssystemen, *Quality-of-Service*-Metadaten, vgl. KERHERVÉ et al. 1996),
- die Klassifikation und Annotation von Daten, um z. B. bestimmte Verwendungszwekke oder Nutzergruppen anzugeben oder auszuschließen (z. B. *content rating* nach dem PICS-Standard des W3C (vgl. MILLER, RESNICK & SINGER 1996) oder
- die Kodierung von Metainformation zur Eingliederung elektronischen Publikationsmaterials in einen Lehr-/Lernkontext (Telelearning, Teleteaching).[102]

Nachfolgend werden Metadatenstandards für bibliographische Daten bzw. für die Aufbereitung von Lehr- und Lernmaterialien vorgestellt, die unmittelbare Relevanz für dynamische elektronische Bücher haben.

9.2.1 Bibliographische Metadaten

Zu den wissenschaftsgeschichtlich ältesten Formen von Metainformation gehört die bibliographische Erschießung informationeller Einheiten, das so lange existiert, wie es Zusammenstellungen einer Vielzahl von Publikationswerken in Bibliotheken gibt. In der Dokumentationswissenschaft hat die Beschäftigung mit Indexierung, Klassifikation, dem Aufbau von Ontologien, der intellektuellen und automatischen Erschließung von Information etc. eine lange Tradition. Seit langem existieren Standardisierungen bibliographischer Datensätze (als Metadaten zu Publikationen) für das elektronische Medium, z. B. der DigiMarc-Standard in den USA. Seit einigen Jahren hat sich in der Forschung der nach einem Workshop in Dublin benannte *Dublin Core*-Standard entwickelt, der eine allgemeine Basis für die bibliographische Beschreibung darstellt (vgl. REW 1998). Er sieht folgende Elemente vor (vgl. DEMPSEY & WEIBEL 1996: Kap. 2, DUBLIN CORE METADATA INITIATIVE 1997):

Datenfeld	*Beschreibung*
Subject	Thema, das von einer Publikation behandelt wird.
Title	Titel (Name) des Objekts (der informationellen Einheit).
Author	Namen der Autoren.
Publisher	Verleger, d. h. Name der Institution, die das Werk verfügbar macht.
Other Agents	Weitere Personen, die erheblich zum intellektuellen Gehalt des Werkes Beitragen haben (Lektoren, Illustratoren, Übersetzer etc.).

[101] Vgl. allgemein GAUS 1995, konkrete Anwendungsszenarien finden sich u. a. bei SMITH, GEFFNER & GOTTSEGEN 1996; CONSORTI, MERIALDO & SINDONI 1996.

[102] Ein Überblick zu elektronisch verfügbaren Metadatenschemata sowie zu Kodierungswerkzeugen befindet sich unter *http://www.metadata.net*.

Datenfeld	*Beschreibung*
Date	Publikationsdatum.
Object Type	Art/Genre des Werkes (Lexikon, Lehrbuch, Drama, Roman etc.).
Form	Art der physikalischen Manifestation.
Identifier	Eindeutiger Bezeichner für das Werk (z. B. ISBN, *Digital Object Identifier, DOI*).
Relation	Beziehungen zu anderen Werken, z. B. die Zugehörigkeitsrelation zwischen einem Bild und einem Buch.
Source	Objekte/Werke, von denen das Werk abgeleitet ist.
Language	Sprache, in der das Werk verfasst ist.
Coverage	Räumliche und/oder zeitliche Erstreckung des Werks.
Rights Management	Information zu Urheberrechten, Verweis auf einen Copyrightvermerk etc.

Tabelle 39: Metadatenfelder im Dublin Core Set

Der Dublin Core wurde bereits in zahlreichen Pilotprojekten in elektronischen Bibliotheken zur Metadatenkodierung verwendet; eine Kodierung mit Hilfe des *Resource Description Framework* ist möglich; für sie stehen sowohl ein RDF-Schema als auch entsprechende Editoren zur Verfügung. [103]

Eine Bewertung dieses Vorschlags ergibt, dass der eingeschränkte Umfang der kodierten Information die Durchsetzung standardisierter APIs zur Nutzung in digitalen Bibliotheken vereinfacht. Als wohldefinierter Standard erleichtert er im Gegensatz zur Speicherung der genannten Information *als Teil des eigentlichen Inhalts einer Publikation* die Nutzung. Die Beschränkung auf ein Basisset von Attributen kann aber nur ein erster Schritt sein, da gerade in dynamischen Publikationen die Inhaltsbestandsänderung (neue Zusammenstellung aus heterogenen Quellen) bzw. die zeitliche Veränderung eine wesentliche Rolle spielt. Im Rahmen des sog. *Warwick Framework* wird der *Dublin Core* erweitert und flexibilisiert zu einem Modell, das unterschiedliche Metadatenbeschreibungen vereinigen kann (vgl. DEMPSEY & WEIBEL 1996: Kap. 3; DANIEL & LAGOZE 1997). Einen anderen Weg geht die *Association of American Publishers*, die Richtlinien für den Online-Informationsaustausch mit besonderer Berücksichtigung des Buchhandels im Internet einerseits bzw. elektronischer Publikationen und Bücher andererseits herausgegeben hat (vgl. AAP 2000). Unter Rückgriff auf verschiedene Kodierungsstandards der ISO (z. B. für Sprachbezeichnungen, Datumsangaben oder Ländernamen) werden 148 Datenfelder zur Beschreibung unterschiedlicher Publikationstypen eingeführt. Dieser Kodierungsvorschlag ist formatunabhängig und soll die Kodierung von Metadaten sowohl als einfachen Text als auch durch XML-Strukturen ermöglichen. Eine Querverbindung zu *Dublin Core* in inhaltlicher, zu RDF in syntaktischer Hinsicht fehlt aber bisher.

Zusammenfassend kann man folgendes festhalten: Bibliographische Metadaten dienen der Beschreibung und Erschließung elektronischer Publikationen und sind ein Ausgangspunkt für die weitergehende Kodierung von Metainformation. Noch nicht abschließend gelöst erschienen Fragen der Granularität der Zuordnung von Metadatensätzen; in der Regel erfolgt bei den aus der Literatur bekannten Beispielen die Zuordnung auf der Speicherungsebene, d. h. die Zuordnung von Metainformation zu einer Datei, die die entsprechende Ressource enthält. Im Rahmen des Modells für dynamische elektronische Bücher soll die Metadatenkodierung aber logisch an die verschiedenen Ebenen der strukturellen und inhaltlichen Auszeichnung gebunden werden, statt an eine arbiträre Speicherungs-

[103] Vgl. auch Beispiele in MILLER 1998 und LASSILA & SWICK 1998: Kap 7.4. Die RDF-Schemadefinition für den Dublin Core findet sich bei BRICKLEY, GUHA & LAYMAN 1998: Kap. 7, eine deutsche Beschreibung der Basiselemente des Dublin Core ist unter *http://www.mpib-berlin.mpg.de/DOK/metatagd.htm* verfügbar.

form (Beschreibung des Gesamtwerks, einer einzelnen Komponente etc.). So wird eine präzise Beschreibung durch Metadaten möglich.

9.2.2 Metadaten für Lehr- und Lernsysteme

Ein zweiter Bereich der Metadatenkodierung, für den verschiedene Standardisierungsvorschläge existieren, sind Lehr- und Lernmedien. Da dieses Anwendungsfeld elektronischer Publikation hier relevant ist, ist es sinnvoll, relevante Standardisierungsbemühungen aus diesem Bereich vorzustellen und zu diskutieren. Zwar gibt es derzeit noch keinen abschließend definierten und akzeptierten Standard für die Aufbereitung von Lehr- und Lernmaterialien, es sind aber eine Reihe von Projekten und Standardisierungsbemühungen erkennbar, die sich der Kodierung von Lehr- und Lernmaterialien und insbesondere ihrer Beschreibung durch Metadaten widmen. Zu ihnen gehören

- die Definition eines *Learning Object Model*, die von der *IEEE P1484.12 Learning Objects Metadata Working Group* im Rahmen des *IEEE Learning Technology Standards Committee* (LTSC) betrieben wird,[104]
- die amerikanische IMS-Initiative (*Instructional Management Systems*), die ebenfalls einen Standard für die Metadatenbeschreibung von Lehr- und Lernmaterialien entworfen hat (vgl. RESMER 1998 und *http://www.ims-project.org*) und schließlich
- das europäische ARIADNE-Projekt (*Alliance of Remote Instructional Authoring and Distribution Networks for Europe*, vgl. *http://ariadne.unil.ch*).

Diese Standardisierungsentwürfe haben einen inhaltlichen Bezugspunkt, der über das Konzept dynamischer elektronischer Bücher hinausreicht. Es ist zu klären, ob und in welchem Umfang die Kennzeichnung von Metadaten für Lernmaterialien im Kontext eines elektronischen Buchs sinnvoll sein kann. Diese Klärung erfolgt anhand des folgenden Katalogs von Zielstellungen, der im Rahmen der *IEEE P1484.12 Learning Objects Metadata Working Group* für die Entwicklung des Learning Object Model definiert wurde. Den einzelnen Anforderungen ist eine Bewertung ihrer Relevanz hinsichtlich des Modells dynamischer elektronischer Bücher gegenübergestellt.

Zielstellungen des Learning Objects Model	*Einschätzung der Relevanz für dynamische elektronische Bücher*
a. To enable learners or instructors to search, evaluate, acquire, and utilize Learning Objects.	Relevant hinsichtlich der angebotenen Erschließungsfunktionen des elektronischen Buchs, soweit diese durch Metadaten optimiert werden können.
b. To enable the sharing and exchange of Learning Objects across any technology supported learning systems.	Nur mittelbar relevant: Die Wieder- und Weiterverwendung von Lernmaterialien eines elektronischen Buchs wird nur am Rande betrachtet; wichtiger erschiene die Erschließung externer Ressourcen durch Recherchemöglichkeiten auf der Basis einheitlich kodierter Metadaten, die aber bisher noch nicht praktisch durchgeführt werden kann.
c. To enable the development of learning objects in units that can be combined and decomposed in meaningful ways.	Dieses Ziel liegt der Verwendung deklarativer Informationsauszeichnung zugrunde und wird durch die einheitliche Metadatenkodierung verstärkt.
d. To enable computer agents to automatically and dynamically compose personalized lessons for an individual learner.	Irrelevant, da elektronische Bücher als Ressource zur Anwendung in Lehr- und Lernsystemen, nicht als solche Systeme an sich betrachtet werden.

[104] Vgl. *http://ltsc.ieee.org*; dabei handelt es sich um ein umfangreiches Vorhaben, das zahlreiche Arbeitsgruppen umfasst, vgl. dazu näher unten Kap. 9.2.2.2.

Zielstellungen des Learning Objects Model	*Einschätzung der Relevanz für dynamische elektronische Bücher*
e. To compliment the direct work on standards that are focused on enabling multiple Learning Objects to work together within a open distributed learning environment.	Insoweit relevant, als die Einordnung des elektronischen Buchs in einen konkreten Nutzungskontext auch der Verwendung eines *open distributed learning environment* entsprechen kann („Kommunikationsinfrastruktur"); das Konzept der Interoperabilität von Komponenten und Diensten entspricht ebenfalls dieser Leitlinie.
f. To enable, where desired, the documentation and recognition of the completion of existing or new learning & performance objectives associated with Learning Objects.	Wird als Gegenstand eines Lehr- und Lernsystems, nicht eines dynamischen elektronischen Buchs hier nicht betrachtet.
g. To enable a strong and growing economy for Learning Objects that supports and sustains all forms of distribution; non-profit, not-for-profit and for profit.	Nur im Rahmen des Ausblicks relevant (Integration in Digitale Bibliotheken und Übernahme deren Infrastrukturdienstleistungen wie Abrechnung, vgl. Kap. 15.2).
h. To enable education, training and learning organizations, both government, public and private, to express educational content and performance standards in a standardized format that is independent of the content itself.	Globales Standardisierungsziel; gilt nicht nur für Metadaten, sondern auch für die zugrundeliegende einheitliche Verwendung von Kodierungsstandards (s. Bemerkung zu Leitlinie c)).
i. To provide researchers with standards that support the collection and sharing of comparable data concerning the applicability and effectiveness of Learning Objects.	Wird nicht betrachtet (s. Bemerkung zu Leitlinie f)).
j. To define a standard that is simple yet extensible to multiple domains and jurisdictions so as to be most easily and broadly adopted and applied.	Als Ziel der Verallgemeinerbarkeit besteht Übereinstimmung mit Anliegen, dynamische elektronische Bücher auf der Basis deklarativer Standards zu modellieren.
k. To support necessary security and authentication for the distribution and use of Learning Objects.	Durch Authentifizierung von Benutzern berücksichtigt. Solche Funktionen sollen dem elektronischen Buch aber prinzipiell von der Infrastruktur einer digitalen Bibliothek bereitgestellt werden.

Tabelle 40: Ziele des Learning Object Metadata Model und ihre Relevanz für elektronische Bücher

Die oben aufgeführten Leitlinien weisen eine Vielzahl von Berührungspunkten mit dem Konzept dynamischer elektronischer Bücher auf und können in Verbindung mit einer standardisierten Kodierungssyntax wie dem *resource description framework* unmittelbar für die Metadatenkodierung des elektronischen Buchs Verwendung finden. Nachfolgend sind werden die wichtigsten Beschreibungselemente aus IMS und IEEE LOM vorgestellt.

9.2.2.1 IMS Metadata Proposal

Auf der Basis des Dublin Core Set ist im Rahmen des *IMS Metadata Project* ein Standard für die Beschreibung von Lehr- und Lernmaterialien definiert worden. Eine Beschreibung durch RDF bzw. die Implementierung durch Middleware-Schnittstellen wird derzeit entwickelt. Das Projekt baut auf Vorarbeiten des *IEEE Learning Technology Standards Committee* auf und wird in Weiterführung einer US-amerikanischen Forschungsinitiative für die Lernmedienstandardisierung von zahlreichen Universitäten und Industriepartnern unterstützt. IMS umfasst vier Mengen von Metainformation:

- *IMS Base Set:* Minimalbeschreibung einer Lernressource.
- *IMS Item Set:* Beschreibung der Medienelemente (Bilder, Texte, Animationen etc.).

- *IMS Tool Set:* Beschreibt Lernressourcen mit eigenständiger Funktionalität, d. h. im wesentlichen Softwaremodule generischer oder spezifischer Art, die der Benutzer bei der Aufgabenlösung zur Verfügung hat (oder benötigt).
- *IMS Module Set:* Beschreibt auf einer höheren Ebene Lernmodule, die aus mehreren Bestandteilen bestehen (d. h. die einzelnen Items innerhalb eines Moduls können mit dem *IMS Item Set* beschrieben werden).

Das IMS Base Set enthält folgende Datenfelder:

Datenfeld	*Beschreibung*
Description	analog Dublin Core
Content Version	Versionsinformation
Format	analog Dublin Core
Publisher	analog Dublin Core
Resource Identifier	analog Dublin Core
Title	analog Dublin Core
Subject [Description, Keywords]	analog Dublin Core
Meta-Meta-Data	Beschreibung der Metadatenkodierung mit den Subfeldern *Metadatenschema, Autor, Erstellungsdatum, letzte Änderung, Containertyp, Validator*
Datum	analog Dublin Core, mit folgenden Subfeldern: - Gültigkeit bis - Verfügbarkeit ab - Publikationsdatum

Tabelle 41: Datenfelder im IMS CORE SET

Die Beschreibung der einzelnen Elemente eines Moduls durch das *IMS Item Set* führt zusätzlich folgende Felder ein:

Datenfeld	*Inhalt*
Author	analog Dublin Core
Price Code	Kostenspezifikation
Rights Management	Urheberrechtsangaben, kodiert durch *Agent* und *Use Rights*

Tabelle 42: Datenfelder im IMS ITEM SET

Die Beschreibung verfügbarer bzw. benötigter Softwaremodule in der Lernumgebung erfolgt im IMS-Projekt durch folgende Datenfelder

Datenfeld	*Beschreibung/Attribute*
Language	analog Dublin Core
User Support	Unterstützung des Benutzers
Learning Level	Kodierung des Lernerniveaus
Prerequisites	benötigte Vorkenntnisse
Price Code	Kostenspezifikation
Rights Management	Urheberrechtsangaben, kodiert durch *Agent, Use Rights*
Platform	Angaben zur benötigten Hardware- und Softwareinfrastruktur (Required Software, Required Hardware)

Tabelle 43: Datenfelder im IMS TOOL SET

Der zentrale Datensatz im Rahmen der IMS Metadaten beschreibt die einzelnen Module, wobei hier die eigentlich pädagogische relevante Information kodiert wird:

Datenfeld	*Beschreibung/Untereinheiten*
Author/Creator	analog Dublin Core
Language	analog Dublin Core
Resource Type	analog Dublin Core
Educational Objectives	Lernziele
Granularity	Granularität/Feinkörnigkeit
Learning Level	Lernerniveau
Organization	zuständige Organisation
Pedagogy	pädagogisches Grundkonzept
Prerequisites	benötigte Vorkenntnisse
Presentation	Art der Präsentation
User Support	Unterstützung des Benutzers
Use Time	geplante Benutzungsdauer
Price Code	Kostenspezifikation
Rights Management	Urheberrechtsangaben, kodiert durch *Agent, Use Rights*
Platform	Angaben zur benötigten Hardware- und Softwareinfrastruktur (*Required Software, Required Hardware*)

Tabelle 44: Datenfelder im IMS Module Set

Der zentrale Begriff des *Moduls* ist weit gefasst: Von einem umfangreichen Lernmodul, das z. B. parallel zu einer Lehrveranstaltung angeboten wird und umfangreichen, in sich strukturiertes Lernmaterial enthält, bis zur Kodierung von Einzelaktivitäten (Lösung einer einzelnen Aufgabe) können die unterschiedlichsten Aktivitäten unter diesem Begriff zusammengefasst werden.

9.2.2.2 IEEE Learning Objects Model Metadata

Einen noch umfassenderen Ansatz zur Kodierung von Metainformation für Lehr- und Lernmedien bzw. Lernsysteme stellt das unter der Schirmherrschaft der IEEE entwickelte *Learning Object Model* dar, das folgende Bereiche abdeckt:

- Allgemeines, Architektur von Lehr- und Lernsystemen
- Lernerbezogene Standardisierung (*Learner Model, Task Model, Student Identifiers, User Interfaces, Quality System for Technology-Based Life-Long Learning*)
- Inhaltsbezogene Standardisierung (*CBT Interchange Language, Course Sequencing, Content Packaging*)
- Daten und Metadaten (*Learning Objects Metadata, Localisation, Semantics and Exchange Bindings, Data Interchange Protocols, HTTP Bindings*)
- Managementsysteme und Anwendungen (*Computer Managed Instruction, Platform and Media Profiles, Tool/Agent Communication, Enterprise Interfaces*)

Mit wenigen Ausnahmen liegen bisher noch keine abschließenden Standards vor, für einige Bereiche existieren nicht einmal Entwürfe. Dies ist insofern bedauerlich, als nicht nur bei der Kodierung der Daten und Metadaten des dynamischen elektronischen Buchs, sondern auch bei dessen Implementierung solche Standards berücksichtigt werden können (einheitliche Benutzerverwaltung, einheitliches API für die Client-Server-Kommunikation, die Integration von Diensten etc.). Die Bereiche *Architekturmodell* und *Metadaten* sind aber bereits beschrieben. Der Entwurf der *Metadatenbeschreibung* liegt Ende 1999 in der Fassung 3.6 vor. Er enthält ein allgemeines Datenmodell für die Erfassung von Metadaten, das auf den Ergebnissen von IMS und Ariadne aufbaut und das Problem der Kodierung von Metadaten in Lehr- und Lernsystemen wie folgt beschreibt:

> This standard will specify the syntax and semantics of Learning Object metadata, defined as the attributes required to fully/adequately describe a Learning Object. The latter, i.e. a Learning Object is defined here as any entity, digital or non-digital, which can be used, reused or referenced during technology-supported learning. [...]
>
> Examples of Learning Objects include multimedia content, instructional content, instructional software and software tools, referenced during technology supported learning. In a wider sense, learning objects could even include learning objectives, persons, organizations, or events.
>
> The Learning Object metadata standards will focus on the minimal set of properties needed to allow these Learning Objects to be managed, located, and evaluated. The standards will accommodate the ability for locally extending the basic properties. [IEEE LTSC 1999: Kap. 2].

Das in LOM definierte *base model* der Metadatenbeschreibung geht von den folgenden neun Basiskategorien der Metadatenbeschreibung aus:

1. General	Fasst alle Merkmale ohne Kontextbezug sowie eine allgemeine Beschreibung der Ressource zusammen.
2. Lifecycle	Bündelt die Merkmale, die sich auf den Lebenszyklus einer Ressource beziehen.
3. Meta-metadata	Beschreibt die Metadaten selbst.
4. Technical	Erfasst technische Merkmale einer Ressource.
5. Educational	Beschreibt die pädagogischen Aspekte.
6. Rights groups	Fasst die Merkmale zusammen, die sich auf die Nutzungsrechte der Ressource beziehen.
7. Relation groups	Stellt Beziehungen zu anderen Ressourcen her.
8. Annotation	Erlaubt die Kodierung von Anmerkungen hinsichtlich des Einsatzes der Ressource.
9. Classification	Behandelt die Zuordnung der Ressource zu einem Klassifikationssystem.

Tabelle 45: Beschreibungskategorien des Learning Object Model

Diese neun Basiskategorien werden im einzelnen nach einem einheitlichen Datenmodell beschrieben; die Einträge des Basisschemas können jeweils durch folgende Elemente beschrieben werden:

- Den *Namen* des Beschreibungselements,
- seine *Definition* bzw. Erklärung,
- die Festlegung der *Kardinalität* eines Beschreibungselements,
- die Definition eines *Wertebereichs* für das Element,
- zusätzliche *Anmerkungen* oder Hinweise sowie
- *Beispiele* für seine Verwendung.

Für die oben aufgeführten Basiskategorien sind in IEEE LTSC 1999: Kap. 5 die einzelnen Bestandteile angegeben; sie wiederzugeben würde zu weit führen, deshalb soll exemplarisch der Bereich 5 – *educational metadata* – dargestellt werden. Die nachfolgende Tabelle ist ein verkürzter Ausschnitt der Definition des IEEE LOM Basisschemas.

Nr. / Name	*Erläuterung*	*Kardinalität, Attributwerte*	*Bemerkung*	*Beispiel*
5. Educational	Lernaspekte der Ressource	1	–	–
5.1. Interactivity Type	Art der Interaktivität der Ressource.	Einzelwert, festes Vokabular: {Active, Expositive, Mixed, Undefined}	In einer expositorischen Ressource verläuft der Informationsfluss von der Ressource zum Lerner (*learning-by-reading*). In einer aktiven Ressource fließt Information von Lerner zur Ressource (*learning-by-doing*).	Expositorische Dokumente sind Aufsätze, Videoclips, graphisches Material und Hypertexte; aktive Dokumente beinhalten Simulationen, Fragebögen und Übungen.
5.2. Learning Resource Type	Spezieller Typ der Ressource (Angabe des wichtigsten Typs zuerst)	Geordnete Liste; offenes Vokabular (u. a. Übung, Simulation, Fragebogen, Diagramm, Abbildung, Graph, Index, Tabelle, Folie, narrativer Text, Test, Experiment, Problembeschreibung, Selbsttest)	Entspricht dem Dublin Core-Element Resource Type mit Anpassung des Vokabulars für LOM.	–
5.3. Interactivity Level	Einordnung des Grades der Interaktivität.	Einzelwert, {0, 1, 2, 3, 4}	0 entspricht *sehr niedrig*, 4 *sehr hoch*.	–
5.4. Semantic Density	Subjekte Einschätzung der Nützlichkeit im Verhältnis zu Dauer, Größe etc.	Einzelwert, {0, 1, 2, 3, 4}	0 entspricht *sehr niedrig*, 4 *sehr hoch*.	–
5.5. Intended end user role	Typ des Benutzers der Ressource	Geordnete Liste, festes Vokabular: {Teacher, Author, Learner, Manager}	[Def. der Rollen]	
5.6. Learning Context	Typischer Lerner	Geordnete Liste, offenes Vokabular; gibt das Niveau an (z. B. Primarstufe, Hochschulausbildung, technische Ausbildung, Berufliche Ausbildung)	–	–
5.7. Typical Age Range	Alter typischer Lerner	Geordnete Liste (vier Einträge)	–	–
5.8. Difficulty	Schwierigkeitsgrad für den typischen Lerner	Einzelwert, {0, 1, 2, 3, 4}	0 bedeutet sehr leicht, 4 sehr schwer.	–
5.9. Typical Learning Time	Zeitraum, der für die Arbeit mit der Ressource benötigt wird	Einzelwert	–	PT1H30M, PT1M45S
5.10. Description	Kommentar zur Verwendung der Ressource	Einzelwert	–	Z. B. Hinweise für den Lehrer, Dozenten
5.11. Language	Sprache des typischen Lerners	Ungeordnete Liste (acht Einträge), LanguageID, d. h. Sprachcode und Subcode nach ISO639/ISO3166	Analog zum xml:lang-Attribut.	Z. B. en, en-GB, de, fr-CA, it

Tabelle 46: Pädagogische Beschreibungskategorien in LOM

Bei der Analyse dieses Ausschnitts aus LOM in Bezug auf die Anwendbarkeit auf dynamische elektronische Bücher lassen sich folgende Schwierigkeiten erkennen:

- Der Begriff *learning object* ist allgemein definiert und auf die verschiedenen Bestandteile dynamischer elektronischer Bücher anwendbar, d. h. sowohl auf den Basis-

text des dynamischen elektronischen Buchs („*learning-by-reading*“) als auch auf die im Buch verfügbaren interaktiven Komponenten („*learning-by-doing*“). Das dynamische elektronische Buch stellt eine Mischung von *learning objects* unterschiedlichen Typs dar.

- Beim Einsatz in einem elektronischen Buch ist zu klären, auf welcher Granularitätsstufe die Beschreibung durch Metadaten sinnvoll sein kann. Dabei ist nach dem Typ des beschriebenen Objekts zu unterscheiden (Basistext; Komponente bzw. Komponententyp). Zusätzlich ist zu untersuchen, wie eng die Metadatenbeschreibung an die logische und inhaltliche Struktur des elektronischen Buchs gekoppelt sein soll.
- Wie die Beispiele in der voranstehenden Tabelle zeigen, existiert auf der Ebene der Standardisierung kein klares Typenmodell für *learning objects* und als Konsequenz kein *verbindliches Beschreibungsvokabular*. Im Rahmen des Referenzprojekts hat man sich auf eine ähnlich phänomenologische Weise beholfen, wie dies bei der offenen Typenliste von LOM zu erkennen ist. Dies schränkt die generelle Verwendung bzw. die automatische Auswertung der Metadaten ein: Ist kein generisches Vokabular z. B. für die Beschreibung einer Simulation verfügbar, so ist die Nutzung der Metadaten entweder nur durch den Menschen oder durch ein an den Einzelfall gebundenes Auswertungssystem möglich.
- Das Modell lässt sich unmittelbar in ein Kodierungsschema wie das *resource description framework* übersetzen; seine Operationalisierbarkeit ist so gewährleistet. Eine genormte XML/RDF-DTD für LOM liegt aber bisher nicht vor; insofern müsste sie für das dynamische elektronische Buch aus dem Standard selbst heraus entwickelt werden.

Die voranstehenden Kapitel des zweiten Teils dieser Arbeit haben mit Bezug zu elektronischen Büchern aufgezeigt, welche Standards für die deklarative Auszeichnung von Information unterschiedlichen Typs zur Verfügung stehen. Das Ziel war dabei, deklaratives Markup für möglichst viele Bereiche der Kodierung und Entwicklung multimedialer dynamischer Bücher zu verwenden; auch wenn noch nicht für alle diskutierten Standards eine unmittelbar verwendbare Softwareinfrastruktur zur Verfügung steht, sollten doch die Vorteile dieser Vorgehensweise deutlich geworden sein:

- Durch deklaratives Markup werden vorhandene Strukturen so ausgezeichnet, dass sie einer maschinellen Nachbearbeitung zugänglich sind
- Soweit man bei der Verwendung deklarativem Markups auf Standards zurückgreift, können Bestandteile dynamischer Bücher interoperabel gemacht werden
- Die Architektur der XML-Sprachfamilie kann grundsätzlich erweitert werden; d. h. durch Einführung zusätzlicher Markupelemente (ggf. in einem eigenen Namensraum) können neue inhaltliche und/oder funktionale Bestandteile elektronischer Bücher eingeführt werden.

Die für elektronische Bücher erforderliche technische Infrastruktur ist dabei nicht oder nur am Rande behandelt worden. Im Kontext der Diskussion des Prototyps eines dynamischen elektronischen Buchs soll in Teil III daher auch untersucht werden, welche Technologien zur Realisierung dynamischer elektronischer Bücher erforderlich sind und welche Softwaretools hierfür verfügbar sind.

Teil III: Aufbau und Entwicklung dynamischer elektronischer Bücher

Die prototypische Realisierung der in Kap. 4 eingeführten Konzepte dynamischer elektronischer Bücher mit den SGML- und XML- basierten Standards der deklarativen Auszeichnung von Information hat nicht die Entwicklung eines neuen Autorensystems zum Ziel, sondern soll anhand des im Referenzprojekt erstellten elektronischen Buchs die Umsetzbarkeit und Anwendbarkeit des Modells für dynamische elektronische Bücher darstellen. Im Vordergrund steht die *Perspektive des Nutzers* eines elektronischen Buchs, weniger die Position eines Entwicklers, dem ein neues Werkzeug an die Hand gegeben werden soll. Nachfolgend wird gezeigt, wie sich das Konzept unter Zuhilfenahme unterschiedlicher Standards und Softwarewerkzeuge umsetzen lässt.

Dabei gelten für den Prototyp die im Rahmen eines *rapid prototyping*-Verfahrens (vgl. THOMPSON & WISHBOW 1992) zulässigen qualitativen und quantitativen Einschränkungen: Da eine Vielzahl von Arbeitsschritten bei der Aufbereitung der Buchinhalte nicht oder nur partiell automatisierbar sind, erfolgte eine quantitative Einschränkung des Ausgangsmaterials (d. h. des Inhalts des im Referenzprojekt zugrunde liegenden Lehrbuchs). Gleichzeitig konnten nicht alle wünschenswerten Dienste vollständig realisiert werden (qualitative Einschränkung). Für die in Kap. 4.4 eingeführten Dienstetypen und ihre Integrations- bzw. Präsentationsvarianten wird jeweils ein Beispiel angegeben. Die Darstellung der Realisierung des dynamischen elektronischen Buch hat folgenden Aufbau:

- Konkretisierung der in Kap. 4.2 eingeführten allgemeinen Überlegungen zur Einführung verschiedener Markupebenen anhand der Materialien des Referenzprojekts (Kap. 10).
- Vorstellung von Technologien, die bei der Realisierung von Buchserver und Buchbetrachtungssystem verwendet werden (Kap. 11).

- Diskussion der Umsetzung des in Kap. 4.5 entwickelten Architekturmodells beim Aufbau eines Buchservers (Kap. 12).
- Erläuterung des Buchbetrachtungssystems auf der Basis allgemeiner Anforderungen an die ergonomische Gestaltung WWW-basierter Informationssysteme (Kap. 13).
- Überblick zu den verschiedenen Typen von Komponenten, die im Referenzprojekt in das elektronische Buch integriert wurden (Kap. 14.1) sowie Beispiele für die in Kap. 4.4.1 eingeführten Dienstetypen (14.2). Der für die Nutzung elektronischer Bücher zentralen Fragestellung der Informationserschließung für elektronische Bücher ist dabei in Kap. 0 ein besonderer Schwerpunkt gewidmet (Erörterung der Nutzung von Information Retrieval-Technologien für dynamisch elektronische Bücher).

10 Informationskodierung

Die Informationskodierung für das dynamische elektronische Buch erfolgt nach den in Kap. 4.2 vorgestellten Leitlinien für die inhaltsorientierte Informationsauszeichnung. Auf der Basis von XML und seiner Tochterstandards entsteht dabei ein Markupmodell, das sich hinsichtlich des Strukturmarkups an Vorgaben wie DocBook und HTML orientiert und für die inhaltsorientierte Auszeichnung neue Elemente zur adäquaten Beschreibung der im Referenzprojekt vorliegenden Daten enthält. Dabei werden die Möglichkeiten von XML als *Metasprache* genutzt, d. h. aus der Analyse der Struktur und der Inhalte des Ausgangsmaterials ergeben sich Dokumententypdefinitionen. Im einzelnen spielen dabei folgende in Teil II der Arbeit diskutierte Standards eine Rolle:

Standard	*Verwendungszweck*	*ausführliche Darstellung*
Extensible Markup Language (*XML*)	Metasprache für den Entwurf von Markupformaten; syntaktische Grundlage aller anderen verwendeten Standards	Kap. 5.2
XML Namespaces	Namensraum-Mechanismus, der die konsistente Kombination unterschiedlicher Markupformate erlaubt, z. B. bei der Einbindung von Metadaten oder der Diensteverwaltung	Kap. 5.2.3
XLink, XPath und XPointer	Adressierungsformate, die für die Einführung von Verknüpfungen und die Transformation der Ausgangsdaten verwendet werden	Kap. 5.2.4 f.
XML Style Language/Transformation Language XSL/T	Transformations- und Präsentationssprache für die Abbildung des Strukturmarkup und der inhaltsorientierten Auszeichnung in das vom Buchbetrachtungssystem akzeptierte Präsentationsformat	Kap. 7.1.2
Hypertext Markup Language (HTML)	Zielformat, das vom Buchbetrachtungssystem angezeigt werden kann und in das die XML-basierten Daten übersetzt werden	Kap. 6.1 f.
Mathematical Markup Language (MathML)	XML-basiertes Kodierungsformat für die Formeldarstellung bzw. als Austauschformat	Kap. 6.6
Cascading Style Sheets (CSS	*style sheet*-Sprache für die Definition von Gestaltungsregeln für Seitentemplates	Kap. 7.2
Resource Description Framework (RDF)	Generischer Standard für die Kodierung von Metadaten	Kap. 9

Tabelle 47: Übersicht zu den verwendeten Standards

Parallel zur Erstellung der Dokumentformate für die inhaltsorientierte Auszeichnung regeln Transformationsskripte (XSL/T) die Abbildung des Ausgangsformats in das Zielformat, das vom Buchbetrachtungssystem angezeigt werden kann (HTML). Im Rahmen der technischen Realisierung mit Hilfe der *Cocoon Publishing Engine* erfolgt diese Umwandlung auf der Seite des Buchverwaltungssystems (Server).

Die nachfolgende Darstellung der verschiedenen Ebenen des Markup orientiert sich an der in Kap. 4.2.5 entwickelten Vorgehensweise und stellt die für das Referenzwerk erforderlichen Markupelemente vor. Dabei wird von der Aufteilung des Datenbestands in *Basismaterial* und *Komponenten* ausgegangen, wie sie sich zum einen aus dem in Kap. 4.5 entwickelten Architekturmodell dynamischer elektronischer Bücher, zum anderen aus den

Erfordernissen des Buchbetrachtungssystems ergeben, d. h. die Definition der Markupelemente erfolgt durch die Schritte

1. Definition des Strukturmarkups (top-down),
2. Festlegen inhaltsorientierter Kodierung (bottom-up),
3. Definition von Verknüpfungen zwischen den Inhaltsbestandteilen,
4. Präsentationsmarkup bzw. Transformationsprozesse,
5. Metadaten und Schnittstellen.

Die dabei entstehenden Dokumentgrammatiken sind in Anhang 17.1 wiedergegeben.

10.1 Strukturmarkup

Zum Strukturmarkup gehört die Einführung von Elementen für die Kodierung der hierarchischen Struktur des Ausgangsmaterials, die nicht nur für die Präsentation (Umwandlung in Überschriften), sondern auch für den Hierarchiebrowser als Steuerkomponente erforderlich sind. Bei der Definition der Marken in der DTD sind dabei verschiedene Alternativen denkbar:

- Einführung einer generischen Gliederungsmarke mit Ebenen- und Beschriftungszuordnung über Attributwerte:

```
<KAPITEL ebene="1" titel="Mechanik">
   <KAPITEL ebene="2" titel="W&auml;gung und Dichte">
      <KAPITEL ebene="3" titel="Mohr-Westphalsche Waage">
         <!-- Inhalte der dritten Gliederungsebene -->
         <!-- ggf. weitere Untergliederungsebenen-->
      </KAPITEL>
      <!-- ggf. weitere Kapitel-->
   </KAPITEL>
</KAPITEL>
```

- Einführung je eines Elements pro Ebene und Zuordnung der Beschriftung als Attribut:

```
<KAPITEL1 titel="Mechanik">
   <KAPITEL2 titel="W&auml;gung und Dichte">
      <KAPITEL3 titel="Mohr-Westphalsche Waage">
         <!-- Inhalte der dritten Gliederungsebene -->
         <!-- ggf. weitere Untergliederungsebenen-->
      </KAPITEL3>
      <!-- ggf. weitere Kapitel-->
   </KAPITEL2>
</KAPITEL1>
```

- Kodierung der Information über je eigene Elementmarken:

```
<KAPITEL1 >
 <TITEL>Mechanik</TITEL>
 <KAPITEL2>
 <TITEL>W&auml;gung und Dichte"</TITEL>
   <KAPITEL3>
   <TITEL>Mohr-Westphalsche Waage</TITEL>
        <!-- Inhalte der dritten Gliederungsebene -->
        <!-- ggf. weitere Untergliederungsebenen-->
   </KAPITEL3>
   <!-- ggf. weitere Kapitel-->
 </KAPITEL2>
</KAPITEL1>
```

Codebeispiel 77: Varianten der Elementdefinition für das Strukturmarkup

Hier wurde die letzte Alternative gewählt, weil dadurch die Verarbeitung in XSL/T etwas einfacher wird. Neben der Ebenengliederung sind zusätzlich generelle Textstrukturmerkmale erforderlich, die nicht unmittelbar an bestimmte Inhalte gebunden sind. Die nachfolgende Tabelle zeigt in Auswahl einige der erforderlichen Marken:

Funktion	*Marke*	*Kommentar*
Textgliederung	<ABSATZ>	Analog zur Paragraphenmarke in HTML bzw. DocBook. Kontrolle über entsprechende Schemadefinition oder ein ist *architectural form* möglich.
Aufzählungen (geordnet, ungeordnet)	<LISTE typ="geordnet" numerierung="1"> <EINTRAG>...</EINTRAG> </LISTE>	Analog zu den Listentypen in HTML.
Tabellensatz	<HTML:TABLE> <!--... --> <HTML:TR> <HTML:TD><!--... --></HTML:TD> <!--... --> </HTML:TR> <!--... --> </HTML:TABLE>	Übernahme des Tabellenmodells von HTML; Tabellensatz als Textstrukturmerkmal nur dann sinnvoll, wenn nicht inhaltsorientiert kodierte Daten erst zur Präsentation in eine Tabelle transformiert werden sollen.

Tabelle 48: Textstrukturmerkmale für das dynamische elektronische Buch

Neben den voranstehend gezeigten Textstrukturkennzeichnungen können für andere Bezugswerke weitere Elemente erforderlich sein. Wenn dabei teilweise Auszeichnungselemente aus vordefinierten DTDs wie HTML verwendet werden, dann deswegen, um Doppeldefinitionen für gleiche Auszeichnungsprobleme zu vermeiden. Das Beispiel der Übernahme des Tabellenmodells aus HTML ist hier gewählt, da eine inhaltsorientierte Beschreibung der Tabelleninhalte mit nachfolgender Abbildung auf ein struktur- bzw. layoutorientiertes Tabellenmodell aufgrund der Singularität tabellarisch dargestellter Inhalte im Referenzprojekt zu aufwendig erscheint: Für jede Tabelle wäre zunächst eine eigene inhaltsorientierte DTD (bzw. DTD-Teile) zu entwerfen, die bei der Transformation anschließend auf das Tabellenformat abgebildet werden würde.[105]

10.2 Inhaltsorientiertes Markup

Das Strukturmarkup wird ergänzt durch inhaltsorientiertes Markup, dessen Erscheinungsformen sich aus einer inhaltlichen und funktionalen Analyse des Ausgangsmaterials ergeben. Insofern ist seine konkrete Ausprägung an den Einzelfall der Publikation gebunden, allenfalls die Vorgehensweise für die Ermittlung inhaltsorientierter Beschreibungen lässt sich verallgemeinern.

Im Referenzwerk liegt eine sich wiederholende Textstruktur vor, die sich für die inhaltsorientierte Beschreibung nutzen lässt, wobei der Modellierungsaufwand für die DTD- bzw. Schemaentwicklung überschaubar bleibt. Auf einer globalen Ebene ergibt sich die Inhaltshierarchie aus der Gliederung des Gegenstandsbereichs Experimentalphysik in die Gebiete *Mechanik – Wärmelehre – Elektrizitätslehre* und *Optik und Atomphy-*

[105] Dies ist durch XSL/T oder CSS2 möglich, aber aufgrund des damit verbundenen Aufwands nur sinnvoll, wenn es sich nicht um Tabellen mit singulären Inhalten, sondern z. B. um Abfragen von Datensätzen handelt; im Rahmen des Referenzprojekts werden Tabellen als Komponenten gesondert dargestellt und mit Hilfe des HTML-Tabellenmodells kodiert.

sik. Unterhalb dieser Ebene findet in der Regel eine weitere phänomenologische Einteilung statt, z. B. bei der Mechanik in

- *Wägung und Dichte,*
- *Schwingungen,*
- *Deformationsverhalten,*
- *Schall,*
- *Oberflächenspannung und*
- *Viskosität und Strömung.*

Die Einzelkapitel sind weitgehend einheitlich aufgebaut und lassen sich durch *inhaltsbezogene* Marken beschreiben: Zunächst werden die theoretischen Grundlagen eines Gebietes eingeführt und im Anschluss daran einzelne Versuche beschrieben. Der Textaufbau für die verschiedenen Experimente ist ebenfalls weitgehend einheitlich, d. h. ausgehend von einer Aufgabenstellung werden die für den Versuch benötigten Geräte sowie die mit ihnen messbaren Phänomene beschrieben; die Darstellung der eigentlichen Versuchsdurchführung schließt sich an.

Dieser einheitliche Textaufbau rechtfertigt die Einführung inhaltsorientierter Auszeichnungselemente, da durch wiederholte Verwendung keine Kodierung für den Einzelfall stattfindet und die inhaltsbezogenen Marken sowohl für die Darstellung als auch – und dies ist der entscheidende Aspekt – als Anknüpfungspunkt für die Kodierung von Diensten und Komponenten genutzt werden können. Aus dieser groben Beschreibung des Textaufbaus ergeben sich folgende Beschreibungselemente:

Element	*Beschreibung*	*Kommentar*
<EXPERIMENT>	Kennzeichnet ein einzelnes physikalisches Experiment.	Anbindung von Komponenten und Diensten, die für das gesamte Experiment relevant sind.
<AUFGABE>	Beschreibt eine Aufgabenstellung innerhalb eines Experiments.	In Verbindung mit Metadatenstandards aus dem Lehr-/Lernbereich ließe sich die Aufgabenbeschreibung in entsprechende Lehr-/Lernumgebungen integrieren.
<GRUNDLAGEN>	Kennzeichnet die Herleitung allgemeiner Grundlagen bezüglich eines ausgewählten Gegenstandsbereichs (z. B. *Mechanik/Schwingungen*).	Dies ist nur eine grobe Kennzeichnung; eine fein differenzierte Ontologie bzw. ein ontologisches Auszeichnungsmodell vorausgesetzt, könnten die Wissenszusammenhänge auch detaillierter beschrieben werden.
<BESCHREIBUNG>	Stellt das Experiment vor und beschreibt die mit ihm zu messenden Größen; dabei erfolgt in der Regel eine versuchsbezogene Vertiefung der allgemeinen Grundlagen.	
<VERSUCH>	Beschreibt die einzelnen erforderlichen Schritte der Versuchsdurchführung.	Z. B. Anbindung interaktiver Versuchsvisualisierungen.

Tabelle 49: Inhaltsorientierte Auszeichnungen auf der Textebene

In der Regel sind die Experimente wie folgt strukturiert:

```
<EXPERIMENT>
    <AUFGABE>…</AUFGABE>
    <GRUNDLAGEN>…</GRUNDLAGEN>
    <BESCHREIBUNG>…</BESCHREIBUNG>
    <VERSUCH>…</VERSUCH>
</EXPERIMENT>
```

Codebeispiel 78: Aufbau der Inhaltsauszeichnung für ein Experiment

Innerhalb der Inhaltskomponenten können die oben eingeführten Strukturmerkmale der Textauszeichnung (Gliederung in Paragraphen, Einführung von Aufzählungen, Tabellen etc.) auftreten und sind in der DTD entsprechend durch Parameterentitäen und unter Rückgriff auf Elemente aus Standard-DTDs (DocBook, HTML) definiert.

Auf der Ebene der Experimente fallen logische Struktur und inhaltliche Auszeichnung zusammen, d. h. jeder Versuch stellt in der Regel eine Gliederungsebene dar; die Auszeichnungsmarken für ein Experiment sind zusätzlich durch Strukturmerkmale zu kapseln (Kennzeichnung eines Experiments als Strukturbestandteil <KAPITEL4> etc.).

Unterhalb der Textebene kann inhaltsorientiertes Markup zusätzlich auf Wort- und Phrasenebene zweckmäßig sein und dazu dienen, unterschiedliche Typen herausgehobener Begriffe zu kennzeichnen und die Verwendung generischer (z. B. <EM>, <STRONG> in HTML) oder präsentationsbezogener Elemente (z. B. <I>, <B> in HTML) ersetzen bzw. optimieren. Im Referenzwerk finden sich folgende Typen für inhaltlich motivierte Hervorhebungen:

- Allgemeine Fachterminologie,
- Bezeichnungen physikalischer Phänomene und Grundbegriffe,
- Bezeichnung von Geräten, die für die Versuchsausführung erforderlich sind und
- Eigennamen bekannter Physiker.

Es ist jeweils zu unterscheiden, ob ein Begriff im Kontext seiner Definition erstmalig eingeführt wird oder ob von seiner Verwendung ausgehend ein Bezug zu dieser Definition eingeführt werden soll als Hypertextverknüpfung. Dies gilt auch für die oben eingeführten inhaltlichen Auszeichnungen auf Textebene: Zu ihnen treten im Regelfall Einzelbegriffe als Ausgangspunkt für Verknüpfungen, die entsprechend kodiert werden müssen.

Ein weiterer Bereich der inhaltlichen Auszeichnung ist die Kodierung der Komponenten. Der Begriff Komponente ist sowohl inhaltlich als Beschreibung zusätzlicher funktionaler Elemente bzw. Programmodule als auch layoutorientiert definiert, d. h. als gesonderte Darstellung bestimmter Bestandteile des Ausgangsmaterials. Je nach Komplexität kann eine unterschiedlich ausführliche Beschreibung der Komponente erforderlich sein. Im Kontext der Analyse des Referenzwerks und der zusätzlich zu seinem Datenbestand erstellten Zusatzmaterialien ergeben sich folgende Beschreibungsmarken:

Element	*Erläuterung*
<KOMPONENTE>	Generische Containermarke, die Inhalte kennzeichnet, die gesondert präsentiert werden sollen (in der Kontextseite des Buchs oder in einem eigenen Fenster).
<ABBILDUNG>	Beschreibt eine Abbildung; grundsätzlich wäre eine – ebenfalls inhaltsbezogene – Feingliederung (Graph, Diagramm, Schaltplan, schematische Zeichnung eines Geräts) denkbar.
<ANIMATION>	Beschreibt eine Animation, zugeordnet zu einem Versuch oder zur Beschreibung eines Phänomens.
<BIOGRAPHIE>	Beschreibt einen Zusatztext, der die Biographie eines Physikers enthält.
<SIMULATION>	Beschreibt eine Simulation, die einem Versuch zugeordnet ist
<UEBUNG>	Zusätzliche Übung, die den Basistext ergänzt.
<TEXT>	Generischer Container für Zusatztext (Vertiefung, Exkurs, Material aus anderen Büchern), als inhaltsorientiertes Markup durch Attribute typisiert; ggf. Einführung zusätzlicher inhaltsorientierter Marken.

Tabelle 50: Inhaltsorientierte Kodierung von Komponenten

Die Komponenten haben ein unterschiedlich differenziertes Inhaltsmodell (*content model*): Bei Multimediakomponenten (Animationen und Simulationen) ist in Abhängigkeit von ihrer technischen Realisierung innerhalb der jeweiligen Komponentenmarke das entsprechende Element für die Einbettung interaktiver Objekte vorzusehen (z. B. <OBJECT>,

<EMBED>, <APPLET>) oder durch Transformation (XSL/T) zu generieren. Darüber hinaus können die Komponenten durch zusätzliche Textbestandteile beschrieben sein, für die ebenfalls inhaltsorientiertes Markup verwendet wird; zu ihnen gehören

- beschreibende Texte,
- Anleitungen (z. B. für die Bedienung einer Simulation),
- zusätzliche Erläuterungen und
- Metadaten (Schlagworte für die Informationserschließung; Hinweise die Autoren; Metadaten zur Kennzeichnung des interaktiven Lernmaterials, s. u.).

10.3 Hypertext-Relationierung

Die Einführung von Verknüpfungen erfolgt nach Vorliegen der Struktur- und Inhaltskodierung für Basistext und Komponenten. Folgende Typen von Verknüpfungen bzw. Zuordnungen sind zu unterscheiden:

1. Verweise innerhalb des Ausgangstexts, die nach Art ihres Bezugspunkts typisiert sind. Hierzu gehören Rückbezugsverweise auf eine inhaltliche Beschreibungskategorie oder einen Strukturbestandteil des Buchs, also Verweise auf Gleichungen, Aufgabenstellungen, die Herleitung eines Phänomens. Der Typ der Verknüpfung ergibt sich aus der Art des Zielpunkts.
2. Verweise aus dem Basistext auf gesondert dargestellte Komponenten, bei denen der Ausgangsanker des Verweises ein Begriff oder eine Textpassage des Buchs ist (Verweise auf Abbildungen, tabellarisches Datenmaterial oder Simulationen).
3. Zuordnungen zwischen Textbereichen und Komponenten, bei denen der Anker der Verknüpfung nicht an einzelne Worte oder Phrasen gebunden ist (sog. *spanning links*). In diesem Fall ist eine Zuordnung von Komponenten zu Struktur- oder inhaltsorientiertem Markup erforderlich. Die Motivation für diesen Typ der Hypertext-Relationierung ergibt sich aus der Beobachtung, dass eine Komponente, z. B. eine Abbildung oder eine Simulation, über einen größeren Inhaltsbereich hinweg für den Leser relevant sein kann und ihm als Verknüpfung angeboten werden sollte. Beispielsweise ist die Abbildung eines Versuchsaufbaus in der Regel für die verschiedenen Bestandteile der Darstellung eines Experiments als zusätzliche, in einem nicht-textuellen Medium repräsentierte Information sinnvoll zu nutzen und sollte jederzeit – auch bei einer längeren Folge von Präsentationseinheiten – abrufbar sein.

Die Kodierung der Verknüpfungen erfolgt in der XML-Ausgangsstruktur mit Hilfe von XLink, es ergeben sich für die angeführten Typen von Verknüpfungen Unterschiede hinsichtlich der Realisierung als einfache (*simple*) bzw. erweiterte (*extended*) Verknüpfungen:

- Einfache 1:1-Verknüpfungen werden als *simple link* in den als Verknüpfungsanker dienenden Elementen berücksichtigt, ggf. unter Angabe eines Verknüpfungstyps. Sobald von einem Ausgangspunkt mehr als eine Verknüpfung ausgeht, ist sie als *extended link* mit einer Sammlung von Lokatoren kodiert.
- *spanning links*, die von mehr als einer Präsentationseinheit aus verfügbar sein sollen, sind als *extended link* einer inhaltsorientierten Marke zugeordnet; aufgrund der Vielfalt der Komponenten (einschließlich der zahlreichen Abbildungen, die ebenfalls als Komponenten realisiert sind), ist meist eine Mehrzahl von Lokatoren erforderlich. Sie werden für die Präsentation im Buchbetrachtungssystem gesondert behandelt (vom

Basistext abgetrennte gesonderte Darstellung und Typisierung durch Verwendung unterschiedlicher Symbole, vgl. unten Kap. 13.3.4).

Bei der Transformation von Verknüpfungen in die Präsentations-DTD HTML werden die XLinks auf HTML-Anker abgebildet. Für einfache Links entsteht aus einem XLink-Attribut eine Ankermarke, bei 1:n-Verknüpfungen wird unter Berücksichtigung des Linkstyps eine Liste von Ankermarken generiert, die im Buchbetrachtungssystem gesondert dargestellt werden.

10.4 Präsentationsmarkup

Die vierte Ebene der Markupkodierung ist die Festlegung von Transformationsregeln und die Angabe eines Zielformats. Die Notwendigkeit für die Definition solcher Regeln ergibt sich aus

- dem technischen Erfordernis, den neu eingeführten XML-Elementen Präsentationsinformation beizuordnen, ohne die sie nicht interpretiert bzw. dargestellt werden können,
- den gestalterischen Anforderungen für die Darstellung der verschiedenen Inhaltselemente (*screen design*, vgl. unten Kap. 13.3),
- den Erfordernissen des Aufbaus des Buchbetrachtungssystems (Zuordnung der Daten zu Fenstern und Darstellungsbereichen),
- und den technischen Möglichkeiten des zugrunde liegenden Viewingsystems als Basis des Buchbetrachtungssystems (Netscape Navigator V. 4.7).

Für den Prototyp des dynamischen elektronischen Buchs erfolgt die Transformation in zwei Stufen über gesonderte XSL-*style sheets* für den Basistext und die zugeordneten Komponenten. In technischer Hinsicht stellt sich das Problem, eine Abbildung umfangreicher Inhaltskomponenten auf einzelne Präsentationseinheiten zu erreichen, d. h. einen seitengerechten Satz für das elektronische Buch zu erzielen: Ohne Erweiterungsmechanismen werden mit den Verarbeitungsfunktionen von XSL/T die Ausgangsdaten (XML) jeweils auf eine Zieleinheit, i. d. R. eine Datei, abgebildet. Durch die limitierten Darstellungsmöglichkeiten ist für das Buchbetrachtungssystem eine weitere Aufteilung in einzelne Präsentationseinheiten erforderlich (vgl. unten Kap. 12.2).

10.5 Einbindung von Metadaten

In Kap. 9 wurden generische Formate für die Kodierung von Metadaten sowie anwendungsbezogene Metadatenmodelle für die Bereiche bibliographische Beschreibung und Lehr- und Lernsysteme vorgestellt. Von ihnen ausgehend ist zu klären, welche Metadaten für die Entwicklung und Verwaltung dynamischer elektronischer Bücher erforderlich sind. Dabei muss man unterscheiden, welche Informationen bereits über inhaltsorientiertes Markup kodiert werden und wie Metainformation davon getrennt werden kann. Der einfachste Ansatz, wie ihn BERNERS-LEE 1997A als pragmatische Herangehensweise vorschlägt, läuft darauf hinaus, jegliche Zusatzinformation, die nicht bereits in einer einmal erfolgten Aufbereitung der Daten kodiert ist, als Metadaten zu bezeichnen (z. B. bibliographische Daten, Daten zum Speicherungsort, nachträgliche Indexierung, Daten, die für Abrechnungsverfahren benötigt werden etc.). Diese allgemeine Unterscheidung zeigt, wie der Begriff Metadatenkodierung als Restkategorie für alle Informationen verwendet wird, die nicht bereits durch einen vorhandenen Standard (oder eine differenzierte Dokumentenstruktur) abgedeckt werden. Mit dem *Resource Description Framework* liegt ein

allgemeiner Kodierungsstandard vor, der in dieser Weise als „Auffangbecken“ verwendet werden kann. Es ist zu erwarten, dass sich nach und nach Standardisierungsvorschläge für unterschiedliche Kategorien von Metainformation herausbilden und RDF-Schemata für sie definiert werden.

Ein wesentliches Merkmal der Unterscheidung von inhaltsorientiertem Markup und der Kodierung von Metadaten liegt darin, dass Metadaten aufgrund der ihnen zugrunde liegenden semantischen Schemata anwendungsunabhängig automatisch analysiert werden können: Während die inhaltsorientierte XML-basierte Modellierung am Einzelfall orientiert ist,[106] bieten Metadatenschemata wie *Dublin Core* und seine Erweiterungen oder die Lehr- und Lernmetadatenschemata (IMS, IEEE LOM) die Möglichkeit, Metadaten unabhängig von den für ein elektronisches Buch definierten Auszeichnungselementen einzuführen und so die Wiederverwendbarkeit der Substanzen des Buchs im Sinne einer nicht nur medien- sondern auch *anwendungs*neutralen Kodierung zu unterstützen. Dies zeigen exemplarisch FOX et al. 1999, die *Dublin Core*-basierte Metadaten für Lernmaterialien im Rahmen einer digitalen Bibliothek einsetzen. Die Frage der Unterscheidung von Daten und Metadaten und ihrer technischen Umsetzung durch Verwendung inhaltsorientierten Markups (Daten, XML) bzw. Metadatenstandards lässt sich bei textbasierten Anwendungen wie elektronischen Büchern nicht eindeutig treffen. Allgemein halten HUC, LEVOIR & NONON-LATAPIE 1997: Kap. 1 fest:

> One may [...] conclude that the definition of metadata (and hence the boundary between metadata and non metadata) does not correspond to any objective reality and that it could only be established in relation to different categories of people using these ‚objects‘ and the associated metadata. The definition of metadata would therefore be the subjective reflection of a user community at a given time.

BERNERS-LEE 1998B zieht daraus folgenden Schluss:

> Sometimes it seems there is a set of people for whom the semantic web is the only graph which they would consider, and another for whom the document tree (or graph if you include links) is all they would consider. But it is important to recognise the difference.

Unter einem *semantic web* versteht BERNERS-LEE die Repräsentation von Information auf der Basis eines RDF-Schemas (Wissensrepräsentation durch einen *knowledge graph*), unter dem *document tree* die Darstellung durch einen XML-Elementbaum. Bei der Entscheidung, welche Informationen für das elektronische Buch als Metadaten kodiert werden sollen, gibt es folgende Kriterien:

- Die Auswahl eines geeigneten *Metadatenformats* (Syntax, hier: RDF),
- der *inhaltliche Bezug* der Metadaten, aufbauend auf einem Metadatenschema (Semantik, hier: Dublin Core/IMS), d. h. die Funktion, die die Metadaten übernehmen,
- der *Anknüpfungspunkt* der Metainformation (Buchebene, Dokumentenebene, logische Einheit/Präsentationseinheit),
- die Art der *Generierung* der Metainformation:
  - Kodierung durch den Autor oder Entwickler,
  - durch Interaktion mit dem Benutzer,
  - Zwischenverwerter (Integration informationeller Einheiten in neue Kontexte etc.),
  - durch den Distributor,

[106] Durch Einführung *sekundärer Strukturierungsebenen*, wie sie die SGML *architectural forms* bieten, ist eine Kontrolle bzw. Normierung inhaltsorientierter Auszeichnungen möglich, die ggf. auch Funktionen der Metadatenbeschreibung übernehmen bzw. diese überflüssig machen kann, vgl. oben Kap. 5.1.2 und LOBIN 2000: 85 ff.

- der *Entstehungszeitpunkt*: *statisch* zum Zeitpunkt der Inhaltsgenerierung bzw. Inhaltskomposition oder *dynamisch* während der Nutzung,
- der *Gültigkeitsbereich*, z. B. die Einschränkung auf individuelle Benutzer oder einen bestimmten Nutzungszeitraum,
- die Speicherung der Metadaten (eingebettet in die beschriebene Ressource; externe Speicherung als eigene Meta-Ressource) und
- die *Nutzung* der Metadaten (welche *Funktion* hat ein Metadatum?).

Nutzung bzw. Funktion sind notwendige Bedingungen für die Einführung von Metadaten. Erst das Vorliegen eines plausiblen Nutzungsszenarios rechtfertigt den Einsatz von Metadaten, wobei die Nutzung nicht nur innerhalb der konkreten Anwendung eines dynamischen elektronischen Buchs stattfinden kann, sondern (gerade bei Multimediakomponenten) über die Einbindung in eine konkrete elektronische Publikation hinausweisen kann (Wiederverwendung). Allerdings befinden sich zahlreiche Metadatenschemata noch in der Entwicklung. Die hier vorgestellten Beispiele haben daher vorläufigen Charakter. Dies gilt besonders für die Metadatenkodierung in Lehr- und Lernmedien, deren Spezifikation sowohl auf der Ebene der Inhalte (Umfang und Struktur von Metadatenschemata) als auch auf der Ebene der Übersetzung in RDF noch nicht abgeschlossen ist.

Die nachfolgende Tabelle gibt eine Übersicht zu den Verwendungsmöglichkeiten von Metadaten im Rahmen des Prototyps eines dynamischen elektronischen Buchs:

Funktion	*Anknüpfung*	*Nutzung*
Bibliographische Beschreibung	• Globale Beschreibung auf Buchebene • Zusätzliche Beschreibungselemente für alle Inhaltsbestandteile, für die sich abweichende Angaben ergeben (z. B. Autor eines Kapitels) • bibliographische Beschreibung von Multimediakomponenten	Erschließung für die Wiederverwendung; Inhaltserschließung durch eine Infrastruktur, in die ein elektronisches Buch eingebettet sein kann, insb. digitale Bibliotheken
Inhaltliche Beschlagwortung	Multimediakomponenten, die zusätzliche textuelle Beschreibung benötigen und sonst nicht erschlossen werden können	Unterstützung der Informationserschließung
Beschreibung von Lernressourcen	Wie bibliographische Beschreibung	Zuordnung von gruppenbezogenen Diensten (Lehr- und Lernsysteme); Wiederverwendung

Tabelle 51: Einsatz von Metadaten

Die nachfolgenden Beispiele verdeutlichen den Einsatz der verschiedenen Metadatenbeschreibungen; die Generierung erfolgt auf der Basis vorhandener RDF-Schemata und mit Hilfe eines Metadateneditors. Im Verarbeitungsprozess des elektronischen Buchs im Vorfeld der Nutzung (Buchserver) werden für alle generierten Inhaltsdateien Verweise auf den entsprechenden Metadatensatz generiert, soweit eine Speicherungseinheit nicht einen Inhaltsbestandteil enthält, der in Abweichung von den global gültigen Beschreibungen eine spezifische Metadatenbeschreibung zugewiesen bekommt.

Das nachfolgende Beispiel zeigt eine top-level-Beschreibung des Referenzwerks in verkürzter RDF-Syntax (Inhalte als Attributwerte der Dublin Core-Elementfelder). Diese Dublin Core-Umsetzung wurde mit dem Metadateneditor *Reggie* erstellt, vgl. *http://metadata.net/dstc/*.

```
<?xml version = "1.0"?>
<rdf:RDF xmlns:rdf  = "http://www.w3.org/TR/1999/02/22-rdf-syntax-ns#"
   xmlns:DC = "http://metadata.net/dstc/DC-10-DE/#">

 <rdf:Description  xml:lang="de"
                   rdf:about="http://aspra9.informatik.uni-leipzig.de/Teubner/physikbuch.xml">
  <DC:Title rdf:value = "Multimediales Physikalisches Praktikum"/>
  <DC:Creator.PersonalName rdf:value = "Dieter Geschke"/>
  <DC:Subject>  <rdf:bag   rdf:_1 = "Experimentalphysik"
                           rdf:_2 = "Anf&auml;ngerpraktikum"
                           rdf:_3 = "Versuchsbeschreibungen"   />
  </DC:Subject>
  <DC:Description rdf:value =   "Das Werk stellt eine multimediale Einf&uuml;hrung in die
                                 Experimentalphysik dar. Es gliedert sich in die Teile Allgemeine
                                 Grundlagen, Mechanik, W&auml;rmelehre, Elektrizit&auml;tslehre und
                                 Optik und Atomphysik.  Etwa 80 Versuche werden ausf&uuml;hrlich
                                 beschrieben."/>
  <DC:Publisher.CorporateName rdf:value = "B. G. Teubner Stuttgart Leipzig"/>
  <DC:Contributor.PersonalName rdf:value = "Christian Wolff"/>
  <DC:Contributor.PersonalName.Address rdf:value = "Universit&auml;t Leipzig, Institut f&uuml;r
Informatik"/>
  <DC:Date.Created rdf:value = "Juli 1999"/>
  <DC:Date.Available rdf:value = "August 2000"/>
  <DC:Type rdf:value = "Elektronisches Buch mit multimedialen Erg&auml;nzungen"/>
  <DC:Format>   <rdf:bag      rdf:_1 = "text/xml"
                              rdf:_2 = "Java"
                              rdf:_3 = "Macromedia Shockwave" />
  </DC:Format>
  <DC:Identifier rdf:value = "http://aspra9.informatik.uni-leipzig.de/Teubner/start.htm"/>
  <DC:Language rdf:value = "DE"/>
  <DC:Rights rdf:value = "Das Werk einschlie&szlig;lich aller seiner Teile ist urheberrechtlich
gesch&uuml;tzt."/>
 </rdf:Description>
</rdf:RDF>
```

Codebeispiel 79: RDF-Kodierung von bibliographischen Daten nach dem Dublin Core-Schema

Ein zweites Beispiel zeigt die Beschreibung einer eingebetteten Komponente als Lernressource; dabei ist zu beachten, dass bei IMS wie auch bei IEEE LOM für zahlreiche Datenfelder noch kein verbindliches Vokabular für die Attributwerte existiert; eine stärkere terminologische Kontrolle ist wünschenswert (z. B. durch Spezifikation geeigneter Ontologien und Klassifikationen). Diese ist jedoch erst mittelfristig zu erwarten. Ein zweiter beachtenswerter Punkt ist die Tatsache, dass in IMS unmittelbar auch die *Dublin Core*-Felder für die bibliographischen Angaben eingesetzt werden.

```
<?xml version = "1.0"?>
<RDF  xmlns = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:DC = "http://metadata.net/dstc/DC-10-EN/#"
       xmlns:IMS = "http://metadata.net/dstc/IMS/#">
  <Description xml:lang="en"
       about="http://www.aspra9.informatik.uni-leipzig.de/Teubner/Komponenten/osziueben.class">
  <DC:Title.Main xml:lang="de">Simulation Elektronenstrahloszilloskop</DC:Title.Main>
  <DC:Creator.PersonalName>
    <Bag><li xml:lang="de">Tino Hoffmann</li><li xml:lang="de">Dirk Walther</li></Bag>
  </DC:Creator.PersonalName>
  <DC:Subject.Description xml:lang="de">Experimentalphysik</DC:Subject.Description>
```

```
  <DC:Subject.Keywords xml:lang="de">Oszilloskop Elektronenstrahloszilloskop Arbeitsger&auml;t
    Versuchsaufbau Zweikanalelektronenstarhloszilloskop</DC:Subject.Keywords>
  <DC:Description xml:lang="de">Modulare Simulation eines Zweikanalelektronenstrahloszilloskops mit
    zwei generischen Tongeneratoren als Datenquellen</DC:Description>
  <DC:Publisher.CorporateName xml:lang="de">B. G. Teubner Stuttgart Leipzig
  </DC:Publisher.CorporateName>
  <DC:Publisher.CorporateName.Address xml:lang="de">Johannisgasse 16, 04109 Leipzig
  </DC:Publisher.CorporateName.Address>
  <DC:Contributor.PersonalName>
    <Bag><li xml:lang="de">Dieter Geschke</li><li xml:lang="de">Wolfgang Schenk</li>
      <li xml:lang="de">Christian Wolff</li></Bag>
  </DC:Contributor.PersonalName>
  <DC:Date.Created xml:lang="de">1998-10-31</DC:Date.Created>
  <DC:Date.PublicationDate xml:lang="de">2000-08-01</DC:Date.PublicationDate>
  <DC:Type xml:lang="de">Simulation</DC:Type>
  <DC:Format xml:lang="de">Java Bytecode (Java Applet, JDK 1.1)</DC:Format>
  <DC:Identifier xml:lang="de" DC:Scheme="URI">
    http://www.aspra9.informatik.uni-leipzig.de/Teubner/Komponenten/osziueben.class
  </DC:Identifier>
  <IMS:Language xml:lang="de" IMS:Scheme="RFC1766">DE</IMS:Language>
  <DC:Relation.IsPartOf xml:lang="de">
    http://aspra9.informatik.uni-leipzig.de/Teubner/komponenten/OsziUeben.html
  </DC:Relation.IsPartOf>
  <DC:Rights.Use Rights xml:lang="de">
    Nur bei Vorliegen einer g&uuml;ltigen Zugangslizenz f&uuml;r das Gesamtwerk g&uuml;ltig.
  </DC:Rights.Use Rights>
  <IMS:Concepts xml:lang="de">Erlernen von Arbeitsger&auml;ten f&uuml;r physikalische Versuche;
    modularer Aufbau; </IMS:Concepts>
  <IMS:InteractivityLevel> sehr hoch (interaktive Simulation) </IMS:InteractivityLevel>
  <IMS:LearningLevel>Abiturniveau Physik; Erstsemester</IMS:LearningLevel>
  <IMS:Meta-Meta-data.Author xml:lang="de">Christian Wolff</IMS:Meta-Meta-data.Author>
  <IMS:Meta-Meta-data.LastModifiedDate xml:lang="de">2000-03-08
  </IMS:Meta-Meta-data.LastModifiedDate>
  <IMS:Meta-Meta-data.Scheme xml:lang="de">IMS</IMS:Meta-Meta-data.Scheme>
  <IMS:Objectives xml:lang="de">Handlungswissen. Sekund&auml;r: Ger&auml;t als Auspr&auml;gung
von Ph&auml;nomenen
    (Bildr&ouml;hre etc.) </IMS:Objectives>
  <IMS:Pedagogy xml:lang="de">Exploratives Lernen; Selbststudium</IMS:Pedagogy>
  <IMS:Platform xml:lang="de">Plattformneutral (Windows, MAC, Linux, UNIX)</IMS:Platform>
  <IMS:Prerequisites xml:lang="de">Allgemeine Grundlagen; Messtechnik</IMS:Prerequisites>
  <IMS:Presentation xml:lang="de">Interaktive Software; WWW-Browser</IMS:Presentation>
  <IMS:UseTime xml:lang="de">Acht Stunden, ggf. in mehreren Bl&ouml;cken (Einsatz unter
verschiedenen
    Aspekten) </IMS:UseTime>
  <IMS:Version xml:lang="de">1.0</IMS:Version>
 </Description>
</RDF>
```

Codebeispiel 80: Beispiel der Metadatenbeschreibung einer Simulation

Einen unmittelbaren Beitrag für die Funktionsweise des hier diskutierten Konzepts dynamischer elektronischer Bücher leisten Metadaten bei der Informationserschließung: Durch die zusätzliche Beschlagwortung mit Metadaten kann bei der Auswertung des Lesekontexts als Unterstützung bei der Suche nach externen Ressourcen auch Information über eingebettete Komponenten angeboten werden, vgl. dazu näher Kap. 14.2.2.5.2. In

Abhängigkeit von Typ und Umfang multimedialer Inhalte in elektronischen Büchern können weitere Funktionen der Metadatenbeschreibung hinzukommen; dazu gehören:

- *quality of service*-Metadaten (*QoS*) bei alternativen Medienformaten (z. B. Videodateien unterschiedlicher Auflösung/Framerate/Bandbreite),
- Integration digitaler Signaturen (Urheberrechtsschutz, Verifikation, Absicherung),
- Metadaten zur Unterstützung der Abrechnung bei kostenpflichtigen Diensten (*electronic commerce*).

XML-basierte Standards werden im Rahmen dieser Arbeit nicht nur für die primäre und sekundäre (Metadaten) Beschreibung der Daten, sondern auch für die Kodierung von Information zur Abwicklung von Kommunikationsprozessen zwischen den Softwarekomponenten des elektronischen Buchs verwendet (Kodierung von Diensten, Informationen über Benutzer etc.). Auch wenn man argumentieren kann, dass es sich bei dieser Art Information um *Metadaten* bezüglich der Nutzung eines elektronischen Buchs handelt, werden sie direkt durch XML-DTDs bzw. Schemata kodiert: Da keine geeigneten vordefinierten Schemata für diese Anwendungszwecke existieren, würde die Kodierung als Metadaten keinen erkennbaren Vorteil ergeben; zudem ist die Ausdrucksmächtigkeit von XML (im Vergleich mit dem Typensystem von RDF) für die benötigten Strukturen ausreichend.

11 Entwicklungswerkzeuge für elektronische Bücher

Nach der Betrachtung der für die Informationskodierung erforderlichen Markupelemente soll nachfolgend ein technologie- bzw. werkzeugorientierter Aspekt der Entwicklung elektronischer Bücher beleuchtet werden: Geeignete Entwicklungswerkzeuge. In Ergänzung der bereits in den Kapiteln zu SGML und XML diskutierten Werkzeuge sollen nachfolgend zwei Typen betrachtet werden:

- Die im Rahmen dieser Arbeit primär relevanten Programmiersprachen *Java* und *JavaScript*, und
- Multimedia-Autorensysteme für die integrierte Produktion von Hypermedia- bzw. Multimediaanwendungen als Oberklasse dynamischer elektronischer Bücher.

Der in Kap. 3.2 entwickelte Kriterienkatalog für elektronische Bücher hat verdeutlicht, dass bei Entwicklung, Verwaltung und Nutzung dynamischer elektronischer Bücher eine Vielzahl unterschiedlicher Softwarewerkzeuge und Technologien benötigt wird. Die nachfolgende Betrachtung, welche Technologien für die Entwicklung dynamischer elektronischer Bücher erforderlich sind, muss von vornherein Einschränkungen unterworfen sein: Ebenso wie in Kap. 3.2 darauf verzichtet wurde, Formate atomarer Medientypen als Bestandteile eines elektronischen Buchs – Zeichensätze, Graphik- oder Audioformate – zu beschreiben,[107] so wird im Folgenden davon abgesehen, die zu ihrer Erstellung notwendigen Werkzeuge (z. B. Textverarbeitungsprogramme, Videoschnittsoftware oder Bildverarbeitungssysteme) zu diskutieren. Es ist davon ausgegangen, dass solche Werkzeuge als notwendige Voraussetzung für die Erstellung der atomaren Medienelemente dynamischer elektronischer Bücher zur Verfügung stehen. Nachfolgend wird untersucht, welche *Entwicklungswerkzeuge* für elektronische Bücher zur Verfügung stehen, insbesondere Autorensysteme für das *multimedia authoring* sowie geeignete Programmiersprachen für elektronische Bücher. Dabei ist einerseits zu klären, wie die Steuerungsfunktionalität elektronischer Bücher realisiert werden kann, andererseits ist aufzuzeigen, welche Werkzeuge sich für die Entwicklung *multimedialer Komponenten* in elektronischen Büchern eignen.

Weitere technologieorientierte Aspekte werden in den Kapiteln 12-14 jeweils im Kontext des konkreten *Teilproblems* der Realisierung dynamischer elektronischer Bücher zu diskutieren sein. Zu ihnen gehören

- die Middleware-Technologien, die für die Realisierung des Architekturmodells dynamischer elektronischer Bücher als verteilter, netzbasierter Anwendung erforderlichen (vgl. unten Kap. 12.1),
- die unterschiedlichen Möglichkeiten der *Speicherung* der Inhalte elektronischer Bücher, vor allem die Alternativen der dateisystem- bzw. der datenbankbasierten Speicherung (vgl. unten Kap. 12.2),
- die für die Kommunikation erforderlichen Netzwerkprotokolle, d. h. vor allem die Technologien im Umfeld des *HyperText Transfer Protocol* (vgl. unten Kap. 12.6),

---

[107] Eine detaillierte Beschreibung aktueller Standards findet sich in STEINMETZ 1999; FÜNFSTÜCK, LISKOWSKY & MEIßNER 2000 geben einen Überblick zu Werkzeugen für die Entwicklung von Multimedia-Anwendungen.

- geeignete Softwarewerkzeuge für die Präsentation elektronischer Bücher (vgl. unten Kap. 13) und
- das softwaretechnische Konzept der Komponentenentwicklung (*component ware*, vgl. unten Kap. 14.1.1).

Im Zentrum der Überlegungen stehen dabei folgende Fragen:

- Welche Rolle können die diskutierten Werkzeuge und Technologien im Referenzprojekt *Multimediales Physikalisches Praktikum* spielen bzw. wie werden sie eingesetzt?
- Welche Rolle spielen sie bei der Realisierung einer Publikationsinfrastruktur, die Inhalte so weit wie möglich deklarativ auszeichnet, und welcher Mehrwert kann mit ihnen aus der so kodierten zusätzlichen Information gezogen werden ?

11.1 Programmiersprachen: Java und JavaScript

Für die Realisierung des Prototyps eines dynamischen elektronischen Buchs spielen Programmiersprachen für alle Entwicklungsaufgaben eine Rolle, für die nicht spezialisierte Entwicklungswerkzeuge wie Autorensysteme oder Editoren eingesetzt werden können. Hier betrifft dies die Aspekte:

- Vorverarbeitung der Daten (Java, Tools (Editoren, Parser, Konverter, Makros in Office-Programmen)),
- Aufbau des Buchservers (Java),
- Erstellung von Simulationen (Java) und
- Steuerung der Benutzerschnittstelle (Buchbetrachtungssystem, JavaScript).

Für die Entwicklung von Animationen und interaktiven Versuchsdarstellungen konnte dagegen ein Autorensystem zum Einsatz kommen.

11.1.1 Java

Wie die voranstehende Liste zeigt, wurde Java (vgl. ARNOLD & GOSLING 1998, WOLFF 1999A) als Programmiersprache für eine Reihe von Entwicklungsaufgaben sowohl im Referenzprojekt als auch für den Prototyp eines dynamischen elektronischen Buchs eingesetzt. Diese Wahl lässt sich sowohl auf *konzeptueller Ebene* mit den spezifischen Eigenschaften dieser Programmiersprache als auch *in praktischer Sicht* mit den verfügbaren Java-basierten Werkzeuge begründen. Hinsichtlich des ersten Aspekts handelt es sich dabei vor allem um folgende Merkmale:

- Java greift wesentliche Eigenschaften der objekt-orientierten Programmierung auf.
- Java ist eine plattform- und architekturneutrale Sprache, d. h. kompilierter Java-Code (Bytecode) kann auf praktisch allen Rechnerplattformen ohne Portierung ausgeführt (interpretiert) werden.
- Mit Java sind durch das *Multithreading*-Konzept nebenläufige Programme möglich, d. h. es können zu einem Zeitpunkt mehrere Ausführungsstränge (*threads*) eines Programms aktiv sein, was gerade für Multimediaanwendungen wichtig ist.
- Java verfügt über ein flexibles Sicherheitsmodell, dessen Sicherheitsmechanismen Javaprogramme für den Einsatz in Datennetzen tauglich machen.
- Bezüglich der Syntax und der lexikalischen Struktur kann Java als eine Weiterentwicklung der höheren Programmiersprache C++ verstanden werden, da Schlüsselwörter, Operatoren, Programmstruktur und die Steuerung des Kontrollflusses große

Ähnlichkeiten mit C++ aufweisen. Java ist bei entsprechenden Vorkenntnissen schnell erlernbar.
- Die inhärenten Performanznachteile von Java, die dadurch bedingt sind, dass es sich um eine interpretierte Sprache handelt (vgl. TYMA 1998), fallen für die in Java realisierten Programme hier nicht ins Gewicht.

Neben diesen Eigenschaften, die für sich genommen kein Alleinstellungsmerkmal für Java bedeuten (vgl. FRANZ 1998), sprechen zudem folgende Überlegungen für den Einsatz von Java:

- Java-Applets als Programme, die sich in einen Webbrowser einbetten lassen, sind besser als mit Autorensystemen generierte Komponenten für die Umsetzung von Simulationen geeignet.
- Die Java-Entwicklungsumgebung (*Java Development Kit, JDK*) bietet eine Vielzahl vordefinierter Klassen für Netzwerkprogrammierung und die Implementierung von Benutzerschnittstellen.
- Im Bereich der Texttechnologie sind die meisten Werkzeuge, die für die Verarbeitung XML-basierter Sprachen zur Verfügung stehen, in Java entwickelt und stehen in vielen Fällen als *open source* für die Anpassung zur Verfügung (vgl. oben Kap. 5.2.7.2, Tabelle 25).
- Die in dieser Arbeit verwendete *publishing engine*, Cocoon, eine in Java als Servlet (s. u. Kap. 12.1.4) realisierte Anwendung, die für den Prototyp eines dynamischen elektronischen Buchs angepasst wurde.

11.1.2 JavaScript

Auf der Ebene des Buchbetrachtungssystems fand mit JavaScript (vgl. FLANAGAN 1997) eine weitere Programmiersprache Verwendung. JavaScript ist eine interpretierte objektbasierte Skriptsprache, die syntaktisch an Java angelehnt ist und deren Anweisungen direkt in einer HTML-Datei kodiert werden.[108] Dies bedeutet, dass JavaScript anders als Java nur im Kontext des World Wide Web und in Kombination mit einem JavaScript-fähigen WWW-Browser einsetzbar ist. Der Code wird vom Browser nach dem Laden des Dokuments interpretiert und ausgeführt. Im Unterschied zu Java verfügt JavaScript über keine strenge Typprüfung, Funktionen ohne Bindung an Objekte sind zulässig. JavaScript nutzt eine vordefinierte Objekthierarchie des WWW-Browsers, die den Zugriff auf Eigenschaften des Browsers bzw. des aktuellen Dokuments zulässt. Durch die Spezifikation eines *event module* in XHTML bzw. die Definition entsprechender Attribute und Elemente in HTML ist die Verwendung von Skriptsprachen wie JavaScript standardisiert (vgl. Kap. 6.1 und 6.2). Tabelle 52 zeigt Unterschiede zwischen Java und JavaScript.

Kriterium	*Java*	*JavaScript*
Programmausführung	Quellcode wird zu Bytecode kompiliert, der von der virtuellen Java-Maschine interpretiert werden kann.	Kein Compiler; der in der HTML-Seite eingebetteter Quellcode wird interpretiert.
Objektorientierung	Vollständig objektorientiert.	Lediglich objekt-basiert, ohne Klassen und Vererbung; Objekte können aber erzeugt werden.

[108] Ursprünglich wurde sie unter dem Namen *LiveScript* von Netscape entwickelt, vgl. *http://developer.netscape.com.* Derzeit unterstützen sowohl der Netscape Navigator als auch der Microsoft Internet-Explorer JavaScript, wenn auch mit Unterschieden, vgl. FLANAGAN 1997: 267 ff.

Kriterium	*Java*	*JavaScript*
Einsatz in HTML	Einbettung als Applet.	Direkte Kodierung in HTML bzw. Einbettung einer JavaScript-Quellcodedatei.
Variablendeklaration	Streng typisiert, Deklaration zur Kompilierungszeit erforderlich.	Keine strenge Typprüfung, Auswertung erst zur Laufzeit.
Objektreferenzen	Müssen zur Compilezeit verfügbar sein, um Typprüfung zu ermöglichen.	Prüfung zur Laufzeit.
Programmfunktionalität	Methoden sind an Klassen bzw. deren Instanzen gebunden.	Funktionen ohne Bezug zu bestimmten Objekten können definiert werden. Gleichzeitig haben die vordefinierten Objekte in JavaScript Methoden.

Tabelle 52: Vergleich von JavaScript und Java

Da für das elektronische Buch die Benutzerschnittstelle durch Anpassung eines Webbrowsers erfolgte (vgl. unten Kap. 13), waren JavaScript-Funktionen für die Fensterverwaltung und die Interaktionssteuerung erforderlich.

11.2 Autorensysteme

Unter einem Autorensystem ist ein Softwarewerkzeug zu verstehen, mit dessen Hilfe sich Medienelemente unterschiedlichen Typs zu einer multimedialen Anwendung zusammenstellen lassen (vgl. BOLES 1995, FREIBICHLER 1997). Der Begriff Autorensystem ist historisch mit der Entwicklung von Werkzeugen für das computergestützte Lernen verbunden, weil solche Systeme den Autoren als Fachleuten bezüglich der *Inhalte*, nicht bezüglich der Lerntechnologien, als einfach zu bedienende Werkzeuge für die Erstellung von Lerneinheiten dienen sollen. Beispiele solcher Systeme reichen bis in die 60er Jahre zurück (vgl. dazu ausführlich SCHULMEISTER 1997: 87 ff.). Mit der Renaissance des Hypertextgedankens und der Einführung direktmanipulativer Benutzerschnittstellen seit Ende der 80er Jahre hat der Begriff Autorensystem eine Erweiterung gefunden, die über den engeren Kontext des computergestützten Lernens hinausreicht und vor allem die Erstellung und Programmierung multimedialer Präsentationen für beliebige Anwendungszwecke umfasst. Autorensysteme sind durch folgende Merkmale gekennzeichnet:

- Sie präsentieren dem Entwickler ihre Funktionalität über eine bestimmte Metapher, die den Erstellungsprozess unterstützen soll, z. B. die Filmmetapher bei *Macromedia Director* oder die Buch- und Autorenmetapher bei *Asymetrix ToolBook*.
- Sie erlauben den Import einer Vielzahl von Medientypen und verfügen wenigstens über einfache Editoren für Medienelemente (Text, Bild, ggf. Video/Audio).
- Eine eingebettete Programmiersprache macht die dynamische Programmierung von Präsentationen möglich; Schnittstellen zu externen Programmen bzw. Datenbanken erlauben die Erweiterung durch Zusatzmodule.
- Sie verfügen über Exporttechniken oder besondere Datenformate, die die Nutzung der mit einem Autorensystem erstellten Anwendungen im World Wide Web zulässt. Dies erfolgt entweder durch die Konvertierung einzelner Präsentationseinheiten in HTML (z. B. Asymetrix *ToolBook Publisher*) oder durch plug-in-Technologien, bei denen die Autorensystemanwendungen als eingebettete Objekte genutzt werden können.

Neben diesen Basiseigenschaften können besondere Merkmale (Planung zeitabhängiger Medien, Einbindung vordefinierter Übungstemplates, Einbettung in generische Lehr- bzw. Lernumgebungen) zusätzliche Schwerpunkte setzen. Drei Beispiele von Autorensystemen sollen diskutiert werden, da sie gute Beispiele für den state-of-the-art im Bereich

der Multimediaentwicklung bzw. des elektronischen Publizierens sind und an ihnen Defizite bezüglich der Entwicklung elektronischer Bücher verdeutlicht werden können. Eine Marktübersicht zu Autorensystemen findet sich bei RIEDER 1999.

11.2.1 Asymetrix ToolBook

Bei ToolBook (vgl. *http://www.asymetrix.com*) handelt es sich um ein seit langem verfügbares Autorensystem für hypermediale Präsentationen auf der Basis einer *Buchmetapher*: Die Anwendungen sind als Bücher (*books*) modelliert, die sich in einzelne Seiten gliedern, die elementaren Präsentationseinheiten einer ToolBook-Anwendung. ToolBook unterscheidet zwischen Vorder- und Hintergrund der Buchseiten, wobei der Hintergrund für die Entwicklung von Templates für eine Mehrzahl gleich strukturierter Seiten genutzt werden kann. Produktions- und Nutzungsmodus sind als *Autoren-* bzw. *Leserebene* getrennt, alle Editierungswerkzeuge sind im Lesemodus ausgeblendet, d. h. Toolbook verwendet *eine* Benutzungsumgebung in verschiedenen Modi sowohl für die Produktion als auch für die Präsentation.

Neben der durch vorgegebene Schemata möglichen Verwendung von Navigationsmustern bietet ToolBook in Weiterführung früherer Hypermedia-Autorensysteme wie z. B. *HyperCard* die Möglichkeit, Hypertextverknüpfungen zwischen den Seiten einer ToolBook-Anwendung einzuführen. Sie lassen sich auch automatisch generieren.

Die verschiedenen Elemente einer ToolBook-Anwendung stehen in einer hierarchischen Beziehung zueinander (eine Anwendung enthält Seiten, Seiten enthalten Medien- und Interaktionselemente etc.). An die verschiedenen Objekte dieser Hierarchie können Skripten der ToolBook-Programmiersprache *OpenScript* gebunden werden, über die sich Darstellung und Funktionalität einer ToolBook-Anwendung steuern lassen. Über solche Skripte lassen sich externe Programmodule (i. d. R. als *dynamic link library*, DLL) einbinden. Die Syntax von *OpenScript* ist stark an die Syntax des Englischen angelehnt, um die Einarbeitung zu erleichtern. Die nachfolgende Abbildung zeigt ToolBook im Autorenmodus anhand einer Beispielseite aus UNIVERSITÄT LEIPZIG 1998.

Für das Referenzprojekt *Multimediales Physikalisches Praktikum* schied ToolBook als primäres Autorensystem aus, da seine Stärken vor allem im Bereich textintensiver Multimediaanwendungen liegen, die Möglichkeiten der Verarbeitung hochstrukturierten wissenschaftlichen Fachtexts aber stark begrenzt sind: In Toolbook lassen sich weder SGML- bzw. XML-basierte Textformate importieren noch ist eine hinreichende Darstellungsqualität für Formelsatz gegeben. Die Entwicklungsmöglichkeiten für komplexe Multimediaanwendungen sind gegenüber anderen Autorensystemen, insbesondere *Macromedia Director*, beschränkt.

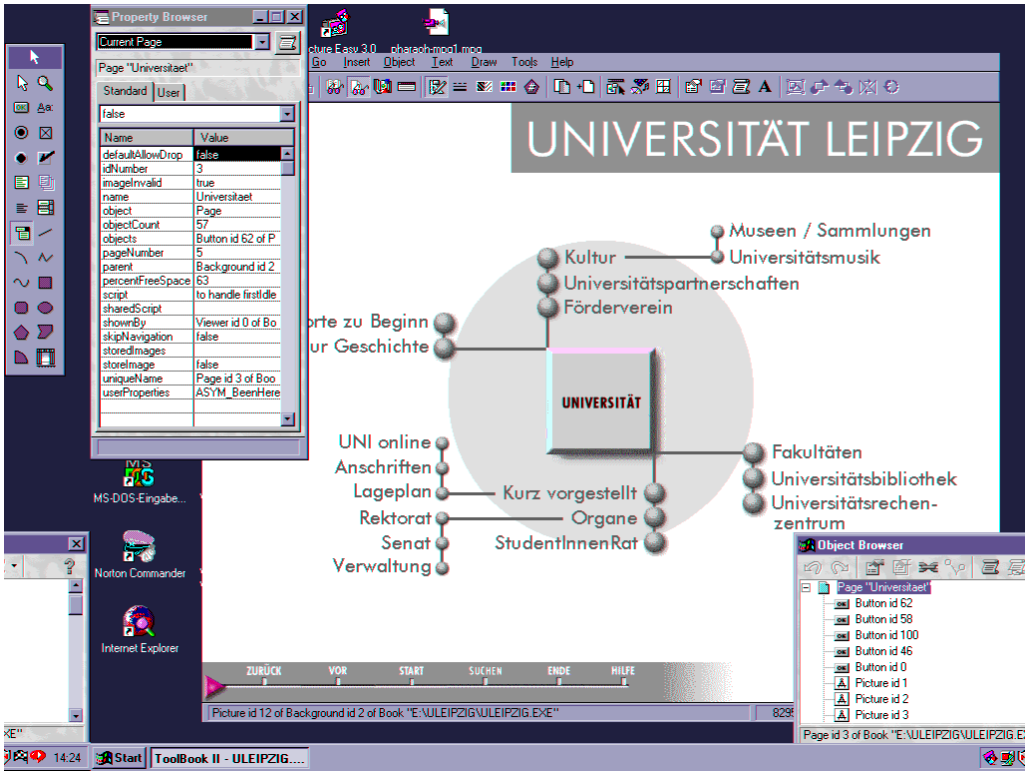

Abbildung 36: ToolBook 6.1 im Autorenmodus, Beispielseite aus UNIVERSITÄT LEIPZIG 1998

11.2.2 Macromedia Director

Macromedia Director (vgl. *http://www.macromedia.com*) ist ein Multimedia-Autorensystem, das auf der Basis der Filmmetapher eine Entwicklungsumgebung für interaktive Multimediaanwendungen mit besonderer Betonung zeitabhängiger Medien bietet (vgl. BOLES et al. 1998B). Die einzelnen Funktionskomponenten orientieren sich an Begriffen aus dem Film- bzw. Kinobereich:

- Die Ablaufplanung einer Director-Anwendung erfolgt über ein *Drehbuch* mit verschiedenen Kanälen als parallele Zeitachsen, in dem sich die Grundkonzeption der Publikation umsetzen und planen lässt,
- die Verwaltung der verschiedenen Medienelemente einer Anwendung als *Darsteller* in einer Besetzungsliste, die in das Drehbuch eingefügt werden können und
- die Präsentation der Anwendung selbst auf der *Bühne* des Systems.

Die Programmierung einer Director-Anwendung erfolgt ereignisbasiert über die Programmiersprache *Lingo*, die ähnlich wie *ToolBook OpenScript* an die Objekte (Darsteller) bzw. die Frames der Kanäle des Drehbuchs gebunden werden. Wie *OpenScript* enthält *Lingo* Elemente einer höheren Programmiersprache (Operatoren, Datentypen, Nachrichtenverarbeitung) mit einer ebenfalls an die natürliche Sprache angelehnten Syntax.

Wie die Erfahrungen im Projektverbund Multimediabuch zeigen, ist Macromedia Director das derzeit vorherrschende Werkzeug für die Multimediaentwicklung (vgl. BOLES et al. 1998B); die Einzelvorhaben dieses Verbunds verwenden fast ausnahmslos Director entweder als primäre Entwicklungsplattform oder als Werkzeug zur Erstellung einzelner Multimediakomponenten. Dabei treten die Nachteile eines proprietären Datenformats, das eine konsistente Verwaltung von Medienbausteinen und ihre einfache komponentenori-

entierte Wiederverwendung erschwert, deutlich zu Tage und behindert einen zyklischen Entwicklungsprozess. Fehlende standardisierte Austauschformate behindern die Weiter- oder Wiederverwendung interaktiver Komponenten, die mit einem Multimedia-Autorensystem entwickelt wurden. Konverter nach SMIL oder HyTime sind bisher nicht Bestandteil ihrer Funktionalität. Dem stehen aber die Vorteile einer integrierten Entwicklungsumgebung entgegen, die auf die Anforderungen der Multimediaentwicklung ausgerichtet ist. Im Referenzprojekt *Multimediales Physikalisches Praktikum* wird Director aufgrund einer negativen Evaluierung hinsichtlich seiner Eignung zur Präsentation komplexer Fachtexte zwar nicht als primäre Entwicklungsplattform verwendet (vgl. dazu BRÖRKENS et al. 1998), dient aber als Autorensystem sowohl für interaktive Multimediaanimationen als auch für die Erstellung interaktiver Versuchsdarstellungen (s. u. Kap. 14.1.3 ff., Abbildung 63 ff.).

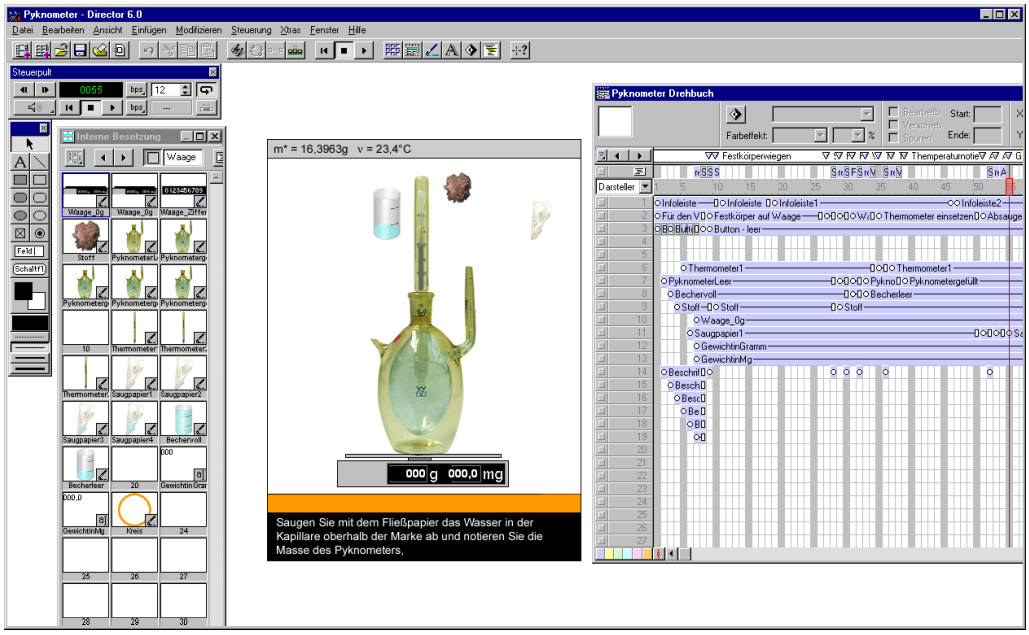

Abbildung 37: Animation Pyknometer im Autorenmodus von Macromedia Director

11.2.3 Authorware

Einen dritten Typus eines Multimedia-Autorensystems stellt Authorware dar, das vor allem für interaktive Tutorien und Anwendungen im Bereich des *Computer Based Training* (CBT) Verwendung findet. Für den Entwickler bietet es den Vorteil, die Erstellung von Tutorien durch *visuelle Programmierung* zu unterstützen: Über einen graphischen Editor können Ablaufpläne für eine Anwendung erstellt und durch die interaktive Spezifikation von Kontrollmechanismen die Programmlogik definiert werden. Die Abbildung 38 zeigt eine Beispielanwendung von Authorware im Autorenmodus; dabei sind mehrere visuelle Skripte geöffnet und können vom Autor direkt-manipulativ verändert werden.

11.3 Fazit

Unter den konkreten Entwicklungsbedingungen des Referenzprojekts konnte kein Multimedia-Autorensystem als primäre Entwicklungsplattform verwendet werden. Dies liegt sowohl an prinzipiellen Defiziten wie der mangelnden Unterstützung neuerer Standards

im Umfeld des World Wide Web (HTML, XML, XSL) und der Verwendung proprietärer Datenformate, die eine Wiederverwendung erschweren, als auch an durch die Besonderheiten wissenschaftlicher Lehrwerke bedingten Anforderungen wie einer geeigneten Darstellung von Formelsatz.

Hinzu kommt die Trennung unterschiedlicher Modellierungsebenen, die die Präsentation der Inhalte von der Speicherungs- bzw. Verwaltungsebene unterscheidet. Sie lässt sich mit XML-basierten Werkzeugen einfacher umsetzen als mit Hilfe eines Autorensystems.

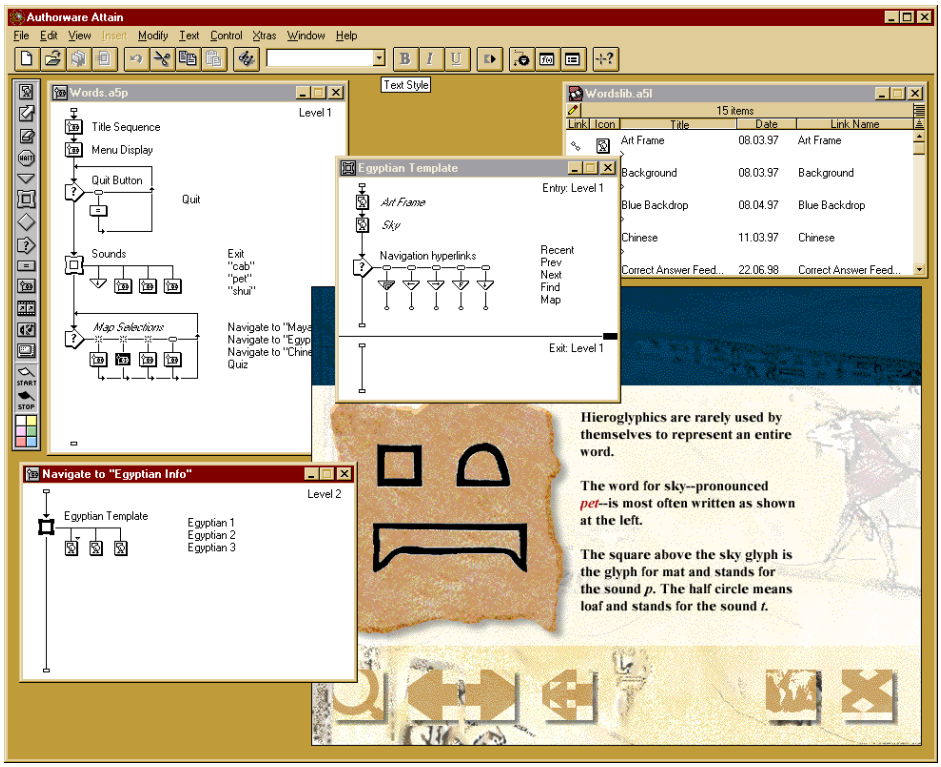

Abbildung 38: Beispielprojekt im Autorenmodus von Authorware

12 Buchverwaltungssystem

In Kap. 11 wurden grundlegende Technologien für die Entwicklung dynamischer elektronischer Bücher vorgestellt, soweit sie den Entwicklungsprozess in seiner Gesamtheit betreffen. Nun ist zu konkretisieren, wie sich das in Kap. 4.5 vorgestellte Architekturmodell realisieren lässt. Für die Basisarchitektur des elektronischen Buchs ergibt sich daraus folgende Konfiguration:

- Realisierung des Buchverwaltungssystems als Anwendung im Kontext eines Webservers und
- Aufbau des Buchbetrachtungssystems als an die Anwendung angepasster Webbrowser mit clientseitig implementierter Steuerung der Benutzerschnittstelle.

Zusätzlich müssen die im Vorfeld des Einsatzes eines dynamischen elektronischen Buchs erforderlichen Schritte diskutiert werden, vor allem die Transformation der Daten in ein für den Buchserver effizient zu verarbeitendes Format.

12.1 Middleware-Konzepte und Client-Server-Programmierung

Die Realisierung von Softwaresystemen als Client-Server-Lösungen oder, allgemeiner, als netzbasierte verteilte Anwendungen, macht aufgrund der Heterogenität der an solchen Anwendungen beteiligten technischen Infrastruktur eine vermittelnde Schicht zwischen Client und Server erforderlich, die i. d. R. als *Middleware* bezeichnet wird. Eine allgemein akzeptierte Definition dieses Begriffs liegt bisher nicht vor, vgl. BERNSTEIN 1996: 90, TRESCH 1996: 249. BERNSTEIN 1996 führt eine einfache etymologische Definition für *middleware* ein:

> To help some customers‘ heterogeneity and distribution problems, [...] vendors are offering distributed system services that have standard programming interfaces and protocols. These services are called *middleware services*, because they sit „in the middle,“ in a layer above the OS and networking software and below industry-specific applications. [BERNSTEIN 1996: 88]

Relativ unbestimmt bleibt auch die von BERG 1999: *Middleware Glossary* in Bezug auf Java-basierte Middleware vorgestellte Definition: “Middleware – Software that runs on a server, and acts as either an application processing gateway or a routing bridge between remote clients and data sources or other servers, or any combination of these.”
Man kann Middleware durch folgende Merkmale beschreiben (vgl. BERNSTEIN 1996: 90):

- Verfügbarkeit auf unterschiedlichen Plattformen,
- Realisierung als verteiltes System,
- Unterstützung standardisierter Protokolle (Internet-Netzwerkstandards) und
- Verfügbarkeit eines standardisierten API für die Entwicklung von Anwendungen auf der Basis einer bestimmten Middleware-Technologie.[109]

[109] Der Aspekt der Standardisierung spielt für die Architektur offener Hypertextsysteme eine große Rolle: WIIL & NÜRNBERG 1999: 432 betonen, dass sich offene Hypertextsysteme der zwei-

Hinsichtlich der mit Middleware zu lösenden Aufgaben unterscheidet TRESCH 1996: 250 folgende Dienstetypen:

- *Kommunikationsdienste* wie den elektronischen Datenaustausch oder E-Mail,
- *Systemdienste* wie Verschlüsselung oder Authentifizierung,
- *Informationsdienste* wie Verzeichnisserver oder Datenbankserver,
- *Ablaufkontrolldienste* wie Transaktionsverwaltung oder Scheduling-Mechanismen,
- *Präsentationsdienste* wie Anwendungen zur Maskengenerierung oder Hypermedia-Verknüpfungen und
- *Berechnungsdienste* wie Anwendungen zur Datenkonvertierung oder zur Sortierung.

Ohne hier bereits technische Realisierungen für einzelne Middleware-Lösungen zu betrachten, wird deutlich, dass sich der Begriff Middleware nicht durch ein spezifisches Anwendungsgebiet, sondern ausschließlich durch seine *Scharnierfunktion* für die Realisierung verteilter Anwendungen definieren lässt.

Sofern man elektronische Bücher nicht als statischen Wissensspeicher, sondern als ein Softwaresystem betrachtet, bei dessen Nutzung eine viele Softwarekomponenten zusammenwirken, stellt sich die Frage, welcher Bezug zwischen der Architektur elektronischer Bücher und dem Einsatz von Middleware besteht. Nachfolgend sollen einige Middleware-Konzepte eingeführt werden, da sie entweder bei der Realisierung des Buchverwaltungssystems als Implementierungswerkzeuge verwendet wurden oder für die Einordnung des Prototyps eine Rolle spielen:

- Das *common gateway interface* (CGI) als älteste Spezifikation für die Realisierung serverseitigen Programmzugriffs im World Wide Web,
- die *remote method invocation* (RMI) als Java-basierter Ansatz für die Umsetzung verteilter Objektsysteme,
- die *common object request broker architecture* (CORBA) als ein umfangreicher Referenzstandard für die sprachunabhängige Realisierung verteilter Objektsysteme und
- *Java Servlets* als serverseitige Erweiterung der Funktionalität eines Webservers um den Programmzugriff.

12.1.1 Common Gateway Interface

Das *common gateway interface* (CGI, vgl. DWIGHT, ERWIN et al. 1996) ist die am längsten bestehende und am weitesten genutzte Möglichkeit, im Rahmen des World Wide Web Programme auf einem Webserver ausführen zu lassen: CGI ist eine einfach strukturierte Schnittstelle für den Austausch von Daten zwischen Client (Browser) und Server (WWW-Server), die nur ein bestimmtes Format für die auszutauschenden Daten festlegt. Der wesentliche Vorteil von CGI ist, dass dieser Standard auf fast allen Plattformen und auf praktisch allen WWW-Servern eingesetzt werden kann. Er ist gleichzeitig nicht an eine bestimmte Programmiersprache gebunden: Serverseitige CGI-Programme können mit beliebigen Programmiersprachen erstellt werden. CGI-Skripte wurden im Rahmen des Referenzprojekts für die Realisierung der Schnittstelle zu einem Volltextrecherchesystem eingesetzt. Für die Implementierung des Buchservers für ein dynamisches elektronisches Buch wird statt serverseitiger CGI-Programme die Java-Servlet-Technologie verwendet.

---

ten Genration gerade dadurch auszeichnen, dass sie für ihre komponentenorientierte Architektur verstärkt auf standardisierte Technologien zurückgreifen. Dies gilt analog für dynamische elektronische Bücher.

12.1.2 Remote Method Invocation

Remote Method Invocation (RMI) bezeichnet ein im Kontext der Programmiersprache Java standardisiertes Verfahren für den Zugriff auf Objekte in verteilten Systemen. Man kann es als Fortführung des seit langem bekannten Prinzips der *remote procedure calls* (RPC) auffassen. Bei *Remote Method Invocation* kommuniziert der Client mit dem Server nicht über ein selbst definiertes Protokoll (wie bei der Kommunikation über Sockets[110]), sondern er benutzt direkt Methoden von Objekten, die auf anderen Rechnern in einer virtuellen Java-Maschine instantiiert sind. Das RMI-API des *Java Development Kit* (ab Version 1.1) bietet alle notwendigen Klassen für die Realisierung verteilter Objektsysteme. Die Anwendung von RMI setzt voraus, dass alle beteiligten Programme (Objekte) einer verteilten Anwendung von RMI *in Java* implementiert sind. Dies ist eine Einschränkung gegenüber der *Common Object Request Broker Architecture* (CORBA, s. u.), bei der die verteilten Objekte in verschiedenen Sprachen implementiert sein können, soweit diese über eine CORBA-Schnittstelle verfügen. Dies muss aber aufgrund der Plattformneutralität von Java kein Nachteil sein. Gegenüber der Verwendung von CORBA hat RMI den Vorteil, dass es wesentlich überschaubarer und einfacher zu implementieren ist und zudem keine weitere proprietäre Software benötigt wird (wie ein *Object Request Broker* (ORB) in CORBA). Die über die Standardentwicklungsumgebung hinaus benötigten Tools für den Einsatz von RMI (eine *registry*, in der über das Netz aktivierbare Objekte registriert werden und ein RMI-Compiler, der die für RMI-Objekte erforderliche Hilfsklassen automatisch generiert) sind im JDK enthalten.

Im Rahmen des Prototyps dynamischer elektronischer Bücher kann RMI dort zum Einsatz kommen, wo Dienste auf verschiedene Rechner verteilt werden und die Kommunikation mit einem Dienst nicht über ein Standardprotokoll wie HTTP oder ein proprietäres, selbst definiertes Protokoll, sondern direkt über den Zugriff auf Objekte und deren Methoden erfolgen soll.

12.1.3 Common Object Request Broker Architecture

CORBA ist ein sprachneutraler Industriestandard für verteilte Objektsysteme. Er wird seit Ende der 80er Jahre von der *Object Management Group* (OMG, vgl. *http://www.omg.org*) entwickelt.[111] RMI – wie oben dargestellt – ist zwar grundsätzlich CORBA-kompatibel, stellt aber eine Lösung für verteilte Objektsysteme dar, die nur in einer Sprache – Java – realisiert werden kann. Bei der Nutzung von CORBA bedient man sich einer sprachneutralen *Interface Definition Language* (IDL): In ihr kann man sprachunabhängig Schnittstellen für die zu realisierende Programmfunktionalität spezifizieren. Durch sog. *language mappings* der IDL für alle Sprachen, in denen CORBA-fähige Objekte entwickelt werden sollen, lassen sich IDL-Spezifikationen übersetzen. Die IDL von CORBA definiert dabei ein eigenes Objektmodell, das sich an die Möglichkeiten von C++ anlehnt.[112]

CORBA ist grundsätzlich sprach- und plattformneutral, d. h. es werden unterschiedliche Plattformen (Solaris, Win95/NT, HP-UX, OpenVMS etc.) und Sprachen (C, C++,

[110] Unter einem Socket versteht man einen Kommunikationsendpunkt eines Programms, über den ein Kommunikationskanal zwischen Rechnern aufgebaut werden kann.

[111] Auf dem Webserver der OMG finden sich umfassende Spezifikationen zur aktuellen Version 3 von CORBA, vgl. einführend hierzu VINOSKI 1998.

[112] Zur Einführung in CORBA vgl. ORFALI, HARKEY & EDWARDS 1998; eine ausführliche Beschreibung der CORBA-Entwicklung mit Java geben VOGEL & DUDDY 1998.

Ada, Java, Smalltalk etc.) unterstützt. Konkret kann das bedeuten, dass über CORBA aus einem unter Solaris laufenden Java-Programm Routinen eines C-Programms auf Win95/NT aufgerufen werden können. Die Voraussetzungen für diese Kommunikation zwischen verteilten Objekten sind

- eine Brücke zwischen einer *native Language* und CORBA (d. h. die *interface definition Language* und ein sprachspezifisches *Language binding* für jede Programmiersprache, in der für eine CORBA-Anwendung Programme implementiert werden sollen) und
- ein *Object Request Broker* (ORB), der die Kommunikation zwischen Objekten vermittelt.

Unterschiedliche ORBs kommunizieren über das *Internet Inter-ORB Protocol* (IIOP). Zugrunde gelegt sind jeweils die Netzwerkstandards der TCP/IP-Familie. Eine CORBA-Installation im Rahmen der *Object Management Architecture* der OMG kann aus folgenden vier Teilen bestehen:

- Der *ORB* als Software-Bus für kommunizierende Objekte,
- *CORBAServices*, die auf Systemebene zusätzliche Dienste zum ORB definieren (z. B. Sicherheitsdienste, Nameserver oder Transaktionen),
- *CORBAFacilities*, d. h. Dienste auf Anwendungsebene wie zusammengesetzte Dokumente und
- *Business Objects*, d. h. Beschreibungen tatsächlich existierender Objekte und Anwendungen.

Der ORB ist Voraussetzung für den Einsatz von CORBA. Er kann funktional unterschiedlich ausgestattet sein, im einfachsten Fall fehlen die oben genannten *Services*, *Facilities* und *Business Objects*, d. h. der ORB ist auf seine Basisfunktion als Software-Bus reduziert, wozu u. a. folgende Aktionen gehören:

- Objekte auf einem *remote host* finden und instantiieren,
- Parameter von einer Programmiersprache in eine andere übersetzen,
- Sicherheitsbeschränkungen überprüfen,
- Metadaten über Anwendungen/Objekte auf dem lokalen System nachschlagen und an einen anderen ORB weiterleiten,
- Methoden von Objekten auf einem remote host (*static* und *dynamic method invocation*) aufrufen,
- erforderliche, aber augenblicklich nicht instantiierte Objekte automatisch starten und
- *callback*-Methoden an die passenden lokalen Objekte weiterleiten.

CORBA stellt den wohl umfassendsten und anspruchsvollsten Standard für verteilte Objektsysteme dar, der vor allem für komplexe Businessanwendungen geeignet ist. Im Rahmen des hier vorgestellten Konzepts dynamischer elektronischer Bücher ist mit den verschiedenen Ebenen des Zugriffs auf Dienste und Komponenten der Aspekt verteilter Daten und Programme zwar ein wesentlicher Gesichtspunkt; auf den Einsatz von CORBA und den damit verbundenen deutlichen Entwicklungsmehraufwand konnte aber verzichtet werden, da die exemplarisch modellierten Dienste andere Standards wie das WebDAV-Protokoll (Speicherung von Daten auf einem *remote host*) vorsehen, mit dem einfacheren RMI-Protokoll arbeiten oder bereits verfügbare Dienste im World Wide Web kapseln, deren Kommunikationsprotokoll mit HTTP fest vorgegeben ist.
Unter dem Gesichtspunkt der Generalisierung des hier vorgestellten Ansatzes und seiner Integration in die Infrastruktur digitaler Bibliotheken könnte die Realisierung funktionaler

Komponenten mit CORBA aber ein interessanter Aspekt sein. Konzeptuell ließe sich eine CORBA-basierte Diensteintegration als Ersatz oder in Ergänzung von RMI- oder HTTP-basierten Diensten in das Architekturmodell für dynamische elektronische Bücher problemlos integrieren.

12.1.4 Java Servlets

Als letzen Typus einer Middleware-Technologie, sollen abschließend *Java Servlets* diskutiert werden. Java-Applets waren bei ihrer Einführung ein wichtiger Schritt in Richtung *active contents* auf der *Clientseite* im World Wide Web, da durch sie im Webbrowser Programme ausgeführt werden können. Sie haben inzwischen eine Entsprechung auf der Seite des Webservers gefunden: Unter Verwendung eines geeigneten Webservers kann man ein Klassenpaket verwenden, mit dem sich in Java entwickelte Programme, sog. Servlets, auf einem Webserver ausführen lassen. Die Unterstützung für Java Servlets ist mittlerweile für fast alle Webserver vorhanden; zur Entwicklung servletbasierter Anwendungen vgl. ausführlich ROßBACH & SCHREIBER 1999.

Die Standardanwendung eines WWW-Servers sieht die statische Verwaltung von Information im Dateisystem vor, d. h. es handelt sich i. d. R. um Dateien, die in einem Filesystem abgelegt sind und über den Webserver abgerufen werden können (Abbildung 39).

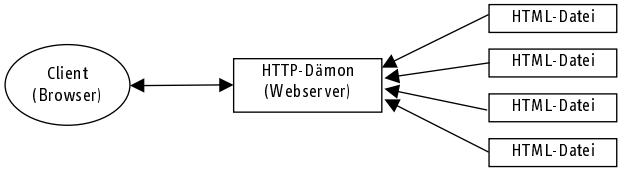

Abbildung 39: Funktionsprinzip eines Webservers

Auf diese Weise können keine Programme aufgerufen werden, deren Ergebnisse als Daten (Information) dem Benutzer (Client) übermittelt werden. Wünschenswert ist daher das in Abbildung 40 gezeigte Zugriffsmodell.

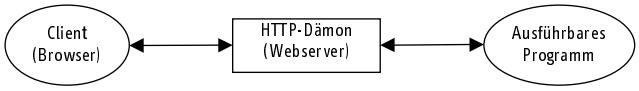

Abbildung 40: Ausführen von Programmen auf einem Webserver

Dies lässt sich – wie oben gezeigt – z. B. durch den Einsatz von CGI-Programmen erreichen. Die Möglichkeit, statt CGI-Programmen Java Servlets einzusetzen, bietet aber v. a. zwei Vorzüge:

- Ein einheitliches API für die Programmierung serverseitiger plattformunabhängiger Anwendungen
- keine Notwendigkeit, wie bei CGI, für jeden Aufruf eines Skripts oder Programms einen eigenen Prozess zu erzeugen und so den Server zu belasten.

Die Servlet-Architektur lässt sich so organisieren, dass Datentypen oder Adressen (URIs) jeweils auf bestimmte Servlets abgebildet werden. Auf diese Weise bleibt dem Benutzer der Einsatz eines Servlet verborgen, da nicht sichtbar ist, dass hinter dem URI nicht eine statische Seite, sondern ein Programm verborgen ist. Im Prototyp eines dynamischen elektronischen Buchs werden Adressen, die auf Dateien vom Typ `.XML` verweisen, durch den Buchserver als Servlet im Kontext eines Apache-Webservers verarbeitet (*JServ servlet engine*, vgl. *http://java.apache.org/jserv/*).

12.2 Aufbau

In Kap. 4.5.2 wurde der Aufbau des Buchservers konzeptuell entwickelt. Dies ist im Folgenden in technologischer Hinsicht zu präzisieren: Der Buchserver ist als Java-Servlet realisiert und setzt einen Webserver mit Servlet Engine voraus; im Rahmen dieser Arbeit wurde hierfür der Apache-Webserver (vgl. *http://www.apache.org*) ausgewählt. Maßgebliche Gründe für diese Entscheidung waren die weite Verbreitung dieses Webservers, die Tatsache, dass er als *open source*-Projekt im Quellcode vorliegt und auf unterschiedlichen Plattformen eingesetzt werden kann sowie die verfügbaren Zusatzmodule für diesen Webserver, die als Basis des Buchservers dienen:

- Die Servlet-Engine *JServ*, die die Einbindung und Verwaltung von Java-Servlets erlaubt,
- ein *WebDAV*-Modul für die benutzer- und gruppenbezogene Speicherung von Information und
- das *Cocoon Publishing Framework*, als ein Servlet zur Verarbeitung und Transformation von in XML kodierten Daten im Rahmen eines HTTP-Servers.

Als Protokoll für die Kommunikation zwischen Buchserver und Buchbetrachter kommt somit HTTP zum Einsatz. Der Buchserver erweitert das *Cocoon Publishing Framework*, das im Rahmen der Weiterentwicklung des Apache Web-Servers als *open source*-Projekt im Quellcode zur Verfügung steht. Es besitzt eine integrierte Architektur für die Verarbeitung und Transformation von XML-Dokumenten durch XSL/T. Die wesentlichen Verarbeitungskomponenten (XML-Parser, XSL/T-Transformationskomponente) können als gekapselte Module beliebig ausgetauscht werden (z. B. Auswahl eines bestimmten XML-Parsers, Wechsel der Transformations-Engine). Abbildung 41 ergänzt Abbildung 25 (Kap. 4.5.2.5) durch Angabe der für die verschiedenen Komponenten und Kommunikationsschnittstellen des Buchservers eingesetzten Technologien.

12.3 Speicherung, Transformation und Aufbereitung von Inhalten

Im Vorfeld der *Nutzung* des elektronischen Buchs als webbasiertes Informationssystem stehen im Fall des Referenzprojekts eine Vielzahl von Konvertierungs- und Aufbereitungsschritten für die Ausgangsdaten; dabei handelt es sich um

- die Erfassung neuer Materialien mit unterschiedlichen Werkzeugen für die einzelnen Medientypen,
- die Entwicklung von multimedialen Komponenten mit Autorensystemen bzw. als Java-Applets und
- um die Konvertierung bereits elektronisch vorliegender Daten aus proprietären Formaten (Textverarbeitungsprogramme etc.) in das XML-basierte Zielformat mit deklarativer Informationsauszeichnung.

Diese Schritte werden hier nicht weiter beschrieben. Die Vorverarbeitung ist hier nur insofern von Interesse, als sie auch XSL/T-basierte Transformationen der Ausgangsdaten umfasst. Gleichzeitig ist mit der Frage der Datenaufbereitung auch das Problem der Datenspeicherung angesprochen.

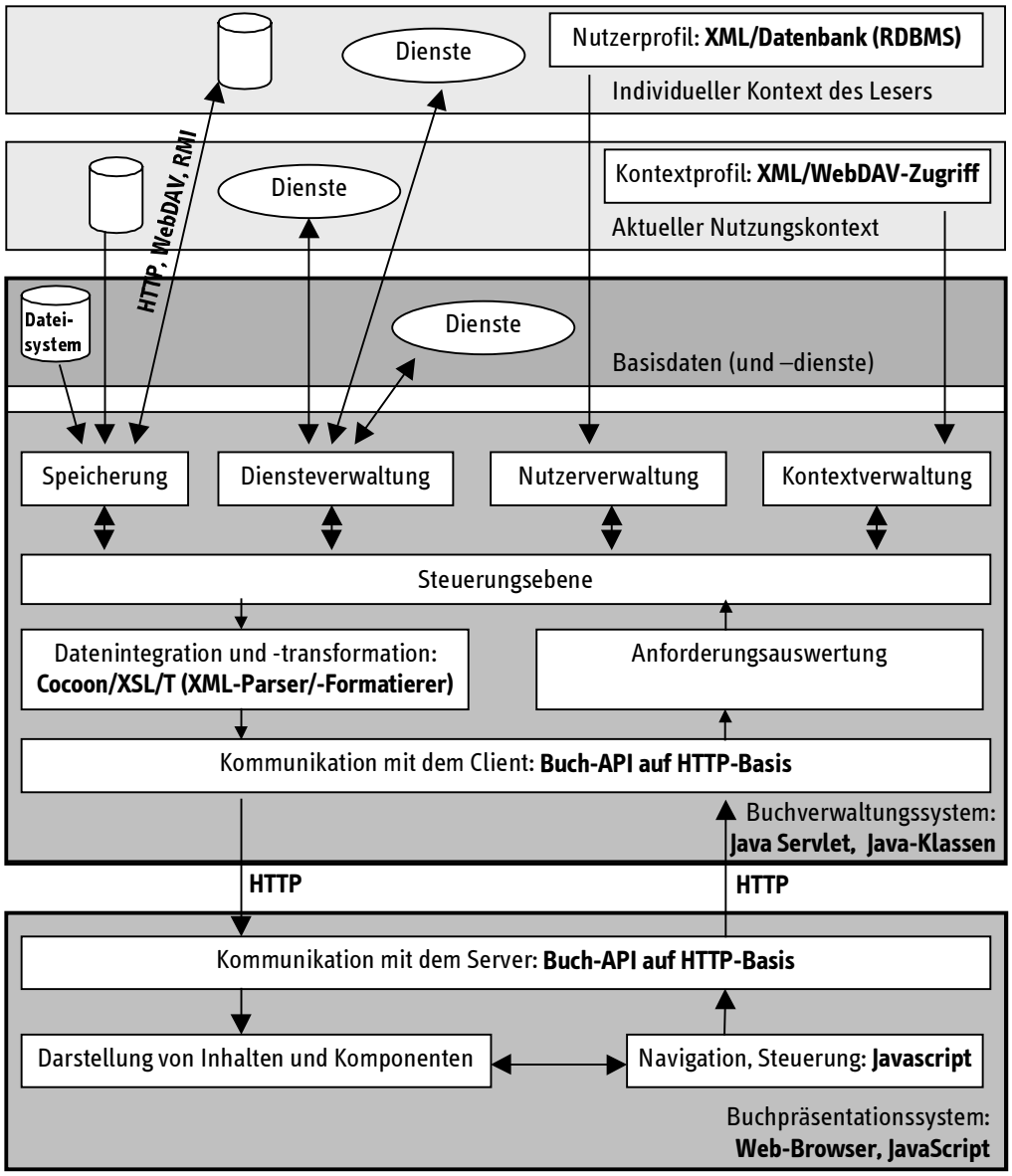

Abbildung 41: Struktur des Buchservers und verwendete Technologien

12.3.1 Speicherungstechnologien

In vielen Fällen werden elektronische Bücher nicht mit Hilfe dedizierter Speichersoftware wie Datenbanken verwaltet, sondern über die Verwendung einfacher Betriebssystemdienste (Dateisystem) oder proprietärer Dateiformate, die alle das Buch betreffende Informationen (Inhalt und Metainformation sowie Layoutdaten) umfassen (z. B. PDF oder die Speicherungsformate von Multimedia-Autorensystemen). Für die Speicherung bieten sich drei grundlegende Alternativen an:

- Speicherung der Daten im Dateisystem,
- Speicherung mit Hilfe einer Datenbank (*relational data base management systems* (RDBMS), *object oriented database management systems* (OODBMS), *multimedia data base management systems* (MMDBMS)) und
- Verwendung dedizierter Dokumentenmanagement-Systeme (*Content Management-Systeme*, CMS).

Ein Merkmal dynamischer elektronischer Bücher ist die Heterogenität der Inhalte (Primärdaten) und der mit dem elektronischen Buch verbundenen Metainformation (Metadaten i. e. S., Verwaltungsinformation des Buchservers). Bei der Speicherung der Daten eines dynamischen elektronischen Buchs sind folgende Kriterien von Belang:

- die unterschiedlichen Datenformate (XML-Dateien des primären Buchinhalts, über Entitäten oder XML-Marken eingebundene Inhalte wie Bilder oder eingebettete Komponenten (Applets, Shockwave-Dateien), Metadaten, Verwaltungsinformation),
- die Granularität der Speicherungseinheiten, d. h. die Frage, ob und wie feinkörnig die XML-Struktur auf Speicherungseinheiten abgebildet wird (Inhalte einzelner Dateien, Abbildung der Dokumentgrammatik auf ein Datenbankmodell),
- die Speicherung der im elektronischen Buch enthaltenen Hypertextverknüpfungen und
- der Zugriff auf die gespeicherten Einheiten (Adressierung, Abfragesprache).

12.3.1.1 Dateisystem

Die einfachste Lösung für die Speicherung des Dokumentenbestands elektronischer Bücher ist die unmittelbare Verwendung des Dateisystems des Betriebssystems;[113] dies ist im World Wide Web der Regelfall, d. h. Webserver liefern Dokumente durch Zugriff auf die im Dateisystem gespeicherten HTML-Dokumente. Diese Lösung hat ungeachtet des Vorteils, dass keine zusätzliche Software für die Speicherung erforderlich ist und die Dokumente ohne weiteres zur Verfügung stehen, eine Reihe von Nachteilen:

- Es kann nicht unmittelbar auf die XML-/HTML-Binnenstruktur zugegriffen werden, sondern diese muss von einem Parser nach dem Laden des Dokuments ermittelt werden.
- Eine dynamische Generierung unterschiedlicher Inhalte ist nur mit großen Aufwand (durch Nachverarbeitung mit Hilfe eines Parsers) möglich.
- Bei einer feinkörnigen Modellierung der Darstellungseinheiten kann sich wegen des geringen Darstellungsumfangs elektronischer Dokumente eine hohe Zahl an Dokumenten ergeben, im Referenzprojekt knapp 2000 HTML-Dokumente (Aufteilung von annähernd 300 Druckseiten auf ca. 1000 Präsentationseinheiten, zusätzlich ca. 800 Seiten mit den verschiedenen Komponententypen (Abbildungen, Animationen, Simulationen etc.)). Durch eine hohe Dokumentanzahl kann sich der Zugriff (z. B. über eine CD-ROM) deutlich verlangsamen, die Verwaltung bringt einen hohen Arbeitsaufwand mit sich.

Die Hauptnachteile liegen auf der Seite der Produktion, Modifikation und Lieferung der elektronischen Dokumente eines Buches, weniger auf der Benutzerseite, da bei Verwendung eines WWW-Browsers als Viewing-System die Form der Speicherung für den Be-

[113] Vgl. dazu ABITEBOUL, BUNEMAN & SUCIU 1999: 171 ff., die die typischen Speicherungstechniken für XML-kodierte Daten diskutieren, allerdings den Schwerpunkt weniger auf *narrative* unstrukturierte Texte, sondern auf datenbanknahe Informationsmodelle (Datensätze) legen.

nutzer ohne Belang ist. Sowohl für das Referenzprojekt als auch für den Prototyp eines dynamischen elektronischen Buchs wurde ungeachtet dieser Nachteile eine dateisystembezogene Speicherung verwendet, da diese Lösung die Distribution des elektronischen Buchs als CD-ROM vereinfacht: Es ist keine lokale Datenbankschnittstelle als *back end* für den Webbrowser erforderlich. Da bei dieser Lösung der Webbrowser auch offline, also ohne Zugriff auf das Internet und Webserver, in der Lage sein muss, das elektronische Buch darzustellen, wäre eine datenbankbasierte Lösung nur schwer zu realisieren gewesen (vgl. BOLES et al. 1998C). Eine datenbankbasierte Lösung wäre für den Prototyp eines dynamischen elektronischen Buchs wünschenswert, aufgrund der überschaubaren Datenmenge konnte darauf aber verzichtet werden.

12.3.1.2 Abbildung auf Datenbankmodelle

Da die heute gebräuchlichen relationalen Datenbanksysteme die Relationenalgebra in erster Normalform realisieren (NF^1), sind sie gut geeignet, große, einfach strukturierte Datenmengen („Massendaten“) effizient zu verwalten. Eine Abbildung der komplexen Struktur eines SGML- oder XML-Dokuments auf ein Relationenschema ist hingegen nicht ohne weiteres möglich: Durch die hierarchische Struktur von XML-Dokumentgrammatiken ist die direkte Abbildung auf ein Relationenschema in erster Normalform nicht direkt möglich und führt zu einer sehr hohen Anzahl von Relationen (vgl. ABITEBOUL, BUNEMAN & SUCIU 1999: 172 ff.).[114]

Ungeachtet der prinzipiellen Probleme bei der Datenmodellierung für SGML-DTDs in relationalen Datenbanksystemen spielen sie bei der dynamischen Generierung elektronischer Publikationen im Kontext von WWW-Informationssystemen eine wesentliche Rolle: Überall dort, wo große Faktendatenbestände einzubinden sind, bietet sich die Verwendung von RDBMS an, z. B. Produktkataloge, Messdatenbestände, Katalogsysteme, Artikelverwaltungen etc. In der Regel wird die Datenbank im Rahmen einer Client-Server-Anwendung über eine auf dem (WWW-)Server angesiedelte Zwischenschicht eingebunden (z. B. CGI-Programm, Applikationsserver wie ColdFusion, Java-Servlets etc., vgl. BENN & GRINGER 1998).

Für die Abbildung des in einer SGML- oder XML-DTD enthaltenen Elementmodells auf ein Speicherungsschema sind objektorientierte Datenbanksysteme besser geeignet als relationale DBMS, da sich die hierarchische Elementstruktur auf das Klassensystem der Datenbank abbilden lässt: Jedes Element der DTD wird auf eine *Klasse* abgebildet, elementare Datentypen von SGML/XML (`#PCDATA`) entsprechend in primitive Typen des Datenbanksystems übersetzt (z. B. `String`). Eine ausführliche Beschreibung der Modellierung von Speicherung und Zugriff auf SGML- und XML-basierter Dokumente geben ABITEBOUL et al. 1997 und ABITEBOUL, BUNEMAN & SUCIU 1999: 176 ff. Objektorientierte Datenbanken bilden daher die Basistechnologie für XML-fähige Datenbanksysteme und datenbankbasierte *Content Management-Systeme*.

[114] Ein Beispiel einer solchen Modellierung ist das Datenmodell für das in WOLFF & QUASTHOFF 1999 diskutierte SGML- bzw. XML-basierte multilinguale Lexikonsystem. Vgl. auch GROSSMAN et al. 1997, die ein Beispiel für die Anpassung eines relationalen Datenbanksystems für die Speicherung in SGML kodierter Texte geben, und SENGUPTA & DILLON 1997, die Erfordernisse an die datenbankbasierte Speicherung von SGML-Dokumenten aufzeigen.

12.3.1.3 Content Management-Systeme

Einen weiteren Typus von Speicherungssystemen für elektronischen Publikationen stellen *Content Management-Systeme* dar, die darauf zugeschnitten sind, Publikationsdaten elektronisch zu verwalten und als *back end* für Publikationssysteme z. B. im Internet zu dienen. Das Content Management-System übernimmt nicht nur die Rolle des Datenspeichers, sondern stellt auch Funktionalität bereit, um verschiedene Fassungen (Versionen) einer Publikation persistent zu speichern. Bei dieser Systemkategorie spielen die Integration offener Standards wie SGML und XML einerseits, die Verwendung adäquater Datenbanktechnologie wie objektorientierter Datenbanken andererseits eine zentrale Rolle: Einheitliche und standardisierte Strukturierungssprachen, wie sie SGML und XML darstellen, bilden die Voraussetzung für eine systematische Abbildung von Publikationsdaten auf ein entsprechendes Datenbankmodell.

Ein Beispiel für ein solches System ist die *Poet Content Management Suite* (vgl. *http://www.poet.de*), die auf der Basis des objekt-orientierten Datenbanksystems *Poet* einen Speicherungs- und Verwaltungsdienst für SGML- und XML-basierte Dokumente anbietet. Das System nutzt die Tatsache, dass sich die hierarchischen Strukturen einer SGML- oder XML-DTD in ein Objektmodell einer objektorientierten Datenbank überführen lassen: Hat der Anwender eines solchen Systems eine für seinen Zweck geeignete DTD definiert, kann das Content Management-System die entsprechenden Klassen zur Speicherung der Publikationsdaten generieren. Dem Benutzer stehen dann zur dynamischen Selektion von Inhalten die entsprechenden Manipulationsverfahren der zugrunde liegenden Datenbank zur Verfügung, z. B. die *Object Query Language* (OQL).

Im Rahmen des *Referenzprojekts* wurden solche Content Management-Systeme lediglich evaluiert, aber nicht als Speicherungsdienst eingesetzt, da sie für eine CD-ROM-basierte Distribution nicht geeignet erschienen bzw. der damit verbundene technische wie ökonomische Aufwand aufgrund des beschränkten Datenumfangs nicht zu rechtfertigen ist (vgl. BOLES et al. 1998C). Das Modell eines dynamischen elektronischen Buchs, wie es in Kap. 4.5 entwickelt wird, ist aber grundsätzlich für unterschiedliche Ausprägungen eines Speicherungs- und Manipulationsdienstes offen. Insofern kann an die Stelle der dateisystemorientierten Speicherung ein Content Management-System als *back end* treten. Wie noch zu zeigen sein wird, ergeben sich zusätzliche Anforderungen an die Funktionalität solcher Verwaltungssysteme, z. B. durch die Einführung von Kooperationsprotokollen wie WebDAV (s. u. Kap. 12.6.2), die ein verteiltes Arbeiten mit Publikationsdaten ermöglichen bzw. standardisieren.

12.3.2 Datentransformation

Kap. 12.3.1 hat unterschiedliche Ansätze zur Speicherung der Inhalte elektronischer Bücher diskutiert; in Kap. 7 wurden verschiedene Sprachen zur Transformation deklarativ kodierter Daten elektronischer Bücher vorgestellt. Nachfolgend sollen die erforderlichen Schritte für die Aufbereitung und Transformation elektronischer Bücher in Weiterführung der im Referenzprojekt erarbeiteten Prozesskette konkretisiert werden. Das Ziel ist dabei, Inhalte und Komponenten so zu transformieren, dass

1. sie in ein präsentationsorientiertes Format umgewandelt werden können, das sich in einem an der Buchmetapher orientierten Buchpräsentationssystem darstellen lässt,
2. die Transformation effizient erfolgen kann, d. h. der Verarbeitungsaufwand für eine Darstellungseinheit bei Anforderung durch den Benutzer gering ist und dabei

3. eine Prozesskette entsteht, die keine oder nur eine geringe Nachbearbeitung der Daten erfordert.

Die erste Anforderung ist eine Konsequenz aus der für das Referenzprojekt getroffenen Designentscheidungen: Danach liegt dem elektronischen Buch die Buchmetapher als Präsentationsform zugrunde und bedingt eine seitengerechte Aufteilung der Daten, um im Client auf Scrolling als Navigationsmodus im Text verzichten zu können (vgl. dazu ausführlich unten Kap. 12.5.1). Die zweite Anforderung ergibt sich aus der Notwendigkeit einer effizienten Aufbereitung der Daten für die Präsentation. Im Idealfall könnte der Buchserver die Gesamtdaten des Buchs im Speicher vorhalten und daraus jeweils den angeforderten Abschnitt errechnen; aufgrund der aufwendigen Berechnung einer XSL-Transformation (Parsen der XML-Daten, Umwandlung des Dokumentbaums durch die Transformationsanweisungen eines XSL/T-*style sheets*) erscheint es aber sinnvoller, die jeweils vom Server zu generierende Informationseinheit möglichst klein zu halten und auf je eine darzustellende Buchseite zu begrenzen.

Der Ausgangspunkt für die Transformation ist das Vorliegen der durch die in Kap. 4.2 und Kap. 10 beschriebenen Schritte in XML kodierten Daten.[115] Eine Transformation ist erforderlich,

1. da man davon ausgehen muss, dass innerhalb des Dokumentbaums die jeweils kleinste *inhaltsbezogene* Beschreibungskategorie auf Textebene umfangreicher sein kann als die kleinste Präsentationseinheit (Bildschirmseite),
2. i. d. R. Inhalte elektronischer Bücher in wenigen, umfangreichen Makroeinheiten vorliegen und
3. in XML kodierte Daten nur bei Vorliegen entsprechender Darstellungsinformation dargestellt werden können (d. h. durch Verarbeitung auf der Basis eines *style sheet*).

Vor allem der erste Punkt ist zu diskutieren: Auf der Basis des im Referenzprojekt entwickelten Buchbetrachtungssystem wird die Buchmetapher für die Gestaltung des Clienten herangezogen und bedingt eine statische Aufteilung der Buchinhalte in einzelne Präsentationseinheiten (Seiten). Die These, dass die kleinste inhaltliche Einheit eines elektronischen Buchs (z. B. die Beschreibung eines Versuchsaufbaus, die Beschreibung der allgemeinen Grundlagen für ein Experiment) in ihrem Umfang durch das Medium bedingt sei und die Autoren sich den Gegebenheiten des Mediums anpassen müssen, ist nicht plausibel: Es ist zwar in gewissem Umfang möglich, die Textstrukturierung an ein Medium anzupassen, grundsätzlich dürfte aber ein Primat der zu beschreibenden Inhalte vorliegen (vgl. LESCH 1999: 144). Am Beispiel des Referenzprojekts kann man wie folgt argumentieren: Die Herleitung eines physikalischen Phänomens als Grundlage für die Durchführung und das Verständnis eines bestimmten Experiments lässt sich mit Blick auf die Präsentation im elektronischen Medium nicht beliebig modifizieren und vor allem nicht auf einen durch das Medium bestimmten Darstellungsumfang reduzieren. Im Referenzprojekt musste zwar der Text des Ausgangswerks weitgehend unverändert bleiben musste, die Möglichkeit einer grundlegenden inhaltlichen Modifikation des Texts für das elektronische Medium (mit Ausnahme der multimedialen Ergänzungen) bestand also nicht. Dies ist aber der Regelfall für die Produktion elektronischer Bücher, wenn man die in der elektronischen Bibliothek der Informatik, InterDoc, oder die für hardwarebasierte elektronische Buchbetrachter produzierten Werke betrachtet.

[115] Das Vorfeld, d. h. die Umwandlung der Daten aus ggf. proprietären Datenformaten, mit denen die Autoren (Hyper-)Texte erstellt haben, bleibt hier außer Betracht; geeignete Konvertierungswerkzeuge oder Exportroutinen sind vorausgesetzt.

Gilt aber dieser Primat des Inhalts und soll eine Aufteilung in feste Präsentationseinheiten erfolgen, so ist zusätzlich zur Abbildung von XML auf ein Präsentationsformat auch eine layoutbezogene Aufteilung der Daten erforderlich. Die hierfür denkbaren technischen Realisierungsalternativen lassen sich nach folgenden Kriterien unterscheiden:

1. Die *Speicherungsform* auf der Serverseite (dateisystemorientierte bzw. datenbankorientierte Speicherung).
2. Das *Zielformat* auf der Clientseite, insb. die Unterscheidung zwischen der direkten Darstellung von XML durch *style sheets* und der Umwandlung in ein Präsentationsformat wie HTML.
3. Die *Zuordnung* der Transformation zum Buchserver zum Buchbetrachtungssystem.
4. Die Zuordnung der Aufteilung der Daten (und der erforderlichen Anpassung hinsichtlich *Adressierung*, Integrität der Hypertextverknüpfungen etc.) zu Buchserver, Buchbetrachtungssystem bzw. dem Vorfeld der Verwaltung durch den Buchserver.

Die Prozesskette des gewählten Verfahrens zeigt Abbildung 42. In ihr wird von den weiteren Komponenten (Benutzerverwaltung, Dienstekoordination etc.) abstrahiert; eine (XML-)Datenbank als Alternative zur dateisystembezogenen Speicherung ist in der Abbildung zusätzlich berücksichtigt, konnte im Rahmen dieser Arbeit aber noch nicht realisiert werden. Das Verfahren arbeitet in zwei Stufen:

1. Die in XML aufbereiteten Daten werden durch ein XSL-/T-*style sheet* in *Präsentationseinheiten* geteilt. Dabei werden die Hypertextverknüpfungen entsprechend angepasst (konsistente Adressierung auch nach der Aufteilung auf mehrere präsentationsbezogene Speicherungseinheiten). Die technische Realisierung erfolgt mit Hilfe des SAXON-XSL/T-Interpreters, der nicht nur eine vollständige Implementierung der XSL/T-Spezifikation umfasst, sondern durch Nutzung des Erweiterungsmechanismus von XSL/T die Integration externer Methoden (durch dynamisches Laden von Java-Klassen) und die Erzeugung einer Mehrzahl von Ausgabedateien zulässt.
2. Auf Anforderung durch den Client hin werden in einem zweiten Schritt die XML-Daten für das elektronische Buch auf HTML/CSS als Präsentationsformat abgebildet und in die Seitenschemata, die der Server für die verschiedenen Darstellungsfenster (Buch, gesonderte Komponentendarstellung) eingebettet. Diese Funktionalität ist durch das *Cocoon Publishing Framework* realisiert, das in modifizierter Form Grundlage der Implementierung des Buchservers ist.

Dabei wird von den erforderlichen Vorverarbeitungsschritten, insbesondere der Konvertierung von Ausgangsdaten aus proprietären Formaten (Texteditoren etc.) abstrahiert. Der erste Schritt ist zeitlich von der tatsächlichen Nutzung des elektronischen Buchs entkoppelt, d. h. er muss als Vorverarbeitungsschritt abgeschlossen sein, bevor das elektronische Buch genutzt werden kann. Die Prozesskette zeichnet sich durch folgende Merkmale aus:

- In den verschiedenen Stufen kommt unterschiedliche Steuerungsinformation in Form von DTDs und XSL-*style sheets* zum Einsatz. Dies ist eine für die Verarbeitung von XML- oder SGML-basierten Publikationen typische Herangehensweise.[116]

[116] Vgl. das Ablaufmodell für den Dokumentationsprozess unter Einsatz verschiedener DTDs, wie ihn MALER & EL ANDALOUSSI 1996: Kap. 3, bes. 67 ff. schildern und vergleichbare Ansätze wie das von RUTLEDGE et al. 1998: 193 ff. geschilderte Prozessmodell für die SGML-/DSSSL-basierte Generierung von SMIL-Dokumenten.

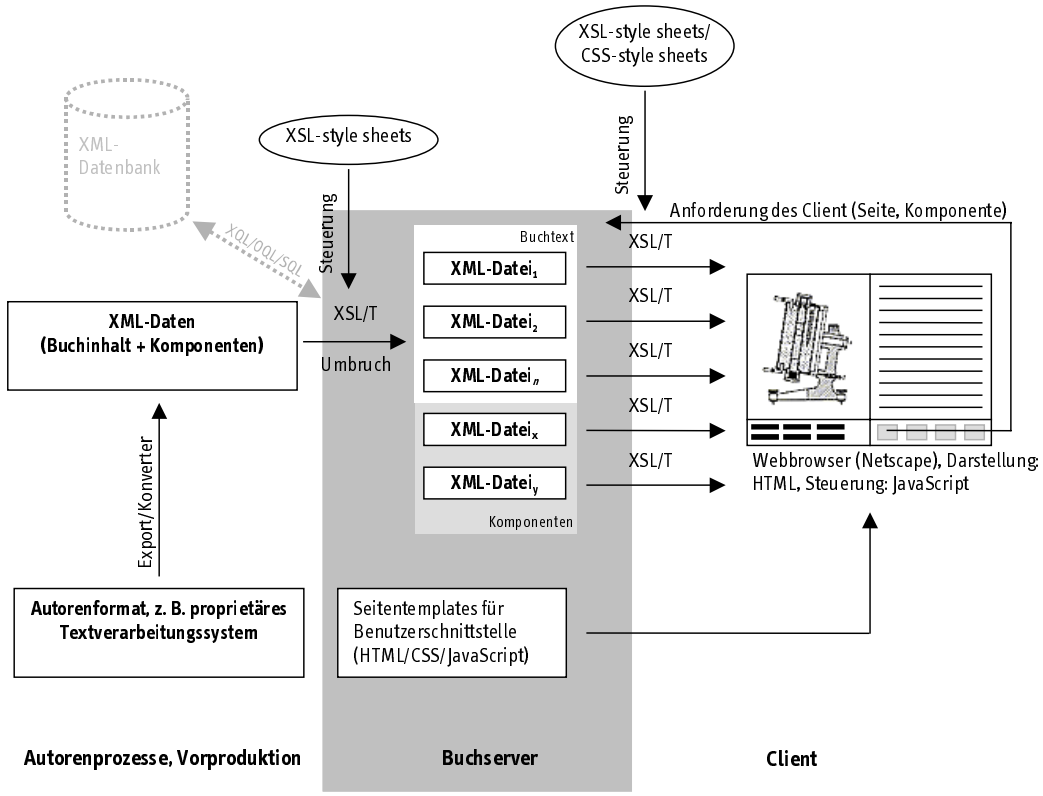

Abbildung 42: Prozesskette der Transformationsschritte bei der Datenaufbereitung für das elektronische Buch

- Der erste Transformationsschritt kann bei Einsatz datenbankbasierter Speicherung bzw. bei Verwendung eines *Content Management-System* entfallen; an die Stelle der Aufteilung auf Speicherungseinheiten (Dateien) tritt dann die Datenselektion bei Anforderung einer Inhaltseinheit durch den Benutzer unter Anwendung der entsprechenden Abfragesprache (z. B. die *Object Query Language* (OQL) im Rahmen des *Poet CMS* oder die *XML Query Language* (XQL) bei XML-Datenbanken).
- Beide Transformationsschritte ließen sich verschmelzen, wenn – bei Beibehaltung der seitenorientierten Darstellung – das Buchbetrachtungssystem über einen Layoutalgorithmus verfügt, der zur Laufzeit die Aufteilung auf die Buchseiten berechnet und die Seiten entsprechend darstellt.

Die Aufteilung der Inhalte auf Präsentationseinheiten erfolgt dabei mit Hilfe eines im Referenzprojekt entwickelten Werkzeugs, das eine Abschätzung der Darstellungsgröße der Buchinhalte ermittelt und eine entsprechende Separierung der Präsentationseinheiten (*offline*, als Vorverarbeitungschritt) vornimmt.

12.4 Informationskodierung

Die Konzeption eines XML-basierten elektronischen Buchs legt es nahe, nicht nur die in Kap. 4.2 diskutierten Ebenen der Inhaltsauszeichnung durch deklarative Standards zu realisieren, sondern auch im Rahmen des Buchservers benötigte Steuerungsinformation durch XML zu kodieren. Gegenüber einem nicht-transparenten binären Format oder der ausschließlichen Speicherung in Datenbanken und dem Zugriff mit Hilfe von Methoden der einzelnen Systemkomponenten, hat dies den Vorteil, dass Information für die Modifi-

kation und den Zugriff durch Buchadministrator oder Benutzer „verständlich“ repräsentiert werden und XML leicht in unterschiedliche Speicherungsformate überführt werden kann. Damit kann die Administration der Dienste und ihrer Zuordnung zu Inhaltseinheiten des dynamischen elektronischen Buchs im einfachsten Fall durch Editieren einer entsprechenden XML-Datei erfolgen und für den Prototyp erübrigt sich die Entwicklung visueller Editoren. Im einzelnen betrifft dies

- die Spezifikation von Diensten,
- die Verwaltung von benutzerbezogenen Informationen und
- die Merkmale der Nutzungskontexte.

Details der Speicherungsformate werden in den folgenden Abschnitten jeweils bei der Diskussion der jeweiligen Komponente des Buchservers erörtert, Dokumentgrammatiken für die Verwaltungsinformationen finden sich im Anhang (Kap. 17.1).

12.5 Steuerungs- und Verwaltungsmodule des Buchservers

Die Steuerungskompomponente des Buchservers überwacht das Zusammenspiel der verschiedenen Funktionseinheiten des Buchservers und wertet die Kommunikation zwischen Buchbetrachtungssystem und Buchserver aus. Dazu gehören

- die *Anforderung von Inhaltseinheiten* durch den Benutzer durch
  - Navigation (Blättern im Buch),
  - Browsing (Verfolgen von Verknüpfungen),
  - aus dem Hierarchiebrowser oder der Rechercheschnittstelle und
  - die Anforderung von Komponenten und Diensten,
- die *Überwachung* der aktuellen Diensteinfrastruktur während der Nutzung (generisch verfügbare Dienste, individuelle Dienste des Benutzers, Nutzungskontext) sowie die Aktivierung von Diensten (Starten von Dienstobjekten) und
- die Schnittstelle zum Speicherungsdienst für die Inhaltsbestandteile des Buchs, d. h. zum Zugriff auf das Dateisystem eines Webservers.

12.5.1 Diensteverwaltung

Die Integration von Diensten in elektronische Bücher setzt ein Diensteverwaltungskonzept voraus, über das Dienste auf der Basis einer deklarativen Spezifikation und nach Verfügbarkeit in die Nutzung des elektronischen Buchs integriert werden können. Die Modellierung von Diensten und ihre Verwaltung ist integraler Bestandteil des Buchverwaltungssystems. Sie orientiert sich an dem für das *Open Hypermedia Protocol* vorgeschlagenen Operationsmodell für die Diensteaktivierung (vgl. MILLARD, REICH & DAVIS 1999: 39). Dabei werden folgende Aufgaben erfüllt:

- Aufbau und Verwaltung der aktuellen Dienstekoordinationsmatrix, die die Zuordnung von Diensten zu Inhaltselementen des elektronischen Buchs festhält,
- Zuordnung der Dienste zu den Inhaltselementen bzw. der generischen Diensteliste bei der Transformation der Buchinhalte vom Ausgangsformat in das Präsentationsformat.
- Aktivierung der Dienste bei Anforderung durch den Benutzer.

Der erste Punkt beinhaltet das Einlesen verfügbarer Dienste aus den drei Ebenen

- allgemein verfügbare Dienste (alle Benutzer),
- Dienste, die für einen gruppenbezogenen Kontext definiert sind und

- individuelle Dienste, die durch einen einzelnen Benutzer spezifiziert sind, insbesondere für die individuelle Annotation der Buchinhalte und die lokale Weiterverarbeitung von Daten.

Die tatsächliche Zuordnung der Dienste erfolgt durch den XML-Parser als Teil des Buchservers, der für jede zu präsentierende Buchseite prüft, ob in ihr Inhaltselemente enthalten sind, für die in der Dienstekoordinationsmatrix ein Dienst spezifiziert ist, und den Dienst über die Generierung entsprechender Verknüpfungen anbindet (HTML-Anker im Präsentationsformat). Beispiele für Dienste und ihre Aktivierung bei der Nutzung des elektronischen Buchs finden sich unten in Kap. 14.2.

12.5.2 Benutzerverwaltung

Neben der Verwaltung der Dienste ist für ein dynamisches elektronisches Buch eine Benutzerverwaltung aus unterschiedlichen Gründen erforderlich:

- Die Identifikation des einzelnen Benutzers erlaubt die Realisierung benutzerbezogener Dienste und ist Grundlage für die Individualisierung des elektronischen Buchs für seinen Benutzer, z. B. durch Speicherung von Daten und die Angabe zusätzlicher Dienste, die den Buchinhalten zugeordnet sein sollen.
- Publikationsinhalte sind in der Regel durch das Urheberrecht geschützt und nicht generell zugänglich; die Identifikation des Benutzers dient der Zuordnung von Nutzungsrechten (Lizenzen) und ggf. auch der Abrechnung von über das elektronische Buch in Anspruch genommenen Diensten (z. B. kostenpflichtige Informationserschließungsdienste wie bibliographische Online-Datenbanken). An die Stelle der Zuordnung eines Individuums kann die Identifikation eines berechtigten Rechners (IP-Nummer) oder einer zugelassenen Nutzungsdomäne treten.
- Durch Identifikation des Benutzers ist die Zuordnung zu einem Gruppenkontext möglich, wenn für das Buch in einem konkreten Einsatzszenario gruppenbezogene Inhalte (z. B. zusätzliche Texte oder Animationen) oder Dienste (z. B. Kommunikationsdienste für die Abwicklung von Übungsaufgaben etc.) ermittelt werden sollen.

Für die Realisierung einer Benutzerverwaltung bieten sich folgende Möglichkeiten:

1. Die einfachste Lösung ist die Verwendung der von der Funktionalität eines Webservers angebotenen Autorisierungsverfahren, z. B. *access control lists* für den vom elektronischen Buch belegten Bereich des Dateisystems des Webservers, kodiert als Konfigurationsanweisungen der Webserververwaltung. Diese scheiden in diesem Kontext aus, da sie sich auf die Identifikation berechtigter Benutzer (oder Benutzergruppen, Rechner, Domänen) hinsichtlich der im Dateisystem des Webservers vorgehaltenen Daten beschränken, hier aber eine genauere Identifikation für die Zuordnung von Inhalten und Diensten erforderlich ist.
2. Eine weitere Lösung ist die Verwaltung entsprechender Zugangslisten als Datei, auf die die Benutzerverwaltung des Buchservers Zugriff hat. Diese Variante ist schlecht skalierbar, da eine sequentielle Suche in einer Benutzerdatei mit hoher Eintragszahl aufwendig ist (und zusätzlich bei XML-kodierten Daten der Aufwand für das Parsen der Datenstruktur hinzukommt).
3. Die dritte Variante ist die Speicherung über eine (relationale) Datenbank; diese Variante erlaubt die Verwaltung größerer Benutzerzahlen, bringt aber den Nachteil einer deutlichen Ausweitung der Infrastruktur des Buchservers mit sich.

Für den Prototyp des dynamischen elektronischen Buchs wurde die letzte Variante gewählt, d. h. die Benutzerverwaltungskomponente des elektronischen Buchs verfügt über eine (JDBC-)Datenbankschnittstelle, über die die Benutzerdaten abgelegt, ermittelt und aktualisiert werden. In der Benutzerschnittstelle dient ein entsprechendes Formular der initialen Eingabe der Benutzerdaten sowie der Autorisierung bei Beginn jeder neuen Sitzung. Die benutzerbezogenen Daten werden darüber hinaus bei der Aktivierung und Zuordnung von Diensten sowie der Speicherung benutzerbezogener Interaktionsdaten benötigt. Als Austauschformat wird wie für die anderen Steuerungsinformationen des Buchservers XML verwendet. Es ist darauf hinzuweisen, dass die Benutzerverwaltung bei Integration des elektronischen Buchs in die Infrastruktur einer digitalen Bibliothek von Komponenten der digitalen Bibliothek übernommen werden sollte.

12.5.3 Kontextverwaltung

Die Kontextverwaltung dient der Integration gruppenbezogener Komponenten und Dienste in das elektronische Buch, wie sie in Kap. 4.4.1.5 eingeführt wurden und für die in Kap. 14.2.5 Beispiele angegeben werden. Sie hat eine Scharnierfunktion zwischen einem Nutzungskontext und seinen Teilnehmern, den Inhalten des Buchs und den durch den Kontext bereitgestellten Komponenten und Diensten und trennt konzeptuell das dynamische elektronische Buch und seine Inhalte von seiner Einbindung in gruppenbezogene Nutzungskontexte. Sie muss die folgenden beiden Aktivitäten ermöglichen:

- die Überprüfung, ob ein bestimmter Kontext für den individuellen Leser zur Verfügung steht und
- der Zugang zum Kontext selbst, d. h. die Bereitstellung der entsprechenden Dienste und Komponenten.

Die Kontextverwaltung als Teil des Buchservers stellt diese Funktionen bereit; für den Zugriff auf Kontextinformation wird das WebDAV-Protokoll verwendet (vgl. unten Kap. 12.6), das den Schreib- und Lesezugriff nicht nur auf die Kontextdefinition, sondern auch auf mit einem Kontext verbundene Ressourcen erlaubt. In der hier gewählten Lösung sind einige vereinfachende Annahmen vorgenommen worden:

- Der individuelle Nutzer muss sich einem Kontext durch Angabe einer Kontextadresse zuordnen.[117]
- Kontextverwalter kann jeder Benutzer des elektronischen Buchs sein, der eine Kontextdefinition auf einem WebDAV-Server bereitstellt.
- Die Kontextdefinition besteht aus einer Adresse, unter der zusätzliche Komponenten für das elektronische Buch verfügbar sind, aus einer Diensteliste sowie den Zugangsrestriktionen (benutzerbezogene und domänenbasierte Autorisierung).

12.6 Kommunikation zwischen Client und Server

Abgesehen von der technischen Grundlage des Hypertext Transfer Protocol (HTTP), das für die Kommunikation zwischen Client und Server verwendet wird, und das die Syntax

[117] In weitergehender Betrachtung des Kontextkonzepts könnte man an die Implementierung geeigneter Suchroutinen denken, die für ein elektronisches Buch Nutzungskontexte ermitteln; dies ist aber eine generische Funktion, die im Rahmen einer digitalen Bibliothek oder eines Telelearning-Systems zu verwirklichen wäre.

für die Kommunikation vorgibt, müssen die verschiedenen Anforderungstypen des Clients einheitlich modelliert sein. Dazu gehören

- die Anforderung der jeweils nächsten Präsentationseinheit aus der aktuellen Leseposition (Blättern vor/zurück),
- die Anforderung einer Inhaltseinheit aus Index, Rechercheschnittstelle oder Hierarchiebrowser (absolute Seitenadresse),
- die Traversierung einer Verknüpfung zur Anzeige im Kontextfenster,
- der Aufruf einer Funktionskomponente des Buchs (Hierarchiebrowser, Retrievalschnittstelle, Hilfe),
- die Aktivierung einer im Kontextfenster eingebetteten oder in separatem Fenster anzuzeigenden Komponente,
- die Anforderung eines Dienstes, der einer Inhaltskomponente zugeordnet ist bzw. die generische Anforderung aus der Diensteliste auf Buchebene und
- die Übermittlung dienstebezogener Daten (Speicherung, Recherche).

Für die Kodierung der Anforderungstypen ist jeweils die Kennzeichnung des Anfragetyps sowie ggf. die Übermittlung einer Attribut-Wert-Liste erforderlich (z. B. Liste von Suchbegriffen, zu speichernde Daten). Dies erfolgt mit Hilfe der *query syntax* des URL-Standards (vgl. KHARE 1999A, 1999B), d. h. Anfragetyp und Anfrageattribute werden an die XML-Adresse angehängt und vom Buchserver analysiert. Liegt keine Erweiterung der Adresse vor, so interpretiert der Buchserver die Adresse unmittelbar und liefert die referenzierte Ressource zurück.

Zusätzlich muss die Möglichkeit bestehen, aus eingebetteten Komponenten mit dem Buchserver bzw. mit Diensten zu kommunizieren. Für die als Java-Applets realisierten Komponenten kann dazu ebenfalls eine HTTP-Verbindung zum Buchserver und die URL-Syntax verwendet werden.

Für die Realisierung eines dynamischen elektronischen Buchs als verteilte Client-Server-Anwendung, wie sie in Kap. 4.5 entwickelt wurde, sind geeignete Netzwerkprotokolle für die Übertragung der Information erforderlich. Nachfolgend soll die Infrastruktur eines TCP/IP-basierten Netzwerks vorausgesetzt werden (untere Schichten im OSI-Modell der Netzwerkarchitektur; Adressierung von Hosts im Internet über IP-Adressen und das *domain name system* etc.); zudem sollen nicht alle für das Internet entwickelten Dienste und ihre zugehörigen Netzwerkprotokolle (z. B. Telnet, FTP) untersucht werden (vgl. dazu KESSLER & SHEPARD 1997 [RFC 2151]), sondern nur die Protokolle, die für die Übermittlung von Daten im World Wide Web eine Rolle spielen. Zugleich gilt es zu unterscheiden zwischen der

- Datenübermittlung zwischen Client und Server, d. h. einem Webbrowser und einem Webserver sowie
- der Übertragung von Daten über die in einem dynamischen elektronischen Buch verfügbaren Dienste, soweit es sich um Daten im Sinne von durch XML-basierte Standards kodierte Information handelt.

Diese Einschränkungen grenzen den Gegenstandsbereich auf das *Hypertext Transfer Protocol* und seine Weiterentwicklungen ein. Nachfolgend werden daher die beiden für den Buchserver verwendeten Netzwerkprotokolle HTTP und WebDAV eingeführt.

12.6.1 HyperText Transfer Protocol

Das *Hypertext Transfer Protocol* (vgl. BERNERS-LEE, FIELDING & FRYSTYK 1996 [RFC 1945], FIELDING et al. 1999 [RFC 2616]) ist als Protokoll für den Austausch von Information zwischen Webbrowser und Webserver das „Rückgrat“ des World Wide Web. HTTP ist ein Protokoll auf der Anwendungsebene im Sinne des OSI-Netzwerkmodells und basiert auf einem einfachen Anfrage-Antwort-Schema. Es sind sowohl Anfragen des Client als auch Antworten des Servers in *Nachrichtenkopf* und *Nachrichtenkörper* gegliedert. Der Nachrichtenkopf enthält den Typ der Anfrage bzw. der Antwort sowie Metainformation, z. B. über den Client oder die vom Server zurückgelieferte Antwort. I. d. R wird für jede Abfolge von Anfrage und Antwort eine neue Netzwerkverbindung aufgebaut (statusloses Protokoll), wie die nachfolgende Abbildung schematisch darzustellen versucht:[118]

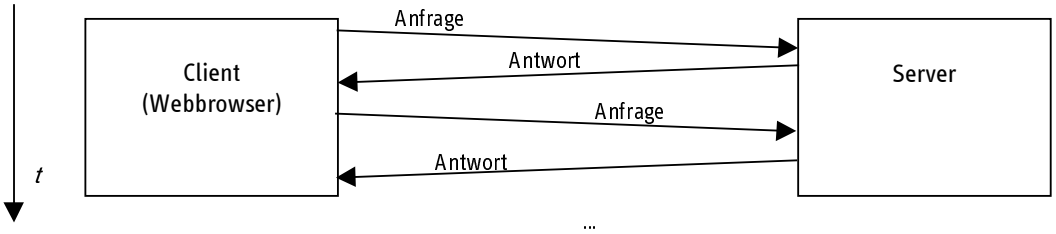

Abbildung 43: Aufbaubau von Netzwerkverbindungen für Anfragen und Antworten in HTTP

Der aktuelle Standard HTTP 1.1 definiert folgende Anfragetypen (FIELDING 1999: Kap. 5.1.1, Kap. 9):

Anfragemethode	*Bedeutung*	*Bemerkung*
GET	Lädt eine Ressource (Adressierung über einen URI), z. B. GET http://www.test.org/test.html HTTP/1.1	–
HEAD	Fordert bezüglich einer Ressource lediglich den Kopf der Antwortnachricht, aber nicht die in der Ressource enthaltenen Daten selbst an.	–
POST	Dient der Übermittlung von Daten vom Client auf den Server, die als bezüglich des angegebenen URIs untergeordnet interpretiert werden.	Anwendung z. B. bei der Verarbeitung der Inhalte von HTML-Formularen.
OPTIONS	Ermittelt Informationen über verfügbare Kommunikationsoptionen bezüglich der angeforderten Ressource.	neu in HTTP 1.1
PUT	Dient der Speicherung der im Körper der Anfragenachricht enthaltenen Daten unter dem angegebenen URI.	neu in HTTP 1.1
DELETE	Weist den Server an, die durch den URI der Anfrage identifizierte Ressource auf dem Server zu löschen.	neu in HTTP 1.1
TRACE	Dient zu Diagnosezwecken, zeigt dem Client, welche Informationen auf dem Server angekommen sind.	neu in HTTP 1.1
CONNECT	Reservierter Methodenname für die Funktionalität von Proxy-Servern.	neu in HTTP 1.1
EXTENSION-METHOD	Erweiterungsmechanismus für neue Anfragetypen.	neu in HTTP 1.1

Tabelle 53: Anfragemethoden in HTTP 1.1

[118] Die aktuelle Spezifikation HTTP 1.1 lässt auch *persistente* Netzwerkverbindungen für eine Mehrzahl von Anfragen und Antworten zu, wird aber nicht durchgängig unterstützt, vgl. FIELDING et al. 1999: Kap. 8.1.

Die Antworten des Servers übermitteln im Kopf der Antwort einen Statuscode, der anzeigt, in welcher Weise der Server in der Lage ist, die Anfrage zu beantworten (z. B. `200 OK` (Anfrage war erfolgreich), `202 ACCEPCTED` (Anfrage wurde angenommen, aber noch nicht verarbeitet), `403 FORBIDDEN` (Zugriff auf die angeforderte Ressource ist nicht erlaubt) oder `404 NOT FOUND` (angeforderte Ressource konnte nicht gefunden werden)), sowie Meta-Informationen über die übermittelten Daten als Folge von Attribut-Wert-Paaren (`Content-Length`, `Content-Type`, `Last-Modified`).

Im durch eine Leerzeile vom Kopf abgetrennten Körper der Antwortnachricht werden die angeforderten Daten übermittelt, d. h. der Inhalt der durch den *request-URI* identifizierten Ressource, z. B. eine HTML-Datei oder die Daten einer Abbildung.

An einer Weiterentwicklung von HTTP wird von Arbeitsgruppen des W3C seit geraumer Zeit gearbeitet (HTTP-NG, vgl. JANSSEN, FRYSTYK NIELSEN & SPREITZER 1998, KOPROWSKI 1998, JANSSEN 1999). Wesentliche Änderungen betreffen die direkte Unterstützung von Middlewareprotokollen wie RMI oder CORBA durch HTTP sowie die logische Gliederung des Protokolls in eine Transport-, eine Nachrichten- und eine Anwendungsschicht (JANSSEN 1999: 70 f., bes. Abb. 1). Die Funktion von HTTP als Datenprotokoll des World Wide Web („*the classic web application*“ – TCWA) ist dann nur noch eine unter vielen möglichen Anwendungsformen für HTTP-NG. Da HTTP-NG bisher nur prototypischen Status hat und in gängigen Webbrowsern noch nicht implementiert ist (vgl. JANSSEN 1999: 72 ff.), kann es im Rahmen dieser Arbeit allerdings nicht berücksichtigt werden, auch wenn gerade die Unterstützung von Middlewareprotokollen bei der Realisierung von Diensten in dynamischen elektronischen Büchern ein wünschenswertes Merkmal ist, das die Kapselung solcher Protokolle durch HTTP-Anfragetypen überflüssig machen könnte (vgl. unten Kap. 14.2).

Das World Wide Web – und HTTP als sein Netzwerkprotokoll – hat sich zu einer universellen Plattform für die Realisierung von Software-Anwendungen entwickelt. Insofern ist es von Bedeutung, Schwachpunkte von HTTP aufzuzeigen. Hinsichtlich der Entwicklung komplexer Informationsdienste wie dynamischer elektronischer Bücher im World Wide Web stellen FIELDING et al. 1998: 85 fest, welche Ziele sich mit Hilfe der verfügbaren Webstandards und -Technologien auf der Basis von HTTP *nicht* erreichen lassen:

- linking all artifacts
- flexible interaction models and hypermedia services
- distributed annotation
- visibility of artifacts over time
- distributed authoring
- distributed coordination and change management.

Einige dieser Desiderata an WWW-Protokolle, vor allem die Möglichkeit, Groupware-Funktionalität für die Bearbeitung von Dokumenten im World Wide Web einzuführen, versucht das IETF WebDAV-Protokoll (*distributed authoring and versioning*, vgl. SLEIN et al. 1997, WHITEHEAD & WIGGINS 1998) auszugleichen. Die damit mögliche Funktionalität z. B. der dynamischen Erstellung von Annotationen oder die Bearbeitung von Dokumenten durch eine verteilte Arbeitsgruppe, ist für dynamische elektronische Bücher relevant.

12.6.2 WebDav – Distributed Authoring and Versioning

Der Ausgangspunkt für den Entwurf eines Kollaborationsprotokolls für das World Wide Web sind die oben genannten Nachteile von HTTP in Verbindung mit den Erfordernissen

komplexer Publikationsprozesse. Im Bereich des elektronischen Publizierens sind verteilte Autorenprozesse die Regel, bei denen eine Vielzahl Beteiligter (Autor, Lektor, Gutachter, Indexierer) an unterschiedlichen Orten und mit unterschiedlichen Funktionen an der Erstellung einer Publikation beteiligt sind. Es fehlt bisher an standardisierten Protokollen und Werkzeugen, die solche Prozesse unterstützen könnten.[119] Im Rahmen dieser Arbeit spielen Autorenprozesse für dynamische elektronische Bücher nur am Rande eine Rolle, da Konzept und Funktionsweise dynamischer elektronischer Bücher aus der Perspektive des Nutzers entwickelt werden. Es ist auf der Basis des in Kap. 4.5 entwickelten Modells dynamischer Bücher allerdings deutlich geworden, dass von der Nutzerseite her Unterstützung für Speicherungs- und Kooperationsdienste, die über die Funktionalität von HTTP hinausgehen, ein Element der Realisierung von Diensten im Rahmen eines dynamischen elektronischen Buchs sein können (individuelle Annotationsdienste, Speicherungsdienst im Rahmen von Arbeitsgruppen eines Nutzungskontexts).

Sind die Möglichkeit zur Versionierung von Ressourcen und ihrer verteilten Bearbeitung (*editing*) und Verwaltung (*document management*) im World Wide Web die wichtigsten konzeptuellen Anforderungen an das WebDAV-Protokoll, so lässt sich ihre konkrete technische Realisierung am ehesten mit den Möglichkeiten der Dateimanipulation in einem Mehrbenutzerbetriebssystem wie UNIX vergleichen: Das WebDAV-Protokoll spezifiziert Zugriffsmöglichkeiten auf Ressourcen im World Wide Web, die es erlauben,

- Ressourcen durch einen Benutzer in einem WebDAV-basierten Kontext aus- und einzuchecken und zu signalisieren, dass sie sich in Bearbeitung befinden,
- über das World Wide Web geladene Ressourcen zu bearbeiten,
- Informationen über einzelne Ressourcen oder Ressourcensammlungen (*container*, z. B. ein Teil eines Dateibaums eines WWW-Servers) zu ermitteln,
- Ressourcen und Behälter zu erstellen,
- Zugriffsrechte und Attribute von Ressourcen und Container zu verwalten,
- Ressourcen und Container zu kopieren, zu verschieben und zu löschen und
- Operationen über zusammengesetzten Dokumenten durchzuführen, die aus einer Mehrzahl einzelner Ressourcen bestehen (vgl. NEUMCKE 1999: 20 ff., WHITEHEAD & WIGGINS 1998: 35 ff., SLEIN et al. 1998: Kap. 5).

Die technologische Realisierung von WebDAV basiert auf bereits eingeführten Technologien, zum einen

- auf der Erweiterung von HTTP um zusätzliche Methoden durch den seit HTTP 1.1 verfügbaren *extension mechanism*,
- zum anderen auf der Verwendung von XML zur Beschreibung von Dokument- bzw. Ressourceneigenschaften (GOLAND 1999: Kap. 1).

Zusätzlich wird das Konzept einer Sammlung (*collection*) von Ressourcen eingeführt, die als Teil einer mit Hilfe eines URI adressierbaren Ressourcenbaums zusammengehörende und durch entsprechende WebDAV-Attribute gekennzeichnete Einzelressourcen enthält. WebDAV bietet die Möglichkeit, alle Bestandteile einer Sammlung gemeinsam zu manipulieren (z. B. für die Bearbeitung zu sperren, zu kopieren, zu löschen etc.). Im Einzelnen führt WebDAV die in der folgenden Tabelle zusammengestellten Erweiterungen der HTTP-Methoden ein (GOLAND 1999: Kap. 8):

[119] Es sei allerdings darauf hingewiesen, dass die Zusammenarbeit bei der Dokumenterstellung ein in der Hypertextliteratur seit langem untersuchtes Thema ist, vgl. oben Kap. 2.2.1.7.

WebDAV-Methode	*Erläuterung*
LOCK	Sperrt eine Ressource bzw. eine Sammlung
UNLOCK	Hebt die Sperre einer Ressource auf
PROPFIND	Ermittelt Attribute einer Ressource oder Ressourcensammlung
PROPPATCH	Setzt oder entfernt Attribute von Ressourcen oder Ressourcensammlungen
MKCOL	Erzeugt eine neue Sammlung auf dem Server
DELETE	Löscht eine Ressource oder Ressourcensammlung
COPY	Erzeugt eine Kopie einer Ressource oder Ressourcensammlung unter einer in der Anfrage enthaltenen neuen Adresse (URI)
MOVE	Verschiebt eine Ressource oder Ressourcensammlung an eine neue Adresse (URI)

Tabelle 54: Zusätzliche HTTP-Methoden, die durch WebDAV eingeführt werden

Das nachfolgende Beispiel zeigt die Anforderung eines Schreibschutzes für ein Dokument auf einem WebDAV-basierten WWW-Server (nach GOLAND 1999: Kap. 8.10.8):

```
# Anfrage:
LOCK /Teubner/Buch/chap01/sub01/p00.htm HTTP/1.1
Host: aspra9.informatik.uni-leipzig.de
Timeout: Infinite, Second-4100000000
Content-Type: text/xml; charset="utf-8"
Content-Length: 2335
Authorization: Digest username="wolff",
   realm="wolff@informatik.uni-leipzig.de ", nonce="...",
   uri="/Teubner/Buch/chap01/sub01/p00.htm ",
   response="...", opaque="..."

<?xml version="1.0" encoding="utf-8" ?>
<D:lockinfo xmlns:D='DAV:'>
   <D:lockscope><D:exclusive/></D:lockscope>
   <D:locktype><D:write/></D:locktype>
   <D:owner><D:href>http://www.informatik.uni-leipzig.de/ifi/abteilungen/asv/wolff.html </D:href>
   </D:owner>
</D:lockinfo>


#Antwort:
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: 2335
<?xml version="1.0" encoding="utf-8" ?>
<D:prop xmlns:D="DAV:">
   <D:lockdiscovery>
      <D:activelock>
         <D:locktype><D:write/></D:locktype>
         <D:lockscope><D:exclusive/></D:lockscope>
         <D:depth>Infinity</D:depth>
         <D:owner><D:href>http://www.informatik.uni-leipzig.de/ifi/abteilungen/asv/wolff.html</D:href>
         </D:owner>
         <D:timeout>Second-604800</D:timeout>
         <D:locktoken><D:href>opaquelocktoken:e71d4fae-5dec-22d6-fea5-00a0c91e6be4</D:href>
         </D:locktoken>
      </D:activelock>
   </D:lockdiscovery>
</D:prop>
```

Codebeispiel 81: Beispiel einer Anfrage-Antwort-Abfolge in WebDAV

Im Körper von Anfrage und Antwort werden in diesem Beispiel in XML kodierte Attribute für den Zugriffsschutz angegeben. Die Antwort des Servers beschreibt, wie lange der Schreibschutz gilt und übermittelt dem Benutzer ein *lock token*, das die Zugriffssperre eindeutig identifiziert.[120]

Die Bearbeitung einer Ressource erfordert im Vergleich mit der Anforderung von Daten über HTTP eine Mehrzahl WebDAV-basierter Interaktionsschritte zwischen Client und Server. Die nachfolgende Abbildung (vgl. WHITEHEAD & WIGGINS 1998: 35, Abb. 1., SUSSMANN 1999: 78, Abb. 2) zeigt schematisch den Ablauf einer solchen Interaktion, bestehend aus

- der Anforderung einer Sperre für eine Ressource (LOCK),
- der Ermittlung von Eigenschaften über die Ressource (PROPFIND),
- dem Laden der Ressource auf den Client (GET),
- der Bearbeitung der Ressource durch den Benutzer,
- dem Zurückspeichern der Ressource (PUT) und schließlich
- dem Aufheben der Zugriffssperre (UNLOCK).

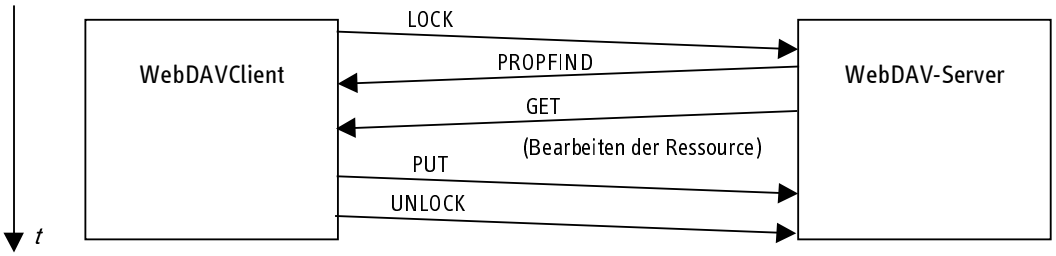

Abbildung 44: Interaktionsablauf in WebDAV

In Abbildung 44 geben die Pfeile nicht wie in Abbildung 43 einzelne Anfragen und Antworten wieder, sondern stellen die wesentliche Richtung des Informationsflusses eines Anfrage-Antwort-Paars dar.

Nicht alle in den Entwürfen des WebDAV-Standards enthaltenen Konzepte sind bereits voll ausgearbeitet oder gar implementiert; die Fragestellung der verteilten Bearbeitung und Versionierung sowie der Zusammenführung von Dokumentteilen, die durch mehrere Autoren bearbeitet werden, sind noch näher zu spezifizieren. Für die Grundfunktionalität der WebDav-Methoden liegen aber bereits eine Reihe von Implementierungen vor. Dazu gehören

- Erweiterungen von Webservern um WebDAV-Module (z. B. für den Apache Webserver (Modul *Apache mod_dav*)),
- eigenständige Browser, die es als WebDAV-Clienten erlauben, auf WebDav-Servern Informationen über Ressourcen zu ermitteln und diese zu bearbeiten (z. B. der als Java-Applikation realisierte *DAVExplorer*, vgl. *http://www.ics.uci.edu/~webdav/*).
- Programmier-APIs, mit deren Hilfe sich WebDAV-Funktionalität in Client-Programme integrieren lässt (z. B. das *DAV4J*-Paket von IBM).

Einen Überblick zu WebDAV-Implementierungen und Fragen ihrer Interoperabilität gibt WHITEHEAD 1999. Die Unterstützung von WebDAV wird von zahlreichen Softwareher-

[120] Man beachte, dass die Spezifikation der Attribute als Metadaten zu Ressourcen im World Wide Web „direkt“ durch eine für WebDAV entworfene XML-DTD erfolgen und nicht etwa auf der Basis von RDF. Man kann die Eigenschaftsspezifikation in WebDAV deshalb als einen weiteren Metadatenstandard betrachten, vgl. GOLAND 1999: Kap. 4.

stellern angekündigt, so von Microsoft für die Integration von WebDAV-Funktionen in Betriebssysteme und Anwendungspaketen (WHITEHEAD & WIGGINS 1998: 490, SUSSMANN 1999: 79).

WebDAV stellt eine Möglichkeit dar, Groupware-Funktionalität im World Wide Web zu realisieren und von an einzelne Rechner gebundenen Konzepten wie lokalen Dateisystemen zu abstrahieren. In diesem Sinn lässt sich WebDAV in die allgemeine Entwicklung zu stärker netzbasierten Anwendungen einordnen. Im Rahmen dieser Arbeit findet WebDAV nicht als generelles Kommunikationsprotokoll Anwendung, sondern wird gezielt für einzelne Dienste im Rahmen eines dynamischen elektronischen Buchs eingesetzt. Gleichzeitig sind aber die Potentiale deutlich, die WebDAV für die Autorenprozesse bei der *Erstellung* elektronischer Bücher spielen kann. Es erscheint geeignet, proprietäre oder nur prototypisch realisierte Arbeitswerkzeuge für das verteilte Editieren wie das BSCW-System (vgl. APPELT & MAMBREY 1999) durch standardisierte Bearbeitungsmechanismen zu ersetzen oder bisher proprietär realisierte Funktionalität kommerzieller Werkzeuge (z. B. Lotus Notes) zu standardisieren, wie das ähnlich bereits mit der Einführung elementarer Internet-Standards und -Dienste (z. B. FTP, HTTP, HTML) der Fall gewesen ist.

13 Buchbetrachtungssystem

Neben dem Buchserver ist das Buchbetrachtungssystem als für die Präsentation des elektronischen Buchs angepasster Webbrowser die zweite wesentliche technische Komponente, die hier zu diskutieren ist. Seine Realisierung für den Prototyp eines dynamischen elektronischen Buchs umfasst die Problembereiche

- Auswahl der Betrachtungssoftware,
- gestalterische und ergonomische Anforderungen an elektronische Bücher und
- die Gestaltung der Benutzerschnittstelle des elektronischen Buchs.

13.1 Betrachtungssoftware

Die Entscheidung, elektronische Bücher auf der Basis von Standards des World Wide Web zu entwickeln, zieht die Auswahl eines geeigneten Betrachtungssystems nach sich, das folgende Kriterien erfüllen muss:

- Programmierbarkeit für die Steuerung der Benutzerschnittstelle (dynamische Fensteraktivierung, Aufbau eines Hierarchiebrowsers, Kommunikation mit dem Server),
- Möglichkeit der Einbettung der interaktiven Komponenten und
- Darstellungsmöglichkeit für XML-/HTML-kodierte Daten (unter Anwendung von *style sheets*).

Es bieten sich folgende Alternativen an:

- Auswahl eines Standardbrowsers (Microsoft Internet-Explorer, Netscape Communicator),
- Verwenden eines Werkzeugs mit entsprechenden Importfunktionen, bzw. der Möglichkeit, einen HTML-Viewer als Komponente einzubetten (z. B. als Ergänzung zu Multimedia-Autorensystemen wie Macromedia Director) und
- Eigenentwicklung auf der Basis HTML-Viewerbibliotheken, wie sie z. B. das Swing-Paket der Java 2-Plattform enthält[121].

Im Referenzprojekt wurde auf der Basis einer Evaluierung die erste Variante (Netscape Communicator ab V. 4.0.4) gewählt,

- da die umfangreiche Funktionalität der Standardbrowser durch die erwähnten Alternativen nicht voll abgedeckt wird (z. B. JavaScript-Unterstützung),
- durch die Verwendung frei verfügbarer Standardsoftware die Distribution vereinfacht wird und
- der Entwicklungsaufwand für die dritte Variante unabsehbar hoch wäre.

[121] Das Paket javax.swing.text.html enthält sowohl Klassen für die Realisierung von HTML-Editoren (u. a. HTMLEditorKit, HTMLEditorKit.HyperlinkEvent) als auch Klassen, die HTML-Dokumente und ihre Strukturbeschreibung kapseln (HTMLDocument, DTD, DTDConstants, HTML), vgl. CHAN 1999: 341 f., 379 ff.

Wie die Diskussion webbasierter Informationssysteme in Kap. 2.2.4 deutlich gemacht hat, eignen sich Webbrowser als Client-Software für viele Typen von Informationssystemen und sind eine geeignete Basis für die Realisierung der Benutzerschnittstelle eines elektronischen Buchs.

13.2 Ergonomische Aspekte multimedialer Bücher

Die Frage nach der benutzerfreundlichen Gestaltung elektronischer Bücher führt unterschiedliche wissenschaftliche und gestalterische Entwicklungslinien in Informatik, Psychologie und Kunst bzw. Design zusammen:

- Die Software-Ergonomie als interdisziplinäre Wissenschaft, die sich mit den Fragen der an die Arbeitssituation angepassten und benutzerfreundlichen Gestaltung von Softwareprodukten befasst (vgl. grundlegend zur ergonomischen Gestaltung NORMAN 1990, zur Software-Ergonomie HERCZEG 1994, EBERLEH, OBERQUELLE & OPPERMANN 1994, BALZERT 1996: 451 ff., SHNEIDERMAN 1998.),
- gestalterische Ansätze aus dem Mediendesign und der Informationsvisualisierung (vgl. TUFTE 1983, 1990, 1997, WURMAN 1996, JACOBSON 1999 und den Sammelband von CARD, MACKINLAY & SHNEIDERMAN 1999),
- Erkenntnisse der traditionellen Typographie und ihrer Anwendung auf elektronische Texte sowie (DILLON 1994, BRINGHURST 1996) und
- struktur- und designbezogene Ansätze, die auf die Besonderheiten des Publizierens im World Wide Web ausgerichtet sind (ROSENFELD & MORVILLE 1998, LYNCH & HORTON 1999, THISSEN 2000).

Die Gestaltung elektronischer Bücher stellt insofern eine besondere Herausforderung dar, als gestalterische Anforderungen auf mehren Ebenen miteinander vereint werden müssen:

- Die Gestaltung des Buchpräsentationssystems und seiner Strukturkomponenten (Steuerungselemente, Fensteraufbau und Funktionskomponenten),
- die Gestaltung des Basisdatenbestands des Buchs,
- die Auswahl von Strategien der Informationsvisualisierung für eingebettete Multimediakomponenten und
- der Entwurf eingebetteter Benutzerschnittstellen für die interaktiven Komponenten.

Voraussetzung für eine systematische Entwicklung ist die Definition einer einheitlichen Grundfunktionalität für das User Interface. Dazu gehören:

- Die generische Navigation innerhalb des Viewers (Blättern, Strukturnavigation über konzeptuelle Erschließung/Inhaltsverzeichnis),
- Steuerungsfunktionen für Zusatzelemente und
- die visuelle Typisierung von Hyperlinks.

Web Design-Werkzeuge lassen in gewissem Umfang die standardisierte Definition von Interfaces für elektronische Publikationen zu (Web-Authoring-Systems wie *MS-Frontpage*, Schemata in Multimedia-Autorensystemen wie *Asymetrix Toolbook*, dedizierte EP-Tools wie *BookWeb*, vgl. *http://www.orthogon.de/doc/products/bookweb.htm*), schöpfen aber nur bedingt die größeren Freiheitsgrade elektronischer Publikationen aus, wie sie in dem hier diskutierten Kontext entstehen. Die Frage einheitlicher Interfacestandards (und ihre Kodierung nach einheitlichen Vorgaben) ist aber nicht nur für die einfachere Nutzung elektronischer Publikationen von Bedeutung, sondern auch für deren Integration auf der Distributionsebene.

Folgt man der These, dass die Interaktions- und Navigationsmöglichkeiten einer Publikation von den generischen (und einfachen) Möglichkeiten der Viewerumgebung abstrahieren sollen (vgl. NIELSEN 1999), so stellt man fest, dass die existierenden Detaillösungen entweder – wie die gerade erwähnten schema-orientierten Lösungen der Standardwerkzeuge – die Gestaltungsmöglichkeiten stark einengen oder – wie das Beispiel *Multimediales Physikalisches Praktikum* – als Resultat Einzellösungen mit individuellen Vorzügen (Seitenkoppelung, Kontextualisierung von Zusatzinformation) sind, die auf idiosynkratischen Implementierungen beruhen und einen hohen Generierungs- und Administrationsaufwand bewirken. Eine Optimierung könnte auf der Verwendung der SGML-Standards zur Definition der Benutzerschnittstelle bzw. ihrer Funktionalität beruhen. Damit wäre ein transparenter und expliziter Aufbau vorgegeben, für den sich die benötigte Funktionalität im Sinne eines *user interface management system* (UIMS) wenigstens semiautomatisch generieren ließe. Eine solche Spezifikation kann auch der Ausgangspunkt für die Sammlung von Interaktionsdaten zur Laufzeit und für die dynamische Präsentation von Inhalten sein: Die Navigations- und Interaktionsstruktur ist deklarativ definiert und für das Viewersystem transparent.

13.2.1 Softwareergonomie als Regelung und Operationalisierung der Gestaltung

Die Software-Ergonomie als Wissenschaft im Schnittfeld von Informatik, Arbeitswissenschaft bzw. Arbeitspsychologie und Gestaltung/Mediendesign hat als primären Gegenstand die Frage nach einer adäquaten Gestaltung der Benutzerschnittstellen von Softwaresystemen, wobei in einem umfassenden Sinn neben den Fragen der Informationsdarstellung und den Interaktionsformen

- die menschengerechte Gestaltung von Software unter Berücksichtigung der physiologischen und psychologischen Fähigkeiten des Benutzers, die Anpassung von Software an die mit ihr zu bewältigenden Aufgaben,
- die Berücksichtigung geeigneter Technologien und
- die Integration der Software in einen organisatorischen Zusammenhang

zu berücksichtigen sind (EBERLEH, OBERQUELLE & OPPERMANN 1994: 1 f.). Ausgangspunkt software-ergonomischer Überlegungen sind Modelle für die Mensch-Maschine-Interaktion, die die an der Softwarenutzung beteiligten Ebenen miteinander in Beziehung setzen wie das IFIP-Modells der Mensch-Maschine-Interaktion (DZIDA 1983 und BALZERT 1996: 454).

Ein grundlegendes Problem der Software-Ergonomie ist die Übertragung der abstrakten Erkenntnisse über Aufgabenstruktur, Fähigkeiten des Benutzers und die zur Verfügung stehenden technischen Realisierungsmöglichkeiten einerseits, und die konkreten Anforderungen für ein Softwaresystem andererseits auf die praktische Umsetzung als Gestaltung einer Benutzerschnittstelle. Um den Entwickler zu unterstützen, wurde eine breite Vielfalt von Hilfsmitteln etabliert (vgl. SMITH 1988). Dazu gehören

- *Normen*, die auf einer allgemein Ebene versuchen, software-ergonomische Mindestanforderungen zu beschreiben,
- *Empfehlungen* und *Designregeln*, die Einzelfälle beschreiben (z. B. den Aufbau eines Formulars und die Gestaltung der Beschriftungstexte seine Felder beschreiben),
- *Gestaltungsleitfäden*, die für eine bestimmte Systemumgebung (z. B. MS-Windows oder OSF Motif) grundlegende Gestaltungsverfahren festlegen (Menüaufbau, Struktur von Dialogmasken, Standardisierung von Beschriftungen etc.) und

- *Softwarewerkzeuge*, die bei den Entwickler bei der (semi-)automatischen Entwicklung von Benutzerschnittstellen unterstützen (vgl. SHNEIDERMAN 1998: 155 ff.).

In den folgenden beiden Kapiteln werden Beispiele für Normen und Leitfäden eingeführt und Richtlinien für die Gestaltung webbasierter Anwendungen mit Hinsicht auf das Modell dynamischer elektronischer Bücher diskutiert.

13.2.2 Normen und Leitfäden

Die bekannteste Ergonomienorm ist die ursprünglich aus der DIN-Norm 66234(8) entwickelte ISO-Norm für die Gestaltung von Dialogsystemen (ISO 9241, Teil 10, 16). Unter dem Globalziel der *Benutzerfreundlichkeit* werden die folgenden sieben softwareergonomischen Gestaltungsziele definiert (DZIDA 1994: 374 ff., bes. 380 f.):

- Aufgabenangemessenheit (*suitability for the task*),
- Selbstbeschreibungsfähigkeit (*self-decriptiveness*),
- Steuerbarkeit (*controllability*),
- Erwartungskonformität (*conformity with user expectation*),
- Fehlerrobustheit (*error tolerance*),
- Individualisierbarkeit (*suitability for individualization*, nur in ISO 9241) und
- Lernförderlichkeit (*suitability for learning*, nur in ISO 9241).

Innerhalb der Norm werden diese Globalziele näher definiert (z. B. *controllability*: „A dialogue is said to be controllable if the user is able to maintain direction over the course of the interaction until the point at which the goal has been met“, Abschnitt 3.3 von ISO IEC 9241/10, zitiert nach DZIDA 1994: 380). Hieraus lässt sich aber kein *konkretes Gestaltungshandeln* für den Einzelfall ableiten. Die Problematik der Anwendung einer solchen Norm liegt insofern weniger in der Akzeptanz der Normierungsziele selbst, sondern in der Frage nach ihrer Operationalisierung. Gerade die gegenüber den mittlerweile klassischen graphischen Benutzerschnittstellen erhöhten gestalterischen Freiheitsgrade in Multimediasystemen und Anwendungen im World Wide Web, zu denen elektronische Bücher zählen, verschärfen dieses Problem noch: Den bekannten und akzeptierten Vorgaben der ergonomischen Normen steht ein großer Gestaltungsfreiraum gegenüber, bei dem es kein eindeutiges Entscheidungsverfahren für die Auswahl eines bestimmten Gestaltungsmerkmals gibt. Zudem bilden sich auf das Medium World Wide Web zugeschnittene Gestaltungsrichtlinien erst allmählich heraus (s. u.).[122]

Auf der Ebene allgemeiner Gestaltungsrichtlinien existieren viele den Globalzielen der Normung vergleichbare Sammlungen von Gestaltungshinweisen. Zu den bekanntesten gehören die „goldenen Regeln des Dialogdesigns“, die Ben SHNEIDERMAN entwickelt hat. Er postuliert drei Grundprinzipien der ergonomischen Gestaltung: Anerkennung der durch die Diversität der Benutzer und der technischen Möglichkeiten bedingten hohen gestalterischen Freiheitsgrade, die goldenen Gestaltungsregeln und Strategien zur Fehlervermeidung (SHNEIDERMAN 1998: 67 ff.). Die Gestaltungsregeln bilden den Kern der Leitlinie (SHNEIDERMAN 1998: 74 f.):

- Streben nach Konsistenz (*strive for consistency*),

[122] Vgl. SHNEIDERMAN 1998: 580: „Hypertext, hypermedia, multimedia, and the World Wide Web are still in the Ford Model T stage of development. Strategies for blending text, sound, images, and video are in need of refinement, and effective rhetorics for hypermedia are only now being created.“

- Abkürzungen für erfahrene Benutzer anbieten (*enable frequent users to use shortcuts*),
- Informatives Feedback anbieten (*offer informative feedback*),
- sinnvolle und abgeschlossene Gliederung von Dialogen (Anfang, Mitte, Ende, z. B. Ende durch Abschlussbestätigung, *design dialogs to yield closure*),
- Einfache Fehlerbehandlung (*offer error prevention and simple error handling*),
- Reversibilität von Aktionen zulassen (UNDO-Funktion, Explorationsmodus etc., *permit easy reversal of actions*),
- den Benutzer als „Herrn des Systems" unterstützen (keine unerwarteten Meldungen, Systemzustände etc., *support internal locus of control*) und
- kurzfristige Gedächtnisbelastung reduzieren (*reduce short-term memory load*).

Bei diesen Regeln ist keine unmittelbare Operationalisierung erkennbar, sie dienen vielmehr als Prüfkatalog, der an die gestalterische Einzelentscheidung (z. B. Wahl eines Steuerungselements für die Aktionsauslösung, visuelle Repräsentation einer Hypertextverknüpfung, Aufbau eines Fenstersystems) angelegt werden muss, um den Entwickler bei der Wahl des richtigen Gestaltungsmittels zu unterstützen. Die am ehesten unmittelbar umsetzbaren Hilfsmittel der Software-Ergonomie finden sich auf der Ebene der systembezogenen Leitfäden, die für eine bestimmte Systemumgebung grundlegende Gestaltungsverfahren definieren. Eine Übersicht darüber findet sich in BALZERT 1996: 468, Tabelle 2.21-1. Sie legen z. B.

- die standardisierte Beschriftung und Anordnung von Menüs (Datei | Bearbeiten | Ansicht ... | Hilfe; in verschiedenen Sprachen,
- die Verwendung, der beschränkten Zahl standardisierter Steuerelemente graphischer Benutzerschnittstellen (Schaltflächen, Auswahllisten, Rollbalken, etc.) und
- die Grundstruktur für Dialogmasken

fest. Der Unterschied zwischen diesen *systembezogenen* Richtlinien und der Gestaltungsproblematik elektronischer Bücher lässt sich gut verdeutlichen: Durch die zusätzlichen Gestaltungsfreiräume von Multimediasystemen ist eine Festlegung auf einige wenige Formen von Steuerelementen nicht möglich oder wünschenswert; bei Anwendungen im World Wide Web ist die Interaktionssyntax eingeschränkt (einfacher Mausklick als generelles Mittel der Aktionsauslösung/Hypertexttraversierung). Entsprechende Gestaltungsleitfäden sind aber weniger an den generell gültigen Erkenntnissen der Software-Ergonomie, als an individuellen Erfahrungen und einzelfallbezogenen Gestaltungslösungen orientiert.

13.2.3 Richtlinien für webbasierte Anwendungen

Erst seit kurzem sind Bemühungen um die Erstellung genereller Gestaltungsrichtlinien für das Design im World Wide Web erkennbar: So hat eine Arbeitsgruppe des W3C *accessibility guidelines* entwickelt, die auf den Ebenen

- *Erstellungswerkzeuge* (*authoring tools*, *Authoring Tool Accesibility Guidelines 1.0*, vgl. TREVIRANUS et al. 1999A; *Techniques for Authoring Tools Accessibility*, vgl. TREVIRANUS et al. 1999B),
- *Anwendungssysteme auf Benutzerseite* (*user agent, User Agent Accessibility Guidelines 1.0*, vgl. GUNDERSON & JACOBS 1999A; *Techniques for User Agent Accessibility Guidelines* 1.0, vgl. GUNDERSON & JACOBS 1999B) und

- *Inhalte* (*web content, Web Content Accessibility Guidelines 1.0,* vgl. CHISHOLM, VANDERHEIDEN & JACOBS 1999A; *Techniques for Web Content Accessibility Guidelines 1.0,* vgl. CHISHOLM, VANDERHEIDEN & JACOBS 1999B)

versuchen, globale Empfehlungen mit einer Liste von Einzelkriterien der Gestaltung zu verbinden und einer Rangordnung für die Dringlichkeit der Gestaltungsziele zuzuordnen. Der Ausgangspunkt der Richtlinienzusammenstellung ist die Zugänglichkeit des World Wide Web für Benutzer mit besonderen Anforderungen oder Behinderungen. Dabei wird aber Wert darauf gelegt, dass sie grundsätzlich für alle Benutzer sinnvoll sein können: „The primary goal of these guidelines is to promote accessibility. However, following them will also make Web content more available to all users, whatever user agent they are using [...] or constraints they may be operating under [...]." (CHISHOLM, VANDERHEIDEN & JACOBS 1999A: Abstract). Im Vorfeld der Definition der Richtlinien stellen die Autoren zwei generelle Ziele auf:

- *Ensuring graceful transformation* und
- *making content understandable and navigable* [CHISHOLM, VANDERHEIDEN & JACOBS 1999A: Kap. 2].

Das erste Ziel betrifft vor allem die bereits diskutierte Trennung von Struktur- und Präsentationsmarkup. Es muss das Ziel sein, Dokumente so mit Markup auszuzeichnen, dass sie sich in Abhängigkeit unterschiedlicher Präsentationsmöglichkeiten adäquat von der Strukturauszeichnung in ein Präsentationsformat transformieren lassen. Die zweite Anforderung bringt das generelle Ziel zum Ausdruck, den Benutzer durch geeignete Orientierungsmöglichkeiten zu unterstützen und graphische Sonderformate (*image maps* etc.) nicht zum alleinigen Navigationsmittel zu machen. In der nachfolgenden Tabelle sind die einzelnen *guidelines for web content accessibility* aus CHISHOLM, VANDERHEIDEN & JACOBS 1999A: Kap. 6 wiedergegeben und jeweils einer Kurzbewertung ihrer Anwendbarkeit für dynamische elektronische Bücher gegenübergestellt:

Richtlinie	*Bewertung*
1. Provide equivalent alternatives to auditory and visual content. Provide content, that, when presented to the user, conveys essentially the same function or purpose as auditory or visual content.	Multimediale Komponenten werden einerseits unmittelbar durch – textuelle – Metadaten ergänzt und sind in der Regel eine zusätzliche Umsetzung textuell beschriebener Sachverhalte (z. B. Beschreibung einer Versuchsdurchführung und ihre Darstellung als Animation).
2. Don't rely on color alone. Ensure that text and graphics are understandable when viewed without color.	Diese Richtlinie kann für das Buchpräsentationssystem und die Mehrzahl der multimedialen Komponenten als erfüllt gelten (z. B. Kennzeichnung typisierter Links durch Text *und* Farbe), wobei allerdings durch den bewussten Einsatz von Farbe in Animationen und Simulationen Ausnahmen möglich sind (z. B. farbliche Darstellung der Zustandsänderung in einer Animation).
3. Use markup and style sheets and do so properly. Mark up documents with the proper structural elements. Control presentation with style sheets rather than with presentation elements and attributes.	Die Trennung von Struktur-, Inhalts- und Präsentationsmarkup ist ein zentrales Anliegen dieser Arbeit; durch den Einsatz eines Parsers und XSL-Prozessors im Buchverwaltungssystem sowie im Vorfeld der Datenaufbereitung ist die syntaktische, durch die Beachtung der beschriebenen Modellierungsschritte die semantische Korrektheit der Auszeichnungsmerkmale gewährleistet.
4. Clarify natural Language use. Use markup that facilitates pronunciation or interpretation of abbreviated or foreign text.	Dies ist ein Ansatzpunkt für das Dienstekonzept (z. B. Nutzung eines Übersetzungsdienstes).

Richtlinie	*Bewertung*
5. Create tables that transform gracefully. Ensure that tables have necessary markup to be transformed by accessible browsers and other user agents.	Dieses Kriterium wird insofern verletzt, als Tabellen entgegen den Details der Richtlinie für die Layoutgestaltung verwendet werden; allerdings greift die Ausnahmeregelung – *do not use tables unless the table makes sense when linearized* (Checkpoint 5.3 zu Guideline 5): Beispiele tabellengesteuerten Layouts sind im Referenzprojekt die Liste verfügbarer Komponenten und die Navigationsleiste, die sich ohne Tabellen sinnvoll linearisieren lässt.
6. Ensure that pages featuring new technologies transform gracefully. Ensure that pages are accessible even when newer technologies are not supported or are turned off.	Durch die Trennung von Basistext und Komponenten ist sichergestellt, dass die Basisdaten auch betrachtet werden können, wenn die Komponente aus technischen Gründen (z. B. fehlendes plug-in) nicht darstellbar ist. Ähnliches gilt für die Dienstekonfiguration: Als flexibles Konzept ist sie von vornherein auf unterschiedliche Nutzungsszenarien angelegt.
7. Ensure user control of time-sensitive content changes. Ensure that moving, blinking, scrolling, or auto-updating objects or pages may be paused or stopped.	Auf der textorientierten Ebene sind solche Merkmale ausgeschlossen (z. B. Präsentationsform ohne Scrolling); auf der Ebene multimedialer Komponenten sind in den Benutzerschnittstellen Elemente für das Anhalten vorgesehen bzw. Animationen und Simulationen laufen ohnehin fast ausschließlich benutzergesteuert ab.
8. Ensure direct accessibility of embedded user interfaces. Ensure that the user interface follows principles of accessible design: device independent access to functionality, keyboard operability, self-voicing etc.	Das Gestaltungsziel konnte mit den verfügbaren Technologien nur bedingt erreicht werden, eine vollständige Tastaturunterstützung war nicht möglich.[123]
9. Design for device-independence. Use features that enable activation of page elements via a variety of input devices.	Dieses Zielkriterium konnte kein wesentliches Gestaltungsziel sein, da aufgrund der technischen Anforderungen an die Umsetzung multimedialer Komponenten die Nutzungsszenarien ohnehin weitgehend feststehen.
10. Use interim solutions. Use interim solutions so that assistive technologies and older browsers will operate correctly.	Auch dieses Kriterium ist nur bedingt anwendbar: Einerseits garantiert eine restriktive Festlegung technischer Randbedingungen die Kompatibilität mit der typischen technischen Infrastruktur der Benutzer, andererseits verlangen gerade multimediale Komponenten neuere Technologien, auf die grundsätzlich nicht verzichtet werden kann.
11. Use W3C technologies and guidelines. Use W3C technologies (according to specification) and follow accessibility guidelines. Where it is not possible to use a W3C technology, or doing so results in material that does not transform gracefully, provide an alternative version of the content that is accessible.	Es handelt sich um ein grundsätzliches Anliegen dieser Arbeit, das bei der Auswahl der Standards für die Kodierung der im elektronischen Buch enthaltenen Daten berücksichtigt wird. Einschränkungen gelten nur insofern, als nicht für alle Standards geeignete Werkzeuge zur Verfügung stehen oder die Standards noch nicht abschließend normiert sind.
12. Provide context and orientation information. Provide context and orientation information to help users understand complex pages or elements.	Durch einheitlichen Aufbau des Buchpräsentationssystem und seiner verschiedenen Erschließungsmöglichkeiten für den Buchinhalt (Hierarchiebrowser; Register/Retrievalschnittstelle, Statusinformation, gestalterische Differenzierung zwischen Fokus- und Kontextseiten) ist dieses Kriterium berücksichtigt.

[123] Im Rahmen des Konzepts eines dynamischen elektronischen Buchs, das in Basisinhalte und Komponenten gegliedert ist, ist eine weitere Ebene zu beachten: Die Navigationsleiste des Buchbetrachtungssystem ist im Sinne dieser Richtlinie ein *embedded interface*, das von der Funktionalität des *user agent* abstrahiert. Innerhalb dieses System verfügt eine multimediale Komponente über eine weitere eingebettete Bedienungsschicht.

13. Provide clear navigation mechanisms. Provide clear and consistent navigation mechanisms – orientation information, navigation bars, a site map etc. – to increase the likelihood that a person will find what they are looking for at a site.	Diese Richtlinie wurde durch konsistenten und einheitlichen Aufbau der Benutzerschnittstelle realisiert (von der Inhaltsdarstellung abgetrennte ortsfeste Navigationsleiste, Verfügbarkeit von Kontextinformation (Seitenposition, Einordnung in die Inhaltshierarchie), festes Modell für die Fensterverwaltung und die Darstellung der Komponenten, verschiedene Erschließungsformen).
14. Ensure that documents are clear and simple. Ensure that documents are clear and simple so they may be more easily understood.	Als Globalziel ist diese Anforderung eine selbstverständliche Leitlinie, für ein Konzept dynamischer elektronischer Bücher aber irrelevant, da die Frage, wie die Inhalte formuliert werden, nicht betrachtet wird.

Tabelle 55: Web Content Accessibility Guidelines und ihre Anwendbarkeit für das Referenzprojekt

Bewertet man diese Richtlinien, so muss man feststellen, dass sich aus ihnen zwar keine konsistente Theorie der Gestaltung und Strukturierung von Dokumenten im World Wide Web ergibt, sie aber in Verbindung mit den ausführlichen Umsetzungshinweisen (CHISHOLM, VANDERHEIDEN & JACOBS 1999B) eine Vielzahl praktischer Hinweise ergeben. Das dabei zugrunde gelegte Ziel, Inhalte für möglichst viele Betrachtungsumgebungen passend oder transformierbar zu gestalten, spielt aber für das Konzept dynamischer elektronischer Bücher eine untergeordnete Rolle, da es hier darum geht, die Gestaltung für eine bestimmte Betrachtungsumgebung zu optimieren.[124]

Zu den Richtlinien für die Inhaltsaufbereitung kommen die Anforderungen an die Gestaltung des *user agent* (GUNDERSON & JACOBS 1999A, 1999B). Sie sind für das Konzept dynamischer elektronischer Bücher mittelbar relevant, da ein bestimmter Typ eines *user agent* vorausgesetzt wird: Ein Webbrowser mit der Möglichkeit der Darstellung von HTML, der Verwendung einer eingebetteten Skriptsprache, der Verfolgung von Hyperlinks und der Informationsübertragung zwischen Client und Server über HTTP. Bei der Realisierung der Benutzerschnittstelle des Buchbetrachtungssystems wird aber von den Navigationsmöglichkeiten des Browsers abstrahiert. Sie werden in der Schnittstelle des elektronischen Buchs ausgeblendet. Wie die Bewertung der *user agent*-Richtlinien hinsichtlich der Gestaltung elektronischer Bücher zeigt, besteht eine deutliche Überlappung mit den oben vorgestellten Richtlinien für die Gestaltung der *Inhalte*.

Guideline	*Bewertung*
1. Support input and output device independence. Ensure that the user can interact with the user agent (and the content it renders) through standard input and output APIs supported by the operating system.	Die technischen Mindestanforderung an das im Referenzprojekt realisierte Buch gewährleisten ein Mindestmaß an Kompatibilität (und Plattformneutralität) für eine Reihe typischer Nutzungsumgebungen.
2. Ensure keyboard access to user agent functionalities. Ensure that the user has access to user agent functionalities through the keyboard since keyboard access is widely supported.	Aus technischen Gründen konnte diese Anforderung nicht durchgängig berücksichtigt werden, auch wenn ihre Anwendung als wünschenswert erscheint – zumindest für die am häufigsten auftretenden Ereignisse der Navigation (Blättern, Aufruf des Hierarchiebrowsers).
3. Ensure user access to all content. Ensure that users have access to all content, notably author-supplied alternative representations of content (descriptions of images, closed captions for video or audio, etc.).	Wie schon oben zur Inhaltsrichtlinie 1 ausgeführt, ist dieses Ziel auf unterschiedlichen Ebenen berücksichtigt (textuelle Beschreibungen von Komponenten, Navigationselementen, typisierten Links etc.).

[124] Die Richtlinien zielen u. a. darauf ab, Information so deklarativ auszuzeichnen, dass sie von unterschiedlichen *information appliances* und unter unterschiedlichen Nutzungsbedingungen interpretiert und dargestellt werden können, vgl. oben Kap. 3.3.

Guideline	*Bewertung*
4. Allow the user to turn off features that may reduce accessibility. Ensure that the user may turn off features that may obscure content or disorient the user.	Soweit die Aktivierung der eingebetteten Komponenten durch den Benutzer erfolgt, ist diese Richtlinie berücksichtigt.
5. Ensure user control over styles. Ensure that the user has control over the colors, text size, speech rate and pitch, and other stylistic aspects of a resource and can override author styles and user agent default styles.	Diese Anforderung steht in direktem Widerspruch zum Gestaltungsmodell des buchorientierten Layout, das mit den verfügbaren Technologien eine Fixierung bestimmter Gestaltungselemente erforderlich macht. Es steht aber auch allgemein im Widerspruch zu jeder expliziten Gestaltung, die durch die Zuordnung gestalterischer Merkmale zu Inhaltselementen bewusst gewählt und insofern nicht beliebig modifizierbar ist.
6. Observe system conventions and standard interfaces. Communicate with other software (dependent user agents, the operating system, plug-ins) through applicable interfaces and observe conventions for the user interface (documentation, installation etc.).	Diese Forderung ist insofern irrelevant, als der *user agent* hier für jede Nutzungsplattform vorausgesetzt wird und von der systemunabhängigen Nutzbarkeit der Inhalte überlagert wird. Sie spielt eine Rolle für das Konzept der Diensteaktivierung, soweit diese über plattformspezifische APIs erfolgen muss.
7. Support applicable W3C technologies and guidelines. Support applicable W3C specifications and in particular their accessibility features.	Vgl. oben Inhaltsrichtlinie 11 in Tabelle 55.
8. Provide navigation mechanisms. Provide navigation mechanisms that meet the need of different users: Serial navigation for context, direct navigation for speed, search functions, structured navigation etc.	Durch das mehrschichtige Strukturierungsmodell, das die genannten Navigationsmöglichkeiten enthält, wird diese Leitlinie generell berücksichtigt.
9. Help orient the user. Provide information about resource structure, viewport structure, element summaries, etc. that will assist the user understand their browsing context.	Erläuterungen zum Aufbau der Benutzerschnittstelle sind integraler Teil des Gestaltungskonzepts; vgl. Inhaltsrichtlinie 13 in Tabelle 55.
10. Notify the user of content and viewport changes. Alert users, in an output device-independent fashion, of changes to content or the viewport.	Durch die benutzergesteuerte Aktivierung der Inhalte ist dies kein relevantes Ziel; eine Ausnahme kann dort vorliegen, wo durch dynamisch integrierte Dienste der Datenbestand automatisch verändert wird.
11. Allow the user to configure the user agent. Allow users to configure rendering, mouse, keyboard, the user interface, etc. to facilitate daily use of the software.	Als allgemeines Ziel der *Adaptierbarkeit* von Softwaresystemen ein klassisches software-ergonomisches Ziel, das aber hier aufgrund der Festschreibung von Gestaltungsparametern für die Umsetzung eines festen Layouts nicht operationalisierbar ist.
12. Provide accessible product documentation and help. Ensure that the user can learn about software features, notably those that relate to accessibility.	Diese Richtlinie ist durch Integration einer einfachen Hilfefunktion zumindest rudimentär berücksichtigt.

Tabelle 56: User agent accessibility guidelines und ihre Anwendbarkeit für das Referenzprojekt

Der Abgleich der *W3C Accessibility Guidelines* mit den Gestaltungsentscheidungen für ein dynamisches elektronisches Buch zeigt für die meisten Richtlinien Übereinstimmung. Mit wenigen Ausnahmen werden die *Accessibility Guidelines* befolgt. Ihre Präzisierung durch einen umfangreichen, im Einzelnen mit Prioritäten versehenen Katalog von Gestaltungstechniken, zu diskutieren, erscheint hier also nicht erforderlich. Wichtiger ist, das Buchbetrachtungssystem für ein dynamisches elektronisches Buch vorzustellen und seine Benutzungsmetapher sowie seine Gestaltungselemente zu diskutieren.

13.3 Aufbau und Gestaltung des Buchbetrachtungssystems

Das Buchbetrachtungssystem stellt dem Benutzer die Inhalte des elektronischen Buchs dar und präsentiert zudem eine Reihe von Grundfunktionen für die Interaktion mit dem elektronischen Buch:

- *Navigationselemente* zur sequentiellen Darstellung aufeinanderfolgender informationeller Einheiten (Blättern),
- *hierarchische Zugriffsmöglichkeiten* auf die Buchinhalt sowie die gezielte Selektion der im Buch enthaltenen multimedialen Elemente,
- Erschließungskomponenten zur gezielten *Suche* im Buch (Register bzw. Volltextrecherche) und
- eine einfache *Hilfefunktion,* die als Informationskomponente Aufbau und Funktionsweise des elektronischen Buchs erläutert.

Zusätzlich zu diesen im Referenzprojekt realisierten Komponenten bietet der Prototyp eines dynamischen elektronischen Buchs den Zugriff auf die für das Buch spezifizierten Dienste sowie die dazu erforderlichen Verwaltungsschnittstellen (Anmelden des Benutzers am Buchserver etc.). Die Buchsteuerung durch die verschiedenen Navigationselemente und Zugriffsverfahren abstrahiert von den vorgegebenen Möglichkeiten des Browsers und trennt Inhaltsdarstellung und Navigation voneinander.

13.3.1 Präsentationsmetapher

Eine entscheidende Frage bei der Realisierung der Benutzerschnittstelle für elektronische Bücher ist die Auswahl einer geeigneten Präsentationsmetapher. Unter einer Metapher versteht man bei der Gestaltung von Benutzerschnittstellen die Übertragung von dem Benutzer vertrauten Konzepten auf das Design des Interfaces (vgl. KRAUSE 1997: 243 ff.). Es entsteht eine *visuelle Sprache*, die in Analogie zu konkreten oder abstrakten Konzepten eines anderen Wirklichkeitsausschnitts stehen (vgl. COOPER 1995: 57 ff., THISSEN 2000: 42). Die bekanntesten Metaphern für die Gestaltung von Benutzerschnittstellen finden sich im Bereich der Betriebssysteme bzw. der Büroautomatisierung: Die *Desktop*-Metapher für die Gestaltung graphischer Benutzungsumgebungen (z. B. Microsoft Windows) und die Schreibmaschinenmetapher für graphisch direkt-manipulative Texteditoren. Im Bereich von Hypermedia und *electronic publishing* finden sich zahlreiche metaphorische Ansätze der Benutzerschnittstellengestaltung:

- Abstrakte Metaphern, die Netze und Bäume als Visualisierungskonzept einsetzen oder diese weiterführen (z. B. *hyperbolic tree views*, vgl. THISSEN 2000:64),
- die Buch- und Schriftrollenmetaphern in Analogie zu traditionellen Publikationsformen und
- piktorielle Metaphern, die als globales Gestaltungsparadigma eine virtuelle Welt aufbauen und den Gegenstandsbereich der Publikation zum Aufbau der Benutzerschnittstelle verwenden (z. B. Labormetapher, vgl. BOLES et al. 1998B, HORNECKER & SCHÄFER 1999).

Im Referenzprojekt wurde beim Entwurf des Layoutmodells von der weit verbreiteten Standardlösung für die Gestaltung von Informationssystemen im World Wide Web abgewichen: Dort erfolgt eine Aufteilung der Inhalte in der Regel durch Definition kleinster logischer Einheiten, deren Darstellungsumfang in der Regel (bedingt durch unterschiedliche technische Voraussetzungen beim Benutzer oder individuelle Einstellungen wie aktu-

elle Fenstergröße) nicht simultan in einem Browserfenster dargestellt werden können und somit *lokales Scrolling* innerhalb eines Dokuments erforderlich machen (Schriftrollenmetapher). Für das Konzept des Buchbetrachtungssystems wurde hier die Analogie zum gedruckten Buch gewählt, d. h. es erfolgt bei der Präsentation eine feste Zuordnung von Inhalten zu Präsentationseinheiten durch eine Aufteilung in einzelne Bildschirmseiten. Die folgende Abbildung zeigt schematisch den Unterschied zwischen beiden Ansätzen:

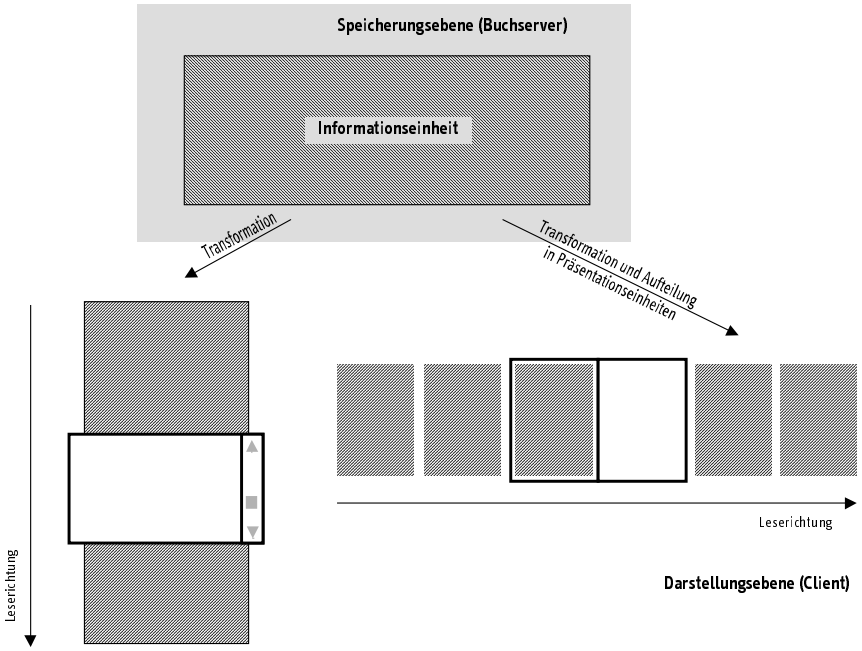

Abbildung 45: Vergleich von Präsentationsmetaphern: Schriftrolle (links) und Buch (rechts)

Die Auswahl der Buchmetapher erfolgt aus folgenden Überlegungen:

- Die Buchbenutzung ist ein vertrautes und bewährtes Konzept der Informationsdarstellung und wird daher für elektronische Bücher häufig verwendet.[125] Die folgende Abbildung zeigt zwei Beispiele elektronischer Bücher bzw. Lernsysteme auf der Basis der Buchmetapher:

[125] Z. B. im Rahmen des *Visual Book*-Projekts, vgl. LANDONI & GIBB 1999, oder im Kontext webbasierter Lernsysteme, vgl. HENZE et al. 1999; wie Kap. 3.3 gezeigt hat, wird die Buchmetapher auch von *digital appliances* aufgegriffen.

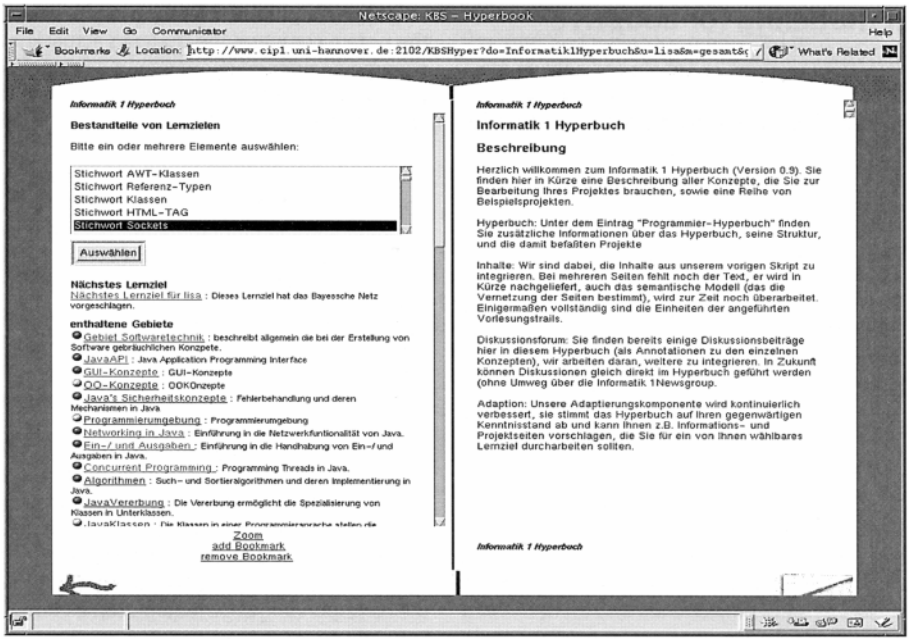

Abbildung 46: Verwendung der Buchmetapher bei HENZE et al. 1999: 27, Abb. 1

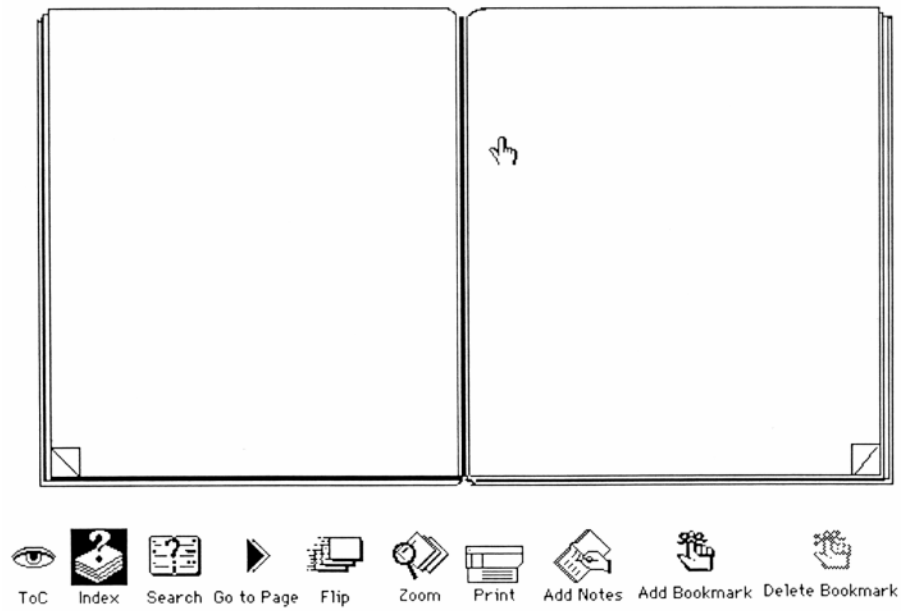

Abbildung 47: Verwendung der Buchmetapher bei LANDONI & GIBB 1999: 12, Abb. 2

- Für einen durch formale Inhalte geprägten Publikationsbestand ist eine solche Darstellung angemessen. Eine stärkere Visualisierung (z. B. als Labormetapher, die die Labors des Physikalischen Praktikums visualisiert) erfordert dagegen nicht nur einen sehr großen Realisierungsaufwand, sondern ist auch sachlich nur bedingt adäquat, da das Erlernen konkreter Handlungsabfolgen (Versuchsdurchführung) nur einer der verschiedenen Aspekte der Realisierung multimedialer Komponenten ist.
- Schließlich dient die Buchmetapher der Lösung des Darstellungsproblems: Eines der größten Probleme bei der Realisierung von Benutzerschnittstellen ist der beschränkt

verfügbare Platz.[126] Bei Berücksichtigung technischer Mindestvoraussetzungen für den praktischen Einsatz (Beschränkung des Buchbetrachtungssystem auf 800 × 600 Pixel) lassen sich auf einer Bildschirmseite nur etwa 1/3 bis 1/4 der Information einer Druckseite darstellen. Insofern kommt der adäquaten Kontextualisierbarkeit von Information große Bedeutung zu. Dies ist im vorliegenden Fall durch die entkoppelbare Ansteuerung der Buchseiten bei gleichzeitigem Verzicht auf Scrolling-Mechanismen realisiert.

Den Vorteilen der Buchmetapher stehen als Nachteile die traditionelle Gestaltung in Anlehnung an das gedruckte Buch und das damit verbundene Fehlen einer graphischen Gestaltungsmetapher („virtuelle Welt") gegenüber. Die Verwendung von HTML/CSS für ein Satzverfahren mit festem Layout hat zu erheblichem Kodierungs- bzw. Analyseaufwand geführt. Soweit sich Ergänzungen verschiedener Art nicht in das Grundkonzept der seitenorientierten Darstellung einfügen lassen, lässt das Gestaltungskonzept für das Buchbetrachtungssystem hilfsweise die Erzeugung zusätzlicher Fenster zu; dieses Konzept wurde aber nur vorsichtig eingesetzt, um die Orientierung des Benutzers an der Nutzung der Buchmetapher zu stärken. In konzeptueller Hinsicht stösst die Übertragung der Buchmetapher für die Gestaltung elektronischer Medien vielfach auf Kritik:

> Vor allem die Versuchung, dem interaktiven Medium einen Seitencharakter aufzudrücken, ist sehr groß. Das eröffnet die Möglichkeit, mit geringem Aufwand eine 1:1 Korrespondenz zwischen Printmedium und interaktivem Medium zu realisieren – nur dass dabei dann leider kein wirklich interaktives Medium entsteht, sondern lediglich ein digitales Printmedium. [GIRMES et al. 1999: 234.]

Dem ist allerdings mit Blick auf die hier gewählte Buchmetapher entgegenzuhalten, dass sich die Korrespondenz mit dem gedruckten Buch letztlich weniger aus der Analogie zum Printmedium, sondern aus inhaltlich bedingten Linearität der Textsequenzen ergibt und dass durch die Integration multimedialer Komponenten die zusätzlichen Freiheitsgrade hinsichtlich der Interaktivität verfügbar bleiben.

Für die Präsentation der Inhalte wird die *Buchmetapher* modifiziert: Die Buchinhalte werden wie bei einem gedruckten Buch als Doppelseite dargestellt, wobei die jeweils rechte Seite als *Fokusseite* die aktuelle Leseposition des Benutzers darstellt, die durch die linke Seite als *Kontextseite* ergänzt wird. Als Default und in Einklang mit der Buchmetapher ist links die je vorangegangene Seite angezeigt. Die Buchmetapher wird dann dadurch erweitert, dass linke und rechte Darstellungsseite voneinander entkoppelt und die linke Seite unabhängig von der rechten mit Inhalten gefüllt werden kann. Die Kontextseite übernimmt somit eine Doppelfunktion:

- Die Information der Fokusseite kann in einem einheitlichen Betrachtungssystem kontextualisiert werden, indem Hypertextverknüpfungen der Fokusseite in der Kontextseite angezeigt werden.
- Die zur aktuellen Inhaltseinheit verfügbaren Komponenten können – soweit der Darstellungsumfang ausreichend ist – in die Kontextseite eingeblendet werden.

Generell erlaubt die Koordination von Kontext und Fokus die gemeinsame Präsentation je zweier beliebiger Informationseinheiten in einem Viewermodell mit klar strukturierter Funktionalität. Die Entkopplungsfunktion ist für die multimedialen Komponenten von

[126] Daher die Vielzahl der Ansätze in der Forschung, durch innovative Informationsvisualisierungsformate die Platzausnutzung zu optimieren (*fish-eye views*, *cone trees*, etc.). Vgl. dazu zahlreiche Beiträge in CARD, MACKINLAY & SHNEIDERMAN 1999.

Bedeutung, da sich die Herleitung der Grundlagen zu einer Simulation über zahlreiche Präsentationseinheiten erstrecken kann und es daher sinnvoll ist, die Multimediakomponente während des Blätterns in der Kontextseite fixiert zu lassen. Die folgenden Abbildungen zeigen Beispiele für die Verwendung der modifizierten Buchmetapher während der Interaktion:

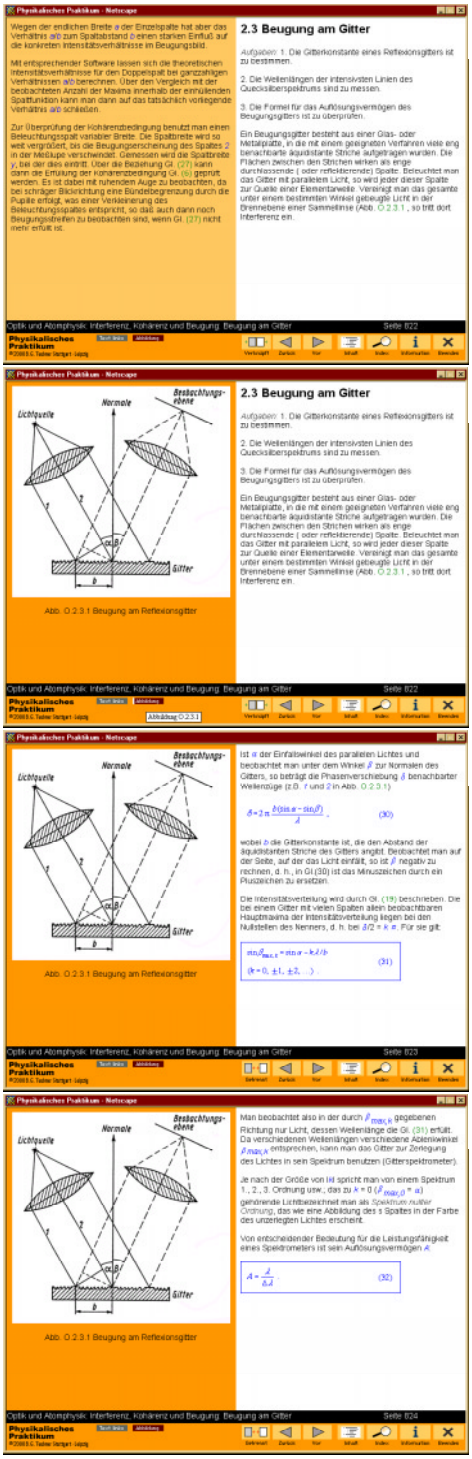

Abbildung 48: Nutzung der Entkopplung von Fokus- und Kontextseite bei der Lektüre

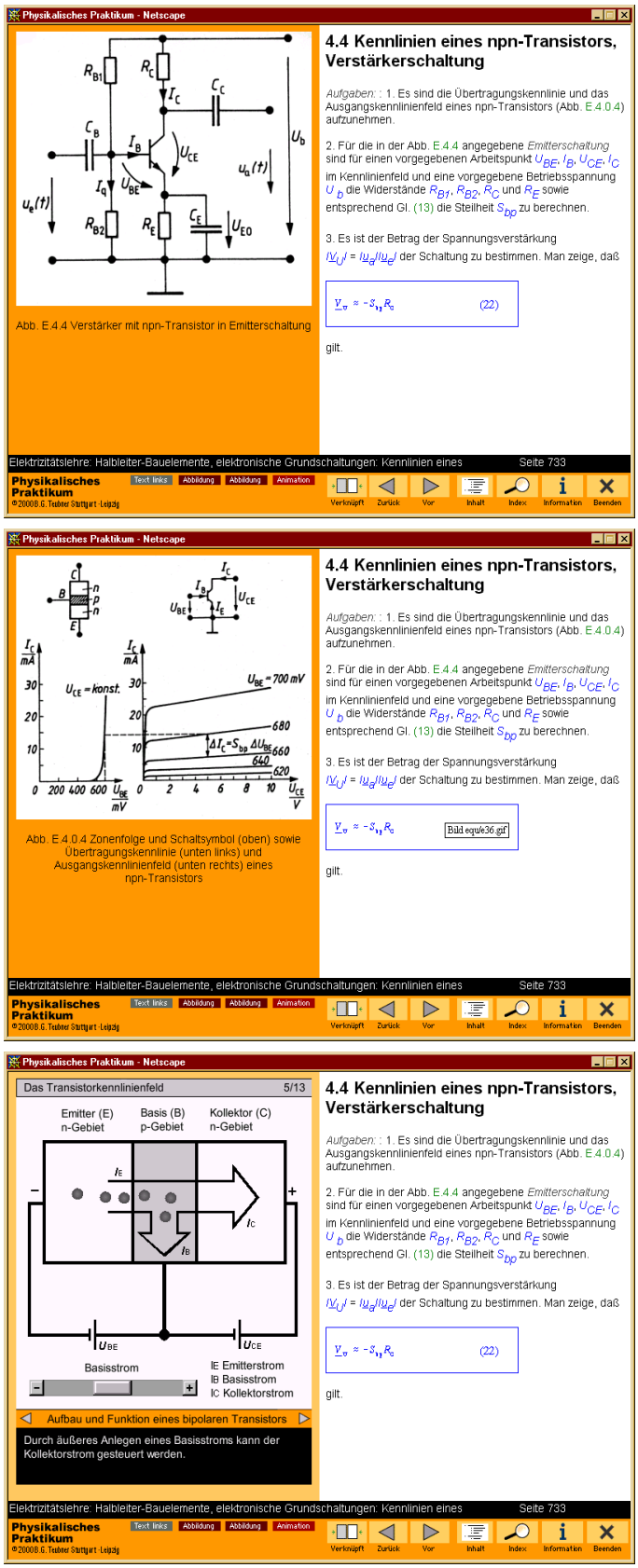

Abbildung 49: Einblenden verschiedener Komponenten in die Kontextseite

Ungeachtet der Nachteile, die diese Vorgehensweise mit sich bringt, u. a. Verletzung einiger oben eingeführter *guidelines* für die Gestaltung von Informationssystemen im World Wide Web sowie ein erheblicher Verarbeitungsaufwand bei der Aufbereitung der Inhalte, ist diese Vorgehensweise aus folgenden Gründen zweckmäßig:

1. In Verbindung mit der Funktionalität entkoppelbarer Fokus- und Kontextseiten ist die grundsätzliche *parallele Darstellbarkeit* beliebiger Inhaltselemente im Rahmen des Präsentationssystems gewährleistet, ohne die Benutzerschnittstelle durch zusätzliche Fenster unübersichtlich werden zu lassen: Findet sich z. B. bei einer längeren formalen Herleitung bei *Gleichung 45* ein Rückbezug zu *Gleichung 17*, so kann der Benutzer diese Stelle in die Kontextseite laden und den Bezug betrachten.
2. Diese Vorgehensweise ist dann sinnvoll, wenn Inhaltselemente sequentiell aufeinander aufbauen und davon auszugehen ist, dass der Benutzer auf lokaler Ebene, d. h. zwischen den einzelnen Präsentationseinheiten einer Inhaltskomponente, zahlreiche Navigationsschritte ausführt, um die dargestellten Inhalte zu verstehen.
3. In Verbindung mit der am *Medientyp* orientierten Aufteilung der Inhalte in Basismaterial (Text und Formeln) und Komponenten ergibt sich ein Layoutmodell, für das zwar die Einzelseitenaufteilung (d. h. Abbildung der Inhaltseinheiten auf Präsentationseinheiten) problematisch ist (Umbruch), das aber innerhalb der einzelnen Präsentationseinheit eine einfache Layoutstruktur aufweist.
4. Lösungen, die ein vollständiges Seitenlayout unter Integration sämtlicher Komponenten vorsehen, sind technisch nur mit großem Aufwand umzusetzen und auch dann nur sinnvoll, wenn sich eine überschaubare Anzahl fester Layoutstrukturen (templates) für die Seiten des Buchs ergibt; es ist aber davon auszugehen, dass aufgrund der Größe und Komplexität der textuellen Inhalte wie der ihnen beigeordneten multimedialen Komponenten eine feste Gestaltungslösung ohne lokales Scrolling nicht zu erreichen ist.

Die Aufteilung auf zwei Buchseiten bedeutet eine starke Eingrenzung des Darstellungsraums für einzelne Medienelemente wie Abbildungen oder Multimediakomponenten; es war mit diesem Ansatz zwar möglich, die meisten Komponenten für die Betrachtung in der Kontextseite einzupassen, diese Strategie ließ sich aber nicht vollständig einhalten. Daher wurde zusätzlich zum Buchbetrachter bzw. dessen Hauptfenster die Möglichkeit zugelassen, Komponenten, die einen größeren Darstellungsraum erfordern, auch in einem separaten Fenster zu öffnen. Dies gilt grundsätzlich auch für alle Abbildungen des Buchs, die in jeweils zwei Darstellungsgrößen vorliegen, einer reduzierten zur Darstellung in der Kontextseite sowie einer erweiterten, in der Details besser erkennbar sind, zur Darstellung in einem zusätzlichen Fenster (vgl. unten Kap. 14.1.7, Abbildung 76 f.).

13.3.2 Definition allgemeiner Gestaltungsparameter

Bei der Festlegung einer Designspezifikation – die konkrete ästhetisch-gestalterische Umsetzung außer Betracht gelassen – kann man folgende Merkmale beobachten:

- Generische vs. kontextabhängige Faktoren, z. B. bei der Definition des Navigationsmodells *allgemeine* Navigationsfunktionen, die generisch erzeugt werden können, gegenüber einer situations- und kontextbedingten Navigation, die zur Lesezeit aus dem aktuellen Kontext heraus erzeugt wird (z. B. Blättern vs. lokales Scrolling innerhalb einer informationellen Einheit).

- Die Festlegung allgemeiner Merkmale der Informationskodierung wie Farbe, Schrift und Größe der gestalterischen Elemente.
- Allgemeine Prinzipien der Layoutgestaltung in Einklang mit software-ergonomischen Anforderungen, d. h. insbesondere die ortsfeste Zuordnung der Inhalts- und Funktionselemente des elektronischen Buchs zu einem Layoutmodell.

Die nachfolgende Übersicht zeigt die Gestaltungsvorgaben, wie sie im Referenzprojekt als Vorlage für den Entwurf des *screen design* verwendet wurden:

Technische Rahmenbedingungen
- Auflösung 800 × 600 gesamt, d. h. effektiv abzüglich Fensterrahmen und Titelleiste (nur oben): ca. 790 × 574 Pixel.
- 256 Farben
- Lieferung der gestalteten Elemente und Bildschirmentwürfe als GIF-Dateien
- Browser als Viewersystem (Netscape 4.X)
- Verwendung von Frames, Tabellen und *Cascading Style Sheets* ist möglich

Allgemeine Designvorgaben
- Einarbeiten gestalterischer Vorgaben des Verlags auf der Basis der gelieferten Beispielbücher bzw. eines CD-ROM-Prototyps
- Umsetzung des im Drehbuch vorgegebenen Seitenaufbaus für das elektronische Buch

Farben
Zusammenstellen geeigneter und mit den Browserfarbpaletten kompatibler Farbschemata für die verschiedenen Bereiche der Anwendung.

Typographie
- Auswahl einer (oder mehrerer) Auszeichnungsschriften für Überschriften etc.
- Auswahl einer geeigneten Satzschrift:
  - gute Lesbarkeit am Bildschirm d. h. im Browser
  - Verfügbarkeit unterschiedlich eng laufender Schnitte, um bei tabellarischen Darstellungen mehr Text darstellen zu können
- Festlegen typographischer Grundkonstanten (Einzüge, Abstände etc.)
- Definition der typographischen Grundkonstanten zur Verwendung als *Cascading Style Sheet*

Gestaltung von Navigationselementen und Symbolen
- Navigationsleiste mit Navigationselementen für
  - Vorblättern
  - Zurückblätter
  - Aufruf des Suchwerkzeugs
  - Aufruf der Hilfe
  - Beenden der Anwendung
  - Kapitel vor/zurück
  - Inhaltsverzeichnis aufrufen
  - Bezug Inhaltsverzeichnis umschalten (linke/rechte Buchseite)
  - Linke und rechte Seite koppeln/entkoppeln
  - Animation klein/groß aktivieren
  - Abbildung klein/groß zeigen

  Annotation aktivieren/ergänzen
  - Erläuterung/Vertiefung/Ergänzung aktivieren
  - Berechnung/Datenanalyse in externer Anwendung aktivieren
  - Lesezeichen aktivieren
  - Optionale Seiten/Verweise aktivieren

Gestaltung von Seiten und Seitenschemata

- Titelseite der CD mit den CI-Elementen der B. G. Teubner Verlagsgesellschaft (Logo, Farbe, Zeichnung des Buchcovers)
- Gestaltung von Bildschirmvorlagen (Schemata):
  - Inhaltsverzeichnis
  - Textdarstellungsseiten:
    - „linke Leseseite“
    - „rechte Leseseite“
  - Darstellungsseiten für Multimediainhalte (Erstellung in Director zur Verwendung mit Shockwave bzw. als Java-Applets) :
    - Design für Schaltflächen
    - Farb- und Schriftvorgaben (in Einklang mit den Textdarstellungsseiten)

Abbildung 50: Gestaltungsvorgaben für die Multimedia-CD-ROM Physikalisches Praktikum

Die gestalterischen Möglichkeiten der Realisierung von Benutzerschnittstellen für Multimediasysteme verlangen nach einer einheitlichen Spezifikation der wichtigsten Darstellungscharakteristika (*screen design*). Sie sollen auch dazu dienen, dem Buch eine wiedererkennbare visuelle Gestalt zu verleihen (und die *corporate identity* des Verlags transportieren). Auf der Basis der Spezifikation

- des Grundkonzepts für das Buchbetrachtungssystem,
- der verschiedenen Typen multimedialer Ergänzungen,
- der erforderlichen verschiedenen Navigationselemente und
- der technischen Randbedingungen (Darstellungsfläche, Farbtiefe etc.).

wurde von einem externen Dienstleister ein *screen design* für das elektronische Buch entwickelt, das die gestalterischen Anforderungen für Buch und Ergänzungen enthält (vgl. STOCKWERK2 1999). Die folgenden beiden Abbildungen zeigen Designentwürfe für den Buchbetrachter und die Benutzerschnittstelle eingebetteter Simulationen:

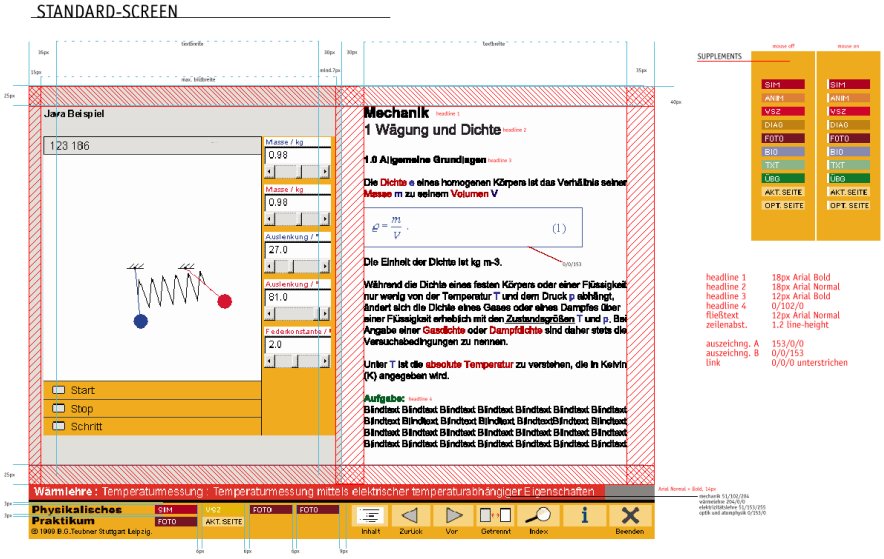

Abbildung 51: Basisdesign für den Buchviewer

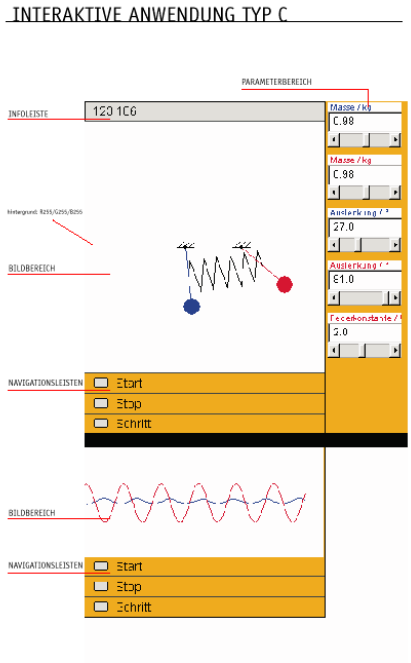

Abbildung 52: Basisdesign für Simulationen

13.3.3 Kodierung und Steuerung

Die Umsetzung des Buch-Viewers erfolgt als Anpassung der Default-Benutzerschnittstelle eines Webbrowsers mit Hilfe der Skriptsprache JavaScript. Die Skripten übernehmen folgende Aufgaben:

- Aufbau und Aktivierung der Fenster des elektronischen Buchs,
- Auswertung der Steuer- und Kontextinformation der Buchseiten, insbesondere der Daten zu den für den aktuellen Lesekontext verfügbaren Komponenten,
- lokale Kommunikation zwischen Fenstern (Steuerung des Hauptfensters (Buchseiten) durch Hierarchiebrowser und Register/Recherchefunktion und
- die Funktionen des Hierarchiebrowsers, d. h. die Darstellung des Inhaltsverzeichnisses als *tree view* mit expandierbaren Knoten.

Dabei lässt sich das elektronische Buch mit wenigen Ausnahmen von verschiedenen Plattformen (MS-Windows, Solaris, Linux) aus nutzen und erfüllt partiell das Kriterium einer plattformneutralen Umsetzung von Informationssystemen. Einschränkungen gelten nur für proprietäre Browser-plug-ins wie *Macromedia Shockwave*, die für die Entwicklung interaktiver Animationen verwendet werden, aber nicht für alle Plattformen verfügbar sind. Die zur Steuerung erforderlichen Skripte sind in den Hauptkomponenten des Buchs (Viewer, Hierarchiebrowser, Register) enthalten. In den Einzelseiten ist lediglich die vom Server bei der Transformation der Ausgangsdaten generierte Kontextinformation (Links auf Medienelemente, sequentieller Seitenkontext, Position) enthalten.

13.3.4 Fensteraufbau des Buchbetrachters

Nachfolgend soll das schon in Kap. 3.1.3 eingeführte Fenstersystem des Buchbetrachters dargestellt werden. Der Benutzer startet das elektronische Buch über eine an der Gestal-

tung des gedruckten Buchcovers orientierte Animation (Abbildung 53), aus der heraus der eigentliche Buchbetrachter (Abbildung 54) aktiviert und geöffnet wird.

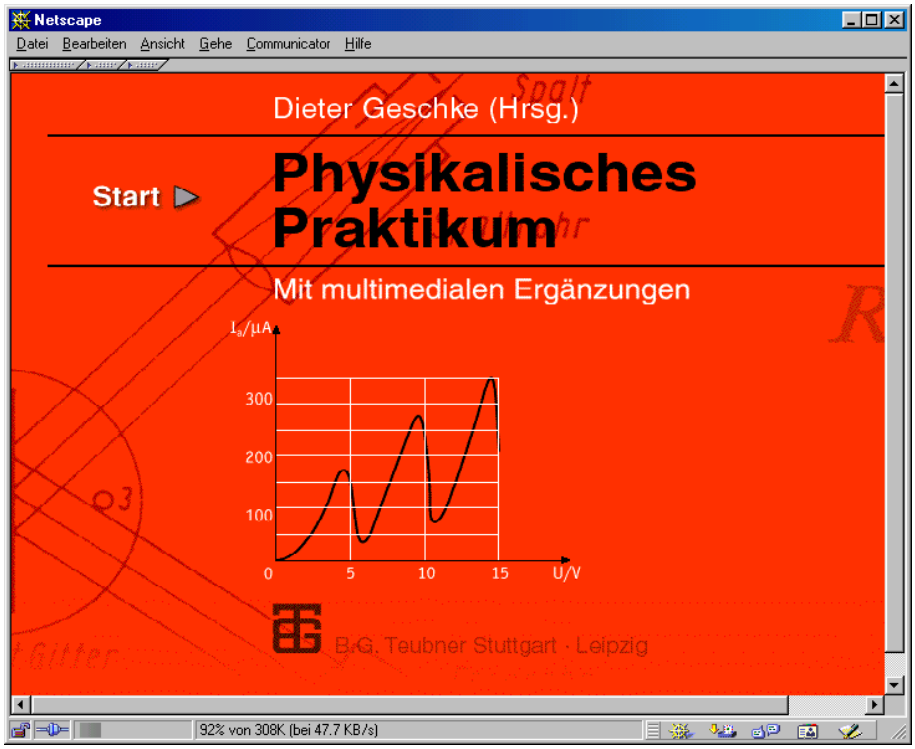

Abbildung 53: Startbildschirm des elektronischen Buchs

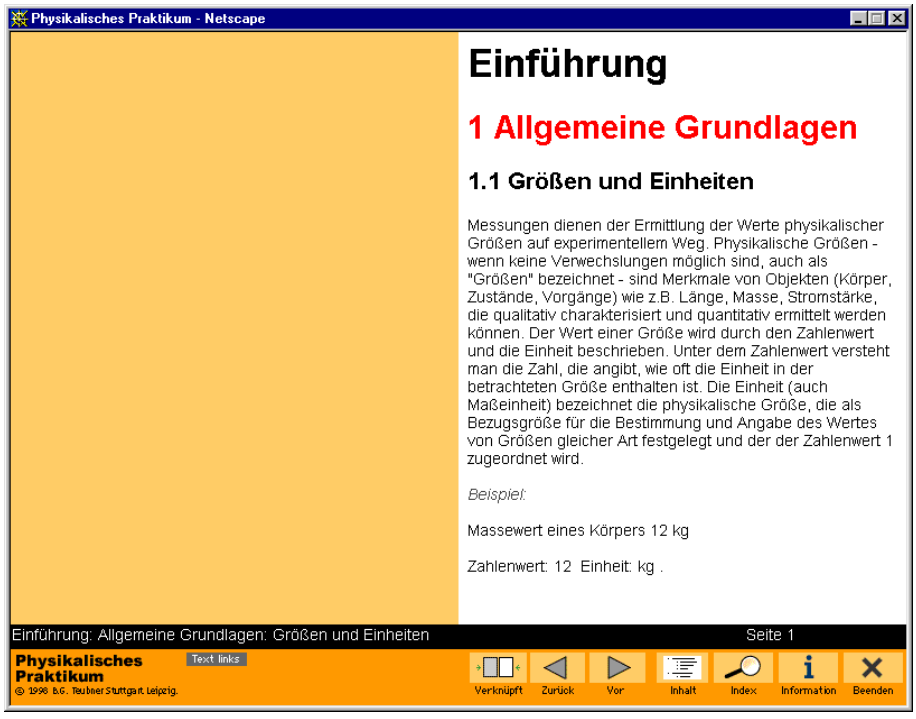

Abbildung 54: Erste Seite des elektronischen Buchs

Am unteren Rand des Betrachtungssystems findet sich links eine Liste mit Symbolen für die im aktuellen Lesekontext verfügbaren Komponenten. Rechts unten, unterhalb der Statusleiste, stellt die Navigationsleiste die generische Interaktionsfunktionalität bereit. Der Buchviewer umfasst also fünf Bereiche (vgl. ZIEMS & NEUMANN 1999: 143f, die für ein Logistik-Lernsystem eine ähnliche Aufteilung vorstellen):

- Fokusseite (rechts)
- Kontextseite (links)
- Navigationsleiste (unten rechts)
- Liste der multimedialen Komponenten (unten links)
- Statusleiste mit Angabe von Inhaltsbereich und Seitenzahl (zwischen Buchseiten und Steuerleisten)

Die technische Realisierung erfolgt mit Hilfe von HTML-Frames, d. h. jeder der fünf Bereiche des Buchbetrachtungssystems ist ein eigener Frame, der vom Buchserver über HTML-Templates (Funktionsbereiche) bzw. durch Transformation der Ausgangsdaten (Inhalt der Buchseiten) gefüllt wird. Die Elterndatei der einzelnen Frames enthält zusätzlich die für die Fensterverwaltung erforderlichen Skripten und globalen Variablen. Durch die ortsfeste Zuweisung eigener Bereiche der Benutzerschnittstelle für Komponentenliste und Navigationsleiste ist die Trennung von Inhaltspräsentation und Steuerung gewährleistet und die entsprechende Funktionalität steht dem Benutzer ohne weitere Interaktion (*scrolling*) zur Verfügung. Jede Buchseite entspricht einer Präsentationseinheit, die vom Buchserver durch Transformation der XML-kodierten Ausgangsdaten erzeugt wird. Die *Navigationsleiste* enthält Elemente

- für die Koppelung bzw. Entkoppelung von linker und rechter Buchseite (Fokus und Kontext),
- für das Blättern im Buch (je eine Seite vor/zurück),
- den Aufruf der hierarchischen Inhaltsübersicht/Inhaltsverzeichnis,
- den Aufruf des Registers bzw. der Volltextrecherche (Online-Version),
- das Abrufen einfacher Bedienungshinweise sowie
- die Möglichkeit, die Anwendung zu beenden.

Zusätzlich lässt sich durch Aktivieren (Klick) des Seitenzählers jede Seite des elektronischen Buchs direkt anspringen, eine Funktion, die zur Vereinfachung von Korrekturarbeiten an den Einzelseiten des Buchs erstellt wurde, sich aber generell für die Nutzuung des Buchs eignet. Die Navigationsleiste ohne und mit aktiviertem Seitennavigator ist in den beiden folgenden Abbildungen dargestellt:

Abbildung 55: Navigationsleiste

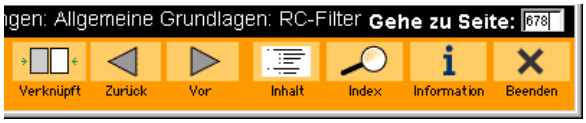

Abbildung 56: Navigationleiste mit aktivierter Seitenansteuerung

Die Liste der zur aktuellen Inhaltseinheit verfügbaren Komponenten enthält als typisierte Verknüpfungen Symbole für jeden Ergänzungstyp, über die sich die Komponente aktivie-

ren lässt. Aus Platzgründen musste auf eine – möglicherweise verständlichere – piktorielle Gestaltung verzichtet werden. Um den Inhalt des Medientypus einer Komponente zu erschließen, werden Tooltips eingeblendet, d. h. eine textuelle Beschreibung des Medienelements, die bei Überfahren des Symbols mit der Maus aktiviert wird. Die nachfolgenden Abbildungen zeigen die Liste der Medienelemente im Buch sowie die verschiedenen Symbole.

Abbildung 57: Liste von Komponenten, die einer Buchseite zugeordnet sind

Abbildung	Blendet eine *Abbildung* ein.
Animation	Blendet eine *Animation* ein.
Biografie	Blendet eine *Biografie* ein.
Fotografie	Blendet eine *Fotografie* ein.
Simulation	Blendet eine *Simulation* ein.
Tabelle	Blendet eine *Tabelle* ein.
Übung	Blendet eine *Übung* ein.
Zusatztext	Blendet einen *Zusatztext* ein.
Text links	Schaltet in der Kontextseite von der Darstellung einer Komponente zur Buchseite zurück.

Tabelle 57: Symbole für die Integration von Medienelementen

Soweit wie möglich sind Komponenten so gestaltet, dass sie bei Aktivierung in der Kontextseite des Buchbetrachters angezeigt werden können. Nur für Komponenten, die grundsätzlich einen größeren Darstellungsbereich benötigen, findet eine Aktivierung in einem zusätzlichen Fenster statt. Beispiele für die eingebetteten Komponenten finden sich unten in Kap. 14.1. So ist gewährleistet, dass eine für den Benutzer übersichtliche Fensterverwaltung gegeben ist: Neben dem eigentlichen Buchbetrachtungssystem werden als zusätzliche Fenster jeweils nur die Navigations- und Recherchefenster (s. u.) sowie ein Inhaltsfenster ohne Seitenteilung geöffnet.

13.3.5 Navigations- und Recherchekomponenten

Zusätzliche Zugangswege zu den Inhalten des elektronischen Buchs sind als separate Fenster im Buchbetrachtungssystem realisiert: Die Inhalte des elektronischen Buchs lassen sich über einen Hierarchienavigator ansteuern (Abbildung 58). In ihm sind die Inhaltsebenen des Buchs durch den Benutzer expandierbar (*tree view*), d. h. der Benutzer kann die Gliederungsebenen des Buchs im Hierarchiebrowser expandieren und kollabieren. Dadurch kann er direkt jede Stelle (genauer: jeden Abschnittsbeginn des Buchs) ansteuern. Die Inhaltshierarchie des Buches wird durch Auswertung des Strukturmarkup des elektronischen Buchs gewonnen, d. h. die Einträge für die verschiedenen Kapitelebenen werden aus dem Datenbestand des elektronischen Buchs extrahiert und bilden die Datengrundlage für einen *tree view*-Algorithmus, der mit Hilfe von JavaScript die Steuerung des Buchbetrachtungssystems erlaubt. Dabei hat der Benutzer zusätzlich die Möglichkeit, mit dem Hierarchiebrowser die linke und die rechte Buchseite getrennt anzusteuern (Wechselschalter im Hierarchiebrowser rechts oben). Auf diese Weise können ähnlich wie bei der Entkoppelung der Buchseiten Fokus- und Kontextseite unabhängig von der Seitenabfolge mit Inhalten gefüllt werden.

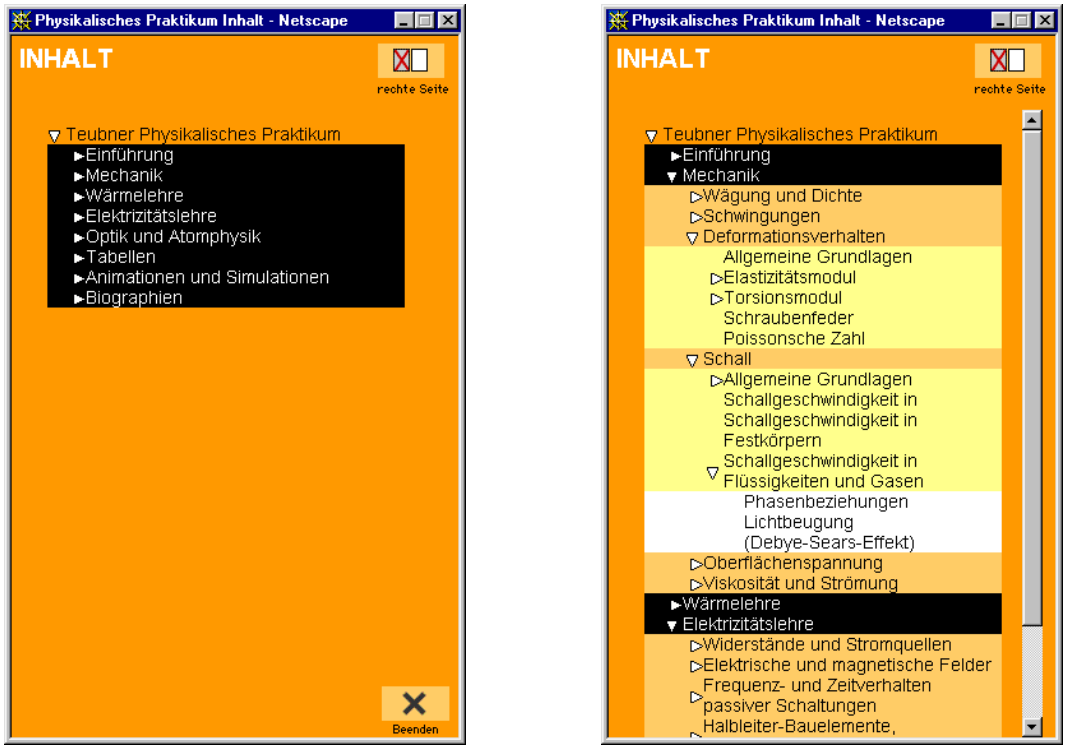

Abbildung 58: Hierarchienavigator (top-Level-Gliederung und expandierte Sicht)

Neben der rein hierarchischen Inhaltsdarstellung (Strukturmarkup) bietet der Hierarchieviewer den Direktzugriff auf Komponenten, d. h. er enthält eine expandierbare Liste aller Simulationen und Animationen sowie der textuellen Zusatzdaten des Buchs, die nicht in die lineare Basisabfolge des Buchtexts eingeordnet sind (Tabellen, Biographien).

Eine zweite Erschließungsmöglichkeit besteht über die auf die Buchinhalte bezogenen Suchfunktionen. Sie sind im Referenzprojekt durch ein *Register* sowie eine *Volltextrecherche* gegeben. Beide werden analog zum Hierarchiebrowser über die Navigationsleiste in einem separaten Fenster aktiviert. Abbildung 59 zeigt die kombinierte Darstellung von Register und Volltextsuche. Die Einträge für das Register werden durch Extraktion begriffsbezogenen inhaltsorientierten Markups automatisch generiert, d. h. ein Skript erzeugt nach Aufteilung der Buchprimärdaten auf einzelne Darstellungseinheiten (erster Transformationsschritt, vgl. oben Kap. 12.3.2) eine Liste von Registereinträgen, die als Hypertextverknüpfungen die Aktivierung der entsprechenden Buchseite im Buchbetrachtungssystem erlauben. Wie im Hierarchiebrowser kann der Benutzer auch hier die rechte und die linke Buchseite getrennt ansteuern. In der Volltextrechercheschnittstelle kann der Benutzer unter Verwendung einer booleschen Anfragelogik eine Suchanfrage stellen und erhält eine Liste passender Buchseiten, die als Hypertextverknüpfung direkt mit den Buchinhalten verbunden sind. Für die Errechnung des Volltextindex sowie die Anfrageverarbeitung kommt das SWISH-E *indexing tool* zum Einsatz (*Simple Web Indexing System for Humans - Enhanced*, vgl. *http://sunsite.berkeley.edu/ SWISH-E/*), das die Indexierung von HTML auf der Basis eines modifizierten booleschen Retrievalmodells erlaubt und das über ein CGI-Skript in das Buch eingebunden ist. Durch die Berücksichtigung von Termfrequenzen bei der Indexierung sind die Suchergebnisse nach Qualität sortiert (*ranking*), was bei einer rein booleschen Zuordnung von Indexierungstermen nicht möglich wäre; zur Informationserschließung siehe unten Kap. 14.2.2.

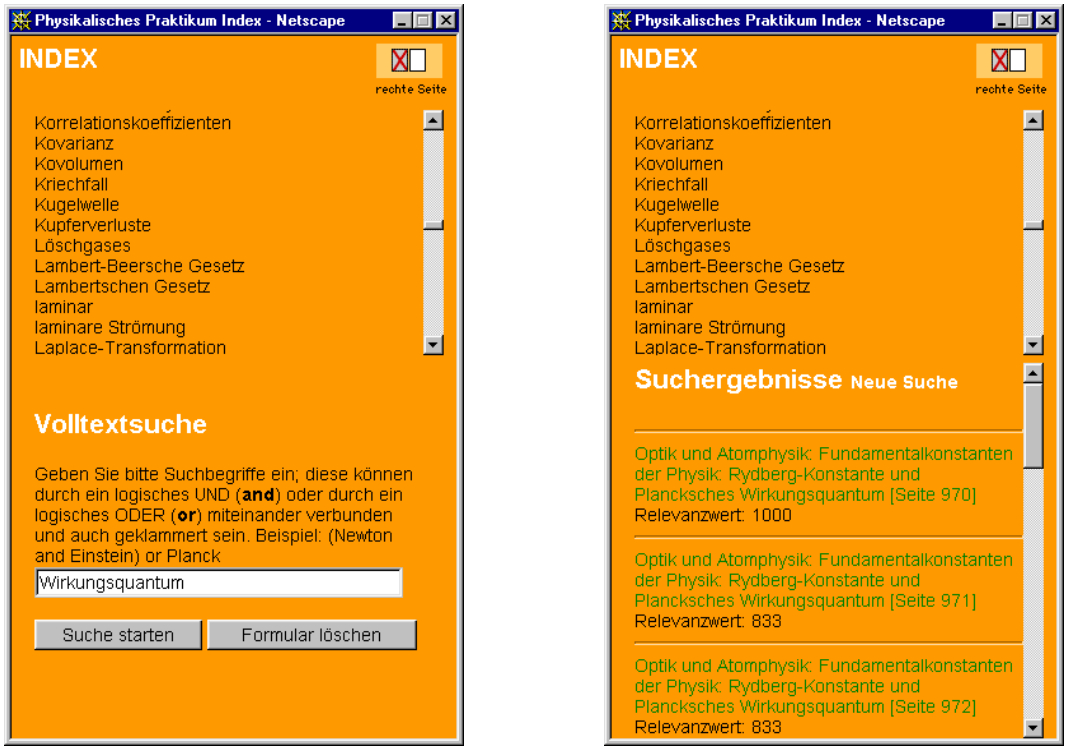

Abbildung 59: Rechercheinterface des Buchs (Anfrage- und Ergebnisdarstellung)

Abschließend zeigt Abbildung 60 im Zusammenhang, wie der Hierarchiebrowser zur Steuerung des Buchs eingesetzt wird; eine analoge Dialogsituation (Steuerungsfenster und Hauptfenster mit den Buchinhalten) ergibt sich auch bei Verwendung des Rechercheinterface.

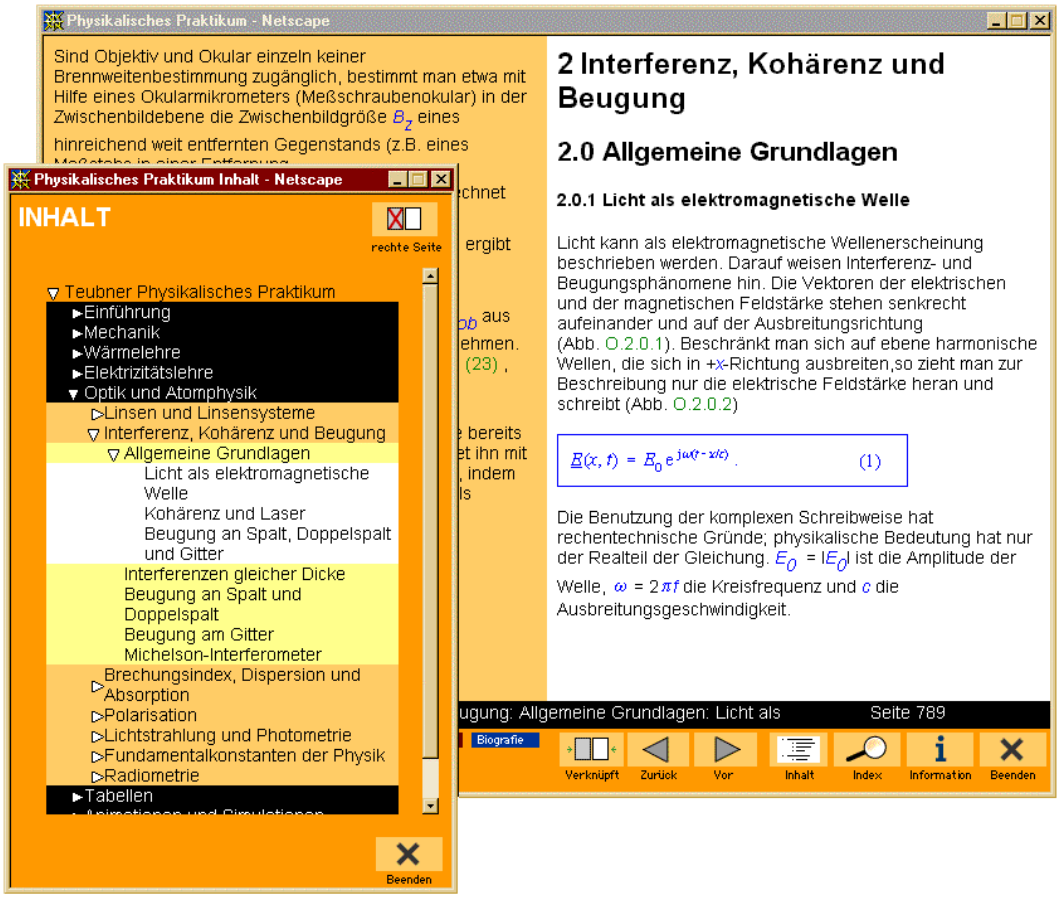

Abbildung 60: Steuerung des Buchs über den Hierarchiebrowser

14 Integration von Komponenten und Diensten

Für das in Kap. 4 eingeführte Architekturmodell für dynamische elektronische Bücher sind Komponenten und Dienste eine wesentliche Erweiterung gegenüber Printprodukten. Daher ist anhand der in Kap. 4 nur konzeptuell eingeführten Typen von Komponenten und Diensten aufzuzeigen,

- welche Komponententypen im Referenzprojekt erstellt wurde und wie sie in das elektronische Buch integriert sind
- und wie die Realisierung des Dienstemodells ausgestaltet sein kann.

14.1 Entwicklung und Integration von Komponenten

Der zweite Inhaltsbestandteil des elektronischen Buchs neben dem eigentlichen Text, die ergänzenden Komponenten, sind hinsichtlich ihrer Präsentation und Integration in das elektronische Buch bewusst weit definiert, d. h. sie umfassen nicht nur für das elektronische Buch neu entwickelte Multimediakomponenten, sondern auch graphische Inhaltsbestandteile des Ausgangswerks (Abbildungen, Diagramme etc.) sowie textuelle Bestandteile (Tabellen, Exkurse, Aufgaben), die sich nicht in das Layoutkonzept des Buchviewers integrieren lassen und die daher eine gesonderte Darstellung erfordern. Die für das elektronische Buch neu generierten Multimediaelemente lassen sich formal, d. h. hinsichtlich ihrer Datenformate bzw. ihrer technischen Realisierung in folgende Typen aufgliedern:

- Photographien,
- textuelle Ergänzungen,
- Java-Applets und
- Macromedia-Shockwave-Anwendungen.

Die letzten beiden Typen stellen die eigentlichen Erweiterungen gegenüber dem gedruckten Buch dar und sind daher ein entscheidender Legitimationspunkt für das multimediale elektronische Buch. Hinsichtlich ihrer Inhalte und Darstellungsform werden folgende Typen unterschieden:

- interaktive Visualisierungen von Versuchsaufbauten,
- Animationen physikalischer Phänomene,
- Animationen von Versuchsabläufen und
- Simulationen physikalischer Phänomene bzw. Geräte.

Die unterschiedlichen Typen der Ergänzungen lassen sich in etwa den folgenden Realisierungsformen zuordnen: Animationen wurden als Macromedia-Anwendungen, Simulationen als Java-Applets realisiert, um die jeweiligen Vorteile eines dedizierten Multimedia-Autorensystems und einer mächtigen objekt-orientierten Programmiersprache ausnutzen zu können.

Für die Erstellung der multimedialen Ergänzungen wurde durch eine Arbeitsgruppe von Physikern und unter Leitung der Buchautoren eine Grobspezifikation erarbeitet (Auswahl geeigneter Multimediakomponenten und ihre Zuordnung zu den Buchinhalten). Diese Grobspezifikationen wurden dann zu Drehbüchern weiterentwickelt, die konkrete Angaben zu den benötigten Medienelementen, der notwendigen Funktionalität und der

Gestaltung der Benutzerschnittstelle einer Komponente enthielten. Auf dieser Basis und unter Berücksichtigung der Designspezifikation erfolgte schließlich die Implementierung der Komponenten. Für die Realisierung der Multimediakomponenten wurden zudem Templates bzw. ein einfaches objektorientiertes Modell entwickelt, auf dessen Basis neue Komponenten realisiert werden können. Insgesamt ließen sich mehr als zwanzig Multimediakomponenten realisieren, wobei deren Komplexität unterschiedlich ist; hinzu kommen vielfältige einfache Medienelemente (Photographien, Text verschiedener Art, s. u.). Bevor anhand von Beispielen die Umsetzung solcher Komponenten diskutiert wird, stellt das folgende Kapitel Technologien für die Realisierung von Softwarekomponenten vor.

14.1.1 Technische Aspekte der Realisierung von Komponenten

Bei der Entwicklung des Architekturmodells in Kap. 4 ist der Begriff *Komponente* in Bezug auf Inhalte und Darstellung eines elektronischen Buchs eingeführt und als Oberbegriff für alle ergänzenden Inhalte des elektronischen Buchs verwendet worden. Nachfolgend soll der *softwaretechnische Aspekt* von Komponenten diskutiert werden. Die Modellierung von Software durch die Spezifikation von Komponenten (*component ware*) ist zwar eine schon länger bekannte, technisch aber erst seit kurzer Zeit breit umgesetzte Idee (vgl. NIERSTRASZ, GIBBS & TSICHRITZIS 1992: 160). Sie hat zum Ziel, Softwarekomponenten auf einem hohen Abstraktionsniveau zu spezifizieren und mit öffentlichen Schnittstellen auszustatten, so dass Komponenten miteinander kombiniert werden können, auch ohne dass ihre Implementierung im Detail bekannt sein muss: Ein Entwickler kann die Komponente in eine Anwendung integrieren, da sie über standardisierte Schnittstellen verfügt. Die Grundkonzepte der objekt-orientierten Programmierung weisen eine für diesen Ansatz zu feine Granularität auf, d. h. man benötigt eine abstraktere Beschreibung der Funktionalität und des Verhaltens von Komponenten. Zu den grundlegenden Eigenschaften einer Softwarekomponente zählen (PIEMONT 1999: 291 ff., MEYER 1999):

- Eine Softwarekomponente ist ein ausführbares Programm,
- sie realisiert eine klar abgegrenzte Programmfunktionalität (z. B. ein GUI-Control oder ein Berechnungsmodul),
- ihre Implementierung ist nach außen nicht transparent (black-box-Modell),
- sie verfügt über öffentliche Schnittstellen, über die auf die Funktionalität der Komponente zugegriffen werden kann (Eigenschaften, Methoden, Ereignisse),
- sie lässt sich in komplexe Programmsysteme integrieren und
- im Unterschied zu Objekten sind Komponenten auf einer abstrakteren Ebene angesiedelt und nur lose miteinander verkoppelt. Dies soll vor allem die einfache Wiederverwendbarkeit von Komponenten gewährleisten.

Zusammenfassend kann man folgende Definition verwenden:

> Eine Softwarekomponente ist ein in sich abgeschlossener Softwarebaustein, der eine Aufgabe aus dem Anwendungsbereich löst und daher für sich allein eine bestimmte Funktionalität bietet. Nach außen hin zeigt sich die Softwarekomponente über eine normierte Schnittstelle. Über diese Schnittstelle kommunizieren Komponenten mit anderen Softwareelementen. [...] Interne Vorgänge innerhalb der Komponente werden jedoch vor den Augen des Entwicklers verborgen (Prinzip des „Information Hiding“). [PIEMONT 1999: 2][127]

[127] Vgl. auch MEYER 1999: 139 „There is no generally accepted definition of components. [...] A software component is a program element with the following properties:
- The element may be used by other program elements (*clients*).

Bisher gibt es keine einheitliche Entwicklungsmethode für *component ware*. Bekannte Konzepte der Objektorientierung werden herangezogen bzw. verallgemeinert (so enthält z. B. die *unified modeling Language* ein Komponentenkonzept, das allerdings nicht sehr differenziert ist, vgl. Booch, RUMBAUGH & JACOBSON 1999: 343 ff.). Für die programmtechnische Realisierung von Komponenten existieren eine Reihe konkurrierender Komponentenarchitekturen, u. a.

- Microsoft ActiveX-Komponenten sowie das Microsoft *component object model* (COM) bzw. das *distributed component object model* (DCOM) als Weiterentwicklung der *object linking and embedding*-Technologie (OLE), vgl. GRAY et al. 1998, EDDON 1999),
- die auf der Basis der Sprache Java entwickelte Komponententechnologie *Java Beans* sowie ihr technologisches Umfeld,
- Standardisierungsversuche für verteilte Objektsysteme der Object Management Group (OMG) im Umfeld von CORBA (*CORBA Business Objects*).[128]

Zusätzlich kann man aufgrund oben genannter Definitionen von Softwarekomponenten auch die mit Multimedia-Autorensystemen erzeugten Anwendungen, die sich in Webbrowser mit Hilfe von plug-in-Software integrieren lassen, als Komponenten auffassen. Dazu zählen auch die im Referenzprojekt mit Macromedia Director erzeugten interaktiven Animationen.

Die standardisierten Komponententechnologien sind wenigstens z. T. untereinander interoperabel, d. h. ein *Java Bean* kann als CORBA-fähiges Objekt ausgestaltet sein oder im Rahmen von ActiveX eingesetzt werden (über eine ActiveX-Bridge, die das Bean als ActiveX-Komponente verpackt). Der Einsatz von *component ware* kann analog zum *three tier*-Modell der Client-Server-Programmierung auf unterschiedlichen Ebenen erfolgen:

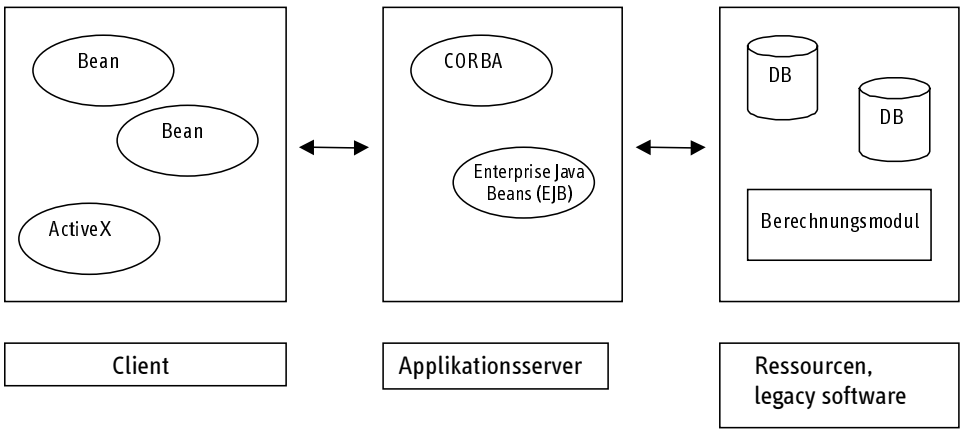

Abbildung 61: Komponenten im three-tier -Modell mit (nach PIEMONT 1999: 296, Abb. 6-1)

Hinsichtlich der Motivation für die Entwicklung interoperabler Komponenten ergibt sich eine bemerkenswerte Übereinstimmung mit dem Einsatz deklarativer Kodierungstechniken im elektronischen Publizieren: Beiden Technologien liegt die Erwartung zugrunde, durch deklarative Auszeichnung bzw. offene Schnittstellen die *Wiederverwendbarkeit* von

- The clients and their authors do not need to be known to the element's authors."

[128] Zum Komponenteneinsatz in CORBA vgl. ORFALI, HARKEY & EDWARDS 1998: 32 ff.; einen Vergleich der wichtigsten Merkmale und Unterschiede dieser drei Ansätze geben KRIEGER & ADLER 1998.

Inhalten bzw. Software zu erreichen (vgl. MEYER & MINGINS 1999). Nachfolgend beschränkt sich die Diskussion der Komponententechnologie auf Java Beans, da im Rahmen dieser Arbeit Java als primäre Entwicklungssprache verwendet wird. Zu den Zielen der Komponentenentwicklung mit Java Beans gehören:

- Die Möglichkeit, Komponenten unterschiedlicher Granularität zu entwickeln, d. h. sowohl Komponenten im Sinne von Bausteinen der Applikationsentwicklung als auch eigenständig zu nutzende Anwendungen, die z. B. zu komplexen Dokumenten kombiniert werden können. In diesem Sinn stellen Beans eine Analogie zu Schnittstellenkonzepten wie OLE (*object linking and embedding*) oder ActiveX dar. Der Regelfall einer Java Bean-Komponente sind Controls mittlerer Größe, für die über die in Java Beans vorgesehenen Schnittstellen ein Maximum an normierter Funktionalität erreicht werden soll.
- Die *Portabilität* der Komponenten als plattformneutrale Anwendungen, ohne dabei die Integration in plattformspezifische Ausführungsumgebungen (z. B. Visual Basic) auszuschließen. Brücken-APIs gewährleisten die Interoperabilität zwischen Java Beans, ActiveX-Komponenten/Microsoft COM und Live Object (*OpenDoc*).
- Ein einheitliches API, das die Obermenge der auf je einer Plattform tatsächlichen Funktionalität spezifiziert. Es muss für jede Plattform gewährleistet sein, dass ein Default-Verhalten für nicht-unterstützte Bean-Funktionalität bereitsteht (Beispiel: Die auf einer bestimmten Plattform nicht mögliche Verschmelzung von Menüs durch Öffnen eines zusätzlichen, räumlich getrennten Menüs ersetzen).

Ein Java Bean ist aufbauend auf diesen Vorgaben wie folgt definiert: „A Java Bean is a reusable software component that can be manipulated visually in a builder tool."[[HAMILTON 1997: 9]. Die Definition impliziert, dass Beans vor allem mit Hilfe visueller IDEs wie Visual Age, Visual Cafe oder Java Workshop erstellt werden können. Die APIs sind aber gleichzeitig für den Programmierer transparent, so dass Beans ohne graphische Entwicklungsumgebungen zu erstellen und zu nutzen sind. Für die Komponentenentwicklung sieht die *Java Beans Specification* folgende Funktionalitätsbereiche vor:

- Eigenschaften (*properties*) und Anpassung (*customization*) von Komponenten,
- Ereignisverarbeitung, um unterschiedliche Komponenten auf einfache Weise miteinander zu verknüpfen und sie kommunizieren zu lassen (*Interoperabilität*),
- Introspektion (*introspection*), so dass ein Entwicklungswerkzeug erkennen kann, wie ein Bean arbeitet und
- *Persistenz* und Verpacken von Komponenten, so dass ein in einem Entwicklungswerkzeug angepasstes bzw. modifiziertes Bean persistent im neuen Zustand gespeichert werden kann.

Beans müssen nicht von einer bestimmten Basisklasse abgeleitet werden, bei sichtbaren Komponenten mit Benutzerschnittstelle gilt lediglich die Voraussetzung, dass sie von der Java-Basisklasse `java.awt.Component` abzuleiten sind. Die wichtigsten Merkmale eines Bean sind:

- seine Eigenschaften (*properties*),
- seine Methoden, die von anderen Komponenten aufgerufen werden können und
- die Ereignisse/Nachrichten, die es auslöst.

Neben der Grundfunktionalität von Java Beans haben sich eine Reihe weiterführender Technologien entwickelt:

- die *InfoBus*-Architektur als flexibler Softwarebus für Komponenten (vgl. COLAN 1998),
- das *Java Activation Framework* (JAF, vgl. CALDER & SHANNON 1999), das es erlaubt, in Java Beans-Komponenten die Aktivierung externer Komponenten und Programme für unterschiedliche Dokumenttypen nach der MIME-Spezifikation (*multipurpose internet mail extensions*, RFC 1521) zu integrieren, eine Funktionalität, die sich z. B. nutzen lässt, um aus einer in ein elektronisches Buch eingebetteten Komponente externe Anwendungen (Dienste) für die Weiterverarbeitung von Daten zu spezifizieren,
- das `BeanContext`-Konzept, das es ermöglicht, Komponenten zu bündeln und einen einheitlichen Zugriff auf eine Mehrzuahl von Komponenten zu ermöglichen und
- *Enterprise Java Beans* (EJB) als Weiterentwicklung des Komponentenkonzepts für den Einsatz im Rahmen verteilter Anwendungen v. a. in Unternehmen und auf Applikationsservern (vgl. PIEMONT 1999: 279 ff., MATENA & HAPNER 1999) mit dem Ziel, EJB als „standard component architecture for building distributed object-oriented business applications in the Java™ programming language." zu etablieren (MATENA & HAPNER 1999: 19). Um die Interoperabilität von EJB zu unterstützen, sind für die EJB-APIs auch CORBA-Mappings vorhanden.

Eine naheliegende Kombination deklarativer Markupsprachen wie XML und Komponententechnologie wie Java Beans ist die Verwendung von XML für die Parametrisierung von Komponenten, z. B. für den Aufbau der Benutzerschnittstelle: Mit Hilfe von XML lassen sich etwa Menüstrukturen beschreiben. Eine solche Spezifikation kann von einem XML-Parser innerhalb der Komponente analysiert und die Benutzerschnittstelle entsprechend konfiguriert werden (vgl. JOHNSON 1999, 2000).

Im Rahmen dieser Arbeit werden mit Java entwickelte Komponenten als *Applets* realisiert. Ein Applet ist eine von der Klasse `java.applet.Applet` des Klassensystems von Java abgeleitete Softwarekomponente, die als ausführbares Programm in eine HTML-Seite eingebettet und von einem Webbrowser mit integriertem Java-Interpreter ausgeführt werden kann. Da die Ausgangsklasse `Applet` von `java.awt.component` abgeleitet ist, erfüllen Applets grundsätzlich die Voraussetzungen, als Komponente im Sinne der *Java Beans Specification* betrachtet zu werden. Da sie über eine vordefinierte Schnittstelle zur einbettenden HTML-Seite verfügen (`PARAM`-Elemente innerhalb der `APPLET`-Marke), über die Parameter eingelesen werden können, kommt nicht XML, sondern die Parameterschnittstelle der Appletklasse zum Einsatz (vgl. unten Kap. 14.2.4). Die Beispielapplets physikalischer Simulationen sind dabei nicht unmittelbar von der Klasse `Applet` abgeleitet, sondern von einer Zwischenklasse, die zusätzlich zu den Grundfunktionen von Applets die notwendigen Routinen für die Parametrisierung der Komponenten durch Angabe von Diensten enthält.

14.1.2 Gestalterische und didaktische Aspekte interaktiver Komponenten

Bevor einzelne Typen interaktiver Komponenten anhand von Beispielen aus dem Referenzprojekt vorgestellt werden, ist grundsätzlich zu fragen, welche Motivation und Umsetzungsstrategie ihre Entwicklung rechtfertigen. An erster Stelle steht die *technologische Realisierbarkeit* von Multimediatechnologie, denn interaktive Simulationen und zeitabhängige Medienpräsentation sind ein für das traditionelle Buch prinzipiell nicht zur Verfügung stehendes Gestaltungselement. Damit ist noch kein prinzipieller Mehrwert verbunden, wenn nicht deutlich wird, welcher konkrete Zweck mit der Realisierung einer Multimediakomponente verbunden ist. Im Rahmen eines elektronischen Lehrbuchs ergibt

sich der konkrete Zweck aus den Inhalten des Buchs einerseits, aus den zu lösenden Bildungsaufgaben und den Lerntätigkeiten des Benutzers andererseits. Dabei ist allerdings nur der zweite Aspekt für eine Klasse elektronischer Bücher, d. h. für elektronische Lehrwerke verallgemeinerbar. GIRMES 1999B: 21 ff. unterscheidet in Hinblick auf Lehr-/Lernsoftware

- *Bildungsaufgaben* (Interesse wecken (1), Aktivieren (2) und Reflexion anregen (3)),
- *elementare Lerntätigkeiten* des Benutzers, die sich den Bildungsaufgaben zuordnen lassen (erkennen (zu Bildungsaufgabe 1), teilnehmen (zu 1), reproduzieren (zu 2), gestalten (zu 2), handeln (zu 2), denken (zu 3), urteilen (zu 3), entscheiden (zu 3)) und
- *Lehraufgaben* der Lernsoftware (motivieren, informieren, üben, anwenden, diagnostizieren).

Diese drei Aspekte können verwandt werden, um zu klären, welchen didaktischen Mehrwert die im Referenzprojekt entwickelten Komponententypen im Rahmen eines elektronischen Buchs übernehmen können. Dabei sollen hier nur Komponenten betrachtet werden, die als interaktive Softwaremodule diejenigen Inhalte eines dynamischen elektronischen Buchs darstellen, die in traditionellen Medien grundsätzlich nicht verfügbar sind (vgl. ZIEMS & NEUMANN 1999: Abb. 8-10, S. 131 ff.):

Komponententyp	*Bildungsaufgabe*	*Lerntätigkeit*	*Lehraufgabe*
Interaktive Visualisierungen von Versuchsaufbauten	Interesse wecken	erkennen	motivieren, informieren
Animationen physikalischer Phänomene	Reflexion anregen	denken, urteilen	motivieren, informieren
Simulationen physikalischer Phänomene	Aktivieren, Reflexion anregen	reproduzieren, handeln, denken, urteilen	üben, diagnostizieren
Simulationen von Arbeitsgeräten	Interesse wecken, Aktivieren	erkennen, handeln	üben, anwenden

Tabelle 58: Interaktive Komponenten und ihre didaktischen Aufgaben

Wie die voranstehende Tabelle zeigt, ergibt sich ein Unterschied zwischen Komponenten, deren Aufgabe stärker auf die eigentliche Versuchsvorbereitung ausgerichtet ist (Realweltbezug; gegenständliche Visualisierung von Versuchsaufbauten, Simulation von Geräten) und Komponenten, die stellvertretend für die Versuchsausführung Phänomene erklären oder simulieren (Animationen und Simulationen physikalischer Phänomene und Experimente).

Die interaktiven Komponenten können nur einen Teil der didaktischen Aufgaben eines elektronischen Lehrbuchs übernehmen; sie sind daher immer im Zusammenhang mit dem Ausgangstext sowie den weiteren – nicht interaktiven – Medienelementen (Abbildungen, Schemata, Funktionsgraphen etc.) zu beurteilen. Außerdem ist zu beachten, dass zusätzliche didaktische Funktionen auch durch die zu einer interaktiven Komponente verfügbaren *Dienste* im Sinne des in dieser Arbeit diskutierten Modells dynamischer Bücher bereitgestellt werden können. Zwei Beispiele sollen dies veranschaulichen:

- Steht für die Weiterverarbeitung einer Gleichung ein Symbolmanipulationssystem als Dienst zur Verfügung, so kann neben dem *Erkennen* als Verständnis der theoretischen Beschreibung eines Zusammenhangs auch das Üben und Anwenden als Lehraufgabe erfüllt werden, wenn der Benutzer die Gleichung nicht nur rezipieren, sondern mit ihr arbeiten und z. B. einen Zusammenhang anhand exemplarischer Daten durchrechnen kann.

- Ist eine Kommunikationsschnittstelle zu einem gruppenbezogenen Kontext als Dienst verfügbar (z. B. als E-Mail-Adresse eines Tutors), so kann der Aspekt des *Anwendens* und *Übens* dahingehend erweitert werden, dass für die Beurteilung und Bewertung z. B. von im elektronischen Buch bereitgestellten Übungsaufgaben auch eine elektronische Prozesskette zur Verfügung steht.

Neben den didaktischen Aspekten, die interaktive Komponenten auszeichnen, ist eine zweite Beurteilungsebene die der gestalterischen Umsetzung. Dies betrifft zum einen die bereits in Kap. 13.3.2 angesprochenen *allgemeinen Gestaltungsparameter*, die nicht nur für das Buchbetrachtungssystem und die Darstellung des Buchtexts i. e. S., sondern auch für die Gestaltung der Benutzerschnittstelle der eingebetteten Komponenten gelten, zum anderen die Frage, welche Prinzipien der Visualisierung für die Umsetzung der Komponenten herangezogen werden. Die folgenden Kriterien können für die Beurteilung herangezogen werden:

- Die verwendete *Visualisierungsstrategie*, d. h. die Frage, ob die multimedialen Inhalte einer Komponente ihren Gegenstandsbereich durch Bilder (oder Bewegtbildsequenzen) unmittelbar *abbilden* („gegenständliche Darstellung“) oder durch die Wahl einer abstrakten visuellen Sprache schematisch darstellen (vgl. WOLFF 1996: 81 ff., REICHENBERGER & STEINMETZ 1999).
- Die *Interaktivität* und *Parametrisierbarkeit* einer Komponente, d. h. die Frage nach Umfang der Interaktionsmöglichkeiten des Benutzers und nach Art und Umfang der Einstellungsmöglichkeiten.

In einer interaktiven Komponente können unterschiedliche Visualisierungsstrategien miteinander kombiniert sein: Eine im Referenzprojekt entwickelte Animation zum *Hall-Effekt* zeigt dem Benutzer z. B. zunächst konkret den Versuchsaufbau (gegenständliche Darstellung) und visualisiert anschließend *schematisch* die bei der Versuchsausführung auftretenden Effekte (s. u. Kap. 14.1.4, Abbildung 67 ff.).

Hinsichtlich der Interaktivität und Parametrisierbarkeit lassen sich *Simulationen* und *Animationen* unterscheiden: Bei Animationen ist der Schwerpunkt auf eine sequentielle Erläuterung eines Phänomens oder Versuchsaufbaus gelegt, bei der dem Benutzer nur relativ wenige Eingriffsmöglichkeiten zur Verfügung stehen (Navigationssteuerung, Wahl fest vordefinierter Parameterwerte). Im Gegensatz dazu ist es ein Kennzeichen einer Simulation, eine möglichst umfassende Steuerbarkeit durch den Benutzer zuzulassen. Eine dritte Beurteilungsebene für Komponenten ist die Frage, inwiefern sie

- durch ihren modularen Aufbau im Sinne der Softwaretechnologie dazu geeignet sind, in verschiedenen Szenarien zum Einsatz zu kommen bzw. mit anderen Komponenten kombiniert werden können und
- inwieweit sich Anknüpfungspunkte für die an das Buch angebundenen Dienste ergeben.

Beide Aspekte sind vornehmlich für Simulationen relevant. Die nachfolgenden Abschnitte stellen Beispiele interaktiver Komponenten aus dem Referenzprojekt vor und sollen die dort erfolgte Typisierung von Komponenten („multimediale Ergänzungen“) verdeutlichen.

14.1.3 Interaktive Visualisierungen von Versuchsaufbauten

Die einfachste Form interaktiver Komponenten im Referenzprojekt sind Visualisierungen von Versuchsaufbauten. Dabei handelt es sich um die Illustration der Bestandteile von Versuchskonfigurationen, die die im gedruckten Buch enthaltenen schematischen Darstellungen ergänzen. Ihre *Interaktivität* beschränkt sich darauf, dass der Benutzer sich Erläuterungen zu den einzelnen Bestandteilen des Versuchsaufbaus anzeigen lassen und zwischen verschiedenen Sichten wechseln kann (unterschiedliche Perspektive, Darstellung von Ausschnitten). Das didaktische Ziel ist hierbei, dem Leser neben der abstrahierenden schematischen Darstellung Informationen über den tatsächlichen Aufbau eines Versuchs im Physiklabor an die Hand zu geben und ein besseres Verständnis für die bevorstehende Versuchsdurchführung zu vermitteln. Die nachfolgenden Abbildungen illustrieren dies am Beispiel des Versuchsaufbaus *Wärmepumpe*, Abbildung 65 zeigt zusätzlich die aus dem gedruckten Ausgangswerk übernommene schematische Darstellung des Versuchsaufbaus.

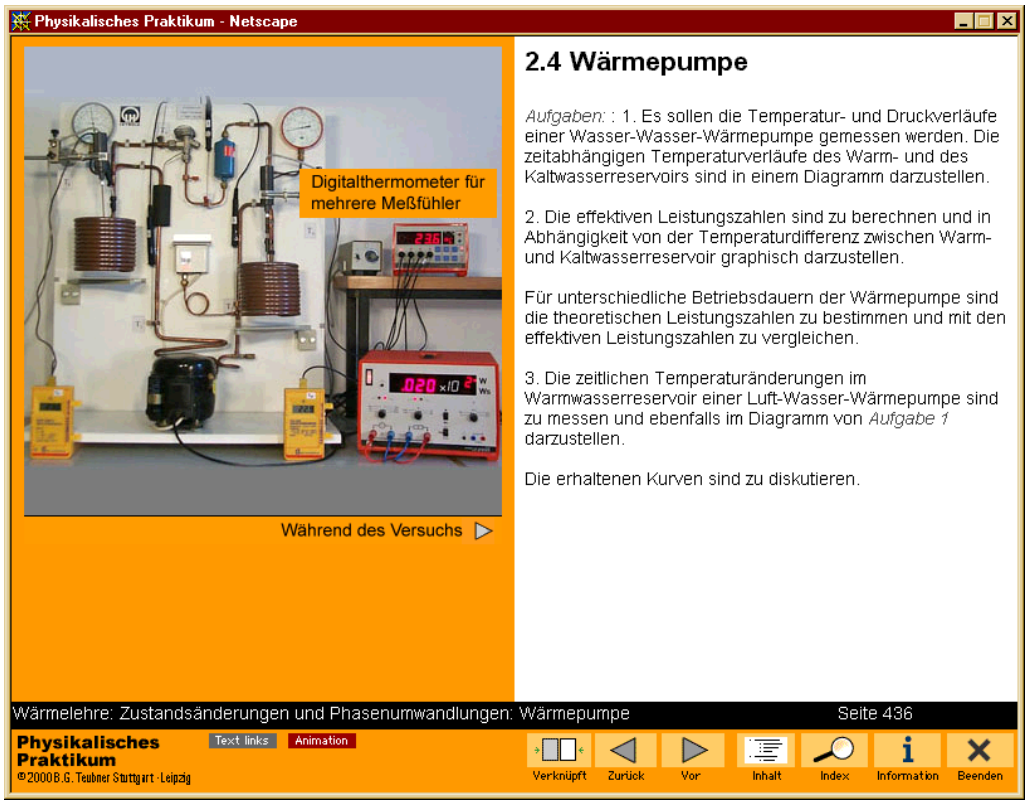

Abbildung 62: Versuchsaufbau Wärmepumpe: Gesamtansicht

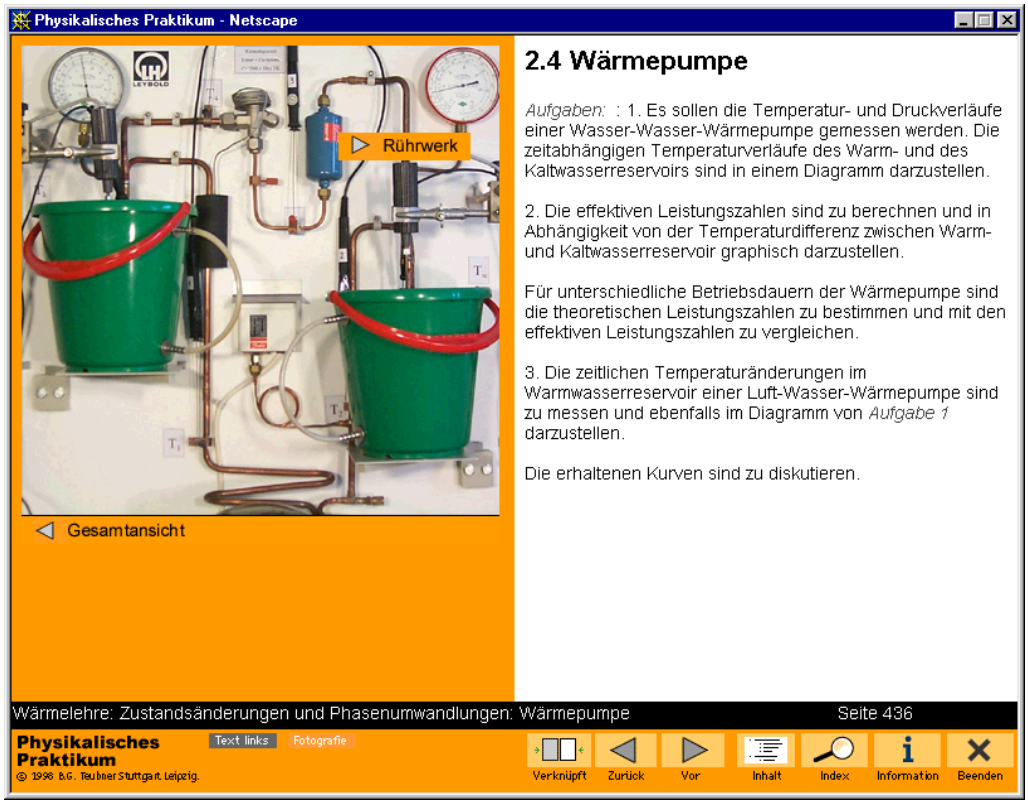

Abbildung 63: Versuchsaufbau Wärmepumpe während der Versuchsausführung

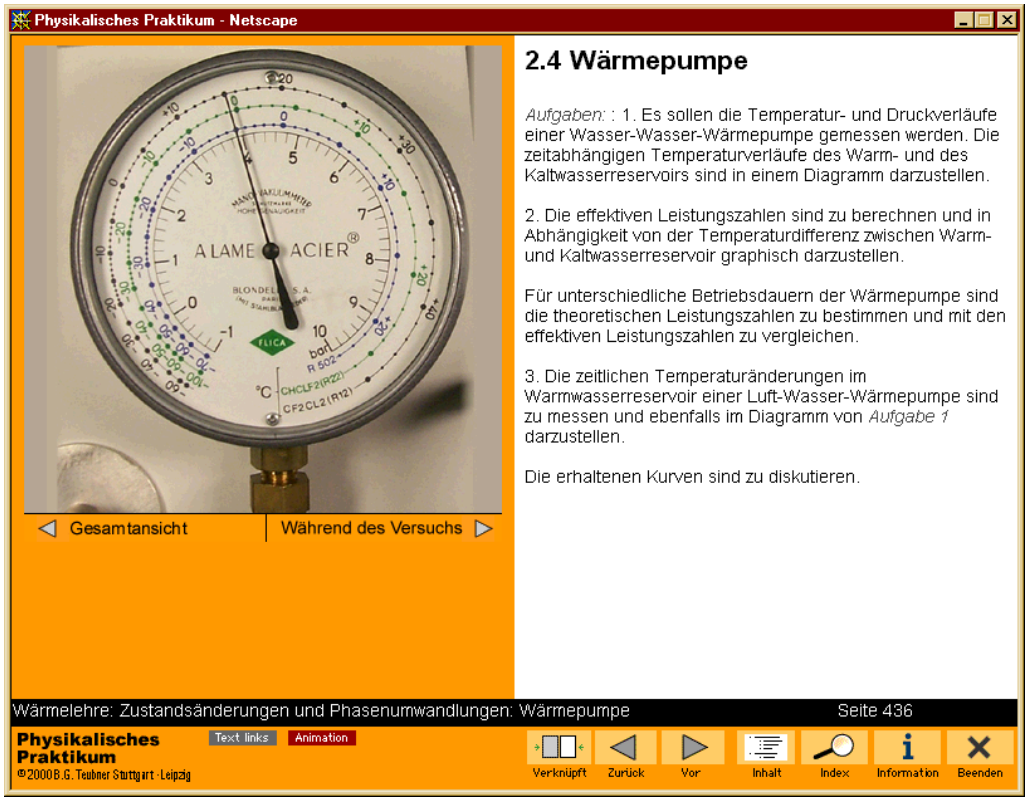

Abbildung 64: Detailansicht zum Versuchsaufbau Wärmepumpe

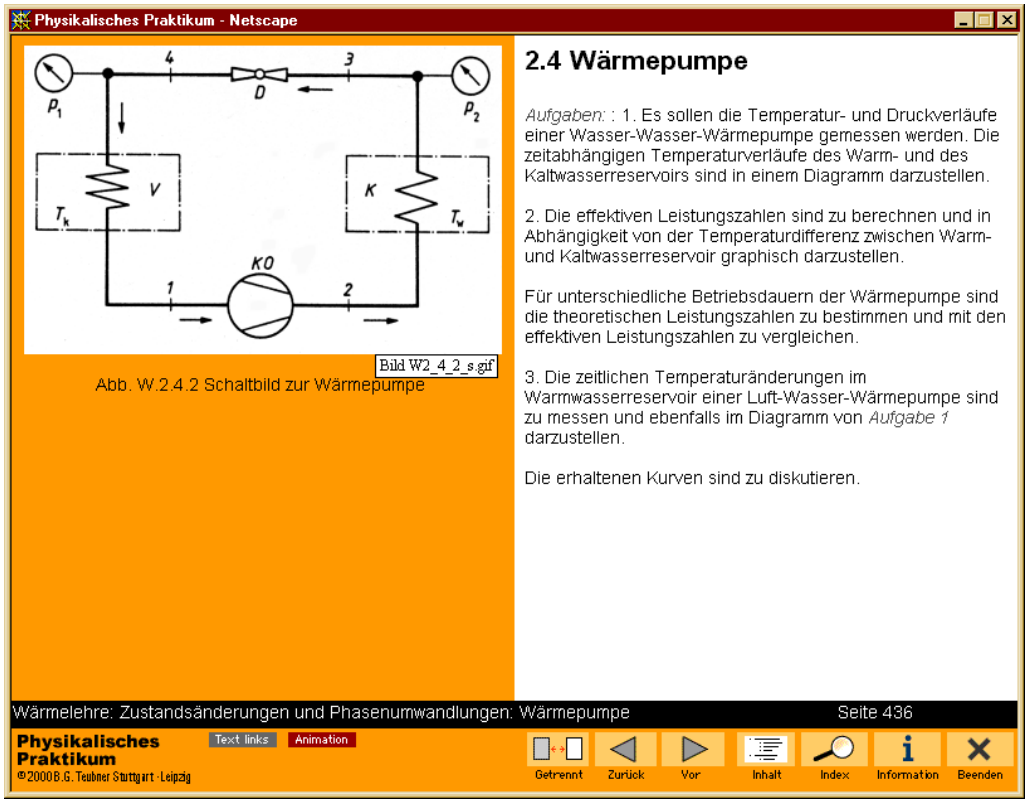

Abbildung 65: Schematische Darstellung des Versuchsaufbaus Wärmepumpe

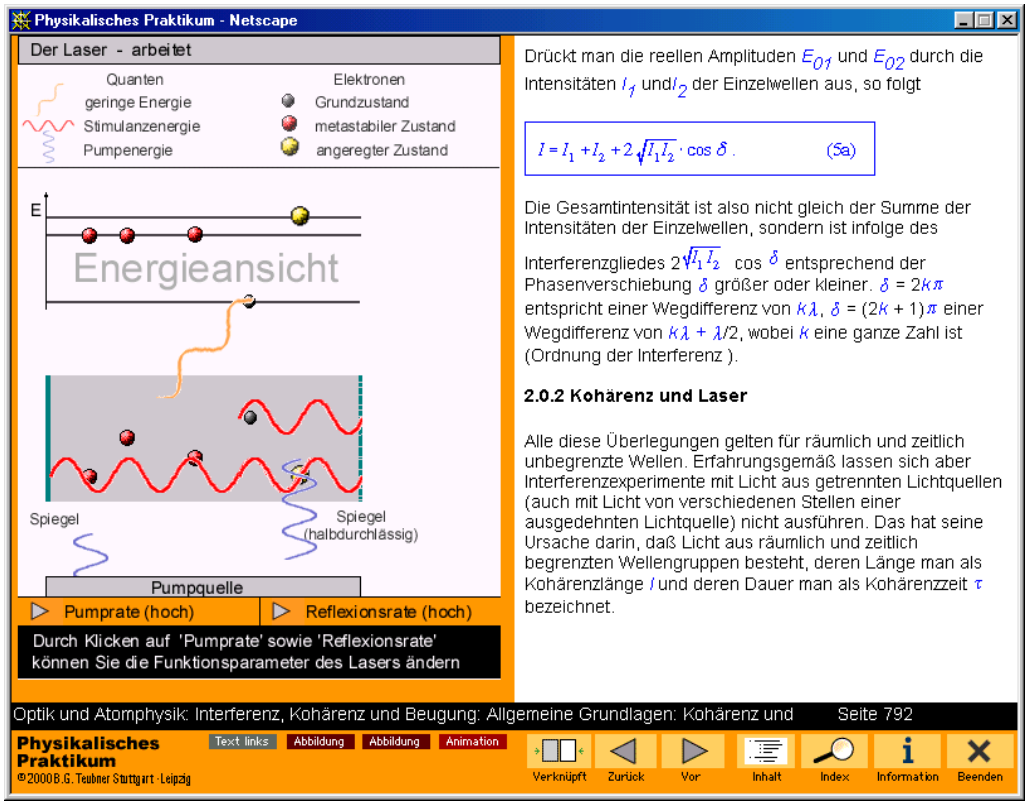

Abbildung 66: Animation eines Lasers

14.1.4 Animationen physikalischer Phänomene

Unter einer Animation ist eine interaktive Abfolge visueller Darstellungen zu verstehen, die ein physikalisches Phänomen darstellt und erläutert und sich einer abstrahierenden graphischen Darstellungssprache bedient. Im Unterschied zur Simulation erfolgt keine rechnerische Modellierung des beschriebenen physikalischen Phänomens. Der Schwerpunkt liegt auf der Herausarbeitung grundlegender Abläufe oder Parameter, die Steuerungsmöglichkeiten durch den Benutzer sind daher eingeschränkt, d. h. der Benutzer kann nur wenige Parameterwerte modifizieren.

In dem in Abbildung 66 gezeigten Beispiel der animierten Darstellung der Funktionsweise eines Lasers kann der Benutzer die Parameter *Pumprate* und *Reflexionsrate* jeweils in drei Stufen einstellen (niedrig, mittel, hoch), die Animation zeigt dann die Auswirkung der Parametersetzung bezüglich des Energieniveaus und des vom Laser ausgestrahlten Lichts. Dabei werden Aufbau und Funktionsweise schematisch visualisiert, d. h. die Animation abstrahiert vom Gegenständlichen, um die auftretenden Phänomene darstellen zu können.

Ein zweites Beispiel, die Animation des Hall-Effekts, zeigt die Kombination aus einer gegenständlichen Darstellungsweise und einer schematischen Animation eines physikalischen Phänomens. Dieser Typus kombiniert damit die Erläuterung eines Versuchsaufbaus (s. o.) mit der Illustration des zugrundeliegenden Phänomens, die durch eine abstrakte visuelle Sprache erreicht wird. Wiederum kann der Benutzer nur zwei Parameter (Magnetfeldrichtung und Stromrichtung) ändern und die sich für den Elektronenfluss ergebenden Auswirkungen in der Animation betrachten. Die Abbildungen zu dieser Animation zeigen den Gesamtaufbau des Versuchs (Abbildung 67), eine durch einen Zoom-Effekt erreichte Ausschnittsvergößerung der Halbleiterprobe (Abbildung 68), die schematische Animation des Hall-Effekts (Abbildung 69) sowie die zusätzlich als Komponente vorhandene und aus dem gedruckten Werk übernommene schematische Abbildung zum Hall-Effekt (Abbildung 70).

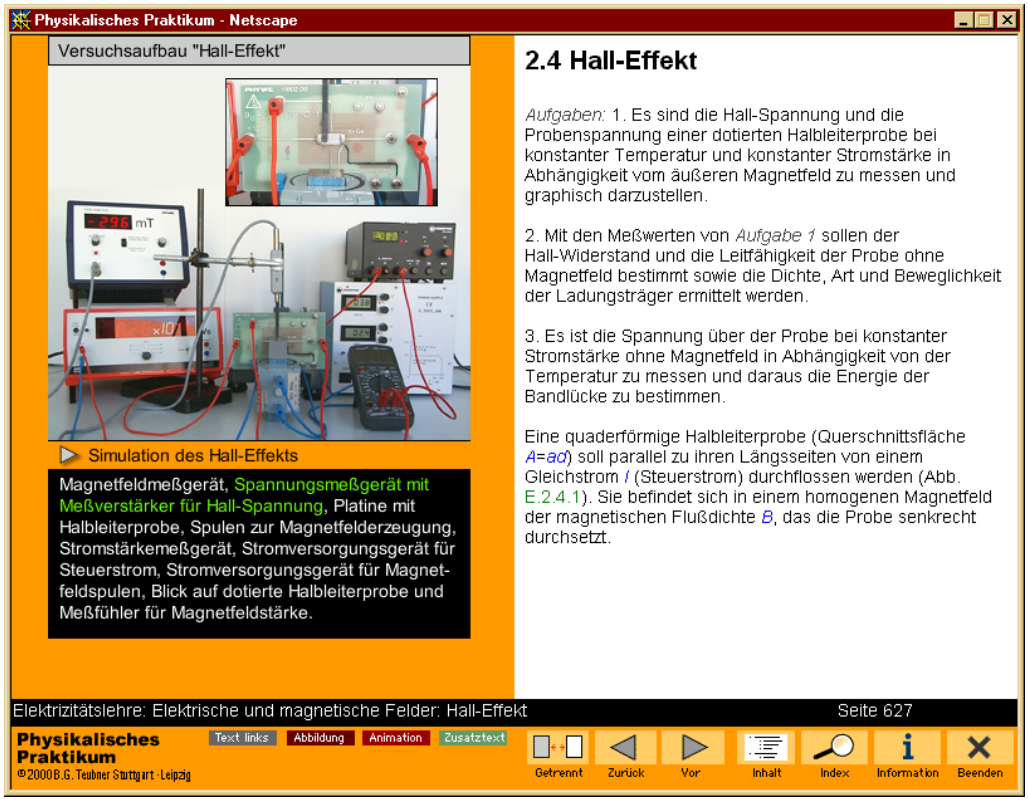

Abbildung 67: Animation zum Hall-Effekt: Versuchsaufbau

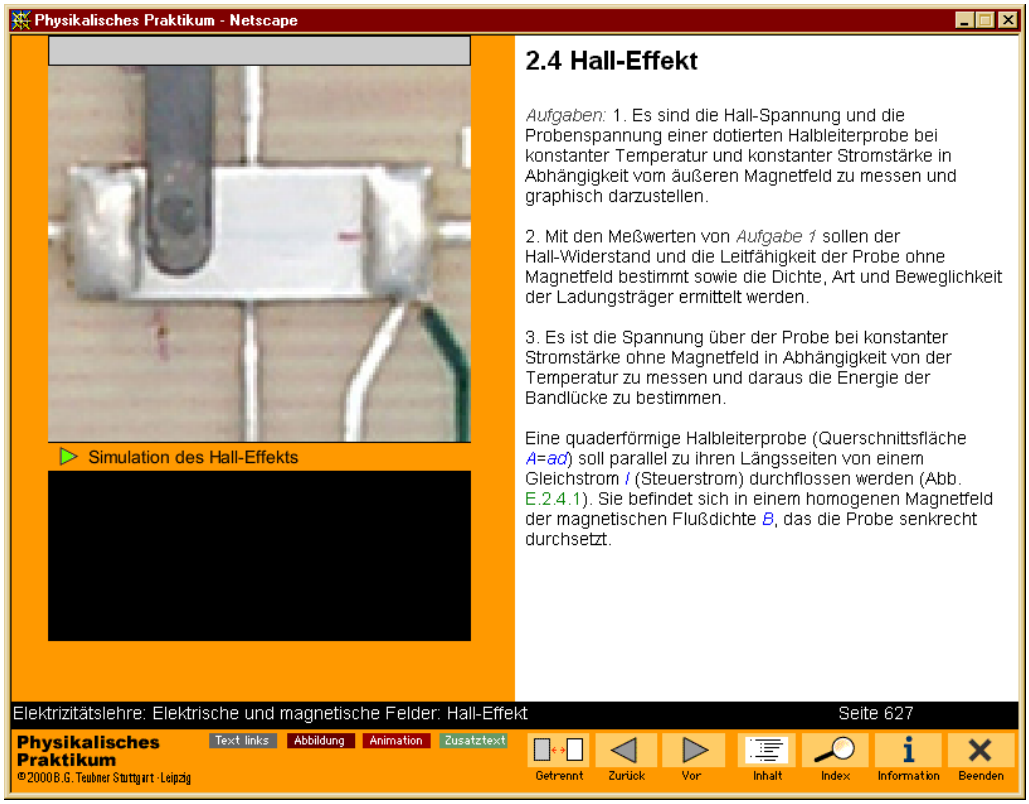

Abbildung 68: Animation zum Hall-Effekt: Ausschnittsdarstellung

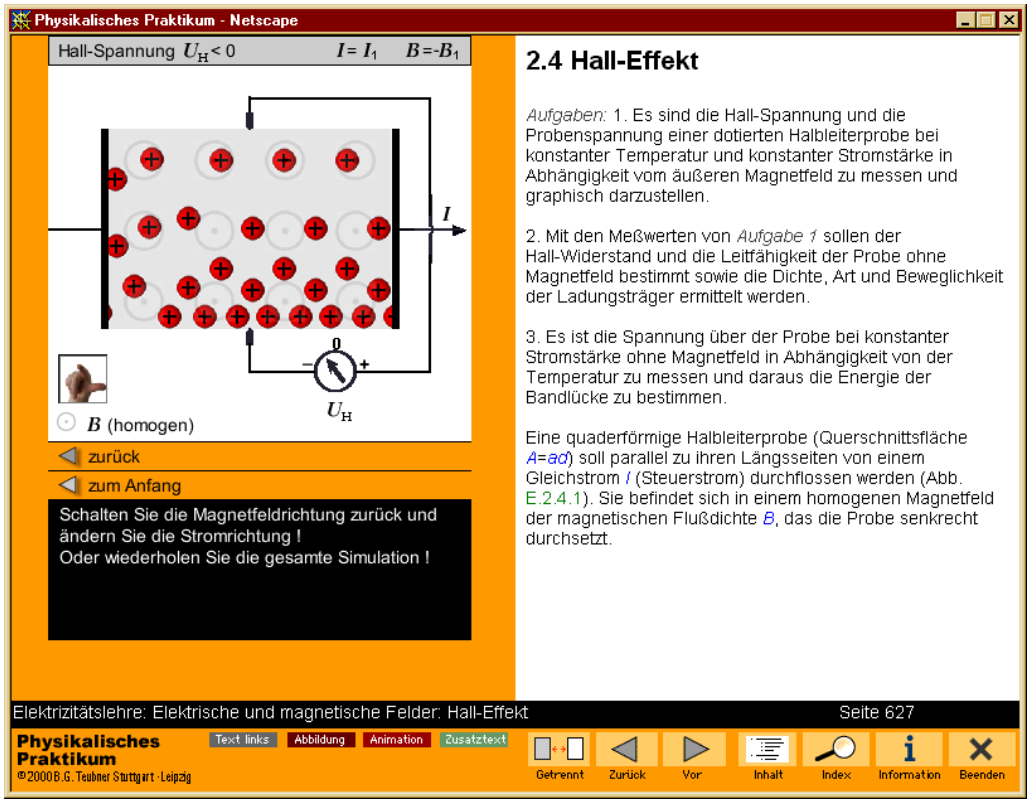

Abbildung 69: Schematische Animation des Hall-Effekts

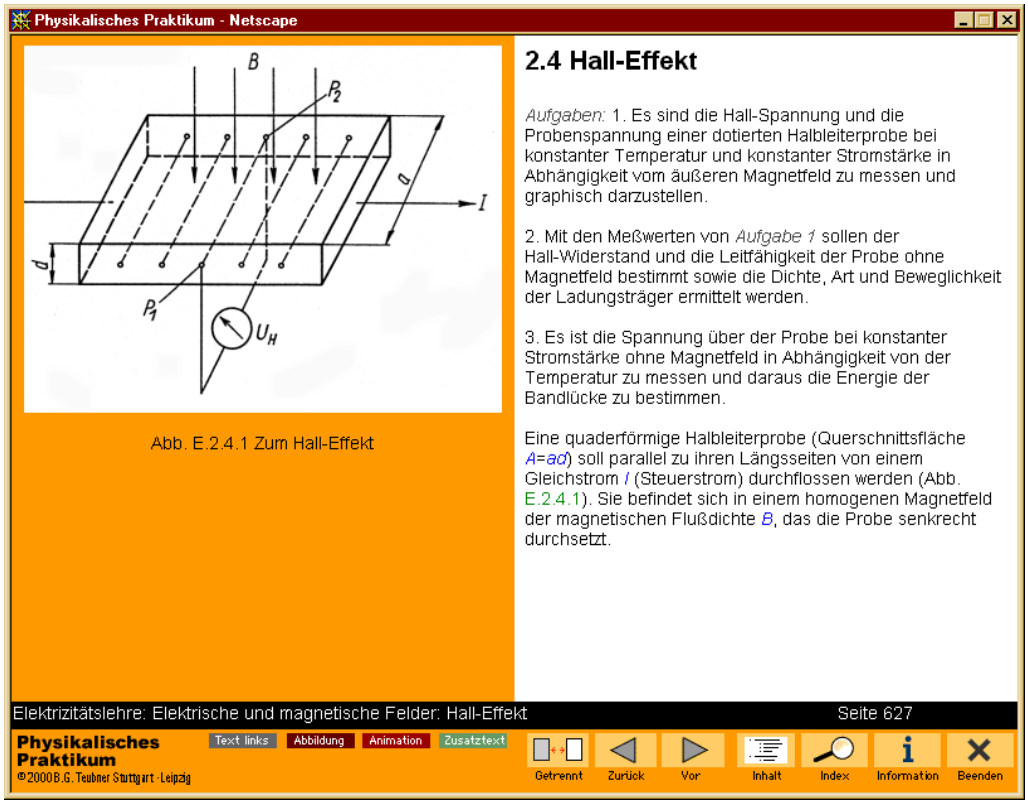

Abbildung 70: Schematische Darstellung des Hall-Effekts

14.1.5 Animationen von Versuchsabläufen

Ein dritter Typus interaktiver Komponenten, die Animation von Versuchsabläufen, ist hinsichtlich seiner Visualisierungsstrategie stärker am Gegenständlichen orientiert, verzichtet aber anders als die Darstellung von Versuchsaufbauten auf die Darstellung aller Details eines Versuchsaufbaus. Diese Form der Animation zeigt in vereinfachter Form die tatsächliche Versuchsausführung, wobei dem Benutzer nur beschränkte Parametrisierungsmöglichkeiten zur Verfügung stehen, d. h. er kann die Animation in ihrem Ablauf nur bedingt beeinflussen. Der Interaktionsverlauf ist von der zeitlichen Abfolge der Arbeitsschritte bei der Versuchsausführung geprägt. Dabei kann der Benutzer entweder unmittelbar zum nächsten Schritt übergehen (einfache Abfolge) oder er muss in der Animation durch Interaktion erst die Voraussetzung für den nächsten Schritt schaffen: Abbildung 72 und Abbildung 73 zeigen eine Animation zur Dichtebestimmung eines Festkörpers mit einem Pyknometer (v. griech. πυκνός: dicht). Um zum jeweils nächsten Schritt zu gelangen, muss der Benutzer die erforderliche Handlung durch Interaktion mit der Animation durchführen (z. B. Wiegen des Festkörpers, Ermittlung der aktuellen Temperatur durch Einbringen des Thermometers in das Pyknometer etc.). Das didaktische Ziel dieses Animationstyps ist es, in stärkerem Maß als bei der einfachen Darstellung eines Versuchsaufbaus die tatsächliche Durchführung von Versuchen als Folge einzelner Handlungsschritte zu visualisieren.

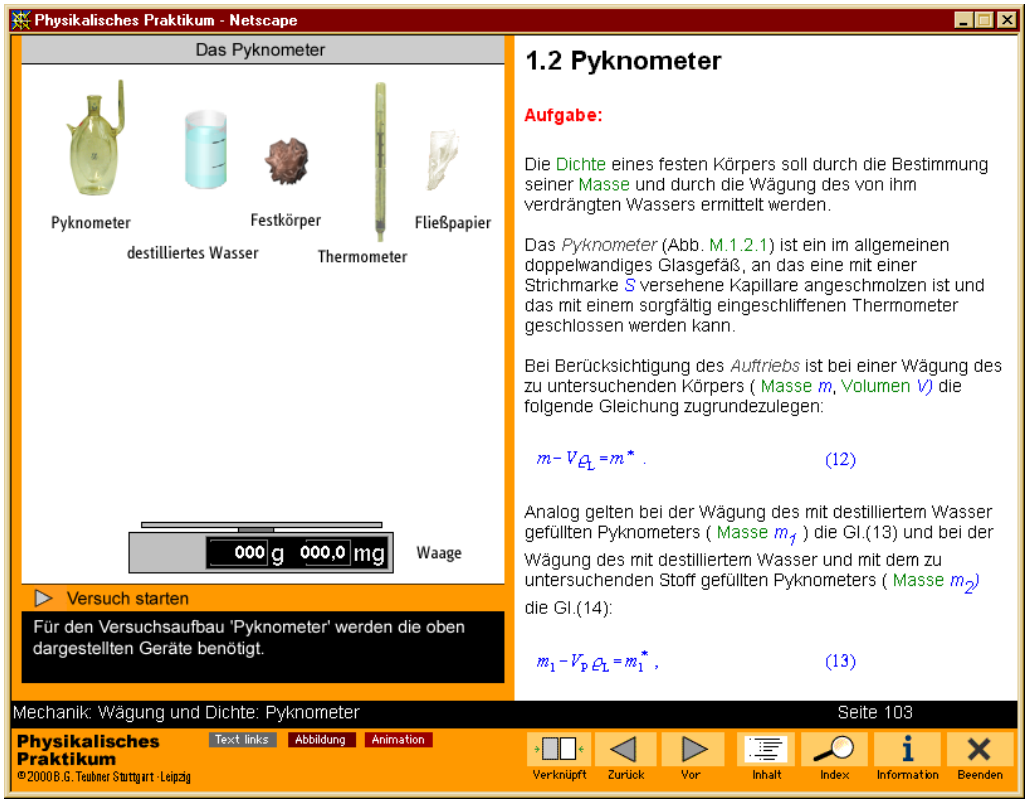

Abbildung 71: Animation zum Pyknometer: Ausgangssituation

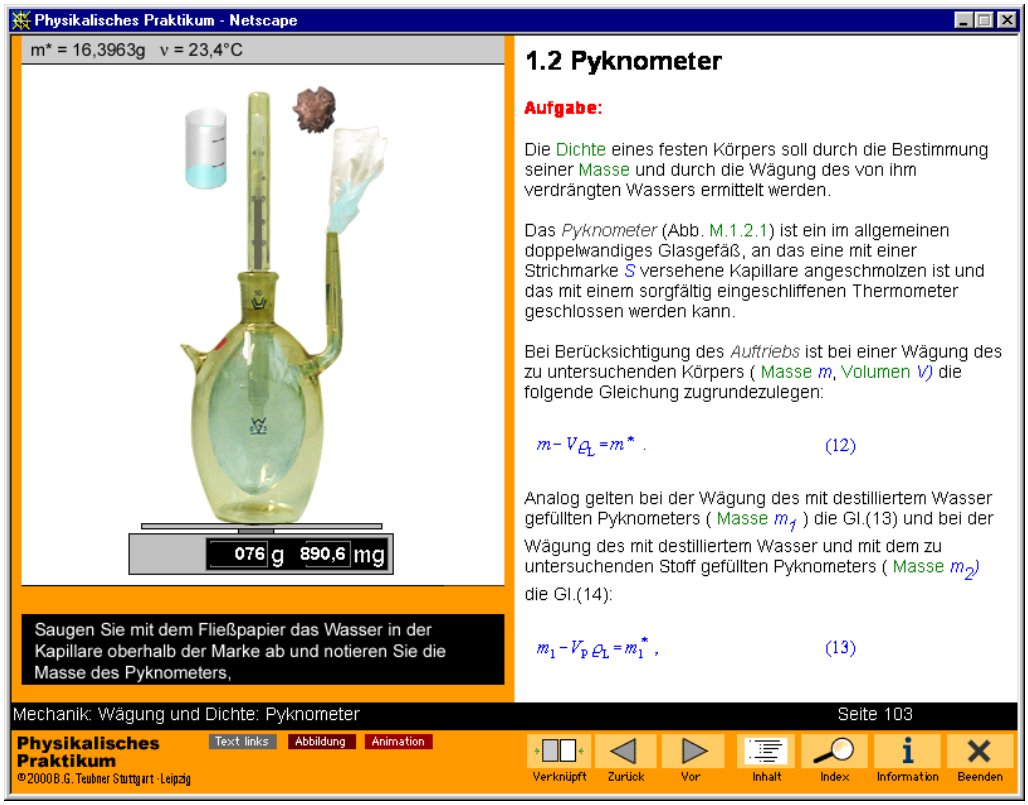

Abbildung 72: Animation zum Pyknometer während der Versuchsausführung

14.1.6 Simulationen physikalischer Phänomene und Geräte

Schließlich wurden *Simulationen physikalischer Phänomene* bzw. *Geräte* entwickelt, bei denen der Benutzer parametergesteuert einen Versuch unter unterschiedlichen Randbedingungen ausführen und simulierte Messergebnisse berechnen kann bzw. den Umgang mit den für die Versuchsausführung erforderlichen Geräten üben kann. Grundlage der Versuchssimulation ist jeweils eine rechnerische Approximation des zugrunde liegenden physikalischen Phänomens. In ihr ist der wesentliche Unterschied zur Animation zu sehen, in der keine parametrisierbare Berechnung erfolgt. Je nach Anwendungsfall sind die Simulationen hinsichtlich ihrer visuellen Darstellung stärker an dem tatsächlichen Versuchsaufbau orientiert (Millikan-Versuch zur Elementarladung) bzw. abstrahieren in ihrer Darstellungsweise vom tatsächlichen Versuchsaufbau (Pendelversuche, Logikbaukasten). Abbildung 73 zeigt die Benutzerschnittstelle der Simulation eines einfachen Pendels. Die Parameter *Pendellänge*, *Auslenkung*, *Dämpfung* und *Masse* können vom Benutzer eingestellt werden, während der Versuchsausführung kann im unteren Teil der Simulation wahlweise die *Auslenkung*, die *Energie* und der *Phasenraum* graphisch dargestellt werden. Zusätzlich besteht die Möglichkeit, eine vergleichende Messung durchzuführen, bei der die Zeitdauer für je 10 Pendeldurchgänge am Nullpunkt bzw. am Umkehrpunkt vom Benutzer ermittelt wird.

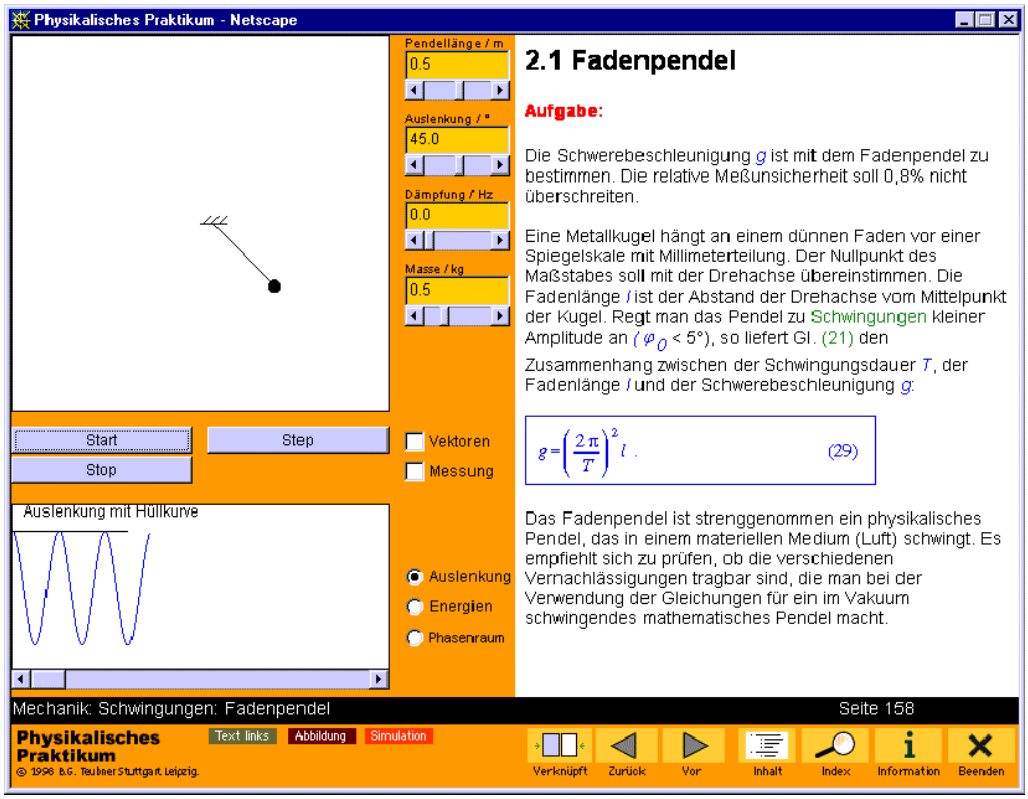

Abbildung 73: Simulation eines einfachen Pendels

Am Beispiel des Elektronenstrahloszilloskops wird ein komplexes Messgerät simuliert, das bei einer Vielzahl von Versuchen eine Rolle spielt. In diesem Fall wird die realitätsnahe Darstellung der Benutzerschnittstelle als direkte Abbildung der Bedienungselemente mit einer schematischen, formularbasierten Eingabeschnittstelle für die Parametrisierung abstrakter Tongeneratoren als Datenlieferanten verbunden. Abbildung 74 und Abbildung 75 zeigen die Simulation des Oszilloskops mit Darstellung der Bedienelemente bzw. mit dem Eingabeformular für die Einstellung der Tongeneratoren. Im Display ist die durch das Verhältnis der Generatorfrequenzen entstehende Lissajous-Figur zu sehen.

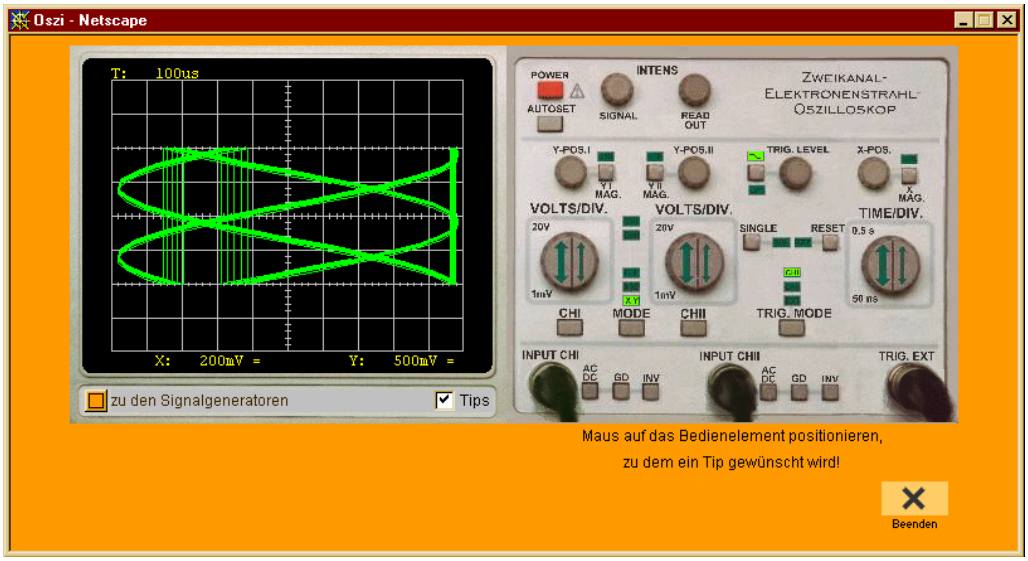

Abbildung 74: Simulation eines Oszilloskops, Bedienelemente zur Oszilloskopsteuerung

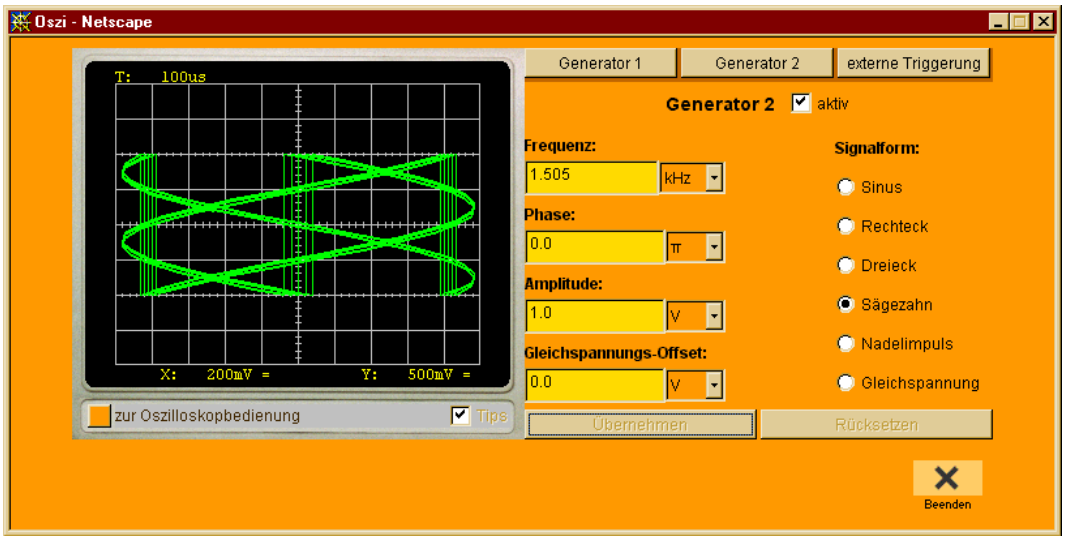

Abbildung 75: Simulation eines Oszilloskops, Formular zur Einstellung der Tongeneratoren

Simulationen sind anders als Animationen ein geeigneter Ansatzpunkt für das in Kap. 14.2 diskutierte Dienstekonzept: Da mit Simulationen durch den Benutzer Daten generiert werden können, stellt sich die Frage der Weiterverwendung dieser Daten durch in das dynamische elektronische Buch integrierte Dienste, z. B. zur Speicherung der Daten, ihrer Verwendung im Rahmen der Erstellung eines Versuchsprotokolls oder bei der rechnerischen Weiterverarbeitung (Visualisierung von Messreihen, soweit dies nicht unmittelbar durch die Simulation erfolgt, statistische Auswertung der Daten). Neben dieser Weiterverarbeitung der von einer Simulation generierten Daten kann auch die unterschiedliche Initialisierung einer Simulation treten, wenn für ihren Einsatz im Rahmen einer bestimmten Ausgangskonfiguration die Parameter vor dem Simulationsstart auf bestimmte Werte gesetzt werden müssen. Auf diese Weise lässt sich eine Simulation an verschiedenen Stellen des elektronischen Buchs mit jeweils unterschiedlichen Ausgangskonfigurationen einsetzen.

Ein weiterer, für die Komponententechnologie typischer Aspekt ist der modulare Aufbau der Simulationen, der es erlaubt, Teilkomponenten einer Simulation mehrfach einzusetzen, wie die nachfolgenden Beispiele verdeutlichen:

- Die Datenvisualisierungskomponente in Abbildung 73 (Fadenpendel) ist so angelegt, dass sie in mehreren Simulationen zur Anwendung kommt, was den Entwicklungsaufwand für jede einzelne Simulation verringert (Aspekt der Wiederverwendung von (Teil-)Komponenten).
- An die Stelle abstrakter Tongeneratoren des Oszilloskops können auch konkrete Messanordnungen treten (z. B. Messung an einem RC-Glied oder in Schwingkreisen). Damit kann diese Komponente neben ihrer primären Anwendung– Erlernen und Üben des Umgangs mit einem komplexen Messgerät – auch für den tatsächlichen Einsatz bei der Versuchssimulation genutzt werden.

14.1.7 Sonstige Komponenten

Neben den interaktiven, multimedialen Komponenten enthält das elektronische Buch im Referenzprojekt eine Reihe weiterer Komponenten; zu ihnen gehören:

- die Abbildungen des Ausgangswerks (Diagramme, schematische Darstellungen von Versuchen, Funktionsgraphen),
- exemplarische Aufgaben mit Hinweisen zum Lösungsansatz,
- Beispiele für Versuchsprotokolle,
- Biographien berühmter Physiker (mit Abbildungen) sowie
- Texte aus externen Materialien (anderen Werken des Verlags).

Diese Ergänzungen sind weder Multimedia- noch Softwarekomponenten; die Bezeichnung Komponente bezieht sich bei ihnen auf den Präsentationsaspekt (Zusatzmaterial zum Basisdatenbestand des elektronischen Buchs). Sie können aber eine qualitative Erweiterung des Buchs darstellen und sind zudem analog zu den Multimediakomponenten in das Nutzungskonzept des elektronischen Buchs eingepasst (Einbindung als *extended links* durch Zuordnung zu Strukturmarkup bzw. inhaltsorientierten Auszeichnungen; Darstellung mit Hilfe typisierter Symbole im Buchbetrachter). Die Darstellung der Abbildungen erfolgt innerhalb der *Kontextseite* des Buchs; der Benutzer kann allerdings für jede Abbildung eine vergrößerte Darstellung in einem eigenen Subfenster anfordern, wie die nachfolgenden beiden Abbildungen zeigen:

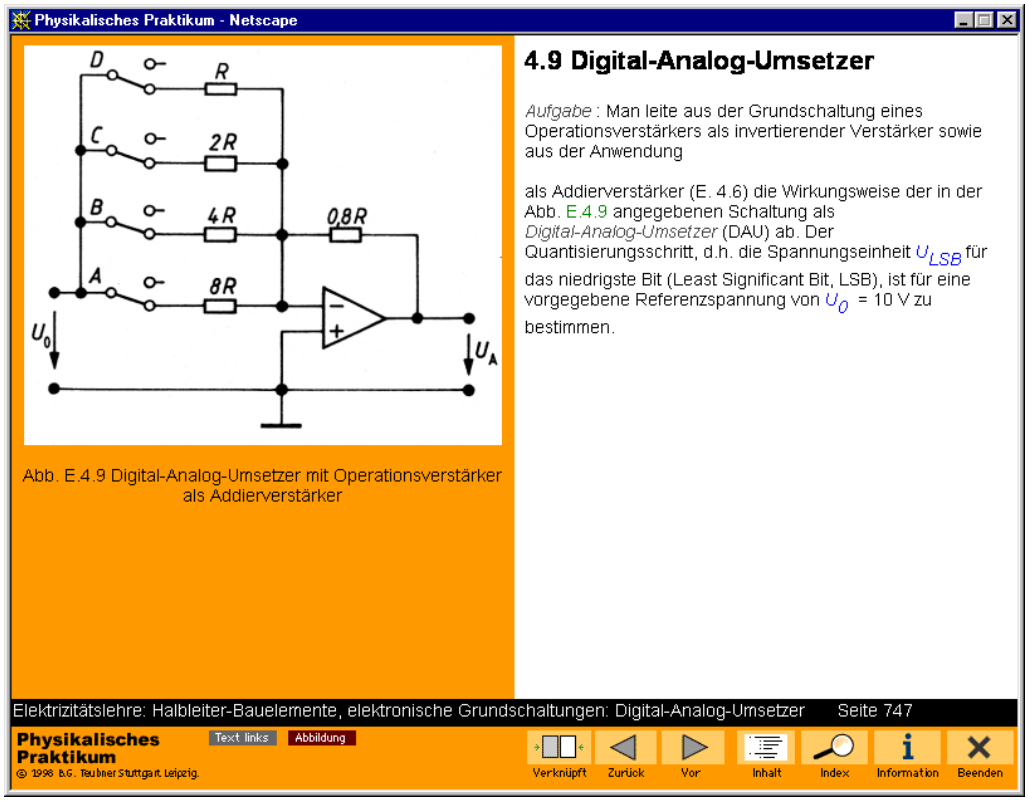

Abbildung 76: In die Kontextseite eingebettete Abbildung

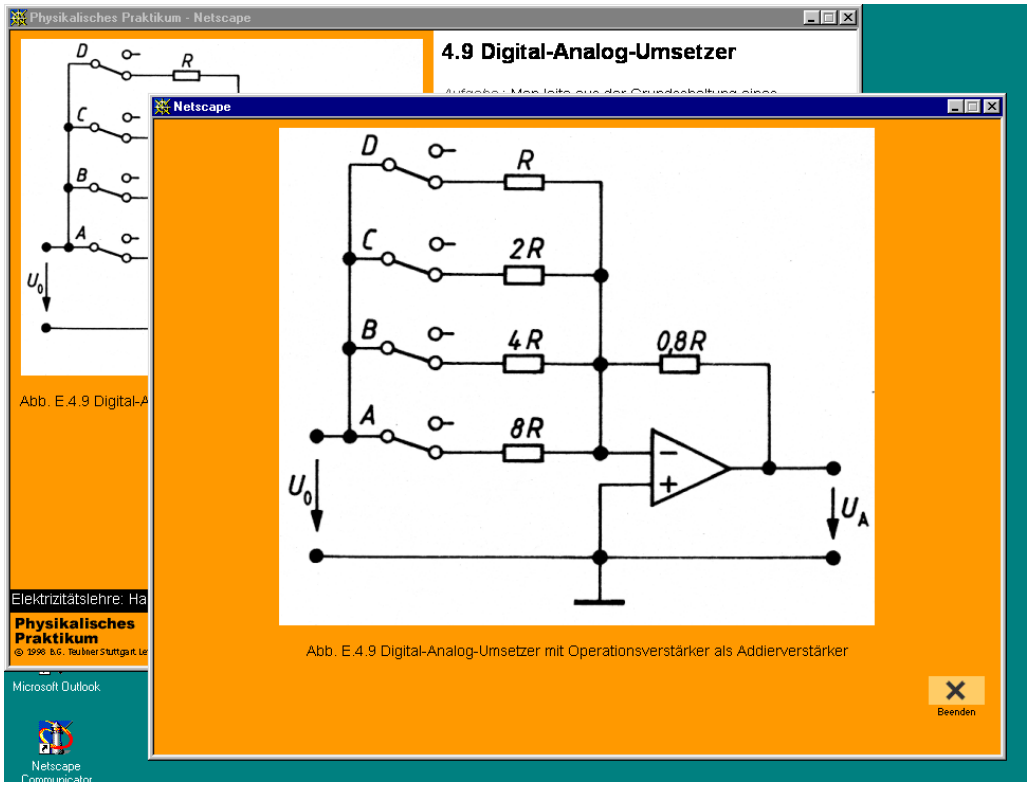

Abbildung 77: Vergrößerte Darstellung einer Abbildung in gesondertem Fenster

In analoger Weise sind auch unterschiedliche textuelle Ergänzungen eingebunden. Die folgenden Abbildungen zeigen dies am Beispiel einer als Zusatztext integrierten Biografie und eines in die Kontextseite des Buchbetrachters eingepaßten Übungsaufgabe.

Abbildung 78: Biographie als textuelles Zusatzelement

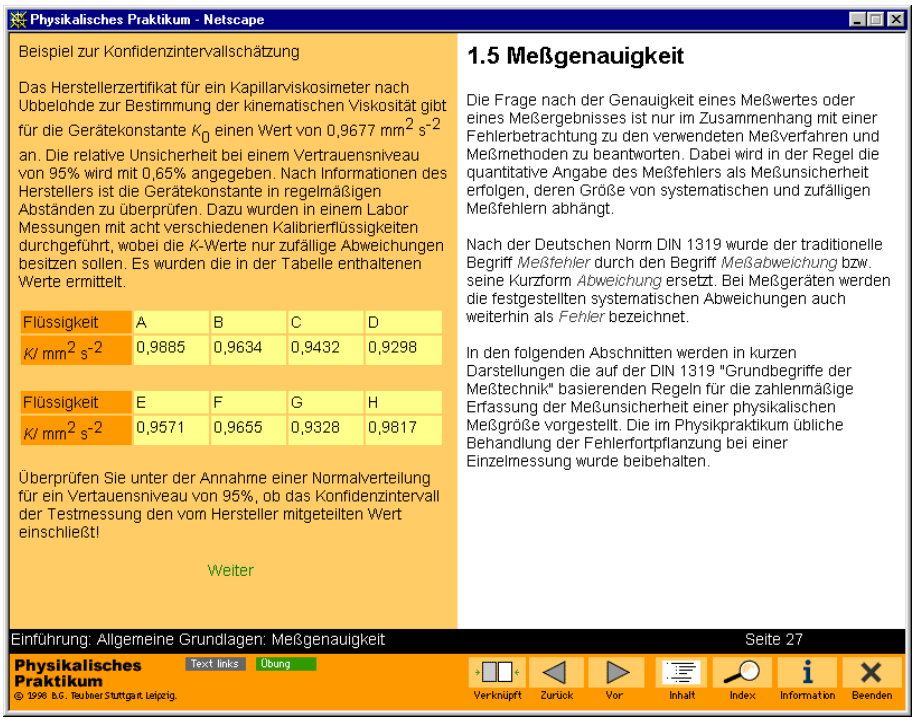

Abbildung 79: In die Kontextseite integrierte Übung

14.2 Integration von Diensten

Die Diensteverwaltung als Komponente des Buchverwaltungssystem ist bereits in Kap. 12.5.1 erläutert worden. Aufbauend auf dem in Kap. 4.4.1 erarbeiteten Modell der Ergänzung dynamischer elektronischer Bücher um (Informations-)dienste soll anhand von Beispielen aufgezeigt werden, wie sich dieses Konzept umsetzen lässt. Wesentliche Kriterien zur Unterscheidung von Diensten sind

- die *Dienstefunktion* (z. B. Speicherung, Informationsdienstleistung),
- der *Anknüpfungspunkt* des Dienstes in Bezug auf die Buchinhalte (Buchebene, Strukturebene, eingebettete Komponenten),
- die *Zuordnung* des Dienstes zu Basisfunktionalität, Nutzungskontext oder individuellem Kontext den dynamischen elektronischen Buchs und
- die technische Realisierung des Dienstes einschließlich seiner Integration in das dynamische elektronische Buch und seine Komponenten (Protokoll, Interface).

Für die Umsetzung des in Kap. 4.4 entwickelten Dienstekonzepts wurden die dort bereits andeutungsweise diskutierten Dienste für die prototypische Realisierung ausgewählt. Unterschieden nach der primären Dienstefunktion handelt es sich um

- einen *Speicherungsdienst*, der verschiedene Aufgaben übernehmen kann (Annotation; Speicherung von aus Komponenten generierten Daten; Speicherung von Daten, die mit Hilfe eines Informationserschließungsdienstes ermittelt wurden),
- einen *Informationserschließungsdienst*, d. h. die Anbindung externer Suchmaschinen an das dynamische elektronische Buch,
- Beispiele für *Informationsdienste*, die generisch und/oder an einzelne Textbestandteile des Buchs gebunden sind und Zusatzinformationen ermitteln (Definition, Übersetzung) bzw. als sekundäre Dienste dem Benutzer Hilfestellung bei der Informationserschließung geben (Kollokationen zu Suchbegriffen),
- einen *weiterverarbeitenden* Dienst, der als Beispiel für einen an bestimmte Inhaltselemente gebundenen Dienst die Instantiierung von Gleichungen ermöglicht sowie
- die Integration des Buchs in einen *konkreten Nutzungskontext* durch Anbieten angepasster *Kommunikationsdienstleistungen*.

Die nachfolgende Tabelle gibt einen Überblick über die Ausgestaltung der Dienste und ihre Einbettung in das dynamische elektronische Buch:

Dienst/Funktion	*Anknüpfungspunkt im Buch*	*Zuordnungsebene*	*Folgedienst*
Speicherungsdienst	Inhaltskomponente (Annotation, Speicherung externer Daten)	individueller Benutzer, Benutzergruppe	(Anzeige als Annotation im Buch; Weiterverarbeitung)
Informationserschließung	Generisch (Buchebene)	generisch (allg. Funktionalität)	Speicherung
Informationsdienste (hier: Definition, Übersetzung)	je nach Zuordnung z. B. Ebene durch markup gekennzeichneter Begriffe	generisch, individueller Benutzer	Speicherung; als sekundäre Dienste auch zur Unterstützung der Informationserschließung
Weiterverarbeitung (Formelmanipulation)	Gekennzeichnete Inhaltskomponenten (hier: Gleichungen in MathML und Präsentationskomponenten (WebEQ-Applets))	individueller Benutzer (Schnittstelle zu lokal verfügbarer Software)	–

Dienst/Funktion	*Anknüpfungspunkt im Buch*	*Zuordnungsebene*	*Folgedienst*
Kommunikationsdienste	Gekennzeichnete Inhaltskomponenten (Aufgaben, Protokolle, Daten aus Simulationen)	Benutzergruppe (Nutzungskontext)	z. B. Benachrichtigung des Benutzers, unabhängig von der Nutzung des elektronischen Buchs

Tabelle 59: Einordnung ausgewählter Dienste

14.2.1 Speicherungsdienste

Die Speicherung zusätzlicher Daten über den Ausgangsbestand an Inhalten und Komponenten hinaus erlaubt es dem Benutzer, Anpassungen am elektronischen Buch vorzunehmen und es im Sinn der entsprechenden ergonomischen Anforderung aus ISO 9241/10 für seine Zwecke zu individualisieren. Je nach Entstehungskontext der zu speichernden Daten ist zu unterscheiden zwischen

- *freien Annotationen*, d. h. der Möglichkeit des Benutzers, Passagen des elektronischen Buchs durch Anmerkungen zu ergänzen,
- der *Speicherung von Daten*, die ein über die Schnittstelle des elektronischen Buchs erreichbarer Dienst generiert hat (z. B. das Ergebnis einer Recherche zu einem Inhaltsbereich des elektronischen Buchs mit Hilfe einer Suchmaschine),
- Daten, die der Benutzer in der Interaktion mit einer im elektronischen Buch enthaltenen Komponente generiert hat (im Fall des Referenzprojekts z. B. die in einer Versuchssimulation errechneten Messwerte).

Zusätzlich zur Unterscheidung nach dem *Entstehungszusammenhang* einer Anmerkung kann man die Speicherung zusätzlicher Daten nach dem Adressatenkreis differenzieren.[129] Neben der Zuordnung zu einem individuellen Benutzer können auch Annotationen, die innerhalb eines Nutzungskontexts für eine Mehrzahl von Benutzern sichtbar sein sollen, über einen solchen Speicherungsdienst realisiert werden und sind ein kennzeichnendes Merkmal für Groupware-Funktionalität (gruppenbezogene Annotationssysteme, vgl. LALIBERTE & BRAVERMAN 1995).

14.2.1.1 Technische Realisierung

Die Speicherung individueller oder gruppenbezogener Zusatzinformation setzt einen *Speicherungsdienst* voraus, wobei sich die Alternativen der Speicherung im lokalen Dateisystem bzw. über Netzwerkdienste anbieten. Die erste Lösung, die für kommerzielle elektronische Bücher kennzeichnend ist, hat den technischen Nachteil, dass sie im Rahmen der Architektur eines browserbasierten Buchbetrachtungssystems nur schwer zu realisieren wäre.[130] Konzeptuell würde so der Ansatz durchbrochen, dynamische elektronische Bücher als diensteintegrierende netzbasierte Anwendungen zu realisieren. Statt

[129] HUMMES, KARSENTY & MERIALDO 1997 modellieren die Problematik der Annotation allgemein als einen Zusammenhang zwischen einer Menge *adressierbarer Entitäten* (z. B. Webseiten), einer Menge von *Anmerkungen*, einer Menge von *Adressaten*, die die Annotationen einsehen können, und einer Menge von *Erzeugern* für Annotationen. Bei Anmerkungen eines individuellen Benutzers sind Erzeuger und Betrachter identisch.

[130] Dies gilt nicht nur für den Zugriff auf das Dateisystem, der nur unter Umgehung des Sicherheitssystems eines Browsers oder durch Zusatztechnologien wie signierte Applets zu realisieren wäre, sondern auch für die konsistente Linkverwaltung.

dessen bietet es sich an, die Speicherung zusätzlicher Daten über einen geeigneten Netzwerkdienst zu ermöglichen. Neben der Einführung eines für die Speicherung von Annotationen benötigten zusätzlichen Protokolls (s. u.) unterscheiden HUMMES, KARSENTY & MERIALDO 1997: Abb. 2 drei verschiedene Formen der Modifikation des Client-Server-Szenarios des World Wide Web bei der Realisierung von Annotationen:

- Integration von Annotationen durch *Modifikation des Servers*,
- Realisierung der Annotationsfunktionalität auf der Seite des *Browsers* (Client) und
- Integration durch ein zwischen Server und Browser geschaltetes *Proxy-Modul.*

Bei der Architektur dynamischer elektronischer Bücher liegt der erste Fall vor. Die in den Buchserver integrierte Diensteverwaltung, die einen Annotationsdienst aktivieren kann, stellt eine Modifikation und Erweiterung der Standardfunktionalität eines Webservers dar. Bezüglich der Realisierung eines Zugriffsprotokolls für die Speicherung von Annotationen ist aufgrund der Weiterentwicklung der Internetprotokolle keine proprietäre Lösung erforderlich: Im Vergleich mit einfacheren Protokollen wie dem *file transfer protocol* (ftp) bietet die in Kap. 12.6.2 eingeführte HTTP-Erweiterung des *distributed authoring and versioning* (WebDAV) eine geeignete Lösung an, da mit diesem Protokoll nicht nur der Schreibzugriff auf Ressourcen im World Wide Web möglich wird, sondern auch gruppenbezogene Lösungen für die Speicherung und Verwaltung von Daten möglich sind. Um einen solchen Speicherungsdienst auf der Basis von WebDAV umzusetzen, müssen mindestens folgende Voraussetzungen erfüllt sein:

- Ein WebServer mit WebDAV-Erweiterung muss zur Verfügung stehen (hier: Erweiterungsmodul `mod_dav` des Apache-Webservers, vgl. *http://www.webdav.org/mod_dav*).
- Der Buchserver muss einen WebDAV-Client als Dienst integrieren und aktivieren können (hier: WebDAV-Client als von der Diensteverwaltung des Buchservers aktiviertes Java-Objekt, das ein API für WebDAV implementiert (DAV4J-Modul, vgl. *http://www.alphaworks.ibm.com/tech/DAV4J*).
- Der Leser muss die Möglichkeit haben, aus der Benutzerschnittstelle des elektronischen Buchs die notwendigen Angaben zur Initialisierung des Speicherungsdienstes zu machen (Dienstespezifikation durch Angabe von Name, Typ und Adresse des Dienstes).
- Die Zuordnung gespeicherter Daten zu Benutzern einerseits (Identifikation, Zugriffsrechte, Benutzerverwaltung), den Inhaltsbestandteilen des elektronischen Buchs (Elemente des struktur- bzw. inhaltsbezogenen Markup) andererseits muss gewährleistet sein, um die erstellten Zusatzdaten in geeigneter Weise in die Benutzerschnittstelle des elektronischen Buchs integrieren zu können.
- Zusammen mit der Dienstespezifikation muss der Buchserver über geeignete Benutzerschnittstellenelemente (z. B. als XML- oder HTML-Templates) verfügen, über die Annotationen in das elektronisch Buch integriert und dargestellt werden können (Repräsentation als Symbol/Ikone sowie Darstellungsfenster für die Darstellung der Annotation).

14.2.1.2 Annotationen

Unter einer Annotation wird jede textuelle oder symbolische Ergänzung des Buchtextes bzw. der in einem elektronischen Buch enthaltenen nicht-textuellen Materialien verstanden. Annotationen stellen eine weit verbreitete „Interaktionsform“ mit Wissen in Form gedruckter oder elektronischer Publikationen dar; die Möglichkeit zur Annotation findet sich nicht nur in den Anforderungskatalogen für elektronische Bücher (vgl. Kap. 3.2),

sondern ist auch in der Hypertextliteratur ein wichtiger Aspekt der Funktionalität von Hypertextsystemen: CHRISTODOULOU, STYLIARAS & PAPATHEODOROU 1998: 6 sehen in ihrem Evaluierungskatalog für Hypermediasysteme Annotationen als ein wesentliches Beurteilungskriterium an. Die Standardfunktionalität von Webbrowsern bietet als eine der Annotation verwandte Funktionalität lediglich die Möglichkeit, über sog. *bookmarks* eine hierarchisch gegliederte Liste von Verweisen auf ausgewählte Adressen im World Wide Web anzulegen. *Bookmarks* sind zudem nicht *Inhalten*, wie dem Text eines elektronischen Buchs, sondern als generisches Ordnungssystem dem Webbrowser zugeordnet. In der Literatur finden sich einige Vorschläge für den Aufbau von Annotationsmöglichkeiten im World Wide Web:

- LALIBERTE & BRAVERMAN 1995 schlagen ein eigenes Protokoll für gruppenbezogene Annotationen im World Wide Web vor, das auf CGI-Skripten zugreift und es dem Benutzer erlaubt, öffentliche Annotationen zu Dokumenten im World Wide Web zu finden.
- RÖSCHEISEN, MOGENSEN & WINOGRAD 1995 stellen ein Annotationssystem vor, das die Zuordnung von Annotationen zu Individuen und Gruppen sowie der öffentlicher Nutzung vornimmt. Ähnlich wie hier der Buchserver bei den Interaktionsschritten prüft, ob ein Inhaltsbestandteil des elektronischen Buchs annotiert ist, sieht dieses System die automatische Prüfung des Vorliegens von Annotationen vor.

Wie die Auswertung von Annotationen im Printmedium zeigt, ist eine breite Vielfalt von Annotationsformen wünschenswert, die den freien Eingriff in den Ausgangstext in Form von Hervorhebungen einschließt: Mehrere Studien haben die Varianten der Annotationstechnik in gedruckten Werken als Strategien der individuellen Informationserschließung und -aufbereitung untersucht, vgl. MARSHALL 1998, O'HARA 1998 und SCHILIT, GOLOVCHINSKY & PRICE 1998. Ein wesentliches Ergebnis ist, dass der häufigste Form der Annotation in der Hervorhebung und Selektion des vorhandenen Materials zu sehen ist und nicht in der *genuinen Textproduktion* durch den Leser. Eine Lösung für solche graphische und symbolische Annotationen in elektronischen Büchern ist das Konzept der Annotation durch *digital ink*, wie es SCHILIT, GOLOVCHINSKY & PRICE 1998 als Annotationsmöglichkeit für den Prototyp eines hardwarebasierten Buchbetrachters vorstellen. In ihm kann der Benutzer über einen Touchscreen freie Annotationen am elektronischen Buchtext vornehmen.

Für das Modell dynamischer elektronischer Bücher bleibt die Annotation aber auf die Textproduktion durch den Leser und die Zuordnung zu Struktur- und Inhaltsmarkup beschränkt:

- Der Benutzer kann über ein Eingabeformular Annotationen zu Bestandteilen des elektronischen Buchs vornehmen. Die logische Zuordnung zwischen Inhalten und Annotationen erfolgt auf der Basis des aktuellen Inhaltsbestandteiles (unterste Ebene der Inhaltshierarchie oder aktueller, durch inhaltsorientiertes Markup ausgezeichneter Textbereich).
- Die Existenz von Annotationen wird in der Benutzerschnittstelle durch Annotationssymbole in einem gesonderten Bereich angezeigt; erstreckt sich der logische Bezug einer Annotation über mehr als eine Präsentationseinheit, so bleibt die Anzeige der Annotation für diesen Bereich bestehen.
- Aktiviert der Benutzer eine schon vorhandene Annotation, so wird sie wie bei der Erstellung in einem Editiermodus (HTML-Formular) angezeigt; der Benutzer hat dann die Möglichkeit der Änderung bzw. des Löschens der Annotation.

- Voraussetzung für die Zusammenführung von Inhalten und Annotationen ist ein Nachschlagen nach vorhandenen Annotationen für jede vom Benutzer angeforderte Struktureinheit des elektronischen Buchs (bzw. auf der Ebene darunter für jedes durch inhaltsorientiertes Markup versehene Inhaltselement).

Die folgenden Abbildungen zeigen das Eingabeformular für Anmerkungen sowie die zusätzliche Symbolleiste des elektronischen Buchs für den Abruf von Anmerkungen (bzw. von anderen gespeicherten Ergänzungen, s. u.).

Abbildung 80: Eingabeformular für Anmerkungen im elektronischen Buch

Abbildung 81: Darstellung vorliegender Annotationen über Symbole im Buchbetrachter

Der hier entwickelte Lösungsansatz lässt sich in Fortführung der von HUMMES, KARSENTY & MERIALDO 1997: Tabelle 2 vorgelegten Taxonomie für Annotationssysteme wie folgt einordnen:

Inhaltsbezug	Strukturmarkup, inhaltsorientiertes Markup
Autorisierung	Benutzerverwaltung des Buchservers
Granularität	Einfacher Text, gekapselte Informationsdienste (s. u.)
Präsentation	Eigenes Interface (Fenster) für Annotationserstellung und -abruf
Benachrichtigung	keine
Lebenszyklus	Abhängig von den Zugriffsrechten des Benutzers auf Speicherungsdienst und elektronisches Buch
Speicherung	Über WebDAV auf einem Webserver, für den der Benutzer Schreibrechte besitzt
Erweiterungen	Generalisierung für weitere Datenformate (z. B. durch Realisierung als Java-Applet mit *drag & drop*-Interface für Datenobjekte und Dokumente.

Tabelle 60: Merkmale des Speicherungsdienstes für Annotationen

Ein zentraler Aspekt ist die Trennung der Dienstefunktionalität von der Realisierung des Buchservers: Als dynamisch einzubindender Dienst ist die Annotationsfunktionalität durch ein vorgefertigtes Softwaremodul, die WebDAV-Erweiterung des Apache-Servers, sowie ein dafür erforderliches Zugriffsprogramm (WebDAV-Client) realisiert. Der Buchserver ist in der Lage, verschiedene Benutzer dieselben Materialien annotieren zu lassen, ohne dass dadurch ein hoher Verwaltungsaufwand für den Server bedingt wäre. Wie für andere Dienste hat der Buchserver eine Scharnierfunktion, d. h. er vermittelt einen Dienst, realisiert diesen aber nicht selbst.

14.2.1.3 Speicherung gekapselter Informationsdienste

Die Grundfunktionalität des Annotationsdienstes, das Speichern von Daten auf einem WebDAV-Server, kann auch als Folgedienst genutzt werden, indem der Benutzer

- Ergebnisse eines anderen Dienstes, z. B. ein Suchergebnis einer Recherche, oder
- Daten, die aus der Interaktion mit einer im elektronischen Buch enthaltenen interaktiven Komponente speichert.

Die Voraussetzung dafür ist, dass der Informationserschließungsdienst so in die Benutzerschnittstelle des elektronischen Buchs eingebettet ist, dass dem Benutzer die Speicherungsmöglichkeit zur Verfügung steht (Abbildung 82).

14.2.1.4 Speicherung komponentenbezogener Daten

Die bisher geschilderten Beispiele für die Nutzung eines Speicherungs- bzw. Annotationsdienstes sind auf Fälle beschränkt, in denen Daten aus einer XML- bzw. browserbasierten Benutzerschnittstelle gespeichert werden (Annotationsformular bzw. Speichern einer externen Ressource). Der Speicherungsdienst soll zudem Daten aus eingebetteten Komponenten verarbeiten können. Dabei kann es sich hinsichtlich der Zuordnung des Dienstes sowohl um einen individuellen Dienst als auch um einen Dienst in einem Gruppenkontext handeln. Voraussetzung für die Realisierung dieser Funktionalität ist

- die Zuordnung des Dienstes zu einzelnen Komponenten bzw. einer Komponentenkategorie (z. B. Simulationen),
- die Initialisierung der Komponente über Parameter, die den Dienst beschreiben und
- die Integration von Benutzerschnittstellenelementen, um dem Benutzer die Möglichkeit zu geben, den Dienst zu steuern.

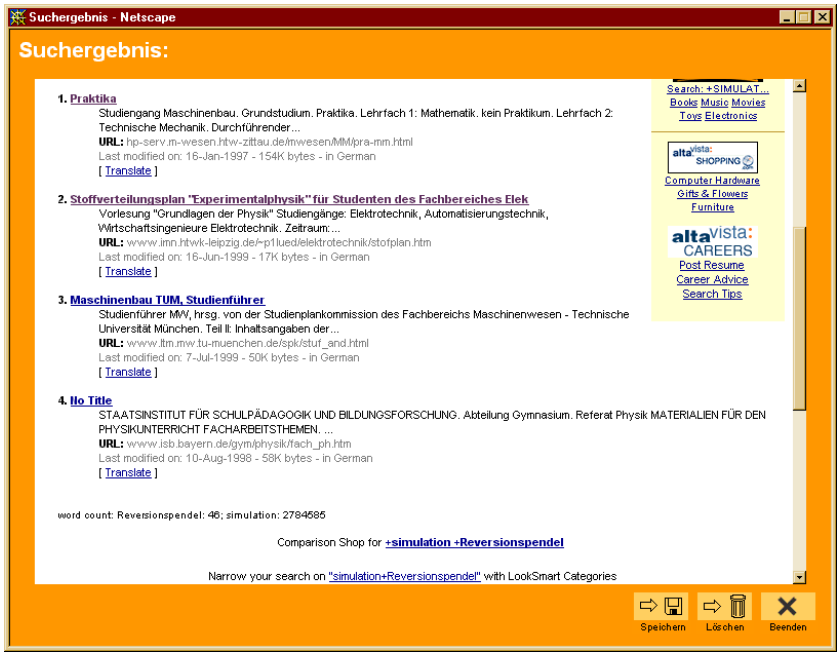

Abbildung 82: Benutzerschnittstelle für einen gekapselten Informationsdienst (Suchmaschine)

Zusätzlich ist zu unterscheiden, ob die Kommunikation der Komponente mit dem Dienst *unidirektional* (wenn z. B. eine Versuchsreihe gespeichert werden soll) oder *bidirektional* verlaufen soll (wenn Daten an den Dienst gesendet werden sollen und Ergebnisse einer Berechnung des Dienstes von der Komponente weiterverarbeitet werden sollen). Die Betrachtung bleibt hier auf den Fall der Speicherung als – aus der Sicht der Komponente – unidirektionalem Dienst beschränkt: Komponenten können den Speicherungsdienst aufrufen und Daten speichern Die gespeicherten Daten stehen wie Annotationen in der Benutzerschnittstelle des elektronischen Buchs zur Verfügung. Ein Nutzungs*kreislauf* kann unabhängig von der Komponente entstehen, wenn die Daten mit inhaltsorientiertem Markup versehen sind und den Markupelementen weitere Dienste zugeordnet sind. Für die Generierung von Datensätzen entsteht folgende Prozesskette:

1. Der Benutzer aktiviert eine Komponente, z. B. eine Simulation; dabei ist für diese Komponente ein Speicherungsdienst vorhanden.
2. Er führt die Simulation durch und erzeugt einen Datensatz, der anschließend gespeichert wird.
3. Der Datensatz wird als Annotation in der Benutzerschnittstelle des elektronischen Buchs angezeigt. Für diesen Inhaltstyp ist ein weiterer Dienst spezifiziert, z. B. ein Visualisierungsmodul oder ein Auswertungsprogramm.
4. Der Benutzer aktiviert die gespeicherten Messdaten und ruft den Weiterverarbeitungsdienst auf.

14.2.2 Informationserschließungsdienste

Unter der Informationserschließung in elektronischen Büchern mit Techniken des Information Retrieval (IR) sind aus der Perspektive des Lesers alle Verfahren und Werkzeuge zu verstehen, die es dem Benutzer erlauben, sich die Inhalte einer elektronischen Publikation außerhalb der von den Autoren vorgesehen Nutzungsreihenfolge zu erschließen. Das bedeutet, auch Inhalte ermitteln zu können, die nicht im Kontext der aktuellen Lese-

position oder von ihr ausgehend durch die in einer Präsentationseinheit verfügbaren Hypertextverknüpfungen unmittelbar zugänglich sind.

Die gezielte Suche stellt einen Gegenpol zur Erschließung durch *browsing* in Hypertexten dar, da die gesuchte Information nicht auf einem durch die Informationsstrukturierung vorgegebenen Weg, sondern durch *Selektion* aus einer *Menge von Informationseinheiten* erfolgt. Damit sind vor allem Verfahren des Information Retrieval angesprochen, mit deren Hilfe elektronisch verfügbare Informationen erschlossen (indexiert) und zugänglich gemacht werden könenn, ohne auf die Navigation durch das elektronische Buch als lineare Folge informationeller Einheiten oder den Zugriff über eine hierarchische Struktur (z. B. Inhaltsverzeichnis) zurückgreifen zu müssen. Neben der Erschließung der Information des elektronischen Buchs spielt die Integration externer Ressourcen (bibliographische Datenbanken, World Wide Web, etc.) eine zentrale Rolle, da durch sie ein elektronisches Buch virtuell erweiterbar wird – der Benutzer kann aus seiner Leseposition heraus externe Quellen erschließen. Es ist davon auszugehen,

- dass eine solche Erschließung sich möglichst nahtlos in die Nutzung des elektronischen Buchs integrieren lassen sollte und
- die im Buch selbst enthaltenen Inhalte eine Hilfestellung bei der Erschließung geben können (z. B. durch Eingrenzung des Recherchekontexts).

Folgende Besonderheiten elektronischer Bücher sind zu beachten:

- Die Informationserschließung über IR-Techniken hat bei der Interaktion mit einem elektronischen Buch als geschlossenem Informationsbestand eine subsidiäre Funktion gegenüber der Erschließung durch hierarchische Zugriffsstrukturen oder Hypertextverknüpfungen.
- Elektronische Bücher stellen anders als Dokumentkollektionen einen weit weniger umfangreichen, aber wesentlich komplexer strukturierten Informationsbestand dar, während das Charakteristikum von IR-Datenbanken das Vorliegen unstrukturierter bzw. semistrukturierter *Massendaten* ist.
- Die Erschließung multimedialer Inhalte macht bei der Erschließung Verfahren erforderlich, die über die üblicherweise bei der Textindexierung angewandten Techniken hinausgehen (Multimedia-Retrieval).

Bevor Lösungsvorschläge für das Informationserschließungsproblem elektronischer Bücher diskutiert werden, ist zunächst der Begriff Information Retrieval einzuführen. Die klassische Definition des IR, wie sie SALTON & MCGILL 1983: 1 f. geben, zeigt den weit gefassten Anspruch dieses Forschungsgebiets, unter den sich viele Aspekte der Modellierung, Entwicklung und Nutzung elektronischer Bücher einreihen lassen:

> Information Retrieval (IR) is concerned with the representation, storage, organization, and accessing of information items. In principle, no restriction is placed on the type of item handled in information retrieval. In actuality, many of the items found in ordinary retrieval systems are characterized by an emphasis on narrative information.

Die zuletzt genannte Betonung narrativer Information ist in der Praxis weiter vorherrschend, gerade auch, wenn man an die bisher textorientierten Inhalte digitaler Bibliotheken denkt. In den letzten Jahren haben aber durch Multimedia-Technologien und die Weiterentwicklung wissensbasierter Technologien die nicht rein textzentrierten Aspekte des IR an Bedeutung gewonnen (vgl. unten Kap. 14.2.2.3).

Man kann argumentieren, dass *electronic publishing* und Information Retrieval wechselseitig aufeinander bezogen sind: Die Nutzung elektronischer IR-Systeme setzt maschi-

nell erfasste und zugängliche Dokumente oder Dokumentsurrogate voraus; umgekehrt sind IR-Techniken für die Erschließung der Inhalte elektronischer Bücher erforderlich.

Das Grundproblem des IR lässt sich wie folgt beschreiben: Ausgehend von einer Menge informationeller Einheiten (Dokumente, Multimediakomponenten etc.) ist für die Informationsbedürfnisse der Benutzer dieser Informationssammlungen eine Abbildungsfunktion zu finden, die aus der Informationssammlung eine zu den Informationsbedürfnissen passende Untermenge auswählt. Folgende Prozesse spielen eine Rolle:

1. Der Prozess der Formulierung des Informationsbedürfnisses durch den Benutzer und die vom System dafür bereitgestellten Hilfestellungen (Anfragesprache, Thesaurus, gekennzeichnete Indexbegriffe, Konzepthierarchien, Ontologien etc.).
2. Die Abbildung der Suchanfrage in eine interne Repräsentation (z. B. ein Vektor gewichteter Anfrageterme).
3. Die Abbildung der Dokumente auf eine interne Repräsentation (z. B. invertierte Datei) durch automatische, intellektuelle oder hybride Verfahren), d. h. die Indexierung der Ausgangsdaten.
4. Die Ähnlichkeitsfunktion für den Abgleich der internen Repräsentation von Informationsbedürfnissen und den informationellen Einheiten (Retrievalfunktion).
5. Die Präsentation der Suchergebnisse in geeigneter Form (z. B. Kurzfassungen als Dokumentsurrogate, Ordnung der Ergebnisse nach Qualität oder einem Metadatenkriterium (z. B. Datum)).

Daraus ergibt sich folgende schematische Systemarchitektur für ein Information Retrieval-System, die sich, wie in der folgenden Abbildung gezeigt, darstellen lässt (vgl. FRAKES 1992A: 7, Abb. 1.1; BAEZA-YATES & RIBEIRO-NETO 1999: 10, Abb. 1.3).

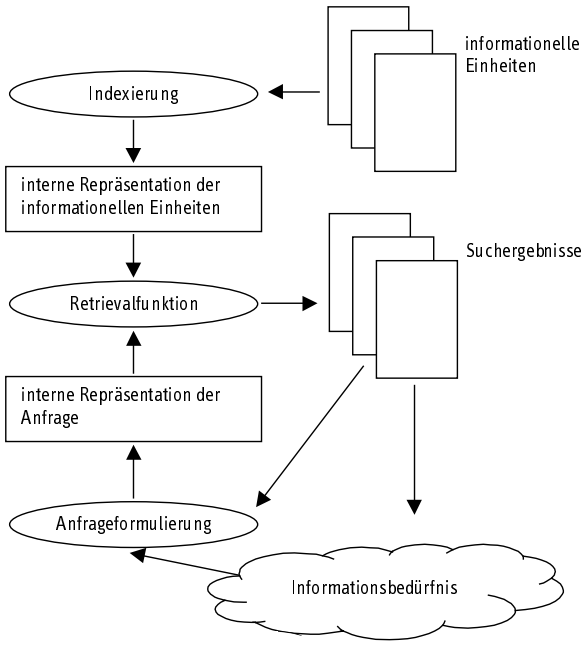

Abbildung 83: Grundprinzip des Information Retrieval

Abbildung 83 stellt dar, dass es sich bei der Informationserschließung oft um *zyklische Prozesse* handelt, bei denen Suchergebnisse die Informationsbedürfnisse beeinflussen und zu weiteren Anfragen führen können, oder direkt für die Modifikation der Ausgangsan-

frage genutzt werden (vgl. WOLFF 1996: 37 ff.). Dies ist Grundlage der *relevance feedback*-Technik, die Relevanzbewertungen des Benutzers zur Modifikation der Anfrage verwendet. Auch bei einer *automatic query expansion*, bei der die Anfrage des Benutzers durch zusätzliche Terme erweitert wird, handelt es sich um einen zyklischen Prozess.

Um die Beschreibungen der in einem IR-System erfassten informationellen Einheiten zu erstellen, verwendet man ein Indexierungsvokabular, d. h. eine Menge an Merkmalen, die meist als *sprachliche Terme* $T = \{t_1, ..., t_n\}$ repräsentiert sind, wobei die Auswahl der Terme wie auch die Beschreibung der Informationseinheiten manuell oder automatisch erstellt werden kann. Dies gilt für das Text- oder Dokumentenretrieval, soweit nicht konzeptbasierte Verfahren eingesetzt werden, bei denen an die Stelle von sprachlichen Begriffen abstrakte Konzepte treten. Im Multimedia-Retrieval können automatisch extrahierte Merkmale der Daten (z. B. Häufigkeitsverteilungen von Farbwerten) die sprachlichen Beschreibungsterme ersetzen, vgl. SCHÄUBLE 1997.

Es existieren Mischformen, wo zu einer *automatischen* Indexierung des Volltextes eine *manuelle* Beschlagwortung hinzukommt. Der Grundbegriff für die Bewertung einer Abgleichsfunktion zwischen Informationsbedürfnis und ermittelten Dokumenten ist die *Relevanz* als Maß der Korrektheit der Zuordnung einer informationellen Einheit zu einem Informationsbedürfnis. Die Relevanz wird in einem benutzerbezogenen Sinn verstanden: Es geht nicht darum, ob das Dokument *formal* zur Anfrage passt, sondern darum, ob das Informationsbedürfnis des Benutzers erfüllt ist, vgl. dazu BLAIR 1990: 72f.. Eine Übersicht zur Entwicklung des Relevanzbegriffs im IR geben PARK 1997, bes. 341 ff. und SCHAMBER, EISENBERG & NILAN 1990, bes. 761 ff.

Aufgrund der Komplexität sowohl der Informationsbedürfnisse als auch der in der Regel semi- und unstrukturierten informationellen Einheiten sowie der damit verbundenen Schwierigkeiten, eine geeignete Repräsentationsform zu finden, ist ein IR-System, das für beliebige Anfragen optimale Ergebnisse liefert, praktisch ausgeschlossen, vgl. SALTON & MCGILL 1983: 10 ff. und FRAKES 1992A: 9 ff. Dieses Grundproblem des IR, dass ein nur *partieller Abgleich* zwischen Informationsbedürfnis und Informationsbestand möglich ist, verdeutlichen Vergleiche zwischen IR-Systemen und anderen Klassen von Informationssystemen (vgl. BLAIR 1990: 7, Tab. 1.1, FRAKES 1992A:9):

	IR-System	*Datenbanksystem*
Informationelle Einheiten	un- oder semistrukturierte Dokumente	Tabellen
Retrievalfunktion	*partial match, best match*	*exact match*
Modell	probabilistisch	deterministisch
Klassifikation	polithetisch	monothetisch
Anfrageformulierung	unvollständig	vollständig
Entscheidungskriterium für den Nachweis	Relevanz	exakter Abgleich

Tabelle 61: Vergleich von Information Retrieval- und Datenbanksystemen

Ausgehend von den oben eingeführten Grundbegriffen des IR werden in den beiden folgenden Kapiteln die wichtigsten Modelle des Information Retrieval sowie Verfahren zur Indexierung vorgestellt werden, bevor die Sonderproblematik der Erschließung elektronischer Bücher als hochstrukturierte Informationsbestände erörtert wird.

14.2.2.1 Retrievalmodelle

Ein Retrievalmodell beschreibt die in Abbildung 83 gezeigten Prozesse, wobei die Art der Repräsentation von Information für Informationseinheiten wie Anfragen und ihr Abgleich (Retrievalfunktion) im Zentrum stehen. In Anlehnung an BAEZA-YATES & RIBEIRO-NETO 1999: 23 f. kann man ein IR-Modell als Quadrupel $\{I, A, F, R(a_i, d_j)\}$ formalisieren,

wobei *I* und *A* die Repräsentation der informationellen Einheiten bzw. der Anfragen an das System darstellen, *F* die Art der Beziehung zwischen *I* und *A* modelliert und *R* eine Ordnungsrelation für die Zuordnung von Dokumenten *d* zu Anfragen *a* definiert (Rankingfunktion). In der IR-Literatur unterscheidet man drei grundlegende Modelle des Information Retrieval:

1. Das boolesche Modell, bei dem den Dokumenten Indexierungsterme Funktion mit binärem Wertebereich zugeordnet werden (0, 1). Dem Benutzer steht eine auf einer (meist erweiterten) booleschen Algebra beruhende Abfragesprache zur Verfügung.
2. Das statistische Modell nimmt auf der Basis von Termhäufigkeiten eine *gewichtete Indexierung* von Dokumenten und Anfragen vor. In seinem wichtigsten Vertreter, dem Vektorraummodell (*vector space model*, vgl. SALTON 1971, SALTON 1989) sind Dokumente und Abfragen als Vektoren gewichteter Terme repräsentiert, die mit Hilfe der Retrievalfunktion abgeglichen werden. Anders als das boolesche Modell erlaubt es eine Sortierung von Suchergebnissen nach dem Wert der Retrievalfunktion (*ranking*).
3. Das probabilistische Modell hat seine Basis in der wahrscheinlichkeitstheoretisch begründeten Annahme, dass der entscheidende Faktor bei der Auswahl von Dokumenten bezüglich einer Anfrage die Abschätzung der Wahrscheinlichkeit ist, dass dieses Dokument als relevant bewertet wird.

Neben diesen Grundmodellen des Information Retrieval existieren eine Vielzahl von Misch- und Sonderformen[131], z. B.

- Retrieval mit Inferenznetzwerken, insbesondere das IR-System INQUERY (vgl. CALLAN, LU & CROFT 1995),
- Retrieval mit Funktionen auf der Basis der *fuzzy logic* (vgl. MIYAMOTO 1998, BAEZA-YATES & RIBEIRO-NETO 1999: 34 ff.), oder
- Retrieval mit Hilfe neuronaler Netzwerke (vgl. BOUGHANEM & SOULE DUPUY 1994).

In der Praxis spielen neben Modellen, die eine andere theoretische Basis als die drei klassischen Verfahren haben, vor allem Kombinationen des Booleschen mit dem statistischen Ansatz eine Rolle. In ihnen wird z. B. versucht, durch Verwendung einer erweiterten Booleschen Algebra, die auch Termgewichte berücksichtigt, zu einem qualitätsorientierten Retrievalergebnis zu kommen. Im Referenzprojekt *Multimediales Physikalisches Praktikum* kommt ein Retrievalsystem zum Einsatz, das dem Benutzer eine einfache Boolesche Anfragesprache bietet, zusätzlich aber aus Termfrequenzen ein gewichtetes Ergebnis berechnet (s. o. Kap. 13.3.5). Auch Suchmaschinen im World Wide Web liegt ein solches kombiniertes Retrievalmodell zugrunde.

14.2.2.2 Indexierung

Voraussetzung für den Einsatz eines IR-Systems ist die *Indexierung* des Informationsbestands, d. h. die Zuordnung von Beschreibungsmerkmalen (Indextermen) zu den einzelnen Informationseinheiten. Grundsätzlich ergeben sich folgende Unterscheidungen für die Indexierung einer Menge von Informationseinheiten:

- *Bezug* der Erschließung: Formale Erfassung (z. B. der bibliographischen Daten) vs. *inhaltliche* Erschließung; nachfolgend ist v. a. die inhaltliche Erschließung von Bedeutung,

[131] Vgl. BAEZA-YATES & RIBEIRO-NETO 1999: 34-65, KORFHAGE 1997: 92 ff., die jeweils einen kurzen Überblick zu „nicht-klassischen“ Ansätzen im IR geben.

- Art der *Repräsentation* der Indexierungsmerkmale: Einzelterme, Thesaurus, Mehrwortbegriffe, Phrasen, Art des Indexierungsvokabulars, kontrolliertes Vokabular,
- *Ausgangsdaten* der Indexierung, d. h. die Frage, was indexiert wird: Das Gesamtdokument (Volltext) oder von vornherein nur ein Surrogat (z. B. ein Abstract) und
- *Durchführung* der Indexierung: Intellektuell, automatisch oder als Mischform.

Die Indexierung wandelt die Informationseinheiten in ein systemgerechtes Repräsentationsformat um. Zu den Hilfsmitteln der Indexierung zählen alle Wissensquellen und Verfahren, die neben den zu indexierenden dokumentarischen Bezugseinheiten herangezogen werden, um die Indexierung qualitativ und quantitativ zu beeinflussen. Neben der Erweiterung der in den Bezugseinheiten enthaltenen Information durch externes Wissen wie eine Klassifikation oder einen Thesaurus sind – bei der Indexierung von Texten – grundsätzliche sprachliche Probleme zu lösen:

Die Indexierung hat das Ziel, einer Bezugseinheit semantisch adäquate Bedeutungsmerkmale zuzuweisen, wobei sie so auszugestalten ist, dass die unterschiedlichen sprachlichen Vorstellungswelten von Benutzer, IR-System und Bezugseinheiten möglichst gut zueinander geführt werden können. Dies setzt eine passende Übersetzung der Ausgangsrepräsentation in den Merkmalsraum der Indexierungssprache voraus. Es treten Schwierigkeiten auf, wenn für graphematische Repräsentation mehrere Bedeutungen hat (*Homonyme*), zu einem Konzept mehrere äquvivalente Bezeichnungen existieren (*Synonyme*) oder spezifischen Begriffen allgemeinere Terme zugeordnet werden sollen (Äquivalenzklassen, Ober-/Unterbegriffsrelationen, vgl. dazu ausführlich GAUS 1995: 51 ff.).

Bei der manuellen Indexierung lassen sich diese Probleme lösen, wenn ein verbindliches Indexierungsvokabular (kontrolliertes Vokabular) vorhanden ist, auf das abweichende Begriffe gleicher Bedeutung übersetzt werden können (gebundene Indexierung), bei der automatischen Indexierung ist eine Auflösung nur bei vorliegenden elektronischen Lexika möglich, die Homonyme und Synonyme explizit ausweisen.

Bei der automatischen Indexierung von Informationseinheiten arbeitet man i. d. R. nicht mit einer festen Liste zulässiger Deskriptoren, sondern indexiert alle auftretenden Merkmale, soweit sie nicht in einer Stoppwortliste explizit von der Indexierung ausgeschlossen sind. Vor der eigentlichen Indexierung als Selektion beschreibender Merkmale und ihrer Gewichtung muss man die in den Informationseinheiten enthaltenen Terme nach einem einheitlichen Verfahren zur Indexierung vorbereiten, was u. a. durch folgende Schritte erfolgt:[132]

- Zerlegung und lexikalische Analyse, d. h. die Aufteilung des Informationsbestands in einzelne Einheiten, die als Indexierungsmerkmale verwendet werden können (Zerlegung von Texten in Sätze, Auffinden der Einzelworte etc.).
- Stoppworteliminierung, d. h. die Entfernung aller Terme, die als nicht bedeutungstragend markiert sind.
- Das Erkennen von Phrasen und Mehrwortbegriffen (bes. im Englischen) oder die Kompositazerlegung für zusammengesetzte Begriffe (bes. im Deutschen).
- Ggf. weitere linguistische Verfahren wie die Grundformenreduktion (*stemming*), die auf der Basis standardisierter Algorithmen (z. B. der Porter-Algorithmus für das Englische, vgl. FRAKES 1992B, bes. 138 ff.), durch Heranziehen eines Lexikons (lexikali-

[132] Die Darstellung orientiert sich an der Indexierung von *Texten*. Die einzelnen Schritte lassen sich aber auch auf andere Repräsentationsformen übertragen (z. B. die Aufteilung eines Videofilms in einzelne Indexierungseinheiten; die Entfernung nicht sinntragender Merkmale wie das Rauschen in einer Audiodatei etc.), vgl. SCHÄUBLE 1997: 91 ff.

sche Grundformenreduktion) oder mit Hilfe einer automatischen morphologischen Analyse erfolgen kann.

Die gewichtenden Verfahren der Indexierung, wie sie paradigmatisch im Vektorraummodell verwirklicht sind, gehen davon aus, dass inhaltlich wichtige Begriffe häufiger auftreten als weniger wichtige und dass daher die Termfrequenz ein Indikator für die Eignung eines Begriffs als Beschreibungsmerkmal ist (vgl. LUHN 1957, 1958). Ein Zusammenhang zwischen der Stärke der Zuordnung eines Term zu seiner Bezugseinheit ergibt sich demnach über die Vorkommenshäufigkeit des Terms, wobei dieser Zusammenhang nur für die sinntragenden Begriffe gültig sein kann, da Stoppwörter sehr häufig auftreten.

Die Berechnung von Termgewichten im statistischen Ansatz erfolgt bei vorliegenden Dokument- und Gesamtfrequenzen der Terme unter der Annahme, dass die Bedeutung eines Terms *k* als Beschreibungsmerkmal für die Informationseinheit i_j

a) proportional zur Auftretenshäufigkeit von *k* in i_j und
b) umgekehrt proportional zur Zahl der Dokumente, in der der Term *k* auftritt.

Das am weitesten verbreitete Verfahren für die Termgewichtung kombiniert daher dokumentbezogene Termhäufigkeiten (*term frequency*, TF) mit der Verteilung des Terms über die gesamte Kollektion (*inverse document frequency*, IDF, vgl. SALTON & MCGILL 1983: 60 ff., BAEZA-YATES & RIBEIRO-NETO 1999: 29 ff.). Das eigentliche Termgewicht errechnet sich aus dem Produkt von (normalisierter) Termfrequenz und inverser Dokumentfrequenz (*TF×IDF-Mass*).[133] Die so ermittelten Gewichte können dann in einer Retrievalfunktion zur Berechnung einer qualitätsorientierten Ergebnisliste bezüglich einer Anfrage verwendet werden. Neben den Basisverfahren der automatischen Indexierung kann man eine Reihe von Zusatzkomponenten bei der Indexierung verwenden; dazu gehören u. a.

- die Berücksichtigung von Interdependenzen zwischen Termen, wobei eine adäquate Beschreibung der Termabhängigkeiten, z. B. durch Klassifikation oder Clustering erforderlich ist,
- die Verwendung linguistischer oder statistischer Verfahren zur Phrasenextraktion und
- die Erweiterung von Anfragen durch Thesaurusklassen, semantische Relationen (Unter-/Oberbegriffe), Synonyme oder Kollokationen (vgl. SALTON 1989: 308).

Bei der Indexierung hochstrukturierter Dokumente, wie sie in elektronischen Büchern vorliegen, können zusätzliche Informationen in den Indexierungsprozess einbezogen werden:

- Strukturinformation aus dem Dokumentmarkup kann zur Auswahl und Gewichtung von Termen verwendet werden.
- Zusätzliche Informationen (wie die Position eines Terms im Kapitel oder Absatz) können das Indexierungsgewicht beeinflussen.
- Die Granularität der Bezugseinheiten liegt nicht auf der Ebene ganzer Dokumente, sondern unterhalb bei logischen (z. B. Abschnitt, Kapitel, inhaltsorientiertes Markup) oder präsentationsbezogenen Einheiten (z. B. eine Bildschirmseite als kleinste Präsentationseinheit).

---

[133] Verschiedene Modifikationen sind denkbar, z. B. kann durch Berücksichtigung des häufigsten Terms eines Dokuments die Termfrequenz normalisiert werden; durch zusätzliche Faktoren und Konstanten kann das Maß an die Gegebenheiten einer bestimmten Kollektion angepasst werden, vgl. SALTON & MCGILL 1983: 63 ff., HARMAN 1992: 371 ff., BAEZA-YATES & RIBEIRO-NETO 1999: 29 f.

- Multimediale Inhalte lassen sich oft nicht direkt (durch Auswertung der Inhalte), sondern nur indirekt (durch Auswerten beschreibender Metadaten) indexieren und für die Recherche erschließen, vgl. unten Kap. 14.2.2.3.

Aufbauend auf den oben eingeführten allgemeinen Modellen und Verfahren des IR sollen ausgewählte Sonderbereiche diskutiert werden, die für die Informationserschließung in dynamischen elektronischen Büchern relevant sind. Es handelt sich um

- die Erschließung multimedialer Datenbestände (Multimedia-IR),
- die Erschließung von Information im World Wide Web durch Information Retrieval Techniken.

Anschließend wird untersucht, welche Möglichkeiten sich für die Unterstützung des Nutzers bei der Informationserschließung während der Interaktion mit einem dynamischen elektronischen Buch anbieten.

14.2.2.3 Information Retrieval in multimedialen Datenbeständen

Neben den Anwendungen des IR im Textbereich gewinnt durch die elektronische Verfügbarkeit multimedialer Daten die Anwendung von IR-Techniken auf die Erschließung, Indexierung und das Retrieval multimedialer Datenbestände an Bedeutung. Unter Multimedia-IR kann man alle IR-Anwendungen verstehen, die über die einfache Erschließung von *Texten* hinausgehen und multimediale Daten als Informationseinheiten erschließen, u. a. Bild, Ton, Video, oder Graphiken. Auch wenn nach wie vor das Textretrieval die mit Abstand wichtigste Anwendung für IR-Techniken bleiben dürfte, zeichnen sich doch vielfältige Verwendungsmöglichkeiten für das Multimedia-IR ab:

- Medizinische Informationssysteme,
- Kriminalistik/Kriminologie (Personenidentifikation),
- Museen und Archive,
- Patentdatenbanken,
- Immobilienbranche (Grundrisse),
- Medienarchive (Bild-, Film- und Radioarchive).

Die Relevanz des Multimedia-Retrieval im Kontext dynamischer elektronischer Bücher ergibt sich aus dem Anteil multimedialer Substanzen in ihnen, d. h. je umfangreicher die multimedialen Bestandteile sind, um so dringlicher ist ihre Erschließung durch geeignete IR-Techniken. Es ist zu unterscheiden, ob die Erschließung lokal für die Nutzung innerhalb des Buches erfolgt oder als Ausgangspunkt für die Integration in eine externe Nutzungsstruktur (Indexierung durch Suchmaschinen, Integration in digitale Bibliotheken etc.). Die Vorgehensweise bei der Erschließung von Multimediadaten hängt von einer Reihe von Faktoren ab; die nachfolgenden Einflussfaktoren dienen als Raster zur Bewertung unterschiedlicher Strategien und Verfahren im Multimedia-IR:

Art der Daten	Einzelmedium (z. B. Filme) oder Multimediadaten im engeren Sinn (Textdokumente mit Bildern und Videos)
Quantitative Aspekte	Anzahl und Umfang der einzelnen Dokumentationseinheiten
Aufbereitungsform	Repräsentation als Rohdaten oder Auszeichnung durch logische bzw. inhaltsbezogene Strukturierung (SGML-Standards)
Indexierungsverfahren	Automatisch, manuell, Mischformen, Art des Matchingalgorithmus
Ansatzpunkt der Indexierung	Zuweisung sprachlich repräsentierter semantischer Attribute, Medienanalyse im engeren Sinn (Bildanalyse, Mustererkennung)

Abbildung 84: Einflussfaktoren im Multimedia-Retrieval

Die wichtigste Unterscheidung ist die Frage nach dem Ansatzpunkt eines Indexierungsverfahrens. Die Inhaltsanalyse als *content-based retrieval* versucht, die Primärdaten selbst zu erschließen, z. B. Videodaten zu analysieren, um Szenen erkennen zu können (vgl. etwa DEMENTHON, KOBLA & DOERMANN 1998) oder durch Spracherkennungsalgorithmen Text als Beschreibungsmerkmale für Sprachdaten zu gewinnen (vgl. SINGHAL & PEREIRA 1999, SCHÄUBLE 1997: 61 ff.). Gerade bei der Analyse von Bilddaten zeigt sich ein breites Kontinuum graphischer Formen, von echten Bilddaten beliebigen Inhalts bis hin zu Graphiken, die sich aus einer wohldefinierten und vorweg bekannten Menge graphischer Konstrukte zusammensetzen (Baupläne, Schaltskizzen etc.). Dem steht eine Bandbreite an Indexierungsverfahren gegenüber, von der manuellen Beschlagwortung mit Attributen (semantische Attribuierung), für die dann die gleichen Regeln und Phänomene gelten wie bei der manuellen Indexierung von Texten, über die teilautomatische Indexierung z. B. durch das automatische Erkennen und Zuweisen von Bildunterschriften und Textbezügen bis hin zur automatischen Bildanalyse, die für eine gegebene Graphik eine passende Repräsentation im Index aufbaut. Es ist davon auszugehen, dass man ein Bild beschreiben kann durch

- *objektive Attribute* wie Format, Größe, Farbmodell etc.,
- durch *semantische Attribute* (Beschlagwortung des Bildinhalts) und durch
- eine *automatische Inhaltsanalyse* (z. B. *spatial constraints*).

Anders als im Textbereich ist es bei der Bilddatenerschließung praktisch nicht möglich, ein allgemeines, d. h. domänenunabhängiges Retrievalsystem mit automatischer Inhaltsanalyse aufzubauen. Dies ist nur auf der Basis einer expliziten Zuweisung semantischer Attribute denkbar. Die folgenden Beispiele sollen die Heterogenität der Erschließungsverfahren verdeutlichen:

- Verwendung von Bildunterschriften zur Bildindexierung (vgl. GUGLIELMO & ROWE 1996) als Variante der Verwendung semantischer Attribute.
- Retrieval von Liniengraphiken mit Hilfe eines statistischen Ansatzes, bei dem Beschreibungsmerkmale durch eine Fouriertransformation graphischer Elemente gewonnen werden (vgl. LORENZ & MONAGAN 1995).
- Ähnlichkeitssuche in Filmdaten auf der Basis räumlicher und zeitlicher Constraints (vgl. PAPADIAS et al. 1999).

HAUPTMANN 1999: 35 identifiziert für die Erschließung von Datenbeständen, die unterschiedliche Medien kombinieren, folgende langfristige Problembereiche:

- Die Art der Anfrageformulierung für die Recherche in multimedialen Datenbeständen insbesondere hinsichtlich einer multimedialen Anfragerepräsentation,[134]
- die Verwendung von *data mining*-Techniken in multimedialen Datenbeständen,
- die Extraktion und Korrelation der aus unterschiedlichen Medien gewonnenen Merkmale,
- die Problematik der Skalierung von IR-Verfahren bei wachsenden Datenbeständen und ihre Auswirkung auf die Retrievaleffektivität und
- der Übergang von einem reinen Mustervergleich (z. B. beim Vergleich von Bilddaten) zu einem Konzeptvergleich, der das „Verstehen“ des Bildinhalts umfasst.

[134] Die visuelle Spezifikation von Suchmustern für Faktendaten (vgl. WOLFF 1996) oder die schematische Definition visueller Suchmuster bei PAPADIAS et al. 1999 sind Beispiele nichttextueller Anfragespezifikationen.

Die angesprochene Heterogenität der Daten trifft auch für elektronische Bücher zu, im Referenzprojekt liegen u. a.

- Softwarekomponenten mit multimedialen Inhalten (Animationen, Simulationen),
- schematische Graphiken (Diagramme, Funktionsgraphen),
- Photographien

als multimediale Datenbestände zum stark strukturierten Text vor. Die Heterogenität der in einem solchen elektronischen Buch enthaltenen Multimediadaten schließt die Verwendung eines *einzelnen* Multimedia-Erschließungsverfahrens aus: Für jeden Medientyp (Film, Bild, Animation etc.) wäre jeweils eine eigene Erschließungskomponente zu definieren, ggf. mit weiterer Differenzierung innerhalb der Medientypen. Für eingebettete Softwarekomponenten ist unklar, wie ein geeignetes Erschließungsverfahren aussehen könnte.[135] Gleichzeitig ist die Anzahl der einzelnen multimedialen Informationseinheiten beschränkt: Im Referenzprojekt ist auch die Häufigkeit des zahlenmäßig stärksten Medientyps, der Abbildungen (Bitmap-Dateien), mit einigen hundert überschaubar. Aus der Perspektive der Erschließung für die Nutzung des einzelnen elektronischen Buchs als „Dokumentenkollektion“ kommt dem Einsatz genuiner Inhaltsanalyseverfahren eine nachrangige Bedeutung zu. Andererseits ermöglicht die Betonung der Verwendung deklarativer Auszeichnungssprachen die Beschreibung von Multimediakomponenten im Sinne der semantischen Attribuierung (z. B. durch Beschlagwortung als Einsatz von Metadaten). Dies hat, soweit standardisierte Auszeichnungsmerkmale zum Einsatz kommen, den Vorteil, dass die Daten für die externe Nutzung z. B. im Kontext einer digitalen Bibliothek erschlossen werden können.

14.2.2.4 Information Retrieval im World Wide Web

Als zweiter Sonderbereich des Information Retrieval wird die Problematik der Informationserschließung im World Wide Web durch Suchmaschinen diskutiert. Das World Wide Web als größter elektronisch verfügbarer Informationsbestand stellt neue Aufgaben für die Informationserschließung; da das Konzept dynamischer elektronischer Bücher die Standards des World Wide Web als Kodierungsgrundlage verwendet und als zentralen Aspekt die Integration von im World Wide Web verfügbaren Diensten in das elektronische Buch beinhaltet, ist es sinnvoll, diesen Sonderbereich der Informationserschließung näher zu betrachten. Es geht vor allem um die Fragestellung, wie man die im World Wide Web verfügbaren Ressourcen für die Nutzung in einem elektronischen Buch erschließen kann. Dabei gilt folgende Annahme: Im World Wide Web lassen sich Ressourcen finden, die den Basisdatenbestand eines elektronischen Buchs erweitern können; ihre Erschließung kann durch den Benutzer aus einer konkreten *Lesesitutation* heraus erfolgen. Am Beispiel des Referenzprojekts können z. B. folgende Informationsbedürfnisse während der Interaktion mit dem elektronischen Buch auftreten, die über die Erschließung von Ressourcen im World Wide Web zu beantworten sind:

- Der Benutzer findet zu einem bestimmten Versuch im elektronischen Buch keine geeignete Multimediakomponente (z. B. die Animation eines physikalischen Versuchs oder die Simulation eines Phänomens) und will im World Wide Web nach entsprechenden Ressourcen recherchieren.

[135] Dies gilt um so mehr, als eingebettete Multimediakomponenten bisher in der Regel nicht auf der Basis der in Kap. 8 diskutierten deklarativen Standards kodiert werden, sondern durch proprietäre Binärformate z. B. eines Multimedia-Autorensystems repräsentiert sind.

- Der Benutzer möchte textuelle Zusatzinformation zu einem bestimmten Problembereich ermitteln (z. B. eine genauere Darstellung einer Herleitung; neuere Literatur zu einem Spezialaspekt).

Es gibt bisher keine differenzierte Theorie der Typisierung von Informationsbedürfnissen im World Wide Web. Unterscheidungen wie sie BEAZA-YATES und & RIBEIRO-NETO 1999: 91 f. treffen (*specific queries, broad queries, vague queries*), haben kaum eine ausreichende Trennschärfe. Erste Benutzerstudien für das Web Retrieval, wie GORDON & PATHAK 1999, nehmen ebenfalls keine Typisierung der Anfragen vor, sondern zeigen lediglich die thematische Bandbreite ihrer Testanfragen auf (vgl. GORDON & PATHAK 1999: 150, Tabelle 2 und 177 (Beispielanfrage) und LEIGHTON & SRIVASTAVA 1999).

Die Betrachtung des World Wide Web als Ressourcenfundus zur Beantwortung von Informationsbedürfnissen ist nur dann sinnvoll und möglich, wenn nicht von vornherein durch Aufbereitung durch Autoren oder die Integration des Buchs in einen spezifischen Nutzungskontext solche Informationen bereits vorhanden sind; dies wurde als Möglichkeit der Integration externer Ressourcen in das dynamische elektronische Buch in Kap. 4.4.1.1 diskutiert. Das World Wide Web ist nur ein – allerdings besonders wichtiger – Informationsdienst unter anderen: In allgemeiner Betrachtung kann jeder elektronisch verfügbare Informationsdienst in solchen Situationen herangezogen werden, z. B. Online-Datenbanken wie INSPEC.

Der hohe Anteil an Daten aus dem akademischen Bereich (Webserver der Universitäten und ihrer Institute) gibt eine hinreichende Anfangsplausibilität für die Verfügbarkeit passender Ressourcen ab – in Bezug auf das Referenzprojekt z. B. die Webserver physikalischer Institute bzw. Experimentallabors. Die Alternativstrategie, relevante Ressourcen intellektuell zu erschließen und als Sammlung relevanter Verknüpfungen in das elektronische Buch zu integrieren, erscheint aufgrund der hohen Änderungsrate von Inhalten und Verweisen im World Wide Web als weniger sinnvoll, von als stabil angenommenen Verweisen auf institutionelle Websites abgesehen; zudem können sich beide Verfahren ergänzen.[136] Nachfolgend soll ausgehend von einigen charakteristischen Daten zum World Wide Web als verteiltem Informationsbestand beschrieben werden, welche Recherchewerkzeuge zur Verfügung stehen. Ihre Integration als Dienst in ein elektronisches Buch ist Gegenstand von Kap. 14.2.2.5.

14.2.2.4.1 Charakteristika des World Wide Web

Die im World Wide Web vorhandenen Daten unterschieden sich erheblich von den traditionellen Dokumentenkollektionen in Information Retrieval-Systemen. Sie sind durch folgende Merkmale gekennzeichnet:

- Die Daten sind auf eine Vielzahl von Hosts im Internet verteilt, aktuelle Schätzungen (NETCRAFT 1999, NETSIZER 1999, OCLC 1999) gehen von einer Anzahl von Webservern bzw. Websites zwischen vier und acht Millionen aus.
- Anzahl und Umfang der Informationseinheiten wachsen sehr dynamisch; gleichzeitig sind die Daten relativ unbeständig, d. h. ihre Verfügbarkeit über einen längeren Zeitraum ist nicht gesichert: In OCLC 1999 wird berichtet, dass 42% aller IP-Nummern, die 1998 auf einen Webserver verwiesen, 1999 nicht mehr auf einen Webserver verweisen.

[136] Dies im Sinne der hier entwickelten Diensteinfrastruktur, wenn etwa für einen bestimmten Nutzungskontext des Buchs Sammlungen relevanter Verweise ins World Wide Web als Zusatzmaterial dynamisch integriert werden.

- Insgesamt haben die Daten ein sehr hohes Volumen, sowohl was die Anzahl der Webseiten (etwa 10^9 Dokumente bis Anfang 2000) als auch ihre Gesamtmenge betrifft.
- Die Daten sind in der Regel semi- oder unstrukturiert, d. h. sie können deklaratives Markup in gewissem Umfang enthalten (HTML-Dateien); es existiert eine hohe Vielfalt an Medientypen und Formaten, wobei Text vorherrschend ist.
- Die Ressourcen liegen in unterschiedlichen Sprachen vor, bei eindeutiger Dominanz des Englischen: Aus verschiedenen Studien ergibt sich der Anteil des Englischen zu ca. 50-80 %, gefolgt von Japanisch und Deutsch mit etwa 4 %, vgl. PIMIENTA et al. 1998.
- Durch die allgemein verfügbaren Publikationsmöglichkeiten im World Wide Web, bei denen in vielen Fällen keine Qualitätskontrolle stattfindet, sind die Inhalte unterschiedlicher (inhaltlicher, formaler) Qualität.

Der häufigste Typ einer Informationseinheit im World Wide Web, eine HTML-Seite, ist im Mittel relativ klein (ca. 8 KB im arithmetischen Mittel, bei einem Median von etwas mehr als 2 KB) und enthält zwischen fünf und zehn Verweise, meist auf andere lokale Ressourcen auf demselben Server.

Diese Charakteristika des World Wide Web sind aus verschiedenen Studien zusammengestellt, vgl. BAEZA-YATES & RIBEIRO-NETO 1999: 369 ff. Eine einheitliche Methodologie für die Bestimmung der wesentlichen Merkmale des World Wide Web beginnt sich – unter dem Einfluss der *Web Characterization Activity* des *World Wide Web Consortium* – erst herauszubilden (vgl. OCLC 1999, LAVOIE & FRYSTYK NIELSEN 1999). PITKOW 1998 gibt einen Literaturüberblick zur den bisherigen Ansätzen der Kenngrößenerfassung und Charakterisierung des World Wide Web.

14.2.2.4.2 Suchmaschinen

Die Informationserschließung im World Wide Web kann grundsätzlich auf unterschiedliche Arten erfolgen:

- Durch direkte Anwahl einer bekannten Adresse (einer Website, einer einzelnen Seite), ggf. in Verbindung mit einer lokalen Suche auf einem Webserver,
- durch die Verwendung eines Verzeichnisdienstes, der Ressourcen aus dem World Wide Web klassifiziert und ordnet,[137]
- durch die Verwendung einer Suchmaschine, die über einen Index eines Teils der im World Wide Web verfügbaren Ressourcen verfügt und auf Anfrage Nachweise (d. h. Adressen und Kurzbeschreibungen) von Dokumenten im World Wide Web liefern kann oder
- durch Metasuchmaschinen, die zwar selbst keinen Index von Webressourcen verwalten, aber die Ergebnisse einer Mehrzahl einzelner Suchmaschinen bündeln.

Zusätzlich existieren Sonderformen wie Agenten (News-Watcher-Agenten, persönliche Informationsfilter etc.), die an spezielle Inhalte oder Benutzerbedürfnisse angepasst sind und Informationen aus dem World Wide Web selektieren oder im zeitlichen Verlauf neue Informationen filtern (vgl. BÖHME 1999). Für die Integration als Dienst in ein elektronisches Buch kommen grundsätzlich alle Kategorien von Suchwerkzeugen in Betracht, da sie in der Regel über eine generische Anfrageschnittstelle verfügen, die sich in ein elek-

[137] Das bekannteste und älteste Beispiel eines solchen Verzeichnisdienstes ist Yahoo (*http://www.yahoo.com*, *http://www.yahoo.de*).

tronisches Buch integrieren lässt. Am Beispiel von Such- und Metasuchmaschinen sollen nun ihr Aufbau sowie ihre Leistungsmerkmale beschrieben werden. Es besteht allerdings das grundsätzliche Problem, dass die (leistungsfähigeren) Suchmaschinen im World Wide Web kommerzielle Softwaresysteme sind, über deren inneren Aufbau nur wenige Details bekannt sind; in Kenntnis der Verfahren des Information Retrieval und durch Analyse der von den Suchmaschinen erzielten Ergebnisse und der dabei verwendeten Anfragelogik lassen sich aber Rückschlüsse bezüglich ihrer Funktionsweise ziehen. Eine Suchmaschine für das World Wide Web ist in der Regel aus folgenden Komponenten aufgebaut:

1. Einem oder mehreren *Erfassungsagenten* (*spider, crawler*), die Dateien aus dem World Wide Web zur Indexierung laden,
2. einer *Indexierungskomponente*, die die Daten auswertet und den Index aufbaut und verwaltet,
3. der *Abfrageschnittstelle*, über die der Benutzer seine Anfrage an die Suchmaschine formulieren kann, und
4. der *Anfrageverarbeitung*, die die Anfrage mit dem Index abgleicht (Retrievalfunktion) und die Ergebnisse an die Benutzerschnittstelle der Suchmaschine weiterleitet.

Die folgende Abbildung zeigt schematisch das Zusammenwirken der einzelnen Komponenten einer Suchmaschine (vgl. BAEZA-YATES & RIBEIRO-NETO 1999: 374, Abb. 13-3):

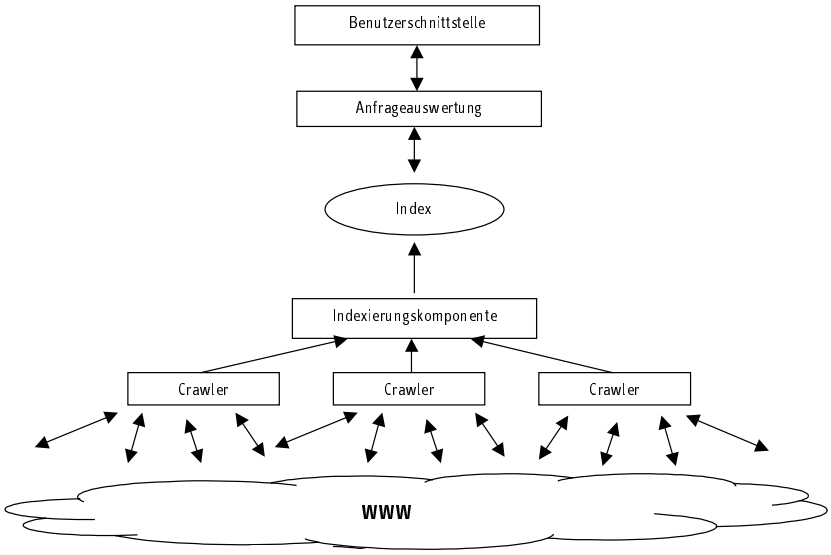

Abbildung 85: Schematischer Aufbau einer WWW-Suchmaschine

Bei dem Crawler als Informationsbeschaffungskomponente einer Suchmaschine handelt es sich um ein Softwaremodul, das nach einer vordefinierten Arbeitsstrategie WWW-Server besucht, und deren Seiten zur Indexierung lädt; man kann ihn als Ausprägung eines Softwareagenten auffassen (vgl. CAGLAYAN & HARRISON 1998: 56 ff.). Dabei ist die technische Realisierung in der Regel als verteiltes System gelöst, bei dem eine Vielzahl von Crawler-Modulen parallel arbeitet; die Strategien für das Durchlaufen des Adressraums des World Wide Web variieren je nach Suchmaschine und können neben dem Laden explizit angegebener Adressen (Startliste von URLs) sowohl eine systematische Abarbeitung des IP-Adressraums als auch eine Vorgehensweise nach verschiedenen top-level-Domänen (.de, .com, .edu etc.) beinhalten. Hinzu kommen Indexierungsheuristiken, die z. B. die Zahl der Referenzen auf eine Website als Indikator für die Aktualisie-

rungsintervalle nutzen und Strategien für die lokale Abarbeitung der Inhalte eines Website.[138] Man geht derzeit von einer Indexierungsleistung in der Größenordnung von 10 Millionen Webseiten pro Tag aus, was bedeutet, dass ein vollständiger Durchlauf ca. 2-3 Monate in Anspruch nimmt.[139]

Die Indexierung baut analog zu der bekannten Architektur eines Information-Retrieval-Systems eine invertierte Datei auf, speichert aber nicht die *Volltexte* der Kollektion (d. h. des World Wide Web), da dies aus Speicher- und Aktualitätsgründen nicht sinnvoll ist. Die Analyse erfolgt auf der Basis des Volltexts jeder Webseite, wobei die üblichen Strategien der Textanalyse (Extraktion lexikalische Einheiten, Eliminierung von Stoppwörtern) zum Einsatz kommen. Zusätzlich können Strukturmerkmale (HTML-Marken für Schlagwörter, Überschriften, Hervorhebungen) bei der Indexierung berücksichtigt sein. Die invertierte Datei enthält als Beschreibungsmerkmale für jedes Dokument in der Regel nicht nur die Adresse (URL) selbst, sondern darüber hinaus eine Kurzbeschreibung des Inhalts (Textanfang, Überschrift, Textextrakt) und formale Angaben (z. B. letztes Aktualisierungsdatum). Zu den Leistungsmerkmalen von Suchmaschinen gehören ihre

- Datenbankgröße und ihr Abdeckungsgrad hinsichtlich des World Wide Web,
- die Qualität und Aktualität der gespeicherten Daten,
- die Anfragemerkmale und das Retrievalmodell sowie
- die Art der Ergebnisaufbereitung.

Aktuelle Statistiken zeigen, dass die umfangreichsten Indizes von Suchmaschinen eine Größe von ca. 200 Mio. Nachweisen erreichen:

	Search Engine Showdown (XI/99, NOTESS 1999)		*Search Engine Watch (XI/99, SULLIVAN 1999)*
Suchmaschine	*Anzahl Webseiten*	*Ohne tote Links*	
Northern Light	200	196	189
Fast Search	192	158	200
AltaVista	191	174	250
Google!	126	124	85
Anzwers	79	75	–
!Won	78	69	–
Excite	71	68	150
Yahoo	68	68	–
AOL	68	–	–
Lycos	55	53	50
InfoSeek	52	50	75
Snap	50	39	–
HotBot	39	–	–

Tabelle 62: Datenbankumfang bekannter Suchmaschinen (Millionen URLs)

Untersuchungen von NOTESS 1999 und LAWRENCE & GILES 1998 ergeben, dass diese Indizes weitgehend disjunkt sind, d. h. der indexierte Datenbestand differiert stark zwischen den verschiedenen Suchmaschinen. LAWRENCE & GILES 1998, 1999 kommen zu folgenden Ergebnissen: Derzeit deckt keine Suchmaschine mehr als 16 % des World Wi-

[138] In der Regel als ein rekursives (*depth-first* oder *breadth-first*) Verfolgen der in den Dateien einer Website enthaltenen Hypertextverknüpfungen.

[139] Vgl. zu den Kennzahlen von Suchmaschinen den *Search Engine Showdown* (NOTESS 1999) sowie die Statistiken des *Search Engine Watch* (SULLIVAN 1999); Informationen über proprietäre Systeme finden sich z. B. bei INKTOMI 1999.

de Web ab; generell werden Sites, auf die viele externe Verknüpfungen verweisen, bei der Indexierung berücksichtigt. Hinsichtlich der Domänen gilt dies auch für kommerzielle Sites (.com) sowie in den USA betriebene Webserver.

Eine ebenfalls hohe Schwankungsbreite existiert bei den „toten Links“, d. h. nachgewiesenen HTML-Seiten, die aufgrund des Alters des Index bereits nicht mehr existieren; je nach Suchmaschine ist von einem Anteil zwischen 3 und 20 % auszugehen (vgl. NOTESS 1999, LEIGHTON & SRIVASTAVA 1999: 880 f., Appendix B, C).

Im Unterschied zu „einfachen“ Suchmaschinen verwalten Metasuchmaschinen keinen eigenen Index. An die Stelle der Crawler-Komponente tritt bei ihnen die Schnittstelle zu verschiedenen Suchmaschinen, die von der Metasuchmaschine abgefragt wird. Die verschiedenen Suchergebnisse werden anschließend von der Metasuchmaschine unifiziert. Das Ziel ist die Optimierung der Retrievaleffektivität, da anzunehmen ist, dass aufgrund der geringen Überlappungsraten in den Inhalten einzelner Suchmaschinen eine solche Kombination von Ergebnissen die Verbesserung der Suchresultate nach sich zieht. Dazu fehlen allerdings bisher empirische Studien.[140] Die Architektur einer Metasuchmaschine hat folgenden Aufbau:

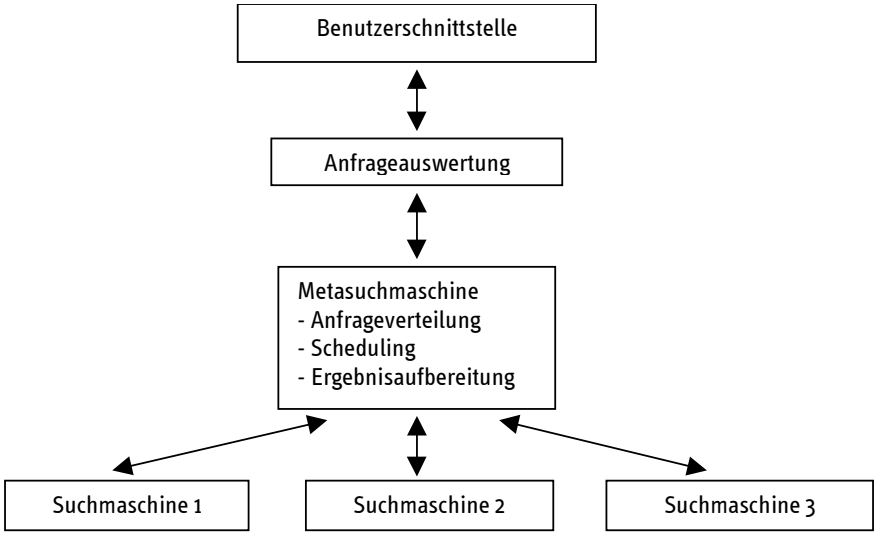

Abbildung 86: Schematischer Aufbau einer Metasuchmaschine

Eine Metasuchmaschine verfügt über Suchagenten für jede Suchmaschine und übersetzt die Anfrage des Benutzers in die jeweilige Abfragesprache der Einzelsuchmaschine, was zur Folge hat, dass Metasuchmaschinen oft nur eine im Vergleich zu einzelnen Suchmaschinen eingeschränkte Mächtigkeit der Anfragesprache aufweisen. Die von jedem Suchagenten ermittelten Ergebnislisten werden nach unterschiedlichen Verfahren gemischt und nach Qualität sortiert (*ranking*). Übersetzung der Anfragesprache und Mischen der Ergebnisse sind die schwierigsten konzeptuellen Probleme, die von einer Metasuchmaschine gelöst werden müssen. Suchmaschinen erlauben in der Regel nur die Berücksichtigung einer begrenzten Anzahl von Ergebnissen pro abgefragter Suchmaschine und verfügen über Zeitschranken, die die Abfrage nicht verfügbarer Suchmaschinen be-

[140] Eine Anfang 2000 begonnene Studie des Autors zur vergleichenden Evaluierung von Such- und Metasuchmaschinen im World Wide Web scheint nach einer ersten punktuellen Auswertung der Ergebnisse keinen klaren Vorteil von Metasuchmaschinen zu ergeben, vgl. WOLFF 2000.

enden (beide Parameter können ggf. vom Benutzer eingestellt werden). Zu den bekannten Metasuchmaschinen zählen

- *SavvySearch* (*http://www.savvysearch.com*, vgl. DREILINGER 1996),
- *MetaCrawler* (*http://www.metacrawler.com*) und
- *MetaGer* (*http://www.metager.de*, für den deutschsprachigen Bereich).

Die von den Suchmaschinen verwandten Retrievalmodelle bauen auf Booleschen und statistischen Verfahren auf; in der Regel besteht für den Benutzer sowohl die Möglichkeit, Operatoren der Booleschen Logik bei der Anfrage zu verwenden, als auch eine natürlichsprachliche Eingabe (mit praktisch beliebiger Länge) ohne Verwendung von Suchoperatoren zu verwenden. Die nachfolgende Tabelle fasst Merkmale der Anfragesprachen und ihre Ausprägungen zusammen (nach NOTESS 1999; vgl. auch SCHWARTZ 1998: 975 f.).

Merkmal	*Ausprägungen*
Defaultstrategie	Konjunktion, Disjunktion
Boolesche Operatoren	AND, OR, NOT, (), sowie die vereinfachte Syntax: +, -
Abstandsoperatoren	NEAR, exakte Phrase
Trunkierung	Rechtstrunkierung, Einzelzeichenmaskierung
Morphologische Expansion	v. a. automatische Pluralbildung im Englischen (nur wenige Suchmaschinen)
Suche in Feldern	Titel, URL, Website
Inhaltliche und sprachliche Eingrenzung	Dokumenttyp, Sprache, Datum (Sprachheuristik i. d. R. über Domänen)
Stoppworteliminierung	ja/nein

Tabelle 63: Recherchemöglichkeiten und Anfragesprachen von Suchmaschinen

Betrachtet man die Ausprägungen der in Tabelle 63 genannten Merkmale für die einzelnen Suchmaschinen, so fällt auf, dass sich bisher kein Konsens hinsichtlich der Interpretation von Benutzeranfragen ohne Operatorenverwendung herausgebildet hat; die Suchmaschinen verwenden unterschiedliche Default-Strategien bei der Anfrageinterpretation. Dies ist vor allem deswegen bemerkenswert, weil Benutzer in der Regel kurze Anfragen ohne Operatoren stellen. Soweit verschiedene Suchmaschinen dies logisch unterschiedlich interpretieren, kann sich kein allgemein akzeptiertes Interpretationsmodell für Suchanfragen herausbilden, mit der Folge, dass Operatoren vielfach falsch eingesetzt werden (vgl. JANSEN, SARACEVIC & SPINK 2000).

Die Ergebnisausgabe der Suchmaschinen erfolgt in der Regel als qualitätssortierte Liste von Adressen mit Kurzbeschreibung der Dokumentinhalte (*ranking*); in den erweiterten Suchmodi kann der Benutzer zusätzliche Sortierkriterien (Website, Datum, „Zitierhäufigkeit“, d. h. Anzahl externer Links auf eine Webseite) einführen.

Die Mehrzahl der bisher vorliegenden Studien zur Retrievaleffektivität von Suchmaschinen beschränkt sich auf einfache Maße wie die Anzahl der nachgewiesenen Treffer hinsichtlich einer Anfrage, ohne aber eine Effektivitätsbewertung im engeren Sinn durchzuführen.[141] Recherchen im World Wide Web stellen im Kontext des IR insofern ein Novum dar, als ein wesentlich größerer Nutzerkreis ohne technische Vorkenntnisse mit Information Retrieval-Systemen interagiert, vgl. dazu KIRSCH 1998, JANSEN et al. 1998 und SILVERSTEIN 1999. KIRSCH 1998: 4 berichtet, dass für die Suchmaschine *Infoseek* nur 1 % aller Anfragen mit Hilfe von *advanced search features* erfolgen. Intelligente Anfrageunterstützungsmechanismen, wie sie zeitweilig von AltaVista oder Lycos angeboten

[141] Ein Überblick über bisherige Evaluierungsstudien für Suchmaschinen im World Wide Web findet sich bei GORDON & PATHAK 1999: 145 ff., insb. 148: Tabelle 1.

worden waren, sind mittlerweile wieder aus dem Angebot verschwunden, vgl. SCHWARTZ 1998: 977, Abb. 1: *Lycos Pro Power Panel* und 980: Abb. 2: *AltaVista Refine.* Nach ersten Studien lässt sich ein „typische Suchmaschinennutzer" wie folgt charakterisieren:

- Es werden kurze Anfragen gebildet (weniger als drei Suchbegriffe im Mittel, vgl. JANSEN et al. 1998, JANSEN, SPINK & SARACEVIC 2000).
- BOOLEsche Operatoren werden kaum verwendet (weniger als 10 % der Anfragen), positive und negative Auftretensoperatoren (+/-) in derselben Größenordnung. Zudem enthalten zahlreiche Anfragen logische Fehler hinsichtlich der Verwendung von Operatoren (JANSEN, SPINK & SARACEVIC 2000: 217).
- Benutzer betrachten nur selten mehr als zwei Ergebnisseiten (ein *cut-off*-Wert bei der Bewertung der Suchergebnisse von weniger als 30, meist wird nur die erste Ergebnisseite betrachtet).[142]

Eine erste umfassende benutzerbezogene Studie zur Retrievaleffektivität haben GORDON & PATHAK 1999 vorgelegt. Sie führen in einem Experiment mit 33 Testpersonen eine vermittelte Recherche an acht verschiedenen Suchmaschinen durch, wobei die Suchergebnisse von den Testpersonen nach einer vierstufigen Skala hinsichtlich ihrer Relevanz bewertet werden. Die Ergebnisse zeigen eine hohe Schwankungsbreite der Suchmaschinen sowohl hinsichtlich der *precision* als auch des *recall*. Bei einem cut-off-Wert von 20 betrachteten Dokumenten erreicht die qualitativ beste Suchmaschine (AltaVista) einen *recall* von nur 15 %, selbst bei 200 betrachteten Dokumenten steigt der Wert nicht über 23 % an (GORDON & PATHAK 1999: 160 ff.).

Auch in dieser Studie wird die bereits erwähnte Beobachtung geringer Überlappung der Ergebnismengen verschiedener Suchmaschinen bestätigt: Von insgesamt 160 Treffern (die ersten 20 Treffer von acht Suchmaschinen) wurden 150 Dokumente jeweils nur von einer Suchmaschine nachgewiesen (GORDON & PATHAK 1999: 170 ff.).

Die bisherigen Ergebnisse zur Bewertung von Suchmaschinen und ihrer Retrievaleffektivität geben eine starke Anfangsplausibilität für die Verwendung von Metasuchmaschinen sowie für die Weiterentwicklung der Retrievalfunktionalität hinsichtlich besserer Benutzerschnittstellen und der Auswertung weiterer Strukturmerkmale der indexierten Texte; die im Rahmen des Konzepts dynamischer elektronischer Bücher diskutierte Einführung mehrerer Ebenen deklarativen Markups stellt insofern eine Tendenz dar, die in der Zukunft eine bessere Grundlage für die Indexierung und den Nachweis mit Hilfe von Suchmaschinen ergeben kann.

14.2.2.5 Informationserschließung für elektronische Bücher

Nach der Einführung von Grundbegriffen des Information Retrieval und der Diskussion relevanter Sonderbereiche ist zu untersuchen, wie sich Erschließungstechniken auf dynamische elektronische Bücher anwenden lassen und welche Besonderheiten dabei zu berücksichtigen sind. Es besteht zwar kein Zweifel darüber, dass Verfahren zur Informationserschließung durch IR-Techniken ein wesentlicher funktionaler Bestandteil elektronischer Bücher sind, bei kommerziellen Lösungen für elektronische Bücher sind diese Mechanismen aber nur schwach ausgeprägt:

[142] SILVERSTEIN et al. 1999: 10 berichten auf der Basis der Analyse mehrerer hundert Millionen Anfragen, dass bei etwa für 85% aller Anfragen nur die erste Ergebnisseite betrachtet wird, bei JANSEN, SPINK & SARACEVIC 2000: 215 liegt dieser Wert bei „nur" 58%.

- In vielen Fällen existiert nur eine einfache lineare Suche, d. h. die Recherche basiert auf sequentieller Stringsuche durch den Informationsbestand ohne eine leistungsfähige Anfragesprache bzw. einen Volltextindex.
- Eine differenzierte und strukturierte Aufbereitung ist nur bei datenbankorientierten Anwendungen (v. a. im Bereich elektronischer Lexika und Enzyklopädien) möglich und erlaubt dann die Suche in Datenfeldern, die Verwendung von Suchoperatoren etc.
- Ist ein Volltextindex vorhanden, so handelt es sich in der Regel um eine einfache Boolesche Suche ohne qualitätsorientierte Ergebnisausgabe (keine Termgewichtung), die Suchoperatoren beschränken sich auf die elementare Boolesche Algebra (AND, OR, NOT).

Die Gründe hierfür finden sich

- in der beschränkten strukturierten Aufbereitung der Informationseinheiten elektronischer Bücher (kein oder beschränktes deklaratives Markup) und
- in den begrenzten Möglichkeiten der für die Informationsaufbereitung vorhandenen Werkzeuge.

Multimedia-Autorensysteme enthalten üblicherweise keine IR-Verfahren zur Aufbereitung der Informationseinheiten und müssen durch zusätzliche IR- bzw. Datenbankfunktionalität ergänzt werden,[143] geeignete IR-Systeme für elektronische Publikationen[144] sind in der Regel auf ein einfaches IR-Modell und die Indexierung von Texten beschränkt bzw. werten Strukturmarkup nur bedingt aus (vgl. oben Kap. 11.2).

14.2.2.5.1 Besonderheiten elektronischer Bücher

Gegenüber den traditionellen IR-Anwendungen wie Online-Datenbanken bibliographischer Nachweise, großen Kollektionen von Dokumenten im Volltext oder digitalen Bibliotheken weisen dynamische elektronische Bücher eine Reihe prinzipieller Unterschiede auf, die für die Informationserschließung Konsequenzen haben:

1. Es handelt sich um einen in sich geschlossenen und zusammengehörigen Informationsbestand mit strukturiertem Markup, das für die Erschließung herangezogen werden kann (z. B. bei der Festlegung hervorgehobener Begriffe als Indexierungsmerkmale).
2. Die Bezugseinheiten der Informationserschließung sind Bestandteil des Informationsbestands; die Wahl der Granularität der Bezugseinheiten kann sowohl auf der Basis logischer Kriterien (z. B. Hierarchieebenen des inhaltlichen Aufbaus, inhaltsorientiertes Markup) als auch auf der Basis von *präsentationsorientierten* Einheiten (z. B. einzelne Darstellungsseiten/Bildschirmseiten) erfolgen.
3. Der Umfang der einzelnen Informationseinheit ist in der Regel klein; dies gilt besonders bei Verwendung der kleinsten *Präsentationseinheit* als Bezugspunkt (Inhalt einer Bildschirmseite).

---

[143] Z. B. durch sog. Xtras wie FileFlex oder DB12 bei Macromedia Director oder Eigenentwicklungen wie im Fall der Toolbook-Anwendung UNIVERSITÄT LEIPZIG 1998, bei der die IR-Funktionalität als Boolesches Retrievalsystem durch eine Eigenentwicklung bereitgestellt wird.

[144] Z. B. kommerzielle Produkte wie der Fulcrum Search Server (*http://www.hummingbird.com/products/dkm/km/searchserver/*), dem Verity Information Server™ (*http://www.verity.com/products/infoserv/*) oder als Free- oder Shareware erhältliche Produkte für Indexierung und Recherche (z. B. FREEWAIS-SF oder SWISH-E, vgl. die Übersicht bei *http://searchtools.com/tools/tools.html*).

4. Die Inhalte sind nicht einem einzelnen Medium zuzuordnen (Text), sondern können unterschiedliche Medien und Repräsentationsformen (Text, Formelsatz, Abbildungen, Diagramme, Daten (z. B. in Tabellen), Multimediaelemente etc.) beinhalten.
5. Die Recherchesituation des Benutzers kann vom System ausgewertet werden, da bei Nutzung eines elektronischen Buchs der informationelle Kontext einer Anfrage aus dem Lesekontext (s. u. 14.2.2.5.2) ermittelt und z. B. für die Anfrageerweiterung und Anfrageunterstützung genutzt werden kann.
6. Das elektronische Buch ist nicht isoliert zu betrachten, sondern die Einbindung in die Infrastruktur z. B. elektronischer Bibliotheken und Informationsdienstleistungen (wie z. B. im World Wide Web angeboten) ist zu beachten.

Bei der Informationserschließung für elektronische Bücher sind zwei unterschiedliche Ansatzpunkte zu unterschieden:

1. die Erschließung der Inhalte des elektronischen Buchs selbst („Suche im Buch“) und
2. die Erschließung zusätzlicher Information aus externen Quellen (World Wide Web, Datenbanken etc.).

Die Nutzungssituation des Lesers, in der er gezielt im Informationsbestand des Buches sein Informationsbedürfnis lösen oder darüber hinaus zusätzliche Ressourcen finden will, ist der Ansatzpunkt für die Informationserschließung aus externen Quellen; man kann sich den Informationsraum, in dem der Benutzer navigiert, als ein Schalenmodell vorstellen, in dessen innerster Schale sich der Informationsbestand des elektronischen Buchs selbst befindet, die explizit dem Buch zugeordneten Informationen und Dienste in einer weiteren Schale, alle externen Ressourcen in einer äußeren Schale. Je weiter man in diesem Modell nach außen geht, um so wichtiger werden genuine IR-Verfahren für die Erschließung, da dann die Erschließungs- und Navigationsmöglichkeiten für den Inhaltsbestand des Buchs (Inhaltssequenz, Hierarchienavigation, Hypertextverknüpfungen) nicht zur Verfügung stehen. Dies versucht die folgende Abbildung zu veranschaulichen:

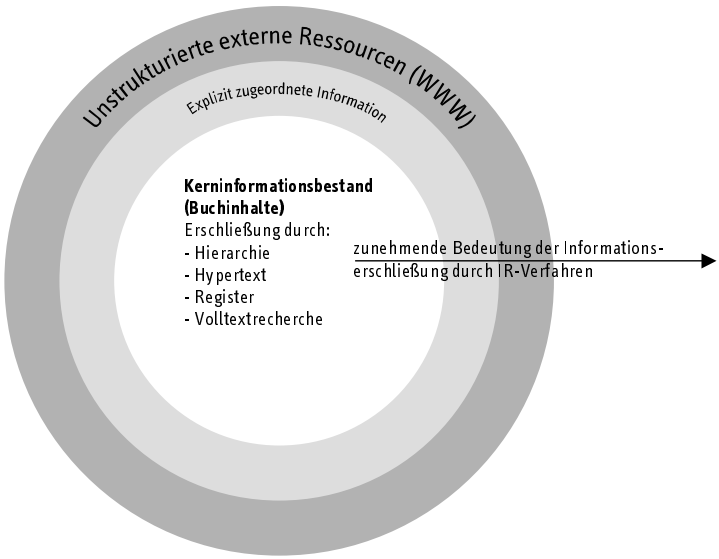

Abbildung 87: Informationsumgebung des Lesers als Schalen abnehmender Strukturierung

Im innersten Bereich stehen unterschiedliche Verfahren für die Informationserschließung zur Verfügung, deren Schnittstelle bereits oben in Kap. 13.3.5 (Abbildung 57 f.) vorgestellt wurde. Der Benutzer hat neben der Hierarchienavigation, der sequentiellen Er-

schließung („Leserichtung“) und dem Verfolgen von Hypertextverknüpfungen zusätzlich die Möglichkeit der Informationserschließung über einen Index sowie die gezielte Volltextrecherche. Es kann angenommen werden, dass aufgrund der Vielfalt der Erschließungsverfahren und der Geschlossenheit der Inhalte eines elektronischen Buchs diese Verfahren für die meisten Informationsbedürfnisse ausreichend sind.[145]

Für die dynamisch eingebundenen Inhalte reduzieren sich diese Möglichkeiten, da sie nicht automatisch in den Index bzw. die invertierte Datei der Volltextrecherche eingebunden werden können; in diesem Fall verbleibt die Erschließung durch Hypertextverknüpfungen.

Der wichtigste zu untersuchende Punkt ist die *Integration externer Ressourcen* über in das Buch als Dienst eingebundene Informationserschließungsdienste; dies soll am Beispiel von (Meta-)Suchmaschinen im World Wide Web diskutiert werden.[146] Da Informationserschließung in externen Ressourcen aufgrund der methodischen Vorannahme, dass die Realisierung aller wünschenswerten multimedialen Ergänzungen für eine Einzelpublikation aus Zeit- und Kostengründen nicht möglich ist und gleichzeitig in vielen Fällen über freie wie proprietäre externe Ressourcen eine sinnvolle Ergänzung denkbar ist, kann man festhalten, dass dies ein passendes Beispiel für eine modulare Ergänzung der Funktionalität elektronischer Bücher darstellt. Dabei müssen folgende Fragen diskutiert werden:

- Wie lässt sich der aktuelle *Lesekontext* des Benutzers für die Unterstützung bei der Anfrageformulierung auswerten?
- Wie können eingebettete Multimediakomponenten beschrieben werden, um einen Beitrag zur Informationserschließung zu leisten?
- Welche zusätzlichen Dienste können für die Unterstützung des Benutzers bei der Anfrageformulierung herangezogen werden (vgl. KRAUSE 1992: 38 ff.)?
- Wie wird der Informationserschließungsdienst in die Benutzerschnittstelle des elektronischen Buchs integriert?
- Welche Möglichkeiten stehen dem Benutzer zur Weiterverarbeitung der Suchergebnisse zur Verfügung?

14.2.2.5.2 Auswertung des Lesekontexts

Der aktuelle Lesekontext dient als Gewinnungsreservoir für potentielle Anfragen, wobei Metainformation wie die aktuelle Dokumenthierarchie bzw. die Beschlagwortung multimedialer Komponenten herangezogen werden kann. Neben der Auswertung explizit formulierter Informationsbedürfnisse durch den Benutzer hat man die zusätzliche Möglichkeit, aus den ohnehin vorgegebenen und elektronisch verfügbaren informationellen Einheiten des elektronischen Buchs Vorschläge zur Definition einer Suchformulierung zu gewinnen. Unter dem aktuellen *Lesekontext* ist die letzte *Position* des Benutzers im elektronischen Buch zu verstehen, die durch Auswertung von Struktur- und Inhaltsmarkup

[145] Diese These bleibt ohne empirische Überprüfung; die Redundanz verschiedener Erschließungsverfahren sowie ihre Analogie zu gedruckten Büchern (Inhaltsverzeichnis, Index) machen sie aber hinreichend plausibel.

[146] Es liegt der Fall eines externen Dienstes vor, der durch Kapselung in das elektronische Buch eingebunden werden kann, vgl. oben Kap. 4.4.6.2; die frei verfügbaren Suchmaschinen stehen prototypisch für Informationserschließungsdienste, die über eine WWW-Schnittstelle verfügen, wie etwa der Recherchedienst *STN Easy* (*http://stneasy.FIZ-Karlsruhe.de*); solche Dienste ließen sich bei entsprechender Adaption des Diensteobjekts alternativ oder zusätzlich einbinden.

sowie durch die Ermittlung der in diesem Kontext eingebetteten Komponenten (Texte, multimediale Inhalte etc.) auf den ihr entsprechenden Inhaltsbereich des elektronischen Buchs abgebildet werden kann. Folgendes ist dabei anzunehmen:

1. Der aktuelle Lesekontext stellt einen Indikator für die Informationsbedürfnisse des Lesers dar.
2. Die Granularität des Kontextes kann bestimmt werden.
3. Es existieren Verfahren der Termextraktion, die nach Präzisierung des Lesekontextes die für den Anfrageaufbau benötigten Terme liefern.

Die erste Annahme beruht auf der Überlegung, dass die aktuelle Leseposition Rückschlüsse auf potentielle Informationsbedürfnisse zulässt: Der Leser setzt sich mit einem bestimmten Informationsbestand auseinander; soweit sein Informationsbedürfnis nicht abschließend gestillt ist, kann man annehmen, dass Zusatzinformation zu dem thematischen Schwerpunkt des aktuellen Lesekontexts gewünscht wird. Es ist zunächst unerheblich, ob das System auch ohne explizite Anfrage Zusatzinformation ermittelt und diese sofort (z. B. als Verweisliste) zur Verfügung stellen kann oder ob dieser Rechercheprozess erst gestartet wird, wenn der Benutzer ihn explizit anfordert. Den Lesekontext innerhalb des elektronischen Buchs verdeutlicht Abbildung 88.

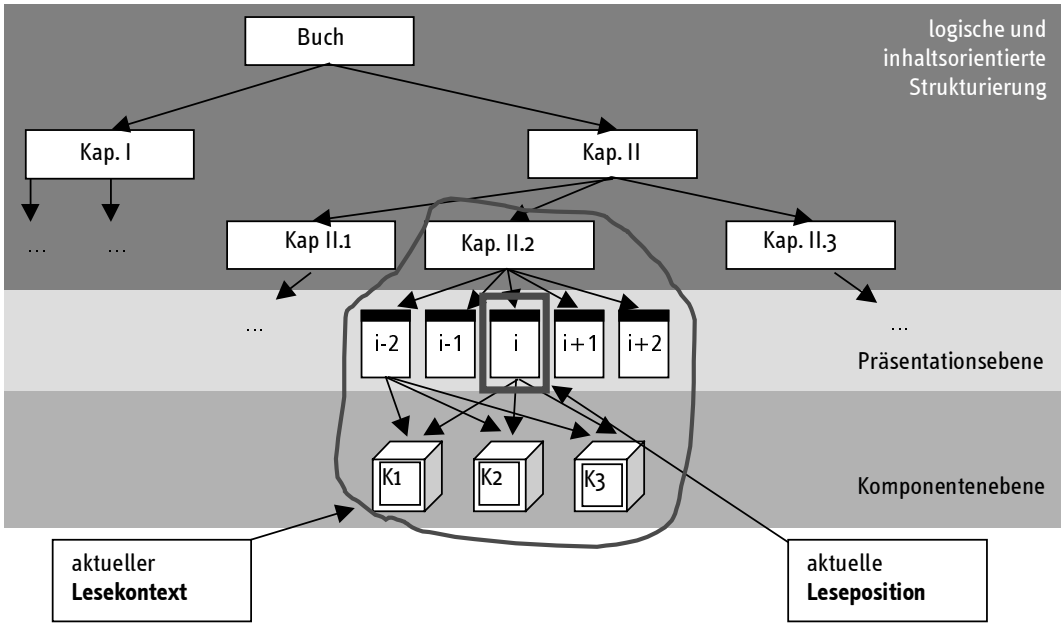

Abbildung 88: Schematische Darstellung des Lesekontexts

Will der Benutzer nach Interaktion mit dem elektronischen Buch aus einem bestimmten Lesekontext heraus zusätzliche Informationen aus externen Ressourcen erschließen, so ist davon auszugehen, dass der aktuelle Lesekontext Hinweise auf Inhalt und Art des Informationsbedürfnisses ergibt. Das Informationsbedürfnis als *anomalous state of knowledge* (vgl. BELKIN, ODDY & BROOKS 1982) des Benutzers ist entstanden ist, weil die gesuchte Information im elektronischen Buch nicht enthalten ist. Es gilt dann die Vermutung, dass sich aus dem Lesekontext ein Hinweis auf das Informationsbedürfnis ermitteln lässt. Dabei liegt die Annahme zugrunde, dass bei der Informationserschließung im Kontext der Interaktion mit elektronischen Texten die Auswertung dieser Ressourcen einen Beitrag

zur Verbesserung der Retrievalqualität leisten kann. Es bieten sich zwei grundsätzlich unterschiedliche Vorgehensweisen an:

1. Die automatische Generierung von Rechercheprofilen durch Auswertung des Lesekontexts mit anschließender Durchführung einer Recherche in externen Informationserschließungsdiensten sowie die Zuordnung der Ergebnisse zum aktuellen Lesekontext durch einen Agenten, der als Proxy zwischen Client (Browser) und Webserver die vom Benutzer verfolgten Inhalte analysiert. Eine solche Vorgehensweise entspricht der Funktionalität von Browsing-Agenten, wie sie JOACHIMS & MLADENIC 1998: 25 ff. diskutieren.[147] Der Vorteil dieser Methode liegt u. a. darin, dass sie sich – nach der Festlegung der Zuordnungspunkte im elektronischen Buch (Granularität des Lesekontexts) unabhängig von der tatsächlichen Interaktion des Benutzers (*offline*) durchführen ließe.[148]
2. Die Analyse des Lesekontexts und Generierung von Suchbegriffen während der Interaktion des Benutzers mit dem elektronischen Buch, um ihn bei der Anfrageformulierung zu unterstützen, ohne aber automatisch Anfragen durchzuführen. In diesem Fall findet die Auswertung des Lesekontexts erst bei Aktivierung der Rechercheschnittstelle durch den Benutzer statt und resultiert in einer Vorschlagsliste.

Im vorliegenden Fall wird das zweite Verfahren angewandt, da keine automatische Generierung von Suchprofilen, sondern die Unterstützung des Benutzers bei der Anfrageformulierung angestrebt wird; aufgrund der Annahme, dass die in der Interaktion mit dem elektronischen Buch auftretenden Informationsbedürfnisse

- sehr unterschiedlicher Natur sein können und
- bedingt durch den eingegrenzten inhaltlichen Kontext relativ spezifisch sind,

ist es sinnvoll, dem Benutzer die volle Kontrolle über den Anfrageprozess zu belassen. Die eigentliche Termextraktion aus einem konkretisierten informationellen Kontext heraus entspricht den Problemen, die sich bei der automatischen Indexierung stellen: Aus einer informationellen Einheit sind diejenigen Terme zu extrahieren, die diese Informationen (und damit das Informationsbedürfnis) besonders genau beschreiben. Dazu bieten sich eine Reihe von z. T. wechselseitig kombinierbaren Verfahren an, die im Kontext der IR-Forschung intensiv untersucht worden sind:

- Verwendung des Volltextes als Anfragestring (s. u.),
- Eliminierung nicht bedeutungstragender Terme (Stoppwörter),
- Grundformenreduktion und/oder Expansion aller Vollformen zur Berücksichtigung morphologischer Varianten im Anfrageprozess (vgl. FRAKES 1992B),
- Phrasenerkennung und Ermitteln von Mehrwortbegriffen (vgl. ZHAI 1997, STRZALKOWSKI 1999) und
- Übersetzen von Termen (multilinguales Retrieval, vgl. FRANZ, MCCARLEY & ROUKOS 1999).

---

[147] Vgl. LIEBERMAN 1998, der einen Überblick zu verschiedenen Informationsagenten für das World Wide Web gibt; das Grundprinzip ist jeweils die automatische Analyse der Interaktion des Benutzers mit verschiedenen Webseiten, deren Inhalte und Verknüpfungen unter Anwendung der bekannten Indexierungsverfahren (TF×IDF-Maß, s. o.) zum Aufbau eines individuellen Interessenprofils dienen, über das relevante Websites erschlossen werden können.

[148] Dies würde bedeuten, dass ein Agent sukzessive für alle Inhaltseinheiten des elektronischen Buchs eine Anfrageformulierung generiert und zu ihr Ressourcen z. B. im World Wide Web über eine Suchmaschine ermittelt.

Für die vorliegende Lösung wird auf solche, über die einfache statistische Indexierung hinausgehende linguistische Verfahren verzichtet; an ihre Stelle tritt die Auswertung des Struktur- und Inhaltsmarkup der elektronischen Ressourcen.

Der aktuelle Lesekontext lässt sich aus der tatsächlich am Bildschirm gezeigten Informationsmenge und der ihr zugeordneten informationellen Einheit (als logische Struktur- bzw. Inhaltseinheit) bestimmen. Drei Fälle sind denkbar:

a) Präsentationseinheit und logische/inhaltliche Strukturierung decken sich, d. h. die logische Einheit wird vollständig angezeigt.
b) Die logische/inhaltliche Einheit umfasst mehr als eine Präsentationseinheit, d. h, um eine logische Einheit rezipieren zu können, ist eine generische Navigationshandlung notwendig (Blättern, *scrolling*).
c) Der aktuelle Präsentationskontext umfasst mehr als eine logische/inhaltliche Einheit.

Im letzten Fall ist zu bestimmen, welcher logischen Einheit das Informationsbedürfnis zuzuordnen ist. Dies kann durch ein Auslösen der Informationserschließung unter direkter Bezugnahme durch den Leser erfolgen (z. B. Anbieten der Informationserschließung über ein per Mausklick aktivierbares Kontextmenü; die Position der Interaktion bestimmt die Zuordnung). Da die logische Strukturierung hierarchisch erfolgt, ist festzulegen, auf welcher Hierarchiestufe die Zuordnung zwischen Informationserschließung, Lesekontext und logischer Struktur erfolgen soll, d. h. wo die Grenzen der automatischen Auswertung des Kontexts liegen sollen. Dies lässt sich auch benutzergesteuert lösen, wenn der Benutzer die Möglichkeit hat, eine Ausweitung des Lesekontexts anzufordern. Die Auswertung des Lesekontexts beinhaltet folgende Schritte:

1. Bestimmung des aktuellen Lesekontexts aus dem aktuellen Darstellungskontext (aktuell präsentierter Ausschnitt innerhalb eines durch inhaltsorientiertes Markup ausgezeichneten Bestandteils des Buchs), im Buchserver als aktuelle Position verfügbar.
2. Bestimmung der logischen Hierarchieposition (durch den Buchserver, der über Methoden des DOM-API den Dokumentbaum nach oben durchwandern kann).
3. Die Extraktion potentieller Suchbegriffe aus Struktur- und Inhaltsmarkup (Überschriften, markierte Fachbegriffe) sowie Metadaten (Beschreibungen multimedialer Komponenten).[149]

Der Ansatz setzt voraus, dass bei der Aufbereitung der Buchinhalte nicht nur zentrale Begriffe explizit gekennzeichnet werden, sondern auch nicht textuell zu erschließende Komponenten durch Metadaten adäquat beschrieben sind bzw. textuelle Attribute wie Abbildungs- und Komponentenbeschriftungen ausgewertet werden können. Die naheliegende Alternative, grundsätzlich den Volltext des elektronischen Buchs zu indexieren und für die Recherche zu verwenden und ggf. den Benutzer Begriffe durch den +-Operator als zwingend erforderlich kennzeichnen zu lassen, ist dagegen nicht vielversprechend: Stichprobenartig wurde untersucht, ob sich Suchergebnisse von Suchmaschinen durch „Volltextanfragen“ verbessern lassen; dazu wurde aus Fachtexten jeweils ein inhaltlich zusammengehörender Textabschnitt gewählt, darin 3 zentrale Begriffe als „unbedingt erforderlich“ (+-Operator der Suchmaschinensprachen) gekennzeichnet und nach Eliminierung von Stoppworten der Volltext für die Recherche verwendet. Die Ergebnisse von 10 Anfragen aus zwei Fachgebieten erbrachte keine klare Veränderung des Suchergebnisses im Vergleich mit Anfragen, die lediglich die besonders gekennzeichneten Begriffe enthielten.

[149] Zur Verwendung von Struktur- und Inhaltsmarkup für die Retrievaloptimierung vgl. MYAENG et al. 1999, SCHRÖDER 1998 und WOMSER-HACKER & ZETTEL 1998.

Aus der Auswertung des Lesekontexts entsteht eine Vorschlagsliste, die vom Benutzer für den Anfrageaufbau bzw. die Anfrageerweiterung genutzt werden kann (s. u.). Der Ansatz ist vergleichbar mit dem in *LiveDoc* realisierten Konzept, wo ebenfalls die dynamische Integration von Wissensressourcen zur Informationserschließung modelliert wird (vgl. MILLER & BONURA 1998).

14.2.2.5.3 Integration in die Benutzerschnittstelle

Der zweite Aspekt der Informationserschließung externer Ressourcen betrifft die Auswahl und Integration solcher Dienste in das elektronische Buch. Es stehen unterschiedliche Typen von Diensten zur Verfügung:

- Frei über das World Wide Web verfügbare Informationserschließungsdienste wie Suchmaschinen und Metasuchmaschinen und
- proprietäre Dienste wie Online-Datenbanken, die nur bei Vorliegen einer Benutzerberechtigung integriert werden können.

Als generischer Dienst kommt nur die erste Kategorie in Frage, da bei Einschränkungen des Zugangs erst der individuelle Benutzer bzw. eine Benutzergruppe den Dienst für ein elektronisches Buch aktivieren kann. Für die Integration in das elektronische Buch ist diese Unterscheidung aber unerheblich. Zu den Voraussetzungen für die Integration eines Informationserschließungsdienstes gehört weiter, dass

- der Dienst entweder generisch vom Buchserver zur Verfügung gestellt wird bzw. der Benutzer über die Diensteverwaltung des elektronischen Buchs einen von ihm gewünschten Informationserschließungsdienst spezifiziert und
- die Funktionalität für die Übertragung der Suchbegriffe an den Informationserschließungsdienst zur Verfügung steht, d. h. eine entsprechende Beschreibung des Dienstes (z. B. Typ, Anzahl und Werte der Attribute von Query-URLs) muss spezifiziert sein (vgl. oben Kap. 4.4.3.2 und 12.5.1).

Nachfolgend soll die Integration von Informationserschließungsdiensten am Beispiel von Suchmaschinen gezeigt werden. Dem Benutzer wird die aus der Auswertung des Lesekontexts generierte Vorschlagsliste angezeigt, deren Begriffe er zur Anfrageformulierung in das Suchformular des integrierten Dienstes eintragen kann. Die weitere Interaktion mit dem Informationserschließungsdienst ist dann von der Benutzerschnittstelle des Dienstes determiniert, die Einbettung in das elektronische Buch dient lediglich der Kapselung, um die Aktivierung von Nachfolgediensten (insbesondere Speicherung) zu ermöglichen. Abbildung 89 zeigt die Benutzerschnittstelle für die externe Suche im World Wide Web. Sie hat folgenden Aufbau:

- Die aus dem aktuellen Lesekontext extrahierten Begriffe finden sich in einer Tabelle auf der linken Seite des Fensters.
- In der rechten Seite befindet sich oben das Suchformular für den externen Suchdienst und unten das Ausgabefenster für die Ergebnisdarstellung
- Die Navigationsleiste (unten) erlaubt neben dem Schließen des Fensters den Zugriff auf einen Folgedienst, die Speicherung der Suchergebnisse.

Im Beispiel wird *Altavista* als Suchdienst eingebunden. Der Benutzer kann die Suchbegriffe direkt durch Mausklick in das Suchformular übertragen. Als Default-Anfragelogik kommt der +-Operator zum Einsatz, d. h. alle Suchbegriffe müssen in den Ergebnisdokumenten enthalten sein.

Abbildung 89: Benutzerschnittstelle für die externe Suche im World Wide Web

14.2.2.5.4 Unterstützungsdienste für die Anfrageformulierung

Die Auswertung des Lesekontexts ist ein Unterstützungsverfahren, das das Vorliegen elektronischer Ressourcen ausnützt, um dem Benutzer Vorschläge für die Anfrageformulierung zu erstellen. Für die Unterstützung des Anfrageprozesses kommen aber auch „sekundäre“ Dienste in Betracht, die dem Benutzer helfen, auf der Basis seiner Selektion von Suchbegriffen die Anfrage weiter zu präzisieren. Ein solcher Dienst ist in dem Sinn sekundär, indem er nicht unmittelbar als Dienst im Kontext des elektronischen Buchs zur Verfügung steht, sondern erst im Kontext der Informationserschließung (also nach der Ermittlung von Begriffen aus dem Lesekontext) eingesetzt wird. Konkret geht es um die Frage, wie dem Benutzer für die vom ihm aus der Vorschlagsliste ausgewählten oder unabhängig davon formulierten Suchbegriffe zusätzlich Information zur Verfügung gestellt werden kann. Es handelt sich um die aus der IR-Forschung bekannte Fragestellung der Anfrageerweiterung (*query expansion*). Diese wird in der Regel mit Blick auf Verfahren betrachtet, die eine Anfrageerweiterung automatisch vornehmen, z. B. durch

- Expansion über Thesaurusrelationen (vgl. KEKÄLÄINEN & JÄRVELIN 1998, MANDALA et al. 1999) oder
- *relevance feedback*-Techniken für die Anfrageerweiterung (und MITRA, SINGHAL & Buckley 1998).

Generell dienen solche Strategien der Verfeinerung von Suchanfragen (nur wenige vom Benutzer angegebene Suchbegriffe, wie bei Recherchen im World Wide Web üblich) bzw.

der Ersetzung zu spezifischer oder zu allgemeiner Begriffe durch Unter- bzw. Oberbegriffe, die aus einem Thesaurus übernommen werden können.

Für den Prototyp eines dynamischen elektronischen Buchs wurde für die interaktive Anfrageerweiterung auf die als elektronischer Dienst verfügbaren Ressourcen des Leipziger Wortschatz-Projekts zurückgegriffen (vgl. LÄUTER & QUASTHOFF 1999, QUASTHOFF & WOLFF 1999, QUASTHOFF & WOLFF 2000 und *http://wortschatz.uni-leipzig.de*). Es handelt es sich um eine Infrastruktur für große monolinguale Sprachdatencorpora (Anfang 2000 ca. 300 Millionen laufende Wortformen, ca. 13 Millionen Beispielsätze, ca. 6 Millionen Wortformen), innerhalb derer zu Begriffen

- signifikante Kollokationen auf Satzebene bzw. Nachbarschaftskollokationen (linke und rechte Nachbarn eines Wortes),
- Sachgebietsangaben bzw. Definitionen und
- visuelle Darstellungen der zu einem Begriff verfügbaren Kollokationen

online abgerufen werden können. Die nachfolgende Abbildung zeigt einen Eintrag aus dieser Online-Ressource zum Begriff *Faraday* einschließlich des Graphen starker Kollokationen:

Große Anfrage
Wort (Wort_nr: 389085): Faraday
Häufigkeitsklasse: 18 (Anzahl: 19)
Beschreibung: englischer Chemiker
englischer Physiker
Links zu anderen Wörtern:
Teilwort von: **Michael Faraday**[12]

Beispiel(e):
Zumindest einen Raum frei von Hochfrequenz-Wellen gibt es: Das Auto ist ein "**Faraday**'scher Käfig", der Strahlung abschirmt. aber nur die von außen. (Quelle: *Frankfurter Rundschau 1993*)
Der in Wissenschaftsakademien in ganz Europa willkommene Gelehrte bastelt sich Jahrzehnte vor **Faraday** einen Blitzableiter und Wetterschutz: ("so baute ich mir einen Käfig von ver goldeten Stangen über mein Hauß, das müste vortrefflich aussehen"). (Quelle: *Frankfurter Rundschau 1992*)
In dem in der Fernsehsendung dargestellten Fall wurde ein entsprechendes Störsignal sogar in ein geschlossenes Auto hineingefunkt, einen **Faraday**'schen Käfig, in dem schon der Betrieb eines hochempfindlichen Autoradios einer äußeren Antenne bedarf. (Quelle: *Frankfurter Rundschau 1991*)

Signifikante Kollokationen für Faraday:
dielektrische (9), dass (8), Davy (7), Royal Society (6), Scheibendynamo (6), Käfig (5), Magnetfeld (5), Oersted (5), Royal (5), Strom (5), Widerstände (5), entdeckte (5), spezifische (5), aufweisen (4)

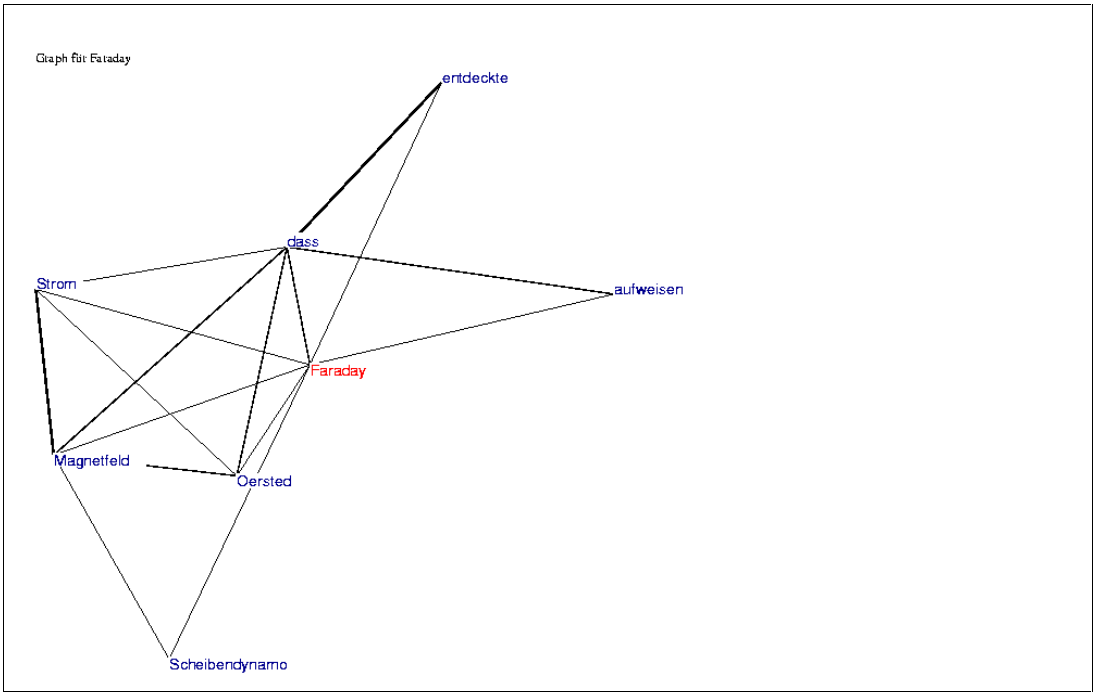

Abbildung 90: Beispieleintrag aus dem DEUTSCHEN WORTSCHATZ (FARADAY)

Der Benutzer kann entweder die Anfrage durch Übertragen eines weiteren Begriffs sofort verfeinern oder vor diesem Schritt weitere Informationen ermitteln, indem er z. B. die Verknüpfung auf den Mehrwortbegriff *Michael Faraday* verfolgt (Abbildung 91) und erst dann einen weiteren Begriff aus dem Eintrag zu *Michael Faraday* in seine Anfrage überträgt:

Große Anfrage
Wort (Wort_nr: 6175205): Michael Faraday
Häufigkeitsklasse: 18 (Anzahl: 12)
Links zu anderen Wörtern:

- ist ein: Person[8978]
- Teilwörter: Michael[8138], Faraday[19]

Beispiel(e):
Bereits im Jahr 1839 demonstrierte der Walliser Richter William Groves, ein Freund von **Michael Faraday**, zum ersten Mal dieses Prinzip. (Quelle: *Sueddeutsche Zeitung 1995*)
Signifikante Kollokationen für Michael Faraday:
Induktion (6), Physiker (6), entdeckte (6), Stromflusses (5), Substanzmenge (5), umgesetzte (4)
Signifikante linke Nachbarn von Michael Faraday:
Physiker (6), Wissenschaftler (5)
Signifikante rechte Nachbarn von Michael Faraday:
entdeckt (5)

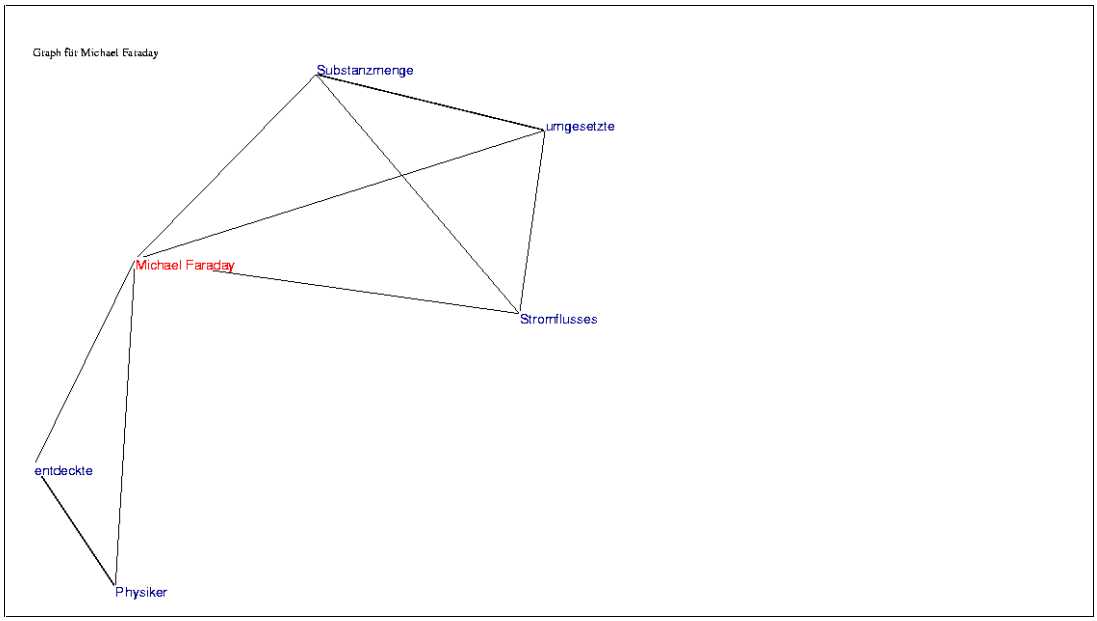

Abbildung 91: Beispieleintrag aus dem DEUTSCHEN WORTSCHATZ (MICHAEL FARADAY)

Dieser zusätzliche Dienst kann in die in Abbildung 90 gezeigte Benutzerschnittstelle für die Suche im WWW integriert werden: Statt einen Suchbegriff direkt in das Suchformular einzutragen, fordert der Benutzer zusätzliche Informationen zu diesem Begriff (Kollokationen, Synonyme, signifikante linke und rechte Nachbarn) aus der Wortschatz-Datenbank an. Die Ergebnisse dieser Recherche werden im Ergebnisbereich des Fensters angezeigt. Wiederum kann der Benutzer durch Mausklick auf die Begriffe zusätzliche Einträge in das Suchformular erzeugen. Abbildung 92 zeigt die Ausgabe verwandter Begriffe für den Suchbegriff *Masse*. Diese als Dienst für die Informationserschließung verfügbaren Daten können den Benutzer in zweierlei Hinsicht unterstützen:

- Unmittelbar als Liste weiterer Vorschläge für Suchbegriffe, die zu einem Ausgangsbegriff generiert werden kann, und
- mittelbar durch Darstellung zusätzlicher Information (Sachgebietsangabe, Definition), die dazu beitragen kann, das Informationsbedürfnis weiter zu präzisieren (um etwa im obigen Beispiel nach dem Faradayschen *Käfig* zu recherchieren), sollte dieser Begriff nicht schon in der aus dem Lesekontext erzeugten Vorschlagsliste enthalten sein.

Es sei darauf hingewiesen, dass dieses Verfahren dort an seine Grenzen stößt, wo

- spezifische Fachbegriffe nicht oder nicht in signifikanter Anzahl im Corpus enthalten und nur wenige signifikanten Kookkurrenzen errechnet werden können oder
- deklaratives Wissen wie Sachgebietsangaben, Thesaurusklassen oder Definitionen nicht in der Wortschatzdatenbank enthalten sind.

14.2.2.5.5 Weiterverarbeitung der ermittelten Informationseinheiten

Durch die Kapselung der Informationserschließungsdienste kann der Benutzer Nachfolgedienste wie die Speicherung als Annotation nutzen, um Ergebnislisten oder die recherchierten Ressourcen abzuspeichern; Voraussetzung ist auch hier, dass ein Speicherungsdienst *als Nachfolgedienst der Informationserschließung* spezifiziert wurde (auf der Ebene generischer Dienstefunktionalität).

Abbildung 92: Benutzerschnittstelle für die Anfrageunterstützung durch ähnliche Begriffe

14.2.2.6 Fazit

Die voranstehende Diskussion der Integration externer Informationserschließungsdienste hat zwei Aspekte des Konzepts der Dienste-Integration in dynamischen elektronischen Büchern verdeutlicht:

- Auf der Ebene der Dienstefunktion die Informationserschließung als einen Aspekt, der dem Benutzer die dynamische Erweiterung des zum größten Teil statischen Inhalts des elektronischen Buchs erlaubt und
- auf der Ebene der Dienste-Integration das Zusammenspiel unterschiedlicher Dienste im Sinne einer Prozesskette, bei der sekundäre Dienste den primären Dienst der Informationserschließung unterstützen.

Die Einbindung der betrachteten Informationserschließungsdienste stellt ein generisches Verfahren dar, das durch Werkzeuge, die über den engeren Kontext des Information Retrieval als Dokumentennachweis hinausgehen (Recherche in Faktenbasen; Anforderung einer e-commerce-Dienstleistung), ergänzt werden könnte.

14.2.3 Informationsdienste

Die Einbettung externer Informationsdienste soll hinsichtlich der Dienstefunktion Beispiele für die dynamische Ermittlung spezifischer Informationen zu Inhalten des Buchs geben und unterscheidet sich von der Informationserschließung dadurch, dass nicht aus einer heterogenen Dokumenten- bzw. Informationskollektion durch eine Retrievalfunkti-

on geeignete Informationseinheiten selektiert werden, sondern eine genau spezifizierte Informationsdienstleistung erbracht wird.

Technisch sind solche Dienste ein weiteres Beispiel für die dynamische Integration eines externen Dienstes durch Zugriff auf über das World Wide Web erreichbare Anwendungen. Als Beispiele für den Prototyp des elektronischen Buchs dienen

- die bereits im Kontext der Informationserschließung diskutierten Dienste des Leipziger Wortschatz-Projekts (insb. Kollokationsermittlung),
- die Einbindung eines Übersetzungsdienstes (hier: *AltaVista Babelfish*).

Technische Grundlage ist in den beiden letzten Fällen eine generische HTTP-Dienstekapsel, der als Argumente Adresse und Parameter des Dienstes übergeben werden und deren Ergebnisse wie beim Annotationsdienst in ein Seitenschema eingebettet werden, das der Buchserver bereitstellt. Optional kann ein follow-up-Dienst wie die Speicherung spezifiziert sein.

Durch den Zugriff auf frei verfügbare Dienste im World Wide Web ist dieser Dienstetyp ein interessantes Beispiel der Zusammenführung verschiedener Ressourcen bzw. Informationsangebote; wie bei der Informationserschließung (Suchmaschinen) reduziert sich der Realisierungsaufwand auf die Spezifikation der Diensteschnittstelle; die technische Lösung ist so konzipiert, dass sie für die Berücksichtigung von Zugangsberechtigungen bzw. Abrechnungsverfahren für kostenpflichtige Dienste erweiterbar ist.

Betrachtet man einen in das elektronische Buch integrierten Informationsdienst isoliert, so bedeutet seine Verfügbarkeit für den Benutzer lediglich eine Vereinfachung der Interaktion, da er z. B. einen Übersetzungsdienst durch ein zusätzliches Browserfenster und die Eingabe bzw. Übertragung der notwendigen Parameter nutzen könnte. Sobald allerdings durch Nachfolgedienste z. B. die Annotation des Buchs durch die über den Informationsdienst gewonnene Information möglich ist, bekommt die Dienste-Integration eine andere Qualität. Dies gilt auch für den Fall, dass solche Dienste wie oben am Beispiel der Wortschatzdienste gezeigt, als sekundäre Dienste den Benutzer bei der Informationserschließung unterstützen.

Ist an einen Begriff im Text des Buchs mehr als ein Dienst angebunden, so handelt es sich um eine 1:n-Verknüpfung. Für die Darstellung der verfügbaren Dienste kann der Benutzer durch Mausklick auf den Begriff im Buch ein Fenster öffnen, das ihm die verfügbaren Dienste anzeigt. Aus dieser Dienstetabelle muss er in einem zweiten Schritt den Dienst aufrufen, der in einem Fenster gekapselt wird, um die Speicherung als Folgedienst bereitstellen zu können.

Die folgenden Abbildungen sollen dies verdeutlichen. In Abbildung 93 ist der Überschrift *Polarisation* (rechte Buchseite) eine Diensteliste zugeordnet, die der Benutzer aktiviert hat. Wählt der Benutzer die Abfrage in der Wortschatz-Datenbank aus, so wird der entsprechende Eintrag in einem separaten Fenster angezeigt (Abbildung 94).
Ein zweites Beispiel zeigt die Aktivierung eines Übersetzungsdiensts. Dabei hat der Benutzer die Diensteliste für die als Fachbegriff gekennzeichnete Nominalphrase *transversale elektromagnetische Welle* aktiviert (1. Textzeile der rechten Buchseite) und den Übersetzungsdienst aufgerufen. Im Ergebnisfenster wird die englische Übersetzung (*transversal electromagnetic wave*) angezeigt (Abbildung 95).

Abbildung 93: Aktivieren der Diensteliste zum Begriff POLARISATION

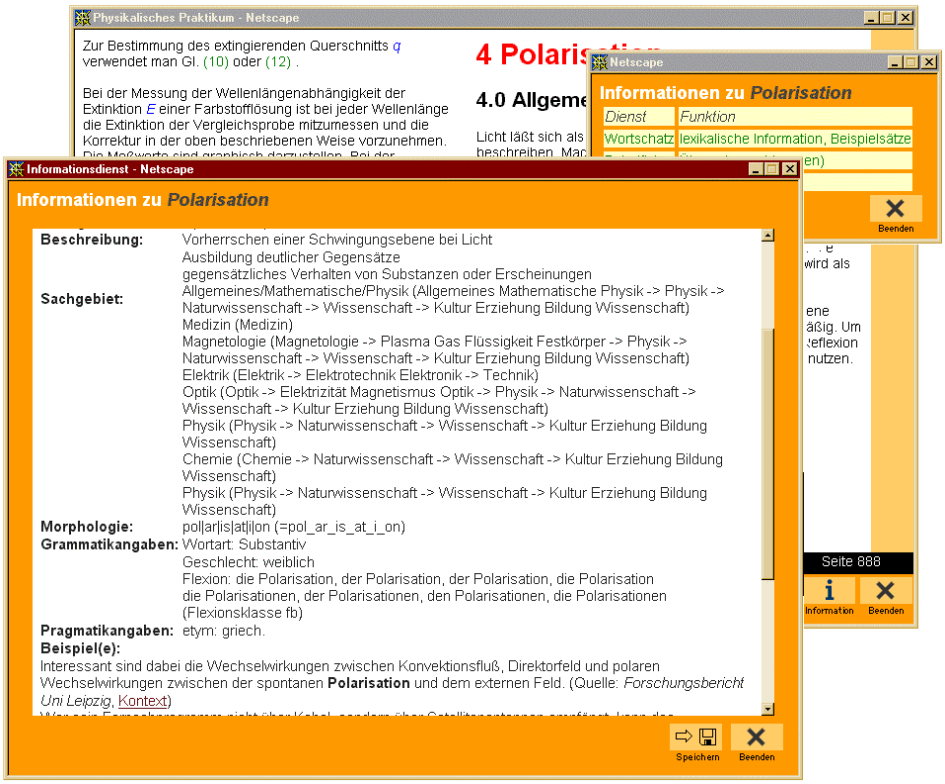

Abbildung 94: Eintrag zu POLARISATION aus der Wortschatz-Datenbank

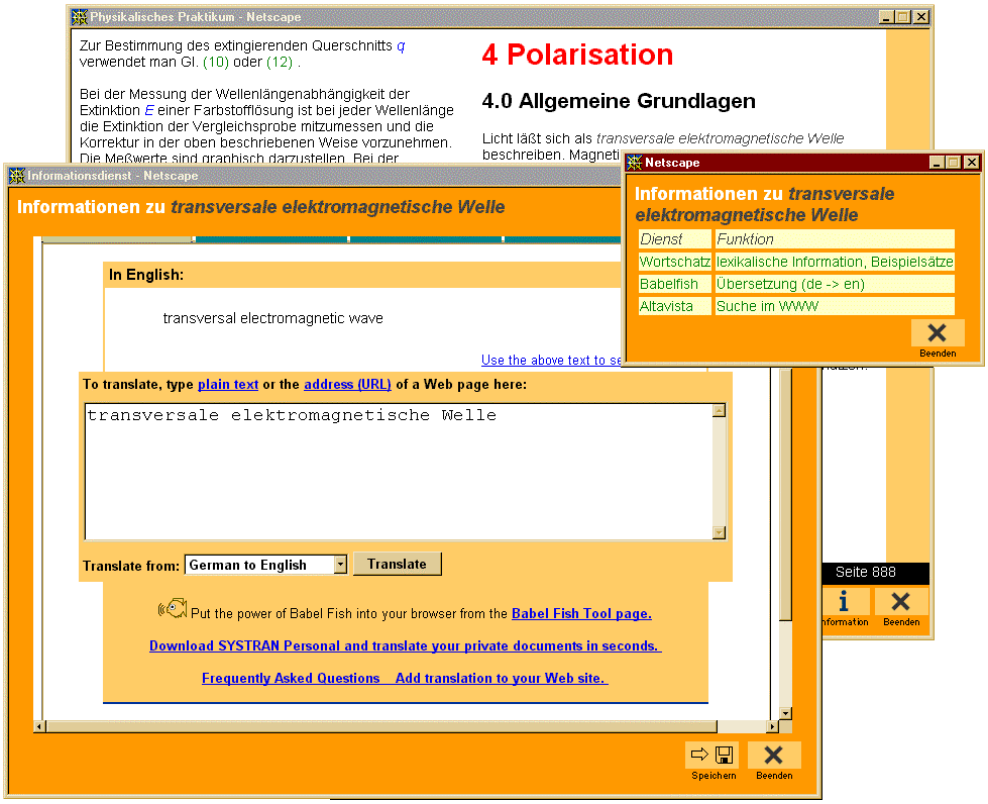

Abbildung 95: Übersetzungsdienst Altavista Babelfish

14.2.4 Weiterverarbeitungsdienste

Im Unterschied zu den bisher genannten Diensten greifen Weiterverarbeitungsdienste Inhalte des Buchs oder aus dem elektronischen Buch dynamisch generierte Daten auf und verarbeiten sie weiter. In technologischer Hinsicht muss man zwischen

1. netzbasierten Diensten und
2. lokalen Softwarekomponenten zu unterscheiden.

Die erste Variante entspricht der Vorstellung, dass es im Rahmen eines Paradigmenwechsels vom PC-zentrierten Einsatz monolithischer Softwarekomponenten zum verstärkten Einsatz verteilter Softwarekomponenten kommen wird. In der Regel ist dies bisher noch nicht das übliche Szenario für die Softwarenutzung. Die zweite Variante stellt auf das Vorhandensein geeigneter Weiterverarbeitungswerkzeuge auf der Seite des Benutzers ab; da die konkrete Nutzungsinfrastruktur aber vom einzelnen Benutzer abhängt, sind solche Dienste in diesem Fall nur individuell zu spezifizieren. Die deklarative Auszeichnung der in einem elektronischen Buch enthaltenen Daten macht eine direkte Anknüpfung von Weiterverarbeitungsdiensten möglich: Wie bei der Informationserschließung können die Inhalte des Buchs unmittelbar für die Weiterverarbeitung genutzt werden; sie sind in diesem Sinn dynamisch, da Inhalte nicht nur im elektronischen Buch *präsentiert* werden, sondern der Zugriff auf die Daten gegeben ist. Exemplarisch soll für den Fall des Referenzprojekts die Integration eines Formelmanipulationssystems diskutiert werden. Konzeptuell ist dabei folgende Vorgehensweise vorgesehen:

1. Der Benutzer ordnet über die Dienstekoordinationsschnittstelle einer Inhaltskomponente eine lokale Softwarekomponente zu.
2. Es existiert ein Austauschformat für die Daten der Inhaltskomponente, das von der lokalen Software verarbeitet werden kann (MathML).
3. Die Diensteaktivierung (Laden der Gleichung im Symbolmanipulationssystem) erfolgt durch eine adäquate Modifikation der Schnittstelle der Inhaltskomponente des elektronischen Buchs, z. B. durch eine Schaltfläche, die beim Laden der Komponente dynamisch generiert wird und über die der Benutzer die Weiterverarbeitung der Daten aktivieren kann.

Hinsichtlich des letzten Punkts ist zu unterscheiden,

- ob der Dienst direkt an die deklarativ ausgezeichneten Daten des Buchs oder an
- eine eingebettete Softwarekomponente angebunden wird.

Da sich in gängigen WWW-Browsern MathML-Markup nur durch zusätzliche Softwarekomponenten darstellen lässt, liegt der letztere Fall vor, d. h. zur Darstellung von Gleichungen werden die in MathML kodierten Daten an ein eingebettetes Applet im Browser übergeben. Dieses muss nach der Spezifikation des Dienstes durch den Benutzer vom Buchserver so parametrisiert werden, dass eine Schnittstelle zur lokalen Softwarekomponente aktiviert werden kann. Konkret liegt dabei folgender Ablauf vor:

1. Die im elektronischen Buch enthaltenen Formeln sind in MathML kodiert, z. B.
```
<GLEICHUNG id="23"> <MATH displaystyle='true'>  <MROW><MO STRETCHY='FALSE'>(</MO>
                    <MI>A</MI><MO>+</MO><MI>B</MI><MO STRETCHY='FALSE'>)</MO>
                    <MO>*</MO><MI>C</MI></MROW></MATH>
</GLEICHUNG>
```
2. Im Rahmen der Transformation von XML nach HTML wandelt der XSL/T-Prozessor des Buchservers die MathML-Marken so um, dass sie durch ein WebEQ-Applet angezeigt werden können. Hat der Benutzer einen Weiterverarbeitungsdienst für die Inhaltskomponente Gleichung angegeben (Dienstekoordinationsliste), so wird das WebEQ-Applet mit entsprechenden Parametern versehen:
```
<APPLET CODEBASE="classes" CODE="webeq.FormelApplet" WIDTH=400 HEIGHT=200>
  <PARAM NAME="Dienst_Name" VALUE="MuPad">
  <PARAM NAME="Dienst_Funktion" VALUE="Symbolmanipulation">
  <PARAM NAME="Dienst_Typ" VALUE="lokal">
  <PARAM NAME="Dienst_Protokoll" VALUE="JAF">
  <PARAM NAME="Dienstparameter_Name" VALUE="eq">
  <PARAM NAME="Dienstparameter_Typ" VALUE="text/mathml">
  <PARAM NAME="parser" VALUE="mathml">
  <PARAM NAME=eq VALUE="
  <math displaystyle='true'><MROW><MO STRETCHY='FALSE'>(</MO>
                            <MI>A</MI><MO>+</MO><MI>B</MI>
                            <MO STRETCHY='FALSE'>)</MO><MO>*</MO><MI>C</MI>
                            </MROW></MATH>
  ">
</APPLET>
```
4. Der Benutzer kann schließlich die Inhalte der Gleichung unmittelbar in das von ihm angegeben Symbolmanipulationssystem übertragen.

Aufgrund bestehender softwaretechnischer Einschränkungen lässt sich dieses Verfahren nicht unmittelbar wie beschrieben umsetzen: Java Applets verfügen nur über eingeschränkte Ausführungsrechte (Zugriff auf das lokale Dateisystem, Programmaufruf etc.),

wenn sie vom Benutzer über ein Netzwerk geladen und in einem Webbrowser ausgeführt werden. Dies ist nur zu umgehen, wenn entweder das Applet als signierter Code mit einer digitalen Unterschrift seines Entwicklers versehen ist und der Benutzer dem Applet erforderliche Rechte (z. B. Schreibrechte) explizit zugesteht oder der Benutzer die Grundeinstellungen seines Webbrowsers so modifiziert, dass die Anforderung zusätzlicher Rechte ebenfalls möglich wird.[150]

Die Verarbeitung von MathML-Markup wird von Symbolmanipulationssystemen bisher nicht unterstützt: Zwar sind die führenden Anbieter von Computeralgebrasystemen, *Wolfram Research* (*Mathematica*) und *Waterloo Maple* (*Maple*) an der Entwicklung von MathML beteiligt, eine Integration entsprechender Schnittstellen zu MathML in diese Systeme steht aber bislang noch aus, vgl. *http://www.wolfram.com/news/archive/mathml.html* und *http://www.maplesoft.com/corporate/press/press_010.html* sowie die Website der Arbeitsgruppe MathML des W3C, *http://www.w3.org/Math/*.

Abbildung 96 zeigt aber die prinzipielle Funktionsweise: Eine in MathML kodierte Gleichung wird im Webbrowser durch ein Applet dargestellt, das das WebEQ Java-API für die Darstellung nutzt. Bei Initialisierung des Applets werden die ihm zugeordneten Dienste ermittelt. Klickt der Benutzer auf die Gleichung, so wird der MathML-Code in einem separaten Fenster zur Weiterverarbeitung angezeigt (stellvertretend für ein MathML-fähiges Symbolmanipulationssystem). Damit stehen deklarativ aufbereitete Inhalt des elektronischen Buchs für die Weiterverarbeitung zur Verfügung.

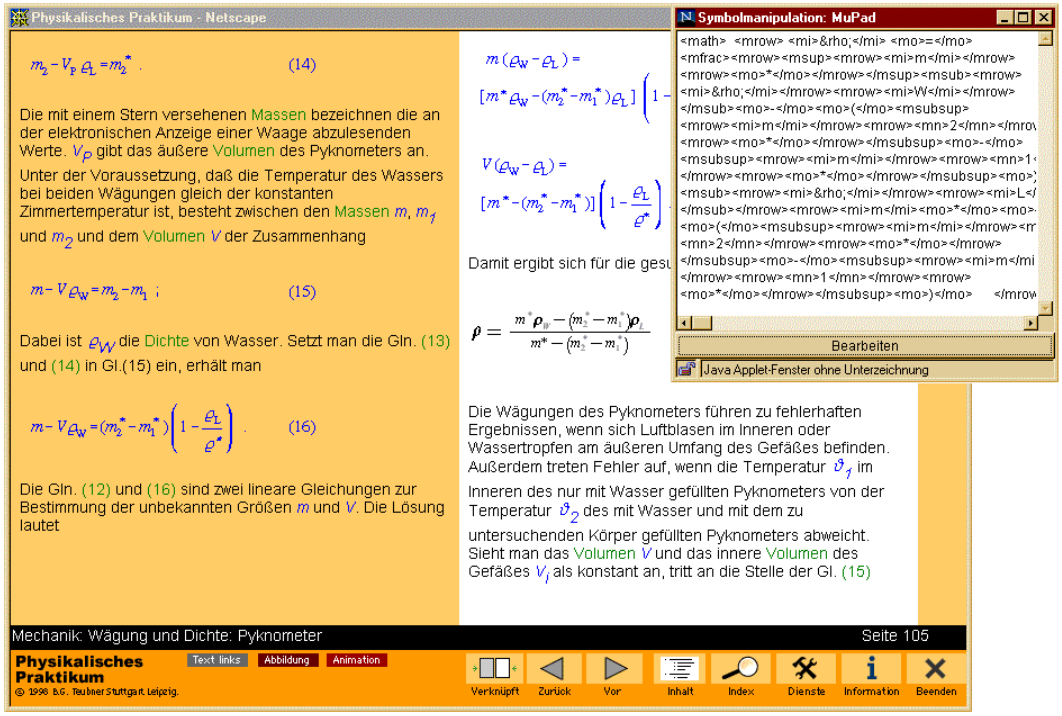

Abbildung 96: Extraktion des deklarativen Formelmarkup zur Weiterverarbeitung

[150] Zu den zugrunde liegenden Konzepten des Sicherheitssystems von Java vgl. GONG 1999; KNUDSEN 1998: 163 ff. beschreibt detailliert die Alternativen beim Einsatz signierter Applets in verschiedenen Webbrowsern.

14.2.5 Nutzungskontext und Kommunikationsdienste

Als letzte Dienstekategorie sollen Dienste mit Bezug zu einem Gruppenkontext betrachtet werden. Sie liegt quer zu den bisher diskutierten Diensten, da sie primär nicht durch die Dienste*funktion*, sondern den *Bezug* eines Dienstes hinsichtlich der an einem Kommunikationsvorgang Beteiligten konstituiert wird. Die in dieser Kategorie erfassten Dienste können identisch mit bereits eingeführten Dienstetypen sein, wenn z. B. ein Speicherungsdienst für Annotationen verwendet wird, der nicht nur für den individuellen Benutzer einsehbar ist, sondern einer Benutzergruppe zur Verfügung steht (eine Anwendung, die einem wesentlichen Ziel des WebDAV-Protokolls entspricht, das für die Realisierung von Speicherungs- und Annotationsdiensten verwendet wird).

Ein Gruppenkontext unterscheidet sich aber nicht nur durch die Mehrzahl der Kommunikationsteilnehmer, sondern auch dadurch, dass er eine weitere sinnvolle Anwendung die Einführung *zusätzlicher Ressourcen* zum elektronischen Buch durch einen Gruppenverwalter (s. u.) ist: Ein Anwendungsbeispiel ist die Einführung zusätzlicher Simulationen und Animationen, die nur für eine bestimmte Benutzergruppe zur Verfügung gestellt wird (z. B. die Teilnehmer einer Lehrveranstaltung, für die das elektronische Buch als Textgrundlage dient). Wie für die anderen Dienstetypen ist auch für Dienste, die im Rahmen eines gruppenbezogenen Kontextes eingeführt werden, sowohl

- die Anbindung an *primäre Inhalte* des elektronischen Buchs (strukturelles und inhaltsbezogenes Markup) als auch
- die Anbindung an *Komponenten* möglich (wie bei den Beispielen Speicherung und Symbolmanipulation angedeutet).

Damit werden aus der Sicht des individuellen Benutzers zwei unterschiedliche Strategien der Informationsanreicherung möglich:

- Bei der Annotation im individuellen oder gruppenbezogenen Kontext erstellt der Benutzer selbst zusätzliches Material, mit dem er das elektronische Buch erweitern will.
- Im Fall der Ergänzung durch den Gruppenverwalter bekommt er zusätzliches Material als Teil des elektronischen Buchs präsentiert, ohne dass dies unbedingt im Interface besonders hervorgehoben werden müsste.

Da dem Gruppenverwalter damit Autorenschaft zukommt, ist es sinnvoll, die nahtlose Integration zusätzlichen Materials auf den Gruppenverwalter einzuschränken.[151]

Bei der Typisierung der für einen Gruppenkontext sinnvollen Dienste ist es offensichtlich, dass sich vor allem solche Dienste als Ergänzung anbieten, die in einem inhaltlichen Bezug zum Einsatz des elektronischen Buches für die ausgewählte Benutzergruppe stehen, mithin Kommunikationsdienste wie E-Mail, Chat-Server oder Diskussionsforen. Als Beispiele für die Verwendung eines gruppenbezogenen Nutzungskontexts sollen nachfolgend

- die Integration zusätzlicher Komponenten durch den Kontextverwalter,
- die Verwendung von E-Mail als Kommunikationsdienst und
- der Einsatz eines Speicherungsdienstes

diskutiert werden. Die Beispiele sollen lediglich das Grundprinzip illustrieren; grundsätzlich könnten beliebige Dienste über den Gruppenkontext eingebunden werden. In

[151] Daraus entstehen allerdings urheberrechtliche Konsequenzen, wenn dynamisch Materialien einem Publikationsbestand zugeordnet werden, was konzeptuell mit der *Transklusion* im Sinne Ted NELSONS vergleichbar ist (vgl. oben Kap. 2.2.1.7).

weitergehender Betrachtung ist es kaum realistisch, dass speziell für einen Gruppenkontext komplexe Dienste entwickelt und bereitgestellt werden. Insofern ist hier an die Integration von CSCW-Systemen als an das elektronische Buch gekoppelte Dienste zu denken, z. B. die Integration einer Installation des CSCW-Tools BSCW als dem elektronischen Buch zugeordnete Software für die Gruppenarbeit, vgl. APPELT & MAMBREY 1999.

14.2.5.1 Einbindung von Komponenten über den Gruppenkontext

Die Einbindung zusätzlicher Komponenten ist durch den Gedanken motiviert, dass zu dem Inhaltsbestand eines elektronischen Buchs wünschenswerte Ergänzungen existieren, die gezielt in die Präsentation des Buchs eingebunden und einem Benutzerkreis zugänglich gemacht werden sollen. Dabei kommen grundsätzlich alle hier diskutierten Typen von Komponenten in Frage, also z. B. zusätzliche Simulationen oder textuelle Exkurse zu einem bestimmten Versuch. Die technische Realisierung der Ergänzung von Materialien im Gruppenkontext setzt folgendes voraus:

- die *Kontextverwaltung* des Buchservers (vgl. oben Kap. 12.5.3),
- einen *Speicherungsort* für die zusätzlichen Materialien, der ohnehin Vorbedingung für den Gruppenkontext ist,
- das *Überprüfen* des Gruppenkontextes durch den Buchserver während der Interaktion mit dem elektronischen Buch und
- eine geeignete *Repräsentation* des zusätzlichen Materials im elektronischen Buch.

Die Zuordnung erfolgt dabei wie bei den primären Komponenten des elektronischen Buchs über strukturelle und inhaltsbezogene Auszeichnungselemente. Während der Interaktion des Benutzers mit dem elektronischen Buch muss der Buchserver jeweils prüfen, ob sich in dem dem Benutzer zugeordneten Gruppenkontext zur aktuell dargestellten Inhaltseinheit zusätzlich Komponenten befinden. Die Zuordnung zwischen Buchinhalt und Komponente im Gruppenkontext ergibt sich dabei über das Adressierungsformat der WebDAV-Kollektion des Gruppenkontexts, die die zusätzliche Ressource enthält.

14.2.5.2 E-Mail als Beispiel eines Kommunikationsdienstes

E-Mail ist ein einfaches Beispiel für einen internetbasierten Kommunikationsdienst. Im Rahmen eines Gruppenkontexts kann er als Möglichkeit dienen, Informationen vom Benutzer an den Gruppenverwalter oder eine andere Person mit besonderer Funktion zu senden (z. B. ein Tutor, der einen bestimmten Versuch betreut).

Da Webbrowser in der Regel softwaretechnisch unmittelbar mit E-Mail-Editoren verbunden sind (z. B. Netscape Messenger, Microsoft Outlook), ist es hier nicht nötig, einen gesonderten E-Mail-Dienst bereit zu stellen,. Dieser Kommunikationsdienst kann für einen Gruppenkontext durch Bereitstellung einer E-Mail-Adresse und ihrer Integration in das elektronische Buch realisiert werden.

Ist für einen Gruppenkontext die E-Mail-Adresse eines Tutors an die Inhaltsmarke <AUFGABE> gebunden, so muss die Diensteverwaltung bei der Generierung einer Buchseite, die eine Aufgabe enthält, eine entsprechende Verknüpfung zur Aktivierung des E-Mail-Dienstes erzeugen, also einen HTML-Anker mit Linkadresse im mailto-Format.

14.2.5.3 Speicherungsdienst

Als drittes Beispiel soll eine weitere Variante eines Speicherungsdienstes diskutiert werden. Grundsätzliche technische und konzeptuelle Fragen der Speicherung und Annotation

sind bereits besprochen worden; hier soll es darum gehen, wie ein zusätzlicher Speicherungsdienst im Rahmen eine Gruppenkontexts zum Einsatz kommen kann.

Die Speicherung von Daten als grundlegende Funktion ist in zahlreichen Diensten, so auch in dem voranstehenden Beispiel der Einbindung spezifischer E-Mail-Adressen enthalten. Insofern ist ein zusätzlicher dedizierter Speicherungsdienst für einen Gruppenkontext nur dann sinnvoll, wenn er sich nicht durch speziellere Dienstetypen realisieren lässt (wie E-Mail, Diskussionsforen, Chat-Server etc.). Ein Beispiel für die Speicherung, die diesem Kriterium gerecht wird, ist die Datenspeicherung aus eingebetteten Komponenten heraus. Im Beispiel des Referenzprojekts ist dies die Speicherung von Messdaten, die durch Simulationen erzeugt werden. Die Speicherung solcher Daten in einem Gruppenkontext ist dann wünschenswert, wenn andere in die Lage versetzt werden sollen, auf die Daten zuzugreifen und sie weiterzuverarbeiten.

14.2.5.4 Diskussion

Die oben vorgestellten Beispiele zeigen, wie die Definition von Gruppenkontexten für dynamische elektronische Bücher genutzt werden kann. Die damit erreichbare zusätzliche Funktionalität für den Benutzer ließe sich auch auf andere Weise bereitstellen,

- indem statt der *Einbindung* in das elektronische Buch Ressourcen und Dienste im World Wide Web bereitgestellt werden und von ihnen auf Abschnitte des elektronischen Buchs verwiesen wird oder
- indem durch direkten Einsatz von CSCW-Systemen ohne Einbindung in ein elektronisches Buch Kommunikationsdienste für eine Gruppe zur Verfügung gestellt werden.

Im Unterschied zu diesen Ansätzen steht hier die Interaktion mit dem elektronischen Buch und die einheitliche Präsentation von Buchinhalten, Komponenten und Diensten im Mittelpunkt.

14.3 Fazit

Die Einführung interaktiver Komponenten und die Anbindung von Diensten an das dynamische elektronische Buch stellen eine wesentliche Erweiterung der Funktionalität elektronischer Publikationen dar. Mit ihnen wird der Benutzer nicht nur in die Lage versetzt, über die reine Textrezeption hinaus mit den Inhalten eines elektronischen Buchs zu interagieren, sondern es werden ihm auch vielfältige Möglichkeiten eröffnet, das Buch für seine Zwecke durch Annotationen und die Erschließung externer Ressourcen zu adaptieren. Die Einführung des Konzeptes eines Gruppenkontexts erlaubt es zudem, elektronische Lehrbücher für den konkreten Einsatz im Rahmen einer Lehrveranstaltung oder allgemeiner für jede gruppenbezogene Nutzung eines Buchs anzupassen.

Die Spezifikation zusätzlicher Dienste und Komponenten setzt dabei Technologien (z. B. einen WebDAV-Server) und Kenntnisse (z. B. von XML und des hier gewählten Formats für die Kodierung und Verwaltung von Diensten) voraus. Es bleibt weiterführenden Entwicklungen überlassen, hierfür benutzerfreundliche Administrationsschnittstellen zu schaffen.

Im folgenden Kapitel wird abschließend erörtert, inwiefern sich das Konzept dynamischer elektronischer Bücher auf andere Anwendungsfelder bzw. Buchinhalte übertragen lässt.

15 Weiterführende Aspekte

Abschließend sollen zwei Aspekte diskutiert werden, die für die Nachhaltigkeit des Konzepts dynamischer elektronischer Bücher von Bedeutung sind. Dazu gehören

- die Frage nach der Generalisierbarkeit des Ansatzes über das am Einzelwerk orientierte Szenario des Referenzprojekts hinaus (Kap. 15.1) und
- das Verhältnis zwischen dynamischen elektronischen Büchern und digitalen Bibliotheken als Infrastruktur für die Nutzung elektronischer Publikationen (Kap. 15.2).

15.1 Generalisierbarkeit

In Kap. 4 wurde versucht, das Architekturmodell für dynamische elektronische Bücher weitgehend unter Abstraktion von einem konkreten Einzelfall zu entwickeln. Die Diskussion des Prototyps in Teil III dieser Arbeit erfolgte in Fortführung des Referenzprojekts an einem konkreten Einzelfall. Abschließend ist zu fragen, inwieweit sich das Modell auf andere elektronische Publikationen übertragen lässt. Dies soll anhand zweier Beispielszenarien aus den Bereichen Informatik (*Lehrbuch einer Programmiersprache*) und Sprachdidaktik erfolgen. Es ist zu untersuchen, ob sich für diese beiden Beispielszenarien analoge Einsatzbereiche für dynamische elektronische Bücher finden lassen. Dies betrifft die Bereiche

- deklarative Auszeichnung von Inhalten in Anwendung der in dieser Arbeit unterschiedenen Markupebenen und der damit verbunden Methode,
- die Entwicklung interaktiver Komponenten sowie die Integration von Diensten sowie
- das in dieser Arbeit entwickelte Architekturmodell einschließlich der gestalterischen Lösung für ein Buchbetrachtungssystem auf der Basis der Buchmetapher.

Die nachfolgenden Tabellen geben jeweils Beispiele für die Übertragung dieser Aspekte auf andere inhaltliche Anwendungsfelder.

	Programmierlehrbuch	*Sprachlehrbuch*
Strukturmarkup	Lineare Progression der Inhaltseinheiten und hierarchischer Aufbau sind typische Merkmale von Lehrbüchern; die Elemente des Strukturmarkup lassen sich insofern übertragen.	
Inhaltsorientierte Auszeichnung	Quellcodebeispiele, Definitionen, eingebettete Komponenten (Beispielprogramme), Übungsaufgaben oder Syntaxregeln. Eine Übernahme zahlreiche Auszeichnungselemente aus den DocBook-DTDs erscheint möglich.	Beschreibung sprachlicher Phänomene durch XML-DTDs (vgl. MILDE 1999, WITT 1999); Markup unterschiedlicher Typen von Texten und Aufgabenstellungen.
Hypertext-Relationierung	Wie im Referenzprojekt können vielfältige Formen der automatischen Verknüpfungsgenerierung über explizite Verweisrelationen im Text erfolgen; auch das Konzept typisierter Verknüpfungen zu Komponenten (s. u.) ist übertragbar.	
Metadaten	Die Metadatenauszeichnung kann analog zum in dieser Arbeit diskutierten Modell erfolgen; Standards wie Dublin Core oder IEEE LOM sind unabhängig von den Inhalten einer elektronischen Publikation bzw. von Lernmaterialien definiert.	

Tabelle 64: Anwendungsbeispiele für die verschiedenen Markupebenen

Wie Tabelle 64 deutlich machen soll, lassen sich die Ebenen des Strukturmarkup, der Hypertext-Relationierung und der Metadatenbeschreibung unmittelbar übertragen und sind für elektronische Bücher unterschiedlichen Inhalts anwendbar, während der entscheidende Zusatzaufwand für die Aufbereitung eines elektronischen Lehrbuchs im Bereich des inhaltsorientierten Markup liegt: Hier sind diejenigen Inhaltselemente auszumachen, die spezifisch für den Einzelfall sind und an die sich für das elektronische Buch spezifische Komponenten und Dienste anknüpfen lassen. Beispiele für solche Komponenten und Dienste gibt die nachfolgende Tabelle.

	Programmierlehrbuch	*Sprachlehrbuch*
Eingebettete Komponenten	Ausführbare Programme, die Strukturen und Phänomene der Sprache bzw. Algorithmen visualisieren (vgl. GLOOR 1993, 1997).	Z. B. Visualisierungen und Animationen linguistischer Phänomene wie etwa bei HANDKE 1999, STORRER 1999 beschrieben.
Weiterverarbeitung	Z. B. Kompilieren und Ausführen von Codebeispielen des elektronischen Buchs, Weiterentwicklung von Programmen, die als Beispiel im elektronischen Buch enthalten sind.	Übernahme von Texten durch den Benutzer zur individuellen Weiterverarbeitung (z. B. Ausarbeitung im Rahmen einer Übungsaufgabe).
Informationsdienste	Zugriff auf Online-Dokumentation, Newsgroups, FAQ-Listen.	Zugriff auf online verfügbare linguistische Ressourcen (Lexika, Übersetzungsdienst).
Speicherung/Informationserschließung	Wie für das Referenzprojekt verdeutlicht, kann man die Speicherung zusätzlicher Information durch den Benutzer und die Informationserschließung aus externen Ressourcen als generische Dienste auffassen, die für Bücher unterschiedlichen Inhalts in gleicher Form eingebunden werden können. Differenzierungen ergeben sich dann, wenn z. B. gezielt themenorientierte Suchmaschinen und Recherchewerkzeuge eingebunden werden (z. B. zum Buchinhalt passende Online-Datenbank).	
Kommunikationsdienste/ Gruppenkontext	Auch gruppenorientierte Dienste, insbesondere soweit sie der Kommunikation bezüglich der Inhalte und Aufgaben eines elektronischen Buchs dienen, sind stärker an generischen Kommunikationsprozessen im Kontext von Lehrveranstaltungen als an den konkreten Inhalten eines Buchs orientiert und lassen sich insofern für dynamische elektronische (Lehr-)Bücher verallgemeinern.	

Tabelle 65: Beispiele für die Integration von Komponenten und Diensten

Das Architekturmodell für dynamische elektronische Bücher lässt sich grundsätzlich verallgemeinern, da es, wie bereits ausgeführt, ohne engen Bezug zum Referenzprojekt entwickelt wurde. Hinsichtlich der für den Prototyp eingesetzten Technologien lässt die Weiterentwicklung webbasierter Informationssysteme eine Reihe von Modifikationen sinnvoll erscheinen:

- Neuere Technologien für verteilte Anwendungen, wie z. B. die Java-basierten Middleware-Konzepte Jini und Java Spaces (vgl. ARNOLD et al. 1999, FREEMAN, HUPFER & ARNOLD 1999) bieten Alternativen für die Realisierung verteilter und kooperativer Dienste,
- die Datenspeicherungs- und Transformationsschritte, wie sie für den Prototyp geschildert wurden, können an Weiterentwicklungen XML-basierter Standards und Werkzeuge angepasst werden.

Hinsichtlich der Verwaltungs- und Steuerungskomponenten des Buchservers liegt die Ausgliederung in die Infrastruktur digitaler Bibliotheken nahe, vgl. dazu unten Kap. 15.2.

Schließlich stellt sich die Frage, ob das Buchbetrachtungssystem mit der Umsetzung einer modifizierten Buchmetapher auf andere elektronische Bücher übertragbar ist. Der

Nachteil des geringen Darstellungsbereichs pro Buchseite bei den im Referenzprojekt gewählten technischen Randbedingungen lässt sich durch Anpassung an höhere Bildschirmauflösungen zumindest relativieren. Die grundsätzliche Designentscheidung, Inhalte als Fokus- und Kontextseite darzustellen und die Kontextseite zusätzlich für die Darstellung den Inhalten zugeordneter Komponenten zu verwenden, erscheint dabei auf andere Inhalte übertragbar. Allerdings sind hier eingehende empirische Untersuchungen erforderlich, die klären könnten, inwieweit dieses Konzept den ergonomischen Anforderungen der Interaktion mit elektronischen Büchern gerecht wird.

Bejaht man die Generalisierbarkeit des Modells dynamischer elektronischer Bücher, stellt sich die Frage, wie sich ihre Modellierung und Entwicklung besser unterstützen lässt. Erfahrungen aus dem Referenzprojekt und darüber hinaus aus dem Projektverbund *Multimediabuch* machen deutlich, dass die Heterogenität der für elektronische Bücher mit multimedialen Inhalten erforderlichen Formate, Standards und Werkzeuge einerseits, der Mangel an integrierenden Entwicklungsmethoden andererseits bestimmende Faktoren für die Erstellung elektronischer Bücher sind. Es existieren bisher kaum integrierte Entwicklungsmethoden für Multimediasysteme; Hinweise auf Planung und Entwicklung von Multimediaanwendungen oder elektronischen Büchern sind in der Regel einzelfallbasiert (vgl. GARRAND 1997, bes. 23 ff., 103 ff.) oder beruhen auf offenen Kriterienkatalogen (vgl. oben Kap. 3.2 und SCHIFMAN, HEINRICH & HEINRICH 1999: 83 ff.).

Auf einer globalen Ebene versucht DEGEN 1996, die Interdisziplinarität der Multimediaentwicklung zu modellieren, indem er sie in die drei Bereiche *Technikdesign*, *Mediendesign* und *Interaktionsdesign* gliedert und deren Interdependenz veranschaulicht. Daneben lassen sich für die globale Planung einer elektronischen Publikation Prozessmodelle aus dem Software-Engineering übertragen, wie dies BULLINGER 1999: 36 vorschlägt: Danach sind für elektronische Publikationen („Medienengineering“) die traditionellen Schritte der Softwareentwicklung allgemein anwendbar: *Planung*, *Definition*, *Entwurf*, *Implementierung*, *Abnahme*/*Einführung* und *Wartung*. Im Einzelnen werden die traditionellen Spezifikationsverfahren des Software-Engineering wie Lasten- und Pflichtenheft durch Methoden ergänzt, die den besonderen Erfordernissen multimedialer Publikationen gerecht werden. Zu ihnen gehört der Entwurf eines Storyboards, das Navigationsmuster und die Aufteilung von Inhalten mit Hilfe einer diagrammorientierten visuellen Sprache festhält (vgl. BURGER 1996).

Hinsichtlich dynamischer elektronischer Bücher ist das in Kap. 4.2 entwickelte Verfahren zur Bestimmung der unterschiedlichen Ebenen zugeordneten Auszeichnungselemente ein Ansatz, der geeignet ist, den globalen Entwicklungsprozess zu unterstützen und gleichzeitig zu konkretisieren: Planungs- und Definitionsphase liefern als Ergebnis die für das elektronische Buch erforderlichen DTDs (primäre Buchinhalte und zugeordnete Komponenten). Für die Komponentenentwicklung kann auf bewährte Verfahren der objektorientierten Modellierung zurückgegriffen werden. Damit steht ein genuin texttechnologisches Verfahren der Inhaltsmodellierung im Zentrum der Entwicklung dynamischer elektronischer Bücher.

15.2 Integration elektronischer Publikationen in digitale Bibliotheken

In dieser Arbeit wurden elektronische Bücher als eigenständige Informationssysteme eingeführt und anhand des Referenzprojekts, also eines einzelnen elektronischen Buchs, diskutiert. Die Nutzung elektronischer Bücher erfolgt in der Praxis aber in der Regel vermittelt über die technische und organisatorische Infrastruktur einer digitalen Bibliothek. Es ist daher zu fragen, welche Teile des hier vorgestellten Architekturmodells auch

von einer digitalen Bibliothek übernommen werden könnten und wie sich nach dem hier diskutierten Modell entwickelte elektronische Bücher in digitale Bibliotheken eingliedern lassen. In digitalen Bibliotheken werden elektronische Publikationen gespeichert und Benutzern über Netzwerke zur Verfügung gestellt. Ihre Entstehung wurde konzeptuell bereits von Vannevar BUSH vorweggenommen, der mit der Konzeption von Memex wohl den ersten Entwurf einer digitalen Bibliothek vorlegte (BUSH 1945). Bereits Mitte der 60er Jahr fordert LICKLIDER 1965:

> [...] any concept of the library which begins with book shelves is sure to encounter trouble [...] we should be prepared to reject the schema of the physical library – the arrangement of shelves, card indexes, check-out desks, reading rooms, and so forth [...]. [zit. nach SALTON 1975: 12].

Es existieren zahlreiche unterschiedliche Interpretationen des Konzepts „digitale Bibliothek“, wie FOX et al. 1995: 24 f. deutlich machen:

> The phrase „digital library“ evokes a different impression in each reader. To some it simply suggests the computerization of traditional libraries. To others, who have studied library science, it calls for carrying out of the functions of libraries in a new way, encompassing new types of information resources; new approaches to acquisition (especially with more sharing and subscription services); new methods of storage and preservation; new approaches to classification and cataloging; new modes of interaction with and for patrons; more reliance on electronic systems and networks; and dramatic shifts in intellectual, organizational and economic practices.
> To many computer professionals, a digital library is simply a distributed text-based information system [...], a collection of distributed information services [...], a distributed space of interlinked information [...], or a networked multimedia information system.

Als wesentliche Definitionsmerkmale (vgl. BORGMANN 1999: 235f.) lassen sich unterscheiden:

- In *technischer* Hinsicht die persistente Speicherung von Wissen in unterschiedlichen Formaten und ggf. verteilt über Netzwerke und die damit verbundenen Problemstellungen (Interoperabilität, Skalierbarkeit, Portierbarkeit, Föderierung etc.) sowie
- hinsichtlich der *Aufgabe* einer digitalen Bibliothek die Bereitstellung von Inhalten für Benutzer (d. h. die *user community*) der digitalen Bibliothek.

Beispiele für digitale Bibliotheken finden sich als

- von Verlagen oder Institutionen angebotene kommerzielle Bibliotheken, in denen vor allem wissenschaftliche Zeitschriften bereitgehalten werden (z. B. *Link*, die digitale Bibliothek des Springer-Verlags (vgl. *http://link.springer.de*) und die digitalen Bibliotheken der ACM bzw. IEEE (*http://www.acm.org/dl* bzw. *http://www.computer.org/publications/dlib/*),
- nationale Infrastrukturprojekte wie die amerikanische *Digital Library Initiative* (DLI, vgl. FOX et al. 1995, BORGMANN 1999) oder MeDoc/InterDoc/GlobalInfo in Deutschland (vgl. ENDRES & FUHR 1998, BOLES et al. 1998A) und
- an universitäre Forschungsgruppen angebundene *digital library*-Projekte (vgl. z. B. PAEPCKE et al. 1999).

Die Merkmale digitaler Bibliotheken und die mit ihnen verbundenen Forschungsschwerpunkte sind ähnlich heterogen wie für elektronische Publikationen und elektronische Bücher (vgl. den Kriterienkatalog in Kap. 3.2 und FOX et al. 1995: 27), in der nachfolgenden Tabelle sind nur einige wesentliche Strukturmerkmale zusammengefasst:

Betreiber	Verlage, Wiss. Gesellschaften Bibliotheken Firmen (Informationsdienstleister) Institutionen
Art der Dokumente	Aufsätze Preprints, Technical Reports Elektronische Bücher Multimediainhalte
Dokumentenkodierung und -strukturierung, Formate	Imaging (Bitmaps, PDF) Deklaratives Markup (HTML, SGML, XML) Proprietäre Formate
Inhaltserschließung	Intellektuelle Beschlagwortung Automatische Indexierung (Volltext) Metadaten (Marc, DigiMarc, Dublin Core etc.)
Zugangssysteme und Einbettung in die Arbeitsumgebung des Benutzers	Proprietär (eigene Viewingsysteme) Internetdienste (z. B. telnet) Eingebettet in das World Wide Web
Koordination heterogener Datenbestände	Sammelagenten (*harvesting*) Brokersysteme
Benutzerkreis	Offen/geschlossen (z. B. Universitätsangehörige) Lokal/verteilt
Zugangskontrolle und Sicherheit	Abrechnungssysteme Lizenzvergabe Rechteverwaltung

Tabelle 66: Einige Merkmale digitaler Bibliotheken

Die Diskussion der einleitend aufgeworfenen Fragen,

- inwiefern sich das her vorgestellte Konzept dynamischer elektronischer Bücher in digitale Bibliotheken integrieren lässt und
- welche Komponenten des Architekturmodells sich auf die Ebene der digitalen Bibliothek verlagern lassen,

erfolgt aufgrund der technologischen Vielfalt und der heterogenen Funktionalität digitaler Bibliotheken auf einer abstrakten Ebene. Der erste Punkt betrifft vor allem die Frage, ob ein nach den hier entwickelten methodischen Vorgaben für die Inhaltsaufbereitung erstelltes elektronisches Buch einschließlich seiner Benutzerschnittstelle und Grundfunktionen Teil des Datenbestands einer digitalen Bibliothek sein kann: In einfachster Form lässt sich das elektronische Buch als webbasiertes Informationssystem ohne Änderung in den Datenbestand einer digitalen Bibliothek integrieren, wobei lediglich Zugangsrestriktionen hinzukommen bzw. die Zugangsmöglichkeiten der digitalen Bibliothek die des elektronischen Buchs als Einzelanwendung ergänzen oder ersetzen. Da die hier gewählten Formate (XML/SGML) auch in vielen digitalen Bibliotheken verwendet werden, ist eine solche Integration nicht problematisch. Wie bereits angedeutet wurde (Kap. 12.3.1), lässt sich die hier verwendete dateisystembasierte Speicherung auch durch eine datenbankorientierte Speicherung ersetzen: Soweit eine digitale Bibliothek eine Datenbank bzw. ein *content management system* für die Speicherungsebene verwendet, ist auch in dieser Hinsicht die Integration möglich.

Die Möglichkeit der Übertragung von Funktionsbestandteilen des Buchservers auf die Ebene einer digitalen Bibliothek, lässt sich stärker differenziert diskutieren:

1. Module, die für eine Mehrzahl elektronischer Bücher in gleicher Weise bereitgehalten werden müssen, lassen sich durch generische Funktionen der digitalen Bibliothek rea-

lisieren. Dazu gehört z. B. die Benutzerverwaltung und damit verbunden die Frage der Zugangsregelung zu den Inhalten eines elektronischen Buchs.

2. In weiterführender Sicht zählen hierzu auch Mechanismen der Abrechnung, die in der Regel Bestandteil der Funktionalität digitaler Bibliotheken sind. Neben Abonnementsystemen, bei denen die Inhalte einer digitalen Bibliothek gegen eine Pauschalgebühr zugänglich werden und werkbezogenen *pay-per-view*-Verfahren, bei denen eine elektronische Publikation (z. B. ein Aufsatz, ein elektronisches Buch) gegen einmalige Zahlung einer Gebühr abgerufen werden können, befinden sich Zahlungsmechanismen mit kleinteiligerem Bezug bislang erst in der Erprobung (Entrichtung eines *micro payment* pro abgerufener Präsentationseinheit, z. B. einer Buchseite). Gerade solche Verfahren ließen sich gut für den Einsatz in dynamischen elektronischen Büchern einsetzen, da nicht nur die Nutzung der Primärinhalte, sondern auch angekoppelter Dienste mit solchen Verfahren differenziert abgerechnet werden könnten.
3. Schließlich verbleiben diejenigen Funktionskomponenten des Buchservers, die spezifisch für ein einzelnes dynamisches Buch und seinen individuellen oder gruppenbezogenen Nutzungskontext sind. Ausgehend vom für jedes elektronische Buch unterschiedlichen inhaltsorientierten Markup, betrifft dies die Anbindung von Diensten und Komponenten und deren Zuordnung zu einem individuellen oder gruppenbezogenen Nutzungskontext. Diese Funktionalität müsste für jedes Buch einer digitalen Bibliothek gesondert erfolgen. Ein weiterführender Gedanke ist allerdings die Überlegung, dass Konzepte wie die Nutzung eines gruppenbezogenen Kontexts bei der Interaktion mit elektronischen Büchern nicht auf ein einzelnes Buch beschränkt sein müssen.

Zusammenfassend kann man festhalten, dass sich dynamische elektronische Bücher nach dem hier vorgestellten Modell in digitale Bibliotheken eingliedern lassen und dass die Bibliothek dabei Teile der Softwareinfrastruktur bereitstellen kann, die im Rahmen dieser Arbeit als Komponenten des Buchservers modelliert sind. In weiterführender Betrachtung ließen sich die Konzepte für die dynamische Integration von Komponenten und Diensten und ihre Zuordnung zu Nutzungskontexten auch für eine Mehrzahl von Publikationen (Büchern, Aufsätzen), wie sie in einer digitalen Bibliothek bereitgehalten werden, verallgemeinern.

16 Zusammenfassung

In dieser Arbeit wurde ein Modell dynamischer elektronischer Bücher entwickelt und anhand eines Lehrbuchs der Experimentalphysik exemplarisch realisiert. Ihm liegen Erfahrungen und Ergebnisse des Forschungsprojekts *Multimediales Physikalisches Praktikum* zugrunde, das im Rahmen des Verbundvorhabens *Weiterentwicklung des wissenschaftlich technischen Lehrwerks zur multimedialen Wissensrepräsentation* des BMB+F (Projektverbund *Multimediabuch*) am Institut für Informatik der Universität Leipzig durchgeführt wurde.

Dynamische elektronische Bücher sind im Kontext des elektronischen Publizierens ein Gegenstandsbereich der angewandten Informatik mit engen Querbezügen zu Informationswissenschaft und Linguistik, insbesondere Texttechnologie. Als Informationssysteme, die über das World Wide Web verfügbar sind, haben sie Berührungspunkte mit Hypertextsystemen, Multimediatechnologie und elektronischen Lehr- und Lernsystemen. Ein dynamisches elektronisches Buch stellt Information mit Hilfe von Hypertext-Techniken dar, geht aber anders als ein (offenes) Hypertextsystem von einem festen Ausgangsdatenbestand aus. Durch die Integration interaktiver multimedialer Komponenten, insbesondere Animationen und Simulationen physikalischer Versuche und Geräte, lassen sich dynamische elektronische Bücher als Multimediasysteme betrachten.

Eine Analyse des state-of-the-art elektronischer Bücher wurde einerseits anhand des Projektverbunds *Multimediabuch*, andererseits unter Berücksichtigung der in digitalen Bibliotheken im World Wide Web verfügbaren elektronischen Bücher vorgenommen (Kap. 3). Dabei zeigte sich, dass elektronische Bücher bisher vor allem als statische HTML-Anwendungen realisiert sind. Aus dieser Analyse ergab sich ein Merkmalskatalog elektronischer Bücher, der die Vielfalt der für ihre Erstellung erforderlichen Verfahren und Technologien verdeutlicht.

Aufbauend auf diesem Merkmalskatalog und auf den im *Referenzprojekt Multimediales Physikalisches Praktikum* gemachten Erfahrungen wurde das Konzept *dynamische elektronische Bücher* eingeführt und für seine Umsetzung ein allgemeines Architekturmodell abgeleitet (Kap. 4). Dynamische elektronische Bücher zeichnen sich durch folgende Eigenschaften aus:

- Für die Kodierung der Information im elektronischen Buch kommen deklarative Standards auf der Basis der *Standard Generalized Markup Language* (SGML) bzw. der *Extensible Markup Language* (XML) zum Einsatz.
- Bei der Informationskodierung werden verschiedene *Markupebenen* für die Beschreibung struktureller Information, die inhaltsorientierte Auszeichnung, die Spezifikation präsentationsorientierter Information und die Beschreibung von Metadaten unterschieden. Für die Umsetzung der Kodierung auf der Basis dieser Markupebenen wird eine geeignete Methode entwickelt.
- Dynamische elektronische Bücher unterscheiden sich von statischen elektronischen Büchern durch die in ihnen eingebetteten interaktiven multimedialen Komponenten und die verfügbaren Dienste.
- Komponenten und Dienste können bei der Nutzung des dynamischen elektronischen Buch erweitert und ergänzt werden. Dies betrifft sowohl die für den individuellen Le-

ser verfügbaren Dienste als auch die Bereitstellung zusätzlichen Materials zum elektronischen Buch über gruppenbezogene Nutzungskontexte.

Im Anschluss an die Diskussion der Eigenschaften dynamischer elektronischer Bücher wurde ein Architekturmodell entworfen, das beschreibt, wie die Softwareinfrastruktur für die Nutzung eines dynamischen elektronischen Buchs beschaffen sein muss:

- In technologischer Hinsicht sieht das Architekturmodell die Realisierung eines dynamischen elektronischen Buchs als Client-Server-Anwendung im World Wide Web vor, bei der das Buchverwaltungssystem als Komponente eines Webservers mit dem Buchbetrachtungssystem als für das elektronische Buch angepasstem Webbrowser kommuniziert.
- Das Buchverwaltungssystem hat dabei die Aufgabe, Inhalte zu speichern und dem Client auf Anforderung bereitzustellen. Es muss die für ein dynamisches elektronisches Buch verfügbaren Dienste und Komponenten verwalten und sie den Buchinhalten zuordnen.
- Das Buchbetrachtungssystem baut die Benutzerschnittstelle des dynamischen elektronischen Buchs auf und enthält die für die Benutzung erforderlichen Steuerungsroutinen und Erschließungskomponenten.

Das Architekturmodell wurde weitgehend ohne Bezug zum Referenzprojekt entwickelt, um seine Generalisierbarkeit für grundsätzlich beliebige Inhalte dynamischer elektronischer Bücher zu gewährleisten.

Im zweiten Teil der Arbeit (Kap. 5-9) wurden die für die deklarative Informationsaufbereitung verfügbaren Standards auf der Basis von SGML und XML vorgestellt und ihre Anwendbarkeit für die Entwicklung dynamischer elektronischer Bücher beschrieben. Neben der Erörterung grundlegender Aspekte von SGML und XML als Metasprachen für den Entwurf von Dokumentgrammatiken führt Kap. 5 insbesondere die in jüngster Zeit vom *World Wide Web Consortium* vorgelegten Substandards von XML (XML *namespaces*, XPath, XPointer, XLink) ein. Als standardisierte Anwendungen von XML wurden neben der als Sprache des World Wide Web weit verbreiteten *Hypertext Markup Language* (HTML) und ihrer Weiterentwicklung als *Extensible Hypertext Markup Language* (XHTML) auch für elektronische Bücher relevante Sonderstandards wie *open eBook* diskutiert (Kap. 6). Darüber hinaus finden in dynamischen elektronischen Büchern Beschreibungssprachen Verwendung, die spezifische funktionsbezogene Markup- und Darstellungsprobleme lösen. Ein Beispiel für diesen Bereich der deklarativen Auszeichnung von Information ist die *Mathematical Markup Language* (MathML).

Neben diesen standardisierten Anwendungen lassen sich drei weitere Gebiete benennen, für die eine Aufbereitung durch deklarative Standards möglich ist. Dazu gehören Transformations- und Präsentationssprachen, die einerseits die Aufgabe haben, Dokumente eines bestimmten Formats mit Hilfe von Dokumentgrammatiken zu übersetzen, andererseits dazu dienen, die Präsentationseigenschaften für eine bestimmte Dokumentgrammatik festzulegen (Kap. 7). Beide Aspekte sind für dynamische elektronische Bücher relevant: Transformationen sind u. a. erforderlich, um inhaltsorientiert ausgezeichnete Daten in darstellungsgerechte Teile zu gliedern und dynamisch Komponenten und Dienste einzubinden. Präsentationsinformation ist erforderlich, um die logische und inhaltsorientierte Auszeichnung in XML darstellen zu können.

Eine zukunftsweisende Entwicklung stellen deklarative Standards für die Aufbereitung und Synchronisation multimedialer, insbesondere zeitabhängiger Daten dar. Relevante SGML- und XML-basierte Standards wie HyTime und die *Synchronized Multimedia In-*

tegration Language (SMIL) wurden eingeführt und hinsichtlich ihrer Anwendbarkeit auf die Entwicklung dynamischer elektronischer Bücher diskutiert (Kap. 8).

Schließlich stellt die Beschreibung von Metainformation einen weiteren Anwendungsbereich für deklarative Standards dar (Kap. 9). Dabei ist zwischen syntaktischen und inhalts- bzw. funktionsbezogenen Aspekten zu unterscheiden. Als ein vom *World Wide Web Consortium* spezifizierter Standard wurde das *Resource Description Framework* vorgestellt, das sich dazu eignet, Metainformation bezüglich dynamischer elektronischer Bücher und ihrer Inhalte zu kodieren. Hinsichtlich der Funktion einer Metadatenbeschreibung sind zwei Bereiche für elektronische Bücher von Bedeutung: Die bibliographische Beschreibung durch Metadaten, die u. a. der Informationserschließung dient: Als Binärdaten vorliegende Komponenten können durch ihre Metadatenbeschreibung erfasst werden. Ein zweiter Bereich der Metadatenkodierung sind Standards, die die lehr- und lernbezogenen Aspekte elektronischer Bücher beschreiben.

Der dritte Teil der Arbeit (Kap. 10-15) konkretisiert das Architekturmodell für dynamische elektronische Bücher am Beispiel des Referenzprojekts. Dabei wurde gezeigt, wie sich die Methode für die Einführung deklarativen Markups auf verschiedenen Ebenen umsetzen lässt (Kap. 10) und dargestellt,

- welche Teile des elektronischen Buchs man durch allgemeine *strukturbezogene* Auszeichnungselemente beschreiben kann,
- für welche *Inhalte* zusätzliche Auszeichnungsmarken erforderlich sind, um den Besonderheiten des konkreten Anwendungsfalls Rechnung zu tragen,
- welche Arten von *Hypertextverknüpfungen* benötigt werden und wie sie darzustellen sind,
- wie die Markupelemente mit *Präsentationsinformation* zu versehen sind und
- welche Form der *Metadatenbeschreibung* Anwendung findet.

Darüber hinaus kommt XML-Markup auch für Verwaltungsfunktionen des Buchservers zum Einsatz (Kodierung von Diensten und Kontexten, Benutzerverwaltung).

Ausgehend von diesem Informationsmodell erfolgte ein kurzer Überblick über Entwicklungswerkzeuge für elektronische Bücher (Kap. 11). Im Rahmen der Darstellung des Buchverwaltungssystems (Kap. 12) wurde das Architekturmodell für dynamische elektronische Bücher konkretisiert. Seine Realisierung erfolgte als Java-Servlet im Kontext eines WWW-Servers. Es baut auf der *Cocoon Publishing Engine* auf und verfügt daher über die erforderlichen Möglichkeiten zur Transformation und Präsentation XML-basierter Daten (XML-Parser, XSL/T-Formatierer). Daneben umfasst das Buchverwaltungssystem eine Reihe von Administrationskomponenten (Benutzer-, Dienste-, Kontextverwaltung).

Der Darstellung des Buchbetrachtungssystems (Kap. 13) als Benutzerschnittstelle des dynamischen elektronischen Buchs geht eine Betrachtung software-ergonomischer Aspekte voraus. Dabei war von besonderer Bedeutung, inwieweit sich Gestaltungsrichtlinien für webbasierte Informationssysteme auf dynamische elektronische Bücher übertragen lassen. Ausgehend von einem Anforderungsprofil für die Gestaltung des elektronischen Buchs wurde die für die Benutzerschnittstelle gewählte modifizierte *Buchmetapher* erläutert: Sie erlaubt die Aufteilung der Inhalte in *Fokus*- und *Kontextseite* und macht es dem Benutzer auf einfache Weise möglich, zum aktuellen Inhalt des Buchs verfügbare Komponenten wie Animationen und Simulationen zu nutzen. Der seitengerechte Satz und eine klare Gliederung der Benutzerschnittstelle in Darstellungs- und Funktionsbereiche erleichtern den Umgang mit dem dynamischen elektronischen Buch. Neben dem Hauptfenster des Buchbetrachtungssystems stehen dem Benutzer mit *Hierarchiebrowser* und

Rechercheschnittstelle zusätzliche Werkzeuge für Navigation und gezielte Suche zur Verfügung.

Die Integration von interaktiven multimedialen Komponenten einerseits, von Diensten andererseits, stellt einen wesentlichen Mehrwert dynamischer elektronischer Bücher sowohl gegenüber ihren gedruckten Pendants als auch gegenüber statischen elektronischen Büchern als 1:1-Entsprechungen der jeweiligen Printfassung dar (Kap. 14). Neben den technologischen Grundlagen wurden auch die didaktischen und präsentationsbezogenen Eigenschaften von Multimediakomponenten erörtert. Bei den verschiedenen Typen von Komponenten handelt es sich um

- interaktive Visualisierungen von Versuchsaufbauten,
- Animationen physikalischer Phänomene,
- Animationen von Versuchsdurchführungen,
- Simulationen physikalischer Phänomene und Geräte.

Die Integration von Diensten in dynamische Bücher eröffnet dem Benutzer zusätzliche Informationsquellen. Am Beispiel von über das World Wide Web verfügbaren Diensten wurde dabei gezeigt, wie sie sich in ein dynamisches elektronisches Buch integrieren lassen und wie sie einzelnen Benutzern bzw. Benutzergruppen zugeordnet werden können. Zu den prototypisch realisierten Diensten gehört ein Speicherungsdienst, der es einzelnen Benutzern bzw. Benutzergruppen erlaubt, die Inhalte des dynamischen elektronischen Buchs durch Annotationen oder durch andere Dienste ermittelte Informationen zu ergänzen und mit Hilfe des WebDAV-Protokolls im World Wide Web zu speichern.

Informationserschließungsdienste, insbesondere Suchmaschinen, erlauben es dem Benutzer, zu den Inhalten des dynamischen elektronischen Buchs zusätzliche Informationen zu ermitteln. Dabei wird die Tatsache, dass die Inhalte des dynamischen elektronischen Buchs deklarativ ausgezeichnet sind und vom Buchverwaltungssystem ausgewertet werden können, für die Unterstützung der Anfrageformulierung genutzt (*query expansion*).

Beispiele für Informationsdienste mit Bezug zu deklarativ ausgezeichneten Inhaltselementen des dynamischen elektronischen Buchs sind ein Online-Lexikon sowie die Übersetzung von Fachbegriffen des Buchs. Mit Hilfe von Weiterverarbeitungsdiensten kann man die Inhalte des dynamischen elektronischen Buchs zusätzlich nutzen.

Abschließend erörtert Kap. 15, inwiefern sich das Modell dynamischer elektronischer Bücher verallgemeinern lässt und wie sich dynamische elektronische Bücher in digitale Bibliotheken einordnen lassen.

Es konnte gezeigt werden, wie sich ein allgemeines Architekturmodell dynamischer elektronischer Bücher exemplarisch realisieren lässt. Neben den Vorteilen, die der Benutzer durch multimediale Komponenten und Dienste erhält, sind die folgenden Merkmale dynamischer elektronischer Bücher ein Ausgangspunkt für Überlegungen, die über diese Arbeit hinausweisen: Die Methode für die Bestimmung der Auszeichnungselemente lässt sich durch texttechnologische und linguistische Verfahren verfeinern. Dabei ist insbesondere an den automatischen Aufbau von Hypertextstrukturen mit Hilfe großer Textcorpora zu denken. Das in dieser Arbeit vorgestellte Architekturmodell dynamischer elektronischer Bücher abstrahiert von den Prozessen der Erstellung der Buchinhalte (Texte, Komponenten). An integrierten Entwicklungsmethoden und Autorenwerkzeugen, die die gesamte Bandbreite XML-basierter Standards berücksichtigen, fehlt es bisher. Insofern liegt hier ein wichtiger Ansatzpunkt für weitergehende Untersuchungen.

17 Anhänge

17.1 Dokumententypdefinitionen

Nachfolgend sind die für das elektronische Buch und seine Verwaltung entwickelten Dokumententypdefinition dokumentiert; sie beschränken sich auf die nötigsten Angaben, die aus dem Kontext des Referenzprojekts ermittelt wurden

17.1.1 Buch

```
<!-- Die Buch-DTD enthaelt die hierarchische Grundstruktur und inhaltsorientierte
     Auszeichnungselemente, aber nicht die Elemente, die aus anderen XML namespaces
     in der Dokumentinstanz importiert werden. Bei diesen Importen handelt es sich um
     u. a. folgende DTDs:
     - HTML f&uuml;r allgemeine Formtierung und Tabellensatz
     - MathML f&uuml; Formelsatz
     - xlink f&uuml;r Verkn&uuml;pfungen

   Bestandteile, die durch Skripte oder den Buchserver automatisch generiert werden,
   wie Inhaltsverzeichnis oder Register, haben hier ebenfalls keine eigenen Elemente
   (anders als in DocBook). Referenzierungs-Ids für die automatische Einfuehrung von
   Hypertextverknuepfungen werden automatisch generiert.

   Die Vorgehensweise wurde von folgenden Aspekten motiviert:
   a) den konkreten Vorgaben des Referenzprojekts,
  b) standardisierten Vorlage-DTDs (v. a. DocBook und HTML) und
  c) dem Ziel, hier nur die im Sinn des Referenzprojekts unbedingt erforderlichen Elemente zu definieren

   Da die Standards nichts ueber die Verwendung von namespaces innerhalb von DTDs aussagen, wird
   wie folgt verfahren:
   -  Benoetigte inhaltsorientierte Elemente erhalten ein freies content model (ANY), innerhalb dessen
      Standardelemente aus anderen DTDs verwendet werden koennen.
   -  Die Alternative, Elemente aus allen potentiell beteiligten DTDs mit Pseudo NS-Präfixen so
      versehen und so explizit zu redefinieren, ist nicht sinnvoll.
   -  Eine zukuenftige umfasende Loesung sollte XML-Schemata mit expliziten Ableitungsrelationen
      verwenden; in zukuenftigen XML-Versionen werden namespaces in der DTD verfuegbar sein und
      eine praezisere Beschreibung erlauben.
   -  Problem: xlink-Namespace bleibt hier enthalten (!?) – implizite Loesung durch Inklusion des
      xlink-Namespace in allen Dokumenten (“Doppelloesung”)
-->

<!DOCTYPE PHYSIKPRAKTIKUM[
<!ENTITY % URI        "CDATA" -- ein URI (wie in HTML definiert)                    -->

<!-- ------------------------------------------------------------------------------- -->
<!-- die folgenden beiden Parameterentitaeten unterscheiden zwischen
     generischem Markup und inhaltsorientierten, buchabhängigen
     Auszeichnungen                                                                -->
```

```
<!ENTITY % INHALT.allgemein "Absatz" --Generischer Container fuer
                                    HTML-block/inline-Elemente          -->

<!ENTITY % INHALT.spezifisch "GRUNDLAGEN | EXPERIMENT |ABSATZ"
  -- ABSATZ: Generischer Container fuer HTML-block-Elemente
     (kann insb. Verweise enthalten)                                    -->
<!ELEMENT ABSATZ  ANY                                                   -->

<!-- Auf der globalen Ebene keine spezifischen Komponentenlinks; Dienste werden
     durch den Buchserver eingebunden                                   -->
<!ELEMENT  PHYSIKPRAKTIKUM (VORWORT?, (KAPITEL1 | KAPITEL)+)           -->
<!ATTLIST    PHYSIKPRAKTIKUM
             adresse  %URI; #REQUIRED
             titel    CDATA #REQUIRED
             autoren  CDATA #REQUIRED                                   -->

<!-- Vorwort: einfaches Modell, koennte konkreter modelliert sein (vgl. DocBook) -->
<!ELEMENT    VORWORT (%INHALT);+                                        -->

<!ELEMENT KAPITEL1   (TITEL,
                      KOMPLINKS?,
                      ((%INHALT.allgemein;) |(%INHALT.spezifisch;))*,
                      KAPITEL2*)                                        -->
<!ATTLIST  KAPITEL1
       autoren  CDATA   #IMPLIED                                        -->

<!ELEMENT KAPITEL2   (TITEL,
                      KOMPLINKS?,
                      ((%INHALT.allgemein;) |(%INHALT.spezifisch;))*,
                      KAPITEL3*)                                        -->
<!ATTLIST  KAPITEL2
       autoren  CDATA   #IMPLIED                                        -->

<!ELEMENT KAPITEL3   (TITEL,
                      KOMPLINKS?,
                      ((%INHALT.allgemein;) |(%INHALT.spezifisch;))*,
                      KAPITEL4*)                                        -->
<!ELEMENT KAPITEL4 (TITEL,
                    KOMPLINKS?,
                    ((%INHALT.allgemein;) |(%INHALT.spezifisch;))*,
                    KAPITEL5*)                                          -->
<!ELEMENT KAPITEL5 (TITEL,
                    KOMPLINKS?,
                    ((%INHALT.allgemein;) |(%INHALT.spezifisch;))*)     -->

<!-- extended link (nach xlink) fuer die Angabe mehrerer Verknuepfungen zur
     einem Inhaltsbestandteil                                           -->
<!-- Musterbasierte einfache Rueckverknuepfungen koennen automatisch
     generiert werden HTML-Standardanker (Voraussetzung: ID-Attribute zur
     Identifikation/XPointer-Verwendung                                         -->
<!ELEMENT  KOMPLINKS BESCHRIFTUNG?, KOMPLINK+>
<!ATTLIST    KOMPLINKS
     xlink:type  (extended)   #FIXED  "extended"
     xlink:role  NMTOKEN      #FIXED  "Komponenten"
     xlink:title  CDATA       #IMPLIED
```

```
    xlink:actuate (  onLoad
                  |  onRequest
                  |  undefined    #IMPLIED    “onRequest”                    -->
    xlink:show    (  new
                  |  replace
                  |  embed
                  |  undefined    #IMPLIED    “new”                          -->

<!-- Typisierte Lokatoren fuer die Linksammlung, Zuordnung im Dokument zu
     Hierarchie oder inhaltsoientierten Marken                               -->
<!ELEMENT  KOMPLINK         EMPTY>
<!ATTLIST  KOMPLINK
           xlink:type       (locator) #FIXED  "locator"
           xlink:href       %URI;     #REQUIRED
           xlink:role       (Abbildung
                            |Animation
                            |Fotografie
                            |Simulation
                            |Tabelle
                            |Text
                            |Uebung)       #REQUIRED
           xlink:title      CDATA         #IMPLIED
           symbol           %URI;         #IMPLIED                  -->

<!ELEMENT  BESCHRIFTUNG     #PCDATA                                   -->
<!ATTLIST  BESCHRIFTUNG
           xlink:type   (title)   #FIXED  "title"                    -->

<!-- Wichtigste Inhaltskomponente: Versuchsbeschreibungen, mit
     standardisiertem Aufbau                                         -->
<!ELEMENT    EXPERIMENT  KOMPLINKS?,
                         AUFGABE,
                         GRUNDLAGEN?,
                         BESCHREIBUNG,
                         VERSUCH                                     -->

<!-- Einschraenkung auf Listen ist nicht sinnvoll, daher allgemeines Modell -->
<!ELEMENT    AUFGABE  KOMPLINKS?, (%INHALT.allgemein;)*              -->

<!ELEMENT    BECHREIBUNGKOMPLINKS?, (%INHALT.allgemein;)*            -->

<!-- GRUNDLAGEN k&ouml;nnen optiunal weitere inhaltsorientierte
     Elemente enthalten.                                             -->
<!ELEMENT  GRUNDLAGEN KOMPLINKS?,
                      #PCDATA,
                      BEISPIEL*,
                      GLEICHUNG*,
                      DEFINITION*,
                      (%INHALT.allgemein;))*                           -->

<!-- ------------------------------------------------------------------------------------ -->
<!-- “kleinere” inhaltsorientierte und textfunktionale Kataegorien, die ueberall
     auftreten koennen                                                  -->
<!ELEMENT    BEISPIEL      KOMPLINKS?, (%INHALT.allgemein;)            -->
```

```
<!Kapsel fuer MathML-Inhalte und als einfacher Text kodierte Formeln          -->
<!ELEMENT  GLEICHUNG    ANY                                                   -->
<!ATTLIST    GLEICHUNG    nr   ID      #IMPLIED                              -->

<!   Kapselt alle Definitionen, Einfuehrungen von Fachterminologie etc.; ID-
     Attribut dient zur Linkgenerierung von der Begriffsverwendung auf die
     Begriffseinfuehrung                                                     -->
<!ELEMENT  DEFINITION    (#PCDATA, FACHBEGRIFF)*                             -->
<!ATTLIST    DEFINITION    nr   ID      #IMPLIED                             -->

<!-- ------------------------------------------------------------------------ -->
<!-- Elemente auf "Wortebene": Fachbegriffe, Symbole etc.                    -->
<!-- aehnlich definition term in HTML; subclassing durch ein XML-Schema
     anhand einer Physik-Ontologie wuneschenswert; hier "nur" einfacher
     Anknuepfungspunkt fuer Dienste etc.                                     -->
<!ELEMENT    FACHBEGRIFF    #PCDATA                                          -->
<!ELEMENT    SYMBOL         #PCDATA                                          -->

<!-- Hier Typisierung, Einschraenkung von Wertebereichen etc. durch Schemata
     sinnvoll                                                                -->
<!MESSWERT  WERT, EINHEIT                                                    -->
<!ELEMENT    WERT          #PCDATA                                           -->
<!ELEMENT    EINHEIT       #PCDATA                                           -->

<!-- Marke fuer einfache Links im Text des Buchs, Typisierung ueber
     das role-Attribut;                                                      -->
<!ELEMENT    VERWEIS    #PCDATA                                            -->
<!ATTLIST    VERWEIS    xlink:type    (simple)    #FIXED      "simple"
                        xlink:href    %URI;       #REQUIRED
                        xlink:title   CDATA       #IMPLIED
                        xlink:role    (Strukturverweis
                                      |Textverweis
                                      |Formelverweis
                                      |Kompverweis) #IMPLIED
                        xlink:actuate
                                      (onLoad
                                      |onRequest
                                      |undefined   #IMPLIED "onRequest"   -->
                        xlink:show    (new
                                      |  replace
                                      |  embed
                                      |  undefined  #IMPLIED "new"         -->
]>
```

17.1.2 Komponenten

```
<!DOCTYPE KOMPONENTE[
<!-- DTD fuer die Komponentencontainer:
     Alle Komponenten sind als eigene Dateien realisiert, die gegen diese DTD
     validiert werden.  Im Prototyp resultiert auch eine gesonderte Darstellung;
     durch einen anderen user agent koennte auch ein komplexeres Layout erreicht
     werden, die Container waeren dann entsprechend in das Hauptddokument
     einzubinden ("Transklusion") und zu formatieren
-->
```

```
<!-- ------------------------------------------------------------------------------------- -->
<!-- Entitaeten zur Kapselung von HTML-Code wie oben in der Buch-DTD                       -->
<!-- ------------------------------------------------------------------------------------- -->
<!ENTITY      %     KOMPINHALT  "(   ABBILDUNG
                                 |   ANIMATION
                                 |   FOTOGRAFIE
                                 |   SIMULATION
                                 |   TABELLE
                                 |   TEXT
                                 |   UEBUNG )"                                          -->

<!-- ------------------------------------------------------------------------------------- -->
<!-- Offenes Modell: Komponente kann auch beliebigen Inhalt enthalten (nicht nur
     Textkomponeneten)                                                                    -->
<!ELEMENT    KOMPONENTE    TITEL, (%KOMPINHALT;),
                           BESCHRIFTUNG?,
                           BESCHREIBUNG?,
                           (%INHALT.allgemein;)*                                        -->
<!ATTLIST    KOMPONENTE    typ    ID     #REQUIRED                                      -->

<!-- Verwendung: Alle diagramatischen/schematischen Darstellungen
     (Diagramme, Funktionsgraphen etc. staerkere Typisierung deknbar                      -->
<!ELEMENT ABBILDUNG   EMPTY>
<!-- Attribute zur Umwandlung in IMG-Marken durch XSL/T                                   -->
<!ATTLIST  ATTLIST ABBILDUNG
           href   %URI;        #REQUIRED
           name   CDATA        #IMPLIED
           hoehe  NMTOKEN #IMPLIED
           breite NMTOKEN #IMPLIED                                                      -->

<!ELEMENT ANIMATION   EMPTY>
<!ATTLIST  ATTLIST ANIMATION
           href   %URI;        #REQUIRED
           name   CDATA        #IMPLIED
           typ    (shockwave|java|smil)
           hoehe  NMTOKEN #REQUIRED
           breite NMTOKEN #REQUIRED                                                     -->

<!-- Verwendung: Alle piktoriellen Darstellungen mit unmittelbarem
     Abbildcharakter                                                                      -->
<!ELEMENT FOTOGRAFIE  EMPTY>
<!-- Attribute zur Umwandlung in IMG-Marken durch XSL/T                                   -->
<!ATTLIST  ATTLIST FOTOGRAFIE
           href   %URI;        #REQUIRED
           name   CDATA        #IMPLIED
           hoehe  NMTOKEN #IMPLIED
           breite NMTOKEN #IMPLIED                                                      -->

<!ELEMENT SIMULATION  (PARAMETER*)>
<!ATTLIST  ATTLIST SIMULATION
           href   %URI;        #REQUIRED
           name   CDATA        #IMPLIED
           typ    (shockwave|java|smil)
           hoehe  NMTOKEN #REQUIRED
           breite NMTOKEN #REQUIRED                                                     -->
```

```
<--  Generische Parametermakre analog param in HTML/APPLET ...                -->
<!ELEMENT PARAMETER EMPTY>
<!ATTLIST ATTLIST PARAMETER
          name  CDATA       #REQUIRED
          wert      NMTOKEN #REQUIRED                                          -->

<--  Generische Textkomponente; hier keine unmittelbare
     Typisierung vorgesehen (aber z. B. durch eingebettete Elemente anderer DTDs
     moeglich (z. B. DTD fuer ein anderes BuchParametermakre analog param
     in HTML/APPLET ...                                                       -->
<!ELEMENT TEXT (%INHALT.allgemein) *)>
<!ATTLIST ATTLIST TEXT
          href    %URI;       #REQUIRED
          name  CDATA       #IMPLIED
                                                                             -->
<--  Tabelle, offenes Modell (fuer HTML-Tabellen, Einbindung ueber namespaces) -->
<!ELEMENT TABELLE  ANY>
<--  allg. Textkomponente                                                     -->
<!ELEMENT TEXT (%INHALT.allgemein) *)>
<--  Uebung, offenes Modell                                                   -->
<!ELEMENT UEBUNG ANY>
]>
```

17.1.3 Buchverwaltung

17.1.3.1 Diensteliste

```
<!-- DTD fuer die Dienstebeschreibung:                                       -->

<!DOCTYPE DIENSTELISTE[

<!ENTITY % URI       "CDATA" -- ein URI (wie in HTML definiert)              -->

<!ELEMENT DIENSTELISTE DIENST+                                               -->

<!ELEMENT DIENST ( NAME,
                   FUNKTION,
                   BEZUG+,
                   ADRESSE,
                   TYP,
                   PROTOKOLL,
                   INTERFACE?,
                   DIENSTPARAMETER*,
                   FOLGEDIENST*,
                   AUTORISIERUNG?,
                   ABRECHNUNG?)                                              -->

<!ELEMENT  NAME            #PCDATA>
<!ELEMENT  FUNKTION        #PCDATA>
<!ELEMENT  BEZUG           #PCDATA>
<!ELEMENT  ADRESSE         (%URI;)>
<!ELEMENT  TYP             #PCDATA >
<!ELEMENT  PROTOKOLL EMPTY>
```

```
<!ATTLIST    PROTOKOLL
             typ          CDATA     #REQUIRED
             adresse      CDATA     #REQUIRED                    -->
<--  Adresse des zum Dienstgehoerenden Interfaceelements, falls es fuer das Buch
     gesondert realisiert ist                                     -->
<!ELEMENT  INTERFACE     EMPTY                                    -->
<!ATTLIST    INTERFACE
             typ          CDATA     #REQUIRED
             adresse      CDATA     #REQUIRED                    -->
<--  Parameterliste für den Dienst, kann verwendet werden, um ein Interface
     zu generieren (z. B. Formularaufbau )                        -->
<!ELEMENT DIENSTPARAMETER EMPTY>
<--  Exaktere Typenliste mit Hilfe von XML Schemata denkbar;
     Einfache Datentypisierung analog Java-Typen; fehlende Werte-Attribute
     sind Eingabeargumente fuer den Benutzer                      -->
<!ATTLIST  DIENSTPARAMETER
           name  CDATA     #REQUIRED
           typ     (int
                   |char
                   |float
                   |boolean
                   |String)     (String)
           wert      CDATA   #IMPLIED>
<!ELEMENT  FOLGEDIENST  #PCDATA>
<!-- hier nicht naeher ausgefuehrt, da noch nicht verwendet:     -->
<!ELEMENT  AUTORISIERUNG    ANY>
<!ELEMENT  ABRECHNUNG       ANY>
]>
```

17.1.3.2 Dienstekoordination

```
<!DOCTYPE  DIENSTEKOORDINATION [
<!ELEMENT  DIENSTEKOORDINATION  DIENSTEANBINDUNG+               -->
<!ELEMENT  DIENSTEANBINDUNG     EMPTY                           -->
<!ATTLIST    DIENSTEANBINDUNG
             bezug          CDATA     #REQUIRED
             dienst         CDATA     #REQUIRED                 -->
]>
```

17.1.3.3 Benutzer

```
<!-- DTD fuer die Verwaltung von Informationen ueber Benutzer:  -->
<!DOCTYPE BENUTZER[

<!-- auch hier koennte ein XML-Schema die Daten praeziser modellieren.
     Der Speicherort, d. h. die Adresse eines WebDAV-Servers, fuer den der
     Benutzer Schreibrechte besitzt (genauer: die Adresse einer Kollektion
     auf einem WebDAV-Server) ist als wichtigste Dienstefunktion den
     benutzerbezogenen Daten zugerechnet.
     Dabei gilt implizit die Annahme, dass die Autorisierungsdaten des
     Benutzers auch fuer den Zugriff auf den WebDAV-Server gelten (und
     somit der Speicherungsdienst in seiner Meta-Funktion zur Bereitstellung
     der Diensteinformation durch die Login-Daten ermittelt werden kann.
     Gleiches gilt fuer die Angabe der Speicherungsinformation eines
     Gruppenkontexts                                             -->
```

```
<!ENTITY % URI        "CDATA" -- ein URI (wie in HTML definiert)          -->
<!ELEMENT BENUTZER (NAME, PASSWORT, EMAIL?, SPEICHERORT)                  -->
<!ELEMENT  NAME           #PCDATA                                         -->
<!ELEMENT  PASSWORT       #PCDATA                                         -->
<!ELEMENT  EMAIL          (%URI;)                                         -->
<!ELEMENT  SPEICHERORT    (%URI;)                                         -->
]>
```

17.1.3.4 Gruppenkontext

```
<!-- DTD fuer Definition und Verwaltung eines Gruppenkontexts
     Er besteht aus einer Adresse, beliebig vielen zugeordneten Benutzzern bzw.
     Domänen, sowie beliebig vielen Dienstespezifikationen
     Die Elemente sind dabei wie oben definiert (wird hier nicht
     wiederholt)                                                          -->
<!ENTITY % DOMAENE "CDATA" -- ein DNS-Name                                -->
<!DOCTYPE  GRUPPENKONTEXT[
<!ELEMENT  GRUPPENKONTEXTADRESSE,
                              BENUTZER*,
                              (%DOMAENE;) *,
                              DIENST*                                     -->
]>
```

17.2 Abkürzungsverzeichnis

ACM	Association for Computing Machinery
m. w. N.	mit weiteren Nachweisen
ACL	Agent Communication Language
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BLOB	Binary Large Object
BOF	Benutzeroberfläche
CACM	Communications of the Association for Computing Machinery
CAP	Computer Aided Publishing
CB-OHS	*component-based open hypermedia system*
CD	Compact Disk
CD-ROM	Compact Disk – Read Only Memory
CGI	Common Gateway Interface
CMS	Content Management System
COM	Component Object Model
CORBA	Common Object Request Broker Architecture
CSCW	Computer Supported Cooperative Work
CSS	Cascading Style Sheet
DB	Datenbank
DCD	Document Content Description
DCOM	Distributed Component Object Model
DLI	*Digital Library Initiative*
DOI	Digital Object Identifier
DOM	Document Object Model
DSSSL	Document Style Semantics and Specification Language
DTD	Document Type Definition
DTP	Desktop Publishing
DVD	Digital Versatile Disk
EBCDIC	
EP	Electronic Publishing
FIPA	Foundation for Physical Agents
FIZ	Fachinformationszentrum
FTP	File Transfer Protocol
GIF	Graphics Interchange Format
GML	Generalized Markup Language
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HDD	Hard Disk Drive
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
HyTime	Hypermedia/Time-based Markup Language
IDE	Integrated Development Environment

IDL	Interface Definition Language
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IFIP	International Federation for Information Processing
IMS	Instructional Management Systems
IR	Information Retrieval
IRC	Internet Relay Chat
IRS	Information Retrieval System
ISO	International Standards Organisation
JAF	Java Activation Framework
JDBC	Java Database Connectivity
JDK	Java Development Kit
JPG	Joint Photographers' Expert Group
KIF	Knowledge Interchange Format
KMS	Knowledge Management System
KQML	Knowledge Querying and Manipulation Language
MARC	Machine Aided Recording and Cataloguing
MB	Megabyte
MHEG	Multimedia and Hypermedia information coding Expert Group
MIME	Multipurpose Internet Mail Extensions
MMDBMS	Multimedia Data Base Management System
NCSTRL	Networked Computer Science Technical Report Library
OASIS	Organization for the Advancement of Structured Information Standards
OEB	Open eBook
ODA	Open Document Architecture
ODBC	Open Database Connectivity
ODIF	Open Document Interchange Format
OHP	Open Hypermedia Protocol
OHS	Offenes Hypermediasystem, *open hypermedia system*
OLE	Object Linking and Embedding
OODBMS	Object Oriented Database Management System
OQL	Object Query Language
ORB	Object Request Broker
OSI	
PDF	Portable Document Format
PI	Processing Instruction
PS	Postscript
QoS	Quality of Service
RAM	Random Access Memory
RDBMS	Relational Data Base Management System
RDF	Resource Description Format
RFC	Request For Comments
RMI	Remote Method Invocation
RMM	Relationship Management Method
RTF	Rich Text Format
SAX	Simple API for XML
SDQL	Standard Document Query Language
SE	Software Engineering
SGML	Standard Generalized Markup Language
SMIL	Synchronized Multimedia Integration Language
SQL	Structured Query Language
SSL	Secure Sockets Layer
STFP	SGML Tree Formatting Process
STTP	SGML Tree Transformation Process
TCP/IP	Transmission Control Protocol/Internet Protocol
TCWA	„*the classic web application*“
TEI	Text Encoding Initiative
UID	Unique ID
UIMS	User Interface Management System
UML	Unified Modeling Language
UNIMARC	
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
USMARC	United States Machine Aided Recording and Cataloguing
WAIS	Wide Area Information System
WebDAV	Web Distributed Authoring and Versioning
WfMC	Workflow Management Coalition
WWW	World Wide Web
XLink	Extensible Link Language
XML	Extensible Markup Language
XQL	Extensible Query Language
XSL	Extensible Style

17.3 Verzeichnis der Tabellen

17.4 Verzeichnis der Abbildungen

17.5 Verzeichnis der Codebeispiele

17.6 Verzeichnis der Websites

Die nachfolgenden Liste enthält die *uniform resource locators* der im Text erwähnten Websites.

http://ariadne.unil.ch
http://aspra15.informatik.uni-leipzig.de/EBook/start.xml
http://aspra9.informatik.uni-leipzig.de/Teubner
http://developer.netscape.com
http://java.apache.org/jserv/
http://jodi.ecs.soton.ac.uk/
http://jodi.ecs.soton.ac.uk/Articles/v01/i02/editorial.shtml
http://link.springer.de
http://ltsc.ieee.org
http://metadata.net/dstc/
http://searchtools.com/tools/tools.html.
http://stneasy.FIZ-Karlsruhe.de
http://sunsite.berkeley.edu/SWISH-E/
http://users.iclway.co.uk/mhkay/saxon/
http://wave.eecs.wsu.edu/CKRMI/OML.html
http://WebEQ.com
http://wortschatz.uni-leipzig.de
http://www.acm.org/dl
http://www.alphaworks.ibm.com/tech/DAV4J
http://www.alphaworks.ibm.com/tech/lotusxsl
http://www.apache.org
http://www.asymetrix.com
http://www.bmbf.de/deutsch/arbeit/aufgaben/leitproj/wissen.htm.
http://www.brainmedia.de
http://www.ccil.org/~cowan/XML/XCatalog.html
http://www.clc-marketing.com/xslp/
http://www.cni.nl/GRiNS
http://www.computer.org/publications/dlib/
http://www.darmstadt.gmd.de/PTF/Ausschreibungen/Multimed.htm
http://www.dlib.org/dlib.html
http://www.dol.uni-leipzig.de
http://www.ebooknet.com
http://www.garshol.priv.no/download/xmltools/cat_ix.html
http://www.global-info.org
http://www.helio.org/products/smil/
http://www.hu-berlin.de/presse/amb/amb98_14.html
http://www.hummingbird.com/products/dkm/km/searchserver/
http://www.hyperwave.com
http://www.ics.uci.edu/~webdav/
http://www.iics.edu/jucs

http://www.imb-jena.de/IMAGE_PROJECT.html
http://www.ims-project.org
http://www.inf-wiss.uni-konstanz.de/FG/IV/KHS/
http://www.iso.ch
http://www.jclark.com/Jade
http://www.jclark.xom/XT
http://www.leitungsbau.de
http://www.macromedia.com
http://www.maplesoft.com/corporate/press/press_010.html
http://www.mathtype.com
http://www.megginson.com/SAX/event.html
http://www.megginson.com/SAX/index.html
http://www.megginson.com/SAX/SAX2/
http://www.xml.org
http://www.metacrawler.com
http://www.metadata.net.
http://www.metager.de
http://www.mevis.de/~guido/MammaVision.html
http://www.microsoft.com
http://www.mpib-berlin.mpg.de/DOK/metatagd.htm
http://www.multibook.de
http://www.multibook.de
http://www.mupad.de
http://www.omg.org
http://www.openebook.org
http://www.orthogon.de/doc/products/bookweb.htm
http://www.poet.de
http://www.real.com
http://www.rocket-ebook.com
http://www.savvysearch.com
http://www.softquad.com/products
http://www.softwareag.com/tamino
http://www.thieme.de
http://www.ub.hu-berlin.de/cdrom/disspub.html
http://www.ubka.uni-karlsruhe.de/eva/
http://www.udanax.com
http://www.verity.com/products/infoserv/
http://www.w3.org/Math/
http://www.w3c/Amaya
http://www.webdav.org/mod_dav
http://www.wolfram.com/news/archive/mathml.html
http://www.xanadu.net/TECH/xuTech.html
http://www.xml.com
http://www.yahoo.com
http://www.yahoo.de
http://xml.apache.org
http://xml.apache.org/cocoon/

17.7 Literaturverzeichnis

ABITEBOUL, Serge et al. (1997). “Querying Documents in Object Databases.” In: International Journal on Digital Libraries 1 (1997), 5-19.

ABITEBOUL, Serge; BUNEMAN, Peter; SUCIU, Dan (1999). Data on the Web. San Francisco et al.: Morgan Kaufman.

ADHVARYU, S. K.; BALBIN, I. (1999). “How ‘Useful’ is Multimedia on the WWW for Enhancing Teaching and Learning?” In: Proc. ICMCS ’98. 1998 IEEE International Conference on Multimedia Computing and Systems, Austin/TX, Juni/Juli 1998, 44-53.

ADLER, Sharon C. (1997). “The ‘ABCs’ of DSSSL.” In: Journal of the American Society for Information Science 48(7) (1997), 597-602.

ADLER, Sharon C. et al. (1997). A Proposal for XSL. *World Wide Web Consortium* Note, August 1997, *http://www.w3.org./TR/NOTE-XSL.html*.

AGOSTI, Maristella et al. (1997). „On the Use of Information Retrieval Techniques for the Automatic Construction of Hypertext.“ In Information Processing & Management 33(2) (1997), 133-144.

AKSCYN, Robert M.; MCCRACKEN, Donald L.; YODER, Elise A. (1988). “KMS: a Distributed Hypermedia System for Managing Knowledge in Organizations.” In: CACM 31(7) (1988), 820-835.

ALLAN, James (1997). “Building Hypertext Using Information Retrieval.” In: Information Processing & Management 33(2) (1997), 145-159.

ALTHEIM, Murray et al. (2000). Modularization of XHTML™. *World Wide Web Consortium* Working Draft, Januar 2000, *http://www.w3.org/TR/xhtml-modularization.*

ALTHEIM, Murray; MCCARRON, Shane P. (2000A). XHTML™ – Module-Based XHTML. *World Wide Web Consortium* Working Draft, Januar 2000, *http://www.w3.org/TR/xhtml11*.

ALTHEIM, Murray; MCCARRON, Shane P. (2000B). Building XHMTL™ Modules. *World Wide Web Consortium* Working Draft, Januar 2000, *http://www.w3.org/TR/xhtml-building.*

ANDERSON, Kenneth M.; TAYLOR Richard N.; WHITEHEAD, E. James, Jr. (1998). “A Critique of the Open Hypermedia Protocol.” In: Journal of Digital Information 1(2) (1998), *http://jodi.ecs.soton.ac.uk/Articles/v01/i02/Anderson/*.

APPARAO, Vidur et al. (edd.) (1998). Document Object Model (DOM) Level 1 Specification Version 1.0. *World Wide Web Consortium* Recommendation, Oktober 1998, *http://www.w3.org/TR/REC-DOM-Level-1*.

APPARAO, Vidur; Glazman, Daniel; Wilson, Chris (edd.) (1999). Behavioral Extensions to CSS. *World Wide Web Consortium* Working Draft, August 1999, *http://www.w3.org/TR/becss.*

APPELT, Wolfgang; MAMBREY, Peter (1999). “Experiences with the BSCW Shared Workspace System as the Backbone of a Virtual Learning Environment for Students.” In: Proc. World Conference on Educational Multimedia, Hypermedia and Telecommunications, ED-MEDIA 99, Seattle, Juni 1999.

ARNOLD, Ken et al. (1999). The Jini™ Specification. Reading et al.: Addison Wesley Longman [The Jini™ Technology Series].

ARNOLD, Ken; GOSLING, James (1998[2]). The Java Programming Language. Reading et al.: Addison Wesley Longman [The Java Series].

ASSOCIATION OF AMERICAN PUBLISHERS [AAP] (ed.) (2000). Guidelines for Online Information Exchange (ONIX). Version 1.0, Januar 2000. Washington/DC: AAP, *http://www.publishers.org/onix.html*.

AUSTIN, John L. (1979). Zur Theorie der Sprechakte (How to do things with words). Stuttgart: Reclam [= Reclam Universal-Bibliothek Nr. 9396].

AYARS, Jeff et al. (2000). Synchronized Multimedia Integration Language (SMIL) Boston Specification. *World Wide Web Consortium* Working Draft, Februar 2000, *http://www.w3.org/TR/smil-boston.*

BAEZA-YATES, Ricardo; RIBEIRO-NETO, Berthier (1999). Modern Information Retrieval. Harlow et al.: The ACM Press/The MIT Press.

BALDONADO, Michelle et al. (1998). “An Extensible Constructor Tool for the Rapid, Interactive Design of Query Synthesizers.” In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1998). Proc. Third ACM Conference on Digital Libraries (DL ’98), Juni 1998, Pittsburgh/PA, 19-28.

BALDONADO, Michelle; CHANG, Chen-Chuan K.; GRAVANO, Luis (1997). “Metadata for Digital Libraries: Architecture and Design Rationale.” In: Proc. Second ACM Conference on Digital Libraries (DL ’97), Juni 1997, Philadelphia/PA, 47-56.

BARKER, Donald I. (1994). “A Technological Revolution in Higher Education.” In: Journal of Educational Technology Systems 23(2) (1994-95), 155-168.

BARKER, Philip (1992). „Electronic Books and Libraries of the Future.“ In: Electronic Library 10(3) (1992), 139-149.

BARKER, Philip; MANJI, Karim (1991). "Designing electronic books." In: Educational and Training Technology International 28(4) (1991), 273-280.

BARNARD, David T.; IDE, Nancy M. (1997). "The Text Encoding Initiative: Flexible and Extensible Document Encoding." In: Journal of the American Society for Information Science 48(7) (1997), 622-628.

BEHME, Henning; MINTERT, Stefan (1998). XML in der Praxis. Bonn et al.: Addison-Wesley.

BELKIN, Nicholas J.; ODDY, R. N.; BROOKS, H. M. (1982). "Ask for Information Retrieval: Part I. Background and Theory." In: Journal of Documentation 38 (1982), 61-71.

BENN, Wolfgang; GRINGER, Ingo (1999). „Zugriff auf Datenbanken über das World Wide Web." In: Informatik Spektrum 21(1), 1-8.

BERG, Cliff (1999). "The State of Java Application Middleware, Part I." In: JavaWorld, März 1999, *http://www.javaworld.com/javawolrd/jw-03-1999/jw-03-middleware.html*.

BERGHEL, Hal (1999). "Value-Added Publishing." In: CACM 42(1) (1999), 19-23.

BERNERS-LEE, Tim (1997A). Metadata Architecture. *World Wide Web Consortium* Personal View, 1997, *http://www.w3.org/DesignIssues/MetaData.html.*

BERNERS-LEE, Tim (1997B). W3C Data Formats. *World Wide Web Consortium* Note, October 1997, *http://www.w3.org/TR/NOTE-rdfarch.html*.

BERNERS-LEE, Tim (1998A). Style Guide for Online Hypertext. *World Wide Web Consortium*, 1998, *http://www.w3.org/Provider/Style/All.html*.

BERNERS-LEE, Tim (1998B). Why RDF Model is Different from the XML Model. *World Wide Web Consortium*, September 1998, *http://www.w3.org/DesignIssues/RDF-XML.html*.

BERNERS-LEE, Tim (1999A). "Challenges of the Second Decade." Keynote Address, Eighth World Wide Web Conference, Canberra, Mai 1999, *http://www.w3.org/Talks/1999/05/www8-tbl/Overview.html*.

BERNERS-LEE, Tim (1999B). Weaving the Web. Unter Mitarbeit von Mark FISCHETTI. San Francisco: HarperSanFrancisco.

BERNERS-LEE, Tim et al. (1994). "The World Wide Web." In: CACM 37(8) (1994), 76-82.

BERNERS-LEE, Tim; FIELDING, Roy T.; FRYSTYK, Henrik (1996). Hypertext Transfer Protocol – HTTP/1.0. Internet Request for Comments N° 1945, Internet Architecture Board, Mai 1996.

BERNERS-LEE, Tim; FIELDING, Roy T.; MASINTER, L. (1998). Uniform Resource Identifiers (URI): Generic Syntax. Internet Request for Comments N° 2396, Internet Architecture Board, August 1998.

BERNSTEIN, Mark (1998). "Patterns of Hypertext." In: GRØNBÆK, Kaj; MYLONAS, Elli; SHIPMAN, Frank M., III. (1998). Hypertext 98. Proc. of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh/PA, Juni 1998. New York: ACM, 21-29.

BERNSTEIN, Philip A. (1996). "Middleware: A Model for Distributed System Services." In: CACM 39(2) (1996), 86-98.

BERRUT, Catherine; MULHEM, Philippe; BOUCHON, Pascal (1995). „Modelling and Indexing Medical Images: The RIME Approach." In: KUHLEN, Rainer; RITTBERGER, Marc (edd. (1995). Proc. HIM'95 Hypertext – Information Retrieval – Multimedia. Konstanz: UVK Informationswissenschaft, 105-115.

BIER, Eric A.; GOODISMAN, Aaron (1990). „Documents as User Interfaces." In: FURUTA, Richard (ed.) (1990). EP 90. Proc. of the International Conference on Electronic Publishing, Document Manipulation and Typography. CUP, 249-261.

BIRON, Paul F.; MALHOTRA, Ashok (edd.) (1999). XML Schema Part 2: Datatypes. *World Wide Web Consortium* Working Draft, Dezember 1999, *http://www.w3.org/TR/xmlscxhema-2/*.

BLACKWELL, J. (1983) „The Impact of Electronic Publishing." In: Electronic Publishing Review 3(4) (1983), 281-302.

BLACKWELL, J. (1984). „The Impact of Electronic Publishing: Review of Comments on First Draft." In: Electronic Publishing Review 4(4) (1984) 289-298.

BLAIR, David C. (1990). Language and Representation in Information Retrieval. Amsterdam et al.: Elsevier.

BÖHME, Timo (1999). Agentensysteme – aktueller Entwicklungsstand und Konzeption eines universellen News-Watcher-Agent. Diplomarbeit, Universität Leipzig, Fakultät für Mathematik und Informatik, Institut für Informatik, März 1999, *http://dol.uni-leipzig.de/pub/1999-37*.

BOLES, Dietrich (1995). „Elektronisches Publizieren – Autorensysteme und Arbeitsumgebungen für Autoren." In: Nachrichten für Dokumentation 46(5) (1995), 273-282.

BOLES, Dietrich et al. (1998A). "Das MeDoc-System – ein elektronischer Publikations- und Nachweisdienst für die Informatik." In: Informatik Forschung & Entwicklung 13 (1998), 110-121.

BOLES, Dietrich et al. (1998B). "Objektorientierte Multimedia-Softwareentwicklung. Vom UML-Modell zur Director-Anwendung am Beispiel virtueller naturwissenschaftlicher Labore." In: APPELRATH, Hans-Jürgen; BOLES, Dietrich; MEYER-WEGENER, Klaus (edd.) (1998). Proc. GI Workshop Multimediasysteme, Magdeburg, September 1998, *http://www-is.informatik.uni-oldenburg.de/~dibo/paper/gi98mm/*.

BOLES, DIETRICH et al. (1998C). Datenverwaltung im Multimediaprojekt Physikalisches Praktikum. Multimediaprojekt Physikalisches Praktikum, Arbeitsbericht O2, OFFIS Oldenburg, Februar 1998.

BOLL, Susanne; KLAS, Wolfgang; WESTERMANN, Utz (1999A). "A Comparison of Multimedia Document Models Concerning Advanced Requirements." Universität Ulm, Fakultät für Informatik, Technischer Bericht Nº 99-01.

BOLL, Susanne; KLAS, Wolfgang; WESTERMANN, Utz (1999B). "Multimedia Document Models – Sealed Fate or Setting Out for New Shores?" In: Proc. ICMCS '99. 1999 IEEE International Conference on Multimedia Computing and Systems, Florenz, Juni 1999, Bd. 1, 604-610.

BOLL, Susanne; KLAS, Wolfgang; WESTERMANN, Utz (1999C). "Exploiting ORDBMS Technology to Implement the ZYX Data Model for Multimedia Documents and Presentations." In: Proc. BTW '99. GI-Fachtagung Datenbanksysteme in Büro, Technik und Wissenschaft, Freiburg, März 1999.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar (1999). The Unified Modeling Language User Guide. Reading/MA et al.: Addison Wesley Longman [The Object Technology Series].

BOOKSTEIN, Abraham (1983). „Information Retrieval: A Sequential Learning Process." In: Journal of the American Society for Information Science 34(5) (1983), 331-342.

BORENSTEIN, N.; FREED, N. (1993). MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies. Internet Request for Comments N° 1521, Internet Architecture Board, September 1993.

BORGMANN, Christine L. (1999). "What are Digital Libraries? Competing Visions." In: Information Processing & Mangement 35 (1999), 227-243.

BORK, Alfred; BRITTON, David R., Jr. (1998). „The Web Is not Yet Suitable for Learning." In: IEEE Computer 31(6) (1998), 115 f.

BORMANN, U.; BORMANN, C. (1991). "Offene Bearbeitung multimedialer Dokumente." In: Informatik-Spektrum 7(2) (1991), 249-260.

BÖRSENVEREIN DES DEUTSCHEN BUCHHANDELS – ARBEITSKREIS *ELEKTRONISCHES PUBLIZIEREN* (1994). Leipziger Empfehlungen zum elektronischen Publizieren, *http://www.darmstadt.gmd.de/BV/leipz.html*

BOS, Bert et al. (edd.) (1998). Cascading Style Sheets, level 2. CSS2 Specification. *World Wide Web Consortium* Recommendation, 12. Mai 1998 [REC-CSS2-19980512], *http://www.w3.org/TR/REC-CSS2*.

BOSAK, Jon (1997). „XML, Java, and the Future of the Web." In: World Wide Web Journal 2(4) (1997), 219-228.

BOTAFAGO, Rodrigo A.; RIVLIN, Ehud; SHNEIDERMAN, Ben (1992). „Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics." In: ACM Transactions on Information Systems 10(2), 142-180.

BOUGHANEM, M.; SOULE DUPUY, C. (1994). "Query Expansion and Neural Network." In: Proc. RIAO '94. Intelligent Multimedia Information Retrieval Systems and Management. New York, Oktober 1994, 519-532.

BOURRET, Ronald et al. (1999). Document Definition Markup Language (DDML) Specification, Version 1.0. *World Wide Web Consortium* Note, Januar 1999, *http://www.w3.org/TR/NOTE-ddml*.

BOUTHILLIER, Larry (1998). "Synchronized Multimedia on the Web." In: Web Techniques Magazine 3(9) 1998, *http://www.webtechniques.com/1998/09/bouthillier/bouthillier.shtml.*

BRADLEY, Neil (2000[2]). The XML Companion. Harlow et al.: Addison-Wesley.

BRAY, Tim; DEROSE, Steve (1997). „Extensible Markup Language (XML). Part 2: Linking." In: World Wide Web Journal 2(4) (1997), 67-82.

BRAY, Tim; FRANKSTON, Charles; MALHOTRA, Ashok (1998). Document Content Description for XML. *World Wide Web Consortium* Note, Juli 1998, *http://www.w3.org./TR/NOTE-DCD.*

BRAY, Tim; HOLLANDER, Dave; LAYMAN, Andrew (1999). Namespaces in XML. *World Wide Web Consortium* Recommendation, Januar 1999, *http://www.w3.org/TR/REC-xml-names*.

BRAY, Tim; PAOLI, Jean; SPERBERG-MCQUEEN, C. M. (edd.) (1998). Extensible Markup Language (XML) 1.0. *World Wide Web Consortium* Recommendation, Januar 1998, *http://www.w3.org/TR/REC-xml.html*.

BRENNER, Walter; ZARNEKOW, Rüdiger; WITTIG, Hartmut (1998). Intelligente Softwareagenten. Grundlagen und Anwendungen. Berlin et al.: Springer.

BRERETON, Pearl; BUDGEN, David; HAMILTON, Geoff (1998). „Hypertext: The Next Maintenance Mountain." In: IEEE Computer 31(12) (1998), 49-55.

BRICKLEY, Dan; GUHA, R.V. (edd.) (1999). "Resource Description Framework (RDF) Schema Specification." *World Wide Web Consortium* Proposed Recommendation, März 1999, *http://www.w3.org/TR/WD-rdf-schema.*

BRINGHURST, Robert (1996[2]). The Elements of Typographic Design. Point Roberts/WA & Vancouver/BC: Hartley & Marks.

BRÖRKENS, Mark et al. (1998). Evaluierungsbericht. Drei Realisierungsalternativen für das Multimediaprojekt *Physikalisches Praktikum.* OFFIS Oldenburg, Februar 1998.

BRÜGGEMANN-KLEIN, Anne (1993A). "Regular Expressions into Finite Automata." In: Theoretical Computer Science 120 (1993), 197-213 [zugl. Universität Freiburg, Institut für Informatik, Technical Report No. 46, Oktober 1993, *ftp://ftp.informatik.uni-freiburg.de/documents/reports/report46/report46.ps.gz.*

BRÜGGEMANN-KLEIN, Anne (1993B). "Unambiguity of Extended Regular Expressions in SGML Document Grammars." Universität Freiburg, Institut für Informatik, Technical Report No. 46, Oktober 1993, *ftp://ftp.informatik.uni-freiburg.de/documents/reports/report33/report33.ps.gz.*

BRUSILOVSKY, Peter (1999). "Adaptive and Intelligent Technologies for Web-based Education." In: Künstliche Intelligenz 4/99, 19-25.

BRYAN, Martin (1998). An Author's Guide to the Standard Generalized Markup Language. Wokingham et al.: Addison-Wesley [unveränderter ND 1992].

BULLINGER, Hans-Jörg (1999). „Neue Geschäftsfelder für Verlage im Internet." In: Bullinger, ENGELBACH & Klostermann (1999), 15-39.

BULLINGER, Hans-Jörg; ENGELBACH, Wolf; KLOSTERMANN, Tanja (edd.) (1999). Verlage im Netz. Produkt- und Prozeßinnovation für neue Geschäftsfelder im Internet für Fachverlage. Proc. IAO-Forum, September 1999. Stuttgart: Fraunhofer IRB Verlag.

BURGER, Cora (1997). Groupware. Kooperationsunterstützung für verteilte Anwendungen. Heidelberg: dpunkt.

BURGER, Cora; MECKLENBURG, Rolf; ROTHERMEL, Kurt (1998). "Internetbasiertes Lernen – Interaktive Animation von Kommunikationsprotokollen auf generischem Wege." In: CLAUS (1998), 176-185.

BURGER, Jeffrey (1996). „Erfolgsbausteine. Die Multimedia-Produktion als Bauherrenmodell." In: Screen Multimedia Januar 1996, 16-24.

BURNARD, LOU (1994). Text Encoding and Information Interchange. Tutorial Material, SIGIR '94, Dublin: Dublin City University [17th International Conference on Research and Development in Information Retrieval].

BUSH, Vannevar (1945). „As We May Think." In: The Atlantic Monthly 76(1) (1945), 101-108 [*http://ccat.sas.upenn.edu/jod/texts/vannevar.bush.html*].

BÜTTEMEYER, Wilhelm (1995). Wissenschaftstheorie für Informatiker. Heidelberg et al.: Spektrum Akademischer Verlag.

CAGLAYAN, Alper K.; HARRISON, Colin G. (1998). Intelligente Software-Agenten. Grundlagen, Technik und praktische Anwendung im Unternehmen. München & Wien: Hanser.

CALDER, Bart; SHANNON, Bill (1999). JavaBeans Activation Framework Specification Version 1.0a. Palo Alto/CA: Sun Microsystems, Mai 1999, *http://java.sun.com/beans/glasgow/JAF-1.0.pdf*.

CALLAN, James P.; LU, Zhihong; CROFT, W. Bruce (1995). "Searching Distributed Collections with Inference Networks." In: SIGIR '95. Proc. 18th International Conference on Research and Development in Information Retrieval, Seattle/WA, Juli 1995, 21-28.

CAPROTTI, O.; CARLISLE, D. P.; COHEN, A. M. (1999). The OpenMath Standard, Version 0.3. The OpenMath Consortium, Esprit Project 24969, August 1999, *http://www.nag.co.uk/projects/OpenMath/omstd/omstd.pdf*.

CARD, Stuart K.; MACKINLAY, Jock D.; SHNEIDERMAN, Ben (edd.) (1999). Readings in Information Visualization. San Francisco/CA: Morgan Kaufman.

CARD, Stuart K.; MORAN, T.P.; NEWELL, Alan. (1983). The Psychology of Human-Computer-Interaction. Hillsdale/NJ: Lawrence Erlbaum.

CARROLL, John M. (ed.) (1991). Designing Interaction. Psychology at the Human-Computer Interface. Cambridge University Press.

CATENAZZI, Nadia; SOMMARUGA, Lorenzo (1994). „Hyper-Book: a Formal Model for Electronic Books." In: Journal of Documentation 50(4) 1994, 316-32.

ÇELIK, Tantek; LINSS, Peter; KUWAMOTO, Sho (edd.) (1999). User Interface for CSS3. *World Wide Web Consortium* Working Draft, September 1999, *http://www.w3.org/TR/css3-userint.*

CHAN, Patrick (1999). The Java Developers Almanac 1999. Reading/MA et al.: Addison Wesley Longman [The Java Series].

CHANG, Shi-Kuo; HASSANEIN, Ehab; HSIEH, Chung-Yuan (1998). "A Multimedia Micro-University." In: IEEE Multimedia 5(3) (1998), 60-68.

CHEN, Chaomei (1997). "Writing with Collaborative Hypermedia: Analysis and Modeling." In: Journal of the American Society for Information Science 48(11) (1997), 1049-1066.

CHEONG, Fah-Chun (1996). Internet Agents. Spiders, Wanderers, Brokers, and Bots. Indianapolis/IN: New Riders Publishing.

CHISHOLM, Wendy; VANDERHEIDEN, Gregg; JACOBS, Ian (edd.) (1999A). Web Content Accessibility Guidelines 1.0. *World Wide Web Consortium* Working Draft, Mai 1999, *http://www.w3.org/WAI/AU/WAI-WEBCONTENT.*

CHISHOLM, Wendy; VANDERHEIDEN, Gregg; JACOBS, Ian (edd.) (1999B). Techniques for Web Content Accessibility Guidelines 1.0. *World Wide Web Consortium* Working Draft, Mai 1999, *http://www.w3.org/WAI/AU/WAI-WEBCONTENT-TECHS.*

CHRISTODOULOU, S. P.; STYLIARAS, G. D. & PAPATHEODOROU, T. S. (1998). "Evaluation of Hypermedia Application Development Management Systems." In: GRØNBÆK, Kaj; MYLONAS, Elli; SHIPMAN, Frank M., III. (1998). Hypertext 98. Proc. of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh/PA, Juni 1998. New York: ACM, 1-10.

CHUA, Tat-Seng; RUAN, Li-Qun (1995). „A Video Retrieval and Sequencing System." In: ACM Transactions on Information Systems 13(4) (1995), 373-407.

CLARK, James (1997). Comparison of SGML and XML. *World Wide Web Consortium* Note, Dezember 1997, *http://www.w3.org/TR/NOTE-sgml-xml-971215.*

CLARK, James (edd.) (1999). XSL Transformations (XSLT) Version 1.0. *World Wide Web Consortium* Recommendation, November 1999, *http://www.w3.org/TR/xslt.*

CLAUS, Volker (ed.) (1998). Informatik und Ausbildung. Proc. GI-Fachtagung Informatik und Ausbildung 1998, Stuttgart März/April 1998. Berlin et al.: Springer.

COLAN, Mark (1998). InfoBus 1.2 Specification. Palo Alto/CA: Sun Microsystems. *http://java.sun.com/beans/infobus/infobus1.2.pdf*.

CONALLEN, Jim (1999). "Modeling Web Applications Architectures with UML." In: CACM 42(10) (1999), 63-70.

CONKLIN, J. (1987). „Hypertext – an Introduction and Survey.“ In: IEEE Computer 20 (1987), 17 ff.

CONNOLLY, Dan (1998). „Let a Thousand Flowers Bloom.“ In: IEEE Internet Computing 2(2) 1998, 22-31 [Interview mit Dan CONNOLLY].

CONNOLLY, Dan; BERNERS-LEE, Tim et al. (1999). „Naming and Addressing: URIs, URLs, ...“ *World Wide Web Consortium*, September 1999, *http://www.w3.org/Addressing/* [Leitseite des W3C zu Fragen der Adressierung].

CONNOLLY, Dan; KHARE, Rohit; RIFKIN, Adam (1997). “The Evolution of Web Documents: The Ascent of XML.” In: World Wide Web Journal 2(4) (1997), 119-128.

CONSORTI, F.; MERIALDO; SINDONI, G. (1996). “Metadata Reference Model for Medical Documentation: A Hypermedia Proposal.” In: Proc. First IEEE Metadata Conference, April 1996, Silver Spring/MA, *http://www.computer.org/conferen/meta96/sindoni/ieee.html*.

COOPER, Alan (1995). About Face: The Essentials of User Interface Design. Foster City/CA: IDG Books.

COVER, Robin (1998). The Essence and Quintessence of XML. Retrospects and Prospects. *http://www.oasis-open.org/html/essence_of_xml.html.*

COVER, Robin (1999). The SGML/XML Web Page. September 1999, *http://www.oasis-open.org/cover/sgml-xml.html*

COWAN, John; MEGGINSON, David (edd.) (1999). XML Information Set. World Wide Web Consortium Working Draft, Dezember 1999, *http://www.w3.org/TR/xml-infoset*.

CRAVENER, Patricia (1998). „Education on the Web: A Rejoinder.“ In: IEEE Computer 31(9) (1998), 107f.

CROZAT, Stéphane; HÛ, Olivier; TRIGANO, Philippe. “A Method for Evaluating Multimedia Learning Software.“ In: Proc. ICMCS ’99. 1999 IEEE International Conference on Multimedia Computing and Systems, Florenz, Juni 1999, Bd. 1, 720-725.

CUMMINGS, Anthony M. et al. (1992). University Libraries and Scholarly Communication. A Study Prepared for The Andrew W. Mellon Foundation. The Association of Research Libraries, November 1992, *http://www.arl.org/scomm/mellon/index.html*.

DANIEL, Ron Jr.; LAGOZE, Carl (1997). “Extending the Warwick Framework: From Metadata Containers to Active Digital Objects.” In: D-Lib Magazine, November 1997, *http://www.dlib.org/dlib/november97/daniel/11daniel.html*.

DATE, C. J.; DARWEN, Hugh (1997[4]). A Guide to the SQL Standard. Reading/MA: Addison Wesley Longman.

DAVIDSON, Andrew et a. (1999): Schema for Object-Oriented XML 2.0. *World Wide Web Consortium* Note, Juli 1999, *http://www.w3.org/TR/NOTE-SOX*.

DAVIES, G.; MAURER, H.; PREECE, J. (1991). „Presentation metaphors for very large hypermedia systems.“ In: Journal of Microcomputer Applications 14(2) (1991), 105-116.

DAVIS, Hugh (1995). “To Embed or Not to Embed ...” In: CACM 38(6) (1995), 108f.

DAVIS, Jim (1995). Annotation Systems. Cornell University, Design Research Institute, *http://dri.cornell.edu/pub/davis/Annotation/others.html.*

DE BRA, Paul; HOUBEN, Gerrt-Jan; WU, Hongjing (1999). „AHAM: A Dexter-based Reference Model for Adaptive Hypermedia.“ In: Proc. Tenth ACM Conference on Hypertext and Hypermedia, Darmstadt, Februar 1999, 147-168.

DE DIANA, I. P. F.; WHITE, T. N. (1994). „Towards an Educational SuperInterface.“ In: Journal of Computer Assisted Learning (1994) (2), 93-103.

DEACH, Stephen (edd.) (1999). Extensible Stylesheet Language (XSL). *World Wide Web Consortium* Working Draft, April 1999, *http://www.w3.org/TR/WD-xsl.*

DECEMBER, John (1994). „Electronic Publishing on the Internet: New Traditions, New Choices“ In: Educational Technology 34(7) 1994, 32-36.

DEGEN, Helmut (1996). „Multimediale Gestaltbereiche als Grundlage für Entwurfswerkzeuge in multimedialen Entwicklungsprozessen.“ In: KRAUSE, Jürgen et al. (edd.) (1996). Herausforderungen an die Informationswirtschaft. Proc. 5. Int. Symposium für Informationswissenschaft, Berlin, Oktober 1996. Konstanz: UVK Informationswissenschaft, 213-226.

DEMENTHON, Daniel; KOBLA, Vikrant; DOERMANN, David (1998). "Video Summarization by Curve Simplification." In: EFFELSBERG, Wolfgang; SMITH, Brian C. (edd.) (1998). ACM Multimedia '98. Proc. 6th ACM International Multimedia Conference, Bristol, September 1998. New York: ACM, 211-218.

DEMPSEY, Lorcan; WEIBEL, Stuart (1996). "The Warwick Metadata Workshop: A Framework for the Deployment of Resource Description." In: D-Lib Magazine Juli/August 1996, *http://www.dlib.org/july96/07weibel.html*.

DEROSE, Steve et al. (edd.) (2000). XML Linking Language (XLink). *World Wide Web Consortium* Working Draft, Januar 2000, *http://www.w3.org/TR/xlink.*

DEROSE, Steve; DANIEL, Ron, Jr.; MALER, Eve (edd.) (1999). XML Pointer Language (XPointer). *World Wide Web Consortium* Working Draft, Dezember 1999, *http://www.w3.org/TR/WD-xptr*.

DESIGN SCIENCE (ed.) (1999). MathType Mathematical Equation Editor User Manual. Version 4.: Long Beach/CA: Design Science [*http://www.mathtype.com*].

DILLON, Andrew (1994). Designing Usable Electronic Text. Ergonomic Aspects of Human Information Usage. London & Bristol/PA: Taylor & Francis.

DILLON, Andrew (1997). "Extending SGML to Accommodate Database Functions: A Methodological Overview." In: Journal of the American Society for Information Science 48(7) (1997), 629-637.

DREILINGER, Daniel E. (1996). "Description and Evaluation of a Meta-Search Agent." M. Sc. Thesis, Colorado State University, Department of Computer Science, Fort Collins/CO, Oktober 1996.

DUBLIN CORE METADATA INITIATIVE (ed.) (1997). "Dublin Core Metadata Element Set: Reference Description." *http://www.purl.org/dc/about/element_set.htm*.

DURAND, David G. (1998). „Hypertext-Related Standards Efforts." In: SIGLINK Newsletter 6(2) (1998), 13-15.

DWIGHT, Jeffry; ERWIN, Michael et al. (1996). Special Edition Using CGI. Indianapolis/IN: Que Corporation.

DYSON, Peter E. „Publishing On the Internet for Fun and Profit." In: Seybold Report on Desktop Publishing (April 1994) 8(8), 3-14.

DZIDA, Wolfgang (1983). „Das IFIP-Modell für Benutzerschnittstellen." In: Office Management Sonderheft 1983, 6-8.

DZIDA, Wolfgang (1985). „Ergonomische Normen für die Dialoggestaltung." In: BULLINGER, H. (ed.) (1985). Software-Ergonomie '85 – Mensch-Computer Interaktion. Stuttgart: Teubner 1985 [= Berichte des German Chapter of the ACM Bd. 24], 430-444.

DZIDA, Wolfgang (1994). „Qualitätssicherung durch software-ergonomische Normung." In: EBERLEH, OBERQUELLE & OPPERMANN (1994), 373-406.

EBERLEH, Edmund; OBERQUELLE, Horst; OPPERMANN, Reinhard (edd.) (1994[2]). Einführung in die Software-Ergonomie. Gestaltung graphisch-interaktiver Systeme: Prinzipien, Werkzeuge, Lösungen. Berlin & New York: Walter de Gruyter [= Mensch Computer Kommunikation Grundwissen Bd. 1/2].

EDDON, Guy (1999). „COM+: The Evolution of Component Services." In: IEEE Computer 32(7) (1999), 104-106.

EISENBERG, Michael B. (1994). "Free from the Constraints of Space and Time: Considering the Opportunities and Challenges for Electronic Publishing." In: Educational Technology 34(7) (1994), 59-64.

ELFREICH, Sigurd; SCHLATTMANN, Marco (1998). Werkzeuge für die Offline-Recherche im elektronischen Buch. Multimediaprojekt Physikalisches Praktikum, Arbeitsbericht O3, OFFIS Oldenburg, März 1998.

ENDRES, Albert et al. (1999). "Professionelle Dienste und kreative Konzepte für die Versorgung mit Informatik-Wissen." In: Informatik Spektrum 22(2) (1999), 136-145.

ENDRES, Albert; FUHR, Norbert (1998). "Students Access Books and Journals through MeDoc." In: CACM 41(4) (1998), 76f.

ENGELBART, Douglas (1995). "Toward Augmenting the Human Intellect and Boosting our Collective IQ." In: CACM 38(8) (1995), 30-33.

ERFLE, Robert (1993). „Specification of Temporal Constraints in Multimedia Documents Using HyTime.“ In: Electronic Publishing: Origination, Dissemination and Design 6(4) (1993), 397-411.

ERFLE, Robert (1994). „HyTime as the Multimedia Document Model of Choice.“ In: Proc. International Conference on Multimedia Computing and Systems. Boston, Mai 1994. Los Alamitos/CA: IEEE Computer Society Press, 445-454.

FALCHUK, Benjamin; KARMOUCH, Ahmed (1998). „Visual Modeling for Agent-Based Applications.“ In: IEEE Computer 31(12) (1998), 31-38.

FANKHAUSER, Peter; MARCHIORI, Massimo; ROBIE, Jonathan (2000). “ XML Query Requirements.” World Wide Web Consortium Working Draft, Januar 2000, *http://www.w3.org/TR/xmlquery-req*.

FARANCE, Frank; TONKEL, Joshua (1998). LTSA Specification. Learning Technology Systems Architecture. Version 4, Mai 1998. Farance Inc, Edutool Division, *http://www.edutool.com/ltsa*.

FAUSEY, Jon; SHAFER, Keith (1997). “All My Data Is in SGML. Now What?” In: Journal of the American Society for Information Science 48(7) (1997), 638-643.

FERRAIOLO, Jon et al. (ed.) (1999). Scalable Vector Graphics (SVG) 1.0 Specification. W3C Working Draft 12 August 1999, *http://www.w3.org/TR/SVG*.

FIELDING, Roy T. (1998). „Web-Based Development of Complex Information Products.“ In: CACM 41(8), 84-92.

FIELDING, Roy T. et al. (1999). Hypertext Transfer Protocol – HTTP/1.1. Internet Request for Comments N° 2616, Internet Architecture Board, Juni 1999.

FININ, Tim et al. (1994). „KQML as an Agent Communication Language.“ In: CIKM ’94. Proc. of the Third International Conference on Information and Knowledge Management, Gaithersburg/MD, November/Dezember 1994, 456-463.

FIPA [Foundation for Intelligent Physical Agents] (1998). FIPA 97 Specification, Version 2.0. Part 2. Agent Communication Language, Oktober 1998. Genf: Foundation for Intelligent Physical Agents, *http://www.fipa.org*.

FIZ KARLSRUHE (1996). Kurzanleitung der Retrievalsprache Messenger. Version S96.3, Dezember 1996. Karlsruhe: FIZ Karlsruhe, STN Servicezentrum Europa.

FLYNN, Peter (1997). “W[h]ither the Web? The Extension or Replacement of HTML.” In: Journal of the American Society for Information Science 48(7) (1997), 614-621.

FOWLER, Martin; SCOTT, Kendall (1998[10]). UML Distilled. Applying the Standard Object Modeling Language. Reading/MA et al.: Addison Wesley Longman [The Object Technology Series].

FOX, Edward A. et al. (1995). „Introduction [to the Special Issue on Digital Libraries].“ In: CACM 38(4) (1995), 22-28.

Fox, Edward et al. (1999). „CRIM: Curricular Resources in Interactive Multimedia.“ In: ACM Multimedia ’99. Proc. 7th ACM International Multimedia Conference, Orlando, Oktober 1999. New York: ACM, 85-90.

FRAKES, William (1992A). “Introduction to Information Storage and Retrieval Systems.” In: FRAKES & BAEZA-YATES (1992), 1-12.

FRAKES, William (1992B). “Stemming Algorithms.” In: FRAKES & BAEZA-YATES (1992), 131-160.

FRAKES, William; BAEZA-YATES, Ricardo (edd.) (1992). Information Retrieval: Data Structures and Algorithms. Englewood Cliffs/NJ: P T R Prentice Hall.

FRANZ, Martin M.; MCCARLEY, J. Scott; ROUKOS, Salim (1999). “Ad hoc and Multilingual Retrieval at IBM.” In: VOORHEES, Ellen M.; HARMAN, Donna K. (edd.) (1999). The Seventh Text Retrieval Conference (TREC-7). Gaithersburg/MD: National Institute of Standards and Technology [= NIST Special Publications 500-242].

FRANZ, Michael (1998). „Java – Anmerkungen eines Wirth-Schülers.“ In: Informatik Spektrum 21(1) (1998), 23-26.

FREEMAN, Eric; Hupfer, Susanne; ARNOLD, Ken (1999). JavaSpaces™ Principles, Patterns and Practice. Reading/MA et al: Addison Wesley Longman [The Jini™ Technology Series].

FREIBICHLER, Hans (1997). „Werkzeuge zur Entwicklung von Multimedia.“ In: ISSING & KLIMSA (1997), 222-240.

FRÖHLICH, Peter; HENZE, Nicola; NEJDL, Wolfgang (1997). "Meta Modeling for Hypermedia Design." In: Proc. Second IEEE Metadata Conference, September 1997, Silver Spring/MA, *http://www.kbs.uni-hannover.de/papers/97/metadata/pfroehlich.html*.

FUCHSSTEINER, Benno; DRESCHER, Klaus; KEMPER, Andreas (1996). MuPAD User's Manual. Stuttgart: Teubner.

FÜNFSTÜCK, Falk; LISKOWSKY, Rüdiger; MEIßNER, Klaus (2000). "Softwarewerkzeuge zur Entwicklung multimedialer Anwendungen. Eine Übersicht." In: Informatik Spektrum 23(1) (2000), 11-25.

GAMMA, Erich et al. (1996). Entwurfsmuster. Elemente wiederverwendbarer objektorientierter Software. Bonn et al.: Addison Wesley Longman.

GARFINKEL, Simson; SPAFFORD, Gene (1997). "Cryptography and the Web." In: World Wide Web Journal 2(4) (1997), 113-126.

GARRAND, Timothy (1997). Writing for Multimedia. Entertaining, Education, Training, Advertising, and the World Wide Web. Boston et al.: Focal Press.

GAUS, Wilhelm (1995[2]). Dokumentations- und Ordnungslehre: Theorie und Praxis des Information Retrieval. Berlin et al.: Springer.

GENESERETH, Michael J. (1995). Knowledge Interchange Format Specification. Stanford University Logic Group, März 1995, *http://logic.stanford.edu/kif/specification.html*.

GENNUSA, Pamela L. (1999). „Evolution and Use of Generic Markup Languages." In: MÖHR & SCHMIDT (1999), 27-50.

GESCHKE, Dieter (ed.) (1998[11]). Physikalisches Praktikum. Stuttgart & Leipzig: Teubner.

GIORDANO, Richard (1994). „The Documentation of Electronic Texts Using Text Encoding Initiative Headers: An Introduction." In: Library Resources & Technical Services 38(4) (1994), 389-401.

GIRMES, Renate (1999B). "Lernsoftware analysieren. Das didaktische Können anderer erschließen." In: GIRMES 1999A, 15-42.

GIRMES, Renate (ed.) (1999A). Lehrdesign und Neue Medien: Analyse und Konstruktion. Münster et al.: Waxmann.

GIRMES, Renate et al. (1999). „Interaktives Lehrdesign - Eine Problemcollage." In: GIRMES 1999A: 219-239.

GLOOR, Peter A. (1993). „Hypermedia-Lernumgebungen für den Informatik-Unterricht." In: it+ti (Informationstechnik und technische Informatik) 35 (1993)(3), 18-26.

GLOOR, Peter A. (1997). Elements of Hypermedia Design: Techniques for Navigation & Visualization in Cyberspace. Basel et al.: Birkhäuser.

GOLAND, YARON et al. (1999). HTTP Extensions for Distributed Authoring – WebDAV. Internet Request for Comments N° 2518, Internet Architecture Board, Februar 1999.

GOLDBERG, Murray W.; SALARI, Sasan; SWOBODA, Paul (1996). „World Wide Web – Course Tool: An Environment for Building WWW-Based Courses." In: Proc. Fifth World Wide Web Conference, Paris, Mai 1996, *http://www5conf.inria.fr/fich_html/papers/P29/Overview.html*.

GOLDFARB, Charles et al. (1997). "A Reader's Guide to the HyTime Standard." HyTime Users' Group, *http://www.hytime.org/papers/htguide.html*.

GOLDFARB, Charles F. (1997). "SGML: The Reason Why and the First Published Hint." In: Journal of the American Society for Information Science 48(7) (1997), 656-661.

GOLDFARB, Charles F. (1999). „Future Directions in SGML/XML." In: MÖHR & SCHMIDT (1999), 3-25.

GOLDFARB, Charles F.; PRESCOD, Paul (1998). The XML Handbook. Upper Saddle River/NJ: Prentice Hall PTR.

GONG, LI (1999). Inside Java™ 2 Platform Security. Architecture, API Design, and Implementation. Reading/MA et al.: Addison Wesley Longman [The Java Series].

GORDON, Michael; PATHAK, Praveen (1999). „Finding Information on the World Wide Web: the Retrieval Effectiveness of Search Engines." In: Information Processing & Management 35 (1999), 141-180.

GOSLING, James; JOY, Bill; STEELE, Guy (1997). Java. Die Sprachspezifikation. Bonn et al.: Addison Wesley Longman [The Java Series].

GRAY, David et al. (1998). “Modern Languages and Microsoft’s Component Object Model.” In: CACM 41(5) (1998), 55-65.

GRØNBÆK, Kaj; WIIL, Uffe Kock (1998). ”Towards a Common Reference Architecture for Open Hypermedia.” In: Journal of Digital Information 1(2) (1998), *http://jodi.ecs.soton.ac.uk/Articles/v01/i02/Gronbak*.

GROSSMAN, David A. et al. (1997). “Integrating Structured Data and Text: A Relational Approach.” In: Journal of the American Society for Information Science 48(2) (1997), 122-132.

GUGLIELMO, Eugene J.; ROWE, Neil C. (1996). "Natural-Language Retrieval of Images Based on Descriptive Captions." In: ACM Transactions on Information Systems 14(3) (1996), 237-267.

GUNDERSON, Jon; JACOBS, Ian (edd.) (1999A). User Agent Accessibility Guidelines 1.0. *World Wide Web Consortium* Working Draft, Oktober 1999, *http://www.w3.org/WAI/AU/WAI-USERAGENT*.

GUNDERSON, Jon; JACOBS, Ian (edd.) (1999B). Techniques for User Agent Accessibility Guidelines 1.0. *World Wide Web Consortium* Working Draft, August 1999, *http://www.w3.org/WAI/AU/WAI-USERAGENT-TECHS*.

HABER, Cornelia; MEYER, Jochen; WEBER, Ricarda (1998). “Aufbau und Realisierung des Me-Doc-Volltextspeichers.” In: Informatik Forschung & Entwicklung 13 (1998), 122-131.

HALASZ, Frank; SCHWARTZ, Mayer (1994). „The Dexter Hypertext Reference Model.“ In: CACM 37(2) (1994), 28-38.

HALLA, Brian (1998). “How the PC Will Disappear.” In: IEEE Computer 31(12) (1998), 134-136.

HAMILTON, Graham (ed.) (1997). Java Beans™ API Specification. Mountain View/CA: Sun Microsystems.

HAMILTON, Graham; CATTELL, Rick; FISHER, Maydene (1998). JDBC. Datenbankzugriff mit Java. Bonn et al.: Addison Wesley Longman [The Java Series].

HAMMWÖHNER, Rainer (1997). Offene Hypertextsysteme: Das Konstanzer Hypertextsystem (KHS) im wissenschaftlichen und technischen Kontext. Konstanz: IVK Informationswissenschaft [= Schriften zur Informationswissenschaft Bd. 32].

HANDKE, Jürgen (1999). „Präsentation und Lernerunterstützung im wissenschaftlichen Lernsystem. The Interactive Introduction to Linguistics.“ In: LOBIN 1999A, 67-88.

HARDMAN, Lynda; BULTERMAN, Dick C.A.; VAN ROSSUM, Guido (1994). „Adding Time and Context to the Dexter Model.“ In: CACM 37(2), 51-62.

HARMAN, Donna (1992). “Ranking Algorithms.” In: FRAKES & BAEZA-YATES 1992: 363-392.

HAUPTMANN, Alexander G. (1999). „Integrating and Using Large Databases of Text, Images, Video, and Audio.“ In: Intelligent Systems 14(5) (1999), 34 f.

HEARST, Marti A. (1999). “User Interfaces and Visualization.” In: BAEZA-YATES & RIBEIRO-NETO (1999), 257-323 [Kap. 10].

HELANDER, Martin (ed.) (1988). Handbook of Human-Computer Interaction. Amsterdam et al.: North-Holland (Elsevier).

HENZE, Nicola et al. (1999). “Adaptive Hyperbooks for Constructivist Training.” In: Künstliche Intelligenz 4/1999, 26-31.

HERCZEG, Michael (1994). Software-Ergonomie. Grundlagen der Mensch-Maschine-Kommunikation. Bonn et al.: Addison-Wesley.

HEYER, Gerhard (1995). “Elements of a Natural Language Processing Technology.” In: HEYER & HAUGENEDER (1995), 15-32.

HEYER, Gerhard; HAUGENEDER, Hans (edd.) (1995). Language Engineering. Essays in Theory and Practice of Applied Natural Language Computing. Wiesbaden: Vieweg.

HEYER, Gerhard; WOLFF, Christian (edd.) (1998A). Linguistik und neue Medien. Proc. 10. Jahrestagung der GLDV. Wiesbaden: Deutscher Universitätsverlag.

HEYER, Gerhard; WOLFF, Christian (1998B). „Zur Relevanz linguistischer Pragmatik bei der Entwicklung von Multimediaanwendungen.“ In: HEYER & WOLFF (1998A), 15-22.

HEYER, Gerhard; WOLFF, Christian (1999). „Strukturierungsmethoden für Hypermediadokumente und ihre Umsetzung.“ In: LOBIN (1999A), 89-119.

HODES, Todd A.; KATZ, Randy H. (1999). “A Document-based Framework for Internet Application Control.“ In: Proc. 2nd Usenix Symposium on Internet Technologies and Systems, Boulder/CO, Oktober 1999.

HOFMANN, Martin; SIMON, Lothar (1995). Problemlösung Hypertext. Grundlagen – Entwicklung – Anwendungen. München: Hanser.

HOFMANN, Tino; WALTHER, Dirk; WOLFF, Christian (1998). Textkonvertierung, Formeldarstellung und Graphikaufbereitung. Multimediaprojekt Physikalisches Praktikum, Arbeitsbericht L4, Universität Leipzig, April 1998.

HÖNING, Sascha (1999). „Eine SGML-basierte bibliographische Datenbank für Nachschlagewerke.“ In MÖHR & SCHMIDT (1999), 123-144.

HORNECKER, Eva; SCHÄFER, Kai (1999). “Gegenständliche Modellierung virtueller Informationswelten.” In: AREND, Udo et al. (edd.) (1999). Proc. Software-Ergonomie ’99. Design von Informationswelten. Walldorf, März 1999. Stuttgart & Leipzig: Teubner [= Berichte des German Chapter of the ACM Bd. 53], 149-159.

HOSCHKA, Philip et al. (1998). Synchronized Multimedia Integration Language (SMIL) 1.0 Specification. *World Wide Web Consortium* Recommendation, Juni 1998, *http://www.w3.org/TR/REC-smil*.

HUC, Claude; LEVOIR, Thierry; NONON-LATAPIE, Michel (1997). “Metadata: Models and Conceptual Limits.” In: Proc. Second IEEE Metadata Conference, September 1997, Silver Spring/MA, *http://www.computer.org/conferen/ proceed/meta97/papers/chuc/chuc.html*.

HUGHES, Merlin et al. (1997). Java Network Programming. Greenwich/CT: Manning.

HUMMES, Jakob; KARSENTY, Alain; MERIALDO, Bernard (1997). “Active Annotations of Web Pages.” In: ALTON-SCHEIDL, Roland et al. (edd.) (1997). Voting, Rating, Annotation – Web4Groups and Other Projects: Approaches and First Experiences. Wien & München: Oldenbourg [= Schriftenreihe der Österreichischen Computer Gesellschaft, Bd. 104], *http://www.eurecom.fr/~hummes/docs/Web4Groups/w4g.ps.gz.*

IEEE LTSC (1999). Learning Object Metadata. IEEE Learning Technology Standards Committee (LTSC), Draft Document V. 3.6, 5. September 1999.

INKTOMI (1999). Search Engine Overview. Dezember 1999, *http://www.inktomi.com/products/portal/search/.*

ION, Patrick; MINER, Robert et al. (edd) (1999). Mathematical Markup Language (MathML™) 1.0.1 Specification. W3C Recommendation, Juli 1999, *http://www.w3.org/TR/REC-MathML*.

ISAKOWITZ, Tomás; BIEBER, Michael; VITALI, Fabio (1998). “Web Information Systems.” In: CACM 41(7) (1998), 78-80.

ISAKOWITZ, Tomás; STOHR, Edward A.; BALASUBRAMIAN, P. (1995). „RMM: A Methodology for Structured Hypermedia Design.“ In: CACM 38(8) (1995), 34-44.

ISHIKAWA, Masayasu et al. (1999). XHTML™ Basic. *World Wide Web Consortium* Working Draft, Dezember 1999, *http://www.w3.org/TR/xhtml-basic*.

ISO/IEC 10179:1996(E). Information technology – Processing Languages – Document Style Semantics and Specification Language (DSSSL) [*ftp://ftp.ornl.gov/pub/sgml/WG8/DSSSL/dsssl96b.pdf*].

ISO/IEC 10744:1997. Information technology — Hypermedia/Time-based Structuring Language (HyTime). 2. Auflage, Mai 1997 [*http://www.ornl.gov/sgml/wg8/docs/n1920/html/n1920.html*].

ISO/IEC JTC1/WG4 N1990 (1990). Information Technology – Text and Office Systems – Using SGML Public Identifiers for Specifying Data Notations, *http://www.ornl.gov/sgml/wg8/document/1990.htm*.

ISSING, Ludwig J. (1997). „Instruktionsdesign für Multimedia.“ In: ISSING & KLIMSA (1997), 194-220.

ISSING, Ludwig J.; KLIMSA, Paul (1997[2]). Information und Lernen mit Multimedia. Weinheim: Beltz Psychologie Verlags Union.

JABLONSKI, Stefan; BÖHM, Markus; SCHULZE, Wolfgang (edd.) (1997). Workflow-Management. Entwicklung von Anwendungen und Systemen. Heidelberg: dpunkt.

JACOBS, Paul S. (ed.) (1992). Text-based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval. Hillsdale/NJ: Lawrence Erlbaum.

JANSEN, Bernard J. et al. (1998). „Real Life Information Retrieval: A Study of User Queries On the Web." In: SIGIR Forum 32(1) (1998), 5-17.

JANSEN, Bernard J.; SPINK, Amanda; SARACEVIC, Tevko (2000). "Real Life, Real Users, and Real Needs: A Study and Analysis of User Queries on the Web." In: Information Processing & Management 36(2) (2000), 207-227.

JANSSEN, William C. (1999). "A »Next Generation« Architecture for HTTP." In: IEEE Internet Computing 3(1) (1999), 69-73.

JANSSEN, William C.; FRYSTYK NIELSEN, Henrik; SPREITZER, Mike (1998). HTTP-ng Architectural Model. *World Wide Web Consortium* Working Draft, Juli 1998, *http://www.w3.org/TR/WD-HTTP-NG-architecture*.

JOACHIMS, Thorsten; Mladenic, Dunja (1998). „Browsing-Assistenten, Tour Guides und adaptive WWW-Server." In: Künstliche Intelligenz 3/1998, 23-29.

JOHNSON, Mark (1999). „XML JavaBeans, Part 1: Make JavaBeans mobile and interoperable with XML." In: JavaWorld, Februar 1999, *http://www.javaworld.com/javaworld/jw-02-1999/jw-02-beans.html*.

JOHNSON, Mark (2000). „Programming XML in Java, Part 1. Create Java Apps with SAX Appeal." In: JavaWorld, März 2000, *http://www.javaworld.com/javaworld/jw-03-2000/jw-03-xmlsax.html*.

KAMPS, Thomas et al. (1999). „SGML für dynamische Publikationen: das Beispiel Fischer Weltalmanach." In: MÖHR & SCHMIDT (1999),173-192.

KEKÄLÄINEN, Jaana; JÄRVELIN, Kalervo (1998). "The Impact of Query Structure and Query Expansion on Retrieval Performance." In: SIGIR '98. Proc. 21st International Conference on Research and Development in Information Retrieval, Melbourne, August 1998, 130-137.

KERHERVÉ, Brigitte et al. (1996). "Metadata Modeling for Quality of Service Management in Distributed Multimedia Systems." In Proc. First IEEE Metadata Conference, April 1996, Silver Spring/MA, *http://www.computer.org/conferen/meta96/kerherve/kerherve.html*.

KESSLER, G.; SHEPARD, S. (1997). A Primer On Internet and TCP/IP Tools and Utilities. Internet Request for Comments N° 2151, Internet Architecture Board, Juni 1997.

KESSLER, Gary C.; ROSENBLAD, Ken; SHEPARD, Steven D. (1999). „The Web Can be Suitable for Learning." In: IEEE Computer 32(2) (1999), 114 f.

KHARE, Rohit (1999A). "Anatomy of a URL (and Other Internet-Scale Namespaces, Part 1)." In: Internet Computing 3(5) (1999), 78-81.

KHARE, Rohit (1999B). "What's in a Name? Trust. Internet-Scale Namespaces, Part II." In: Internet Computing 3(6) (1999), 80-84.

KIMBER, W. Eliot; WOODS, Julia A. (1997). "Application of HyTime Hyperlinks and Finite Coordinate Spaces to Historical Writing, Analysis, and Presentation." In: Journal of the American Society for Information Science 48(7) (1997), 603-613.

KIMBER, W. Elliott (1997). A Tutorial Introduction to SGML Architectures." ISOGEN International Corp., *http://www.isogen.com/papers/archintro.html*.

KIRSCH, Steve (1998). „Infoseek's Experiences Searching the Internet." In: SIGIR Forum 32(2) (1998), 3-7.

KISS, Tibor (1998). „RECALL – Demonstrating a Systems Architecture for Repairing Errors in Computer Aided Language Learning." In: HEYER & WOLFF (1998A), 23-31.

KLING, Rob; LAMB, Roberta (1996). "Analyzing Alternate Visions of Electronic Publishing and Digital Libraries." In: PEEK & NEWBY (1996), 17-54.

KNUDSEN, Jonathan (1998). Java Cryptography. Sebastopol/CA et al.: O'Reilly.

KNUTH, Donald E. (1986). The TEXbook. Providence/RI: American Mathematical Society & Reading/MA et al.: Addison-Wesley.

KOMMERS, Piet A. M.; FERREIRA, Alcindo F.; KWAK, Alex W. (1998). Document Management for Hypermedia Design. Berlin et al.: Springer.

KOPROWSKI, Gene (1998). „HTTP-Next Generation is in the Works.“ In: IEEE Computer 31(12) (1998), 18-20.

KORFHAGE, Robert R. (1997). Information Storage and Retrieval. New York et al.: Wiley.

KORPELA, Jukka (1998). „Lurching Toward Babel: HTML, CSS, and XML.“ In: IEEE Computer 31(7) (1998), 103-106.

KRAUSE, Jürgen (1992). „Intelligentes Information Retrieval. Rückblick, Bestandsaufnahme und Realisierungschancen.“ In: KUHLEN, Rainer (ed.) (1992). Experimentelles und praktisches Information Retrieval. Festschrift für Gerhard Lustig. Konstanz: Universitätsverlag [= Schriften zur Informationswissenschaft, Bd. 3], 35-58.

KRAUSE, Jürgen (1997). “Visual formalisms und »natural mapping«.“ In: KRAUSE, Jürgen; WOMSER-HACKER, Christa (edd.) (1997). Vages Information Retrieval und graphische Benutzungsoberflächen. Beispiel Werkstoffinformation. Konstanz: UVK Informationswissenschaft [= Schriften zur Informationswissenschaft, Bd. 28], 239-257.

KRIEGER, David; ADLER, Richard M. (1998). „The Emergence of Distributed Component Platforms.“ In: IEEE Computer 31(3) (1998), 43-53.

KROCK, Eric (1999). “Introduction to Cross-Browser, Cross-Platform, Backwardly Compatible JavaScript and Dynamic HTML.” Mountain View/CA: Netscape Communications Corporation, *http://developer.netscape.com/docs/technote/dynhtml/xbdhtml/xbdhtml.html.*

KRÜGER, Manfred (1999). „SGML – Praxis des langen Wegs.“ In: MÖHR & SCHMIDT (1999), 51-76.

KUHLEN, Rainer (1991). Hypertext. Ein nicht-lineares Medium zwischen Buch und Wissensbank. Berlin et al.: Springer.

KUHLEN, Rainer; RITTBERGER, Marc (edd.) (1995). Hypertext – Information Retrieval – Multimedia. Synergieeffekte elektronischer Informationssysteme. Proc. HIM ‘95, Konstanz. Konstanz: Universitätsverlag [= Schriften zur Informationswissenschaft Bd. 20].

LABROU, Yannis; FININ, Tim; PENG, Yun (1999). „Agent Communication Languages: The Current Landscape.“ In: IEEE Intelligent Systems 14(2) (1999), 45-52.

LADWIG, Enno (1999). “Beim Lesen erleuchtet.” In: Screen Business Online Oktober 1999, 102-105 [Test Rocket-E-Book].

LALIBERTE, Daniel; BRAVERMAN, Alan (1995). “A Protocol for Scalable Group and Public Annotations.” In: Computer Networks and ISDN Systems 27(6) (1995), 11-18 [= Proc. Third World Wide Web Conference, Darmstadt, April 1995, *http://www.igd.fhg.de/www/www95/proceedings/papers/100/scalable-annotations.html*].

LANDONI, Monica; GIBB, Forbes (1999). “The Importance of Visual Rhetoric in the Design and Production of Electronic Books: The Visual Book.” In: THAI-ETIS. Proc. European Symposium on Telematics, Hypermedia, and Artificial Intelligence in Education and Training for the New Professions in the Information Society, Varese, Juni 1999. Varese: Università degli Studi dell’ Insubria [o. S.].

LASSILA, Ora (1998). „Web Metadata: A Matter of Semantics.“ In: IEEE Internet Computing 2(4) (1998), 30-37.

LASSILA, Ora; SWICK, Ralph R. (edd.) (1999). Resource Description Framework (RDF) Model and Syntax. *World Wide Web Consortium* Recommendation, Februar 1999, *http://www.w3.org/TR/REC-rdf-syntax*.

LÄUTER, MARTIN; QUASTHOFF, UWE (1999). „Kollokationen und semantisches Clustering.“ In: Gippert, Jost; Olivier, Peter (edd.) (1999). Multilinguale Corpora. Codierung, Strukturierung, Analyse. Proc. 11. GLDV-Jahrestagung, Frankfurt/M. Prag: Enigma Corporation, 34-41.

LAVOIE, Brian; FRYSTYK NIELSEN, Henrik (1999). Web Characterization Terminology & Definitions Sheet. World Wide Consortium Working Draft, Mai 1999, *http://www.w3.org/1999/05/WCA-terms/*.

LAWRENCE, Steve; GILES, C. Lee (1998). „Searching the World Wide Web.“ In: Science 280 (1998), 98-100.

LAWRENCE, Steve; GILES, C. Lee (1999). „Accessibility and Distribution of Information on the Web.” In: Nature 400 (1999), 107-109 [*http://www.wwwmetrics.com/*].

LAYMAN, Andrew et al. (1998). XML-Data. *World Wide Web Consortium* Note, Januar 1998, *http://www.w3.org/TR/1998/NOTE-XML-data.*

LECOLINET, E. et al. (1998). "An Integrated Reading and Editing Environment for Scholarly Research on Literary Works and Their Handwritten Sources." In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1998). Proc. Third ACM Conference on Digital Libraries (DL '98), Juni 1998, Pittsburgh/PA, 144-151.

LEGGETT, John J. (1994). "Viewing Dexter with Open Eyes." In: CACM 37(2) (1994), 76-86.

LEIGHTON, H. Vernon; Srivastava, Jaideep (1999). „First 20 Precision among World Wide Web Search Engines." In: Journal of the American Society for Information Science 50(10) (1999), 870-881.

LENAT, Douglas B. et al. (1990). "CYC: Toward Programs with Common Sense." In: CACM 33(8), 30-49.

LENNON, Jennifer A. (1997). Hypermedia Systems and Applications. Berlin et al.: Springer.

LESCH, Sigrid (1999). „Prozeßoptimierung beim Aufbau einer virtuellen Bibliothek im Internet – Theorie und Praxis." In: BULLINGER, ENGELBACH & KLOSTERMANN 1999, 141-149.

LESK, Michael (1995). The Seven Ages of Information Retrieval. International Federation of Library Associations and Institutions [IFLA], IFLA Universal Dataflow and Telecommunications Core Programme, UDT Occasional Paper N° 5, 1995, *http://www.ifla.org/udt/op/udtop5/udtop5.htm*.

LESK, Michael (1997). Practical Digital Libraries. San Francisco/CA: Morgan Kaufman.

LEWIS, Ted (1998). "Information Appliances: Gadget Netopia." In: IEEE Computer 31(1) (1998), 59-68.

LICKLIDER, J. C. R. (1965). Libraries of the Future. Cambridge/MA: The MIT Press.

LIE, Håkon Wium (1998). Using XSL and CSS Together. *World Wide Web Consortium* Note, September 1998, *http://www.w3.org/TR/NOTE-XSL-and-CSS.*

LIE, Håkon Wium; BOS, Bert (1999). Cascading Style Sheets, Level 1. *World Wide Web Consortium* Recommendation, Dezember 1996, überarbeitete Fassung Januar 1999, *http://www.w3.org/TR/REC-CSS1*.

LIE, Håkon Wium; SAARELA, Janne (1999). „Multipurpose Web Publishing Using HTML, XML, and CSS." In: CACM 42(19) (1999), 95-101.

LIEBERMAN, Henry (1998). „Beyond Information Retrieval: Information Agents at the MIT Media Lab." In: Künstliche Intelligenz 3/1998, 17-23.

LOBIN, Henning (ed.) (1999A). Text im digitalen Medium. Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering. Opladen & Wiesbaden: Westdeutscher Verlag.

LOBIN, Henning (1999B). „Intelligente Dokumente. Linguistische Repräsentation komplexer Inhalte für die hypermediale Wissensvermittlung." In: LOBIN (1999A), 155-178.

LOBIN, Henning (2000. Informationsmodellierung in SGML und XML. Berlin et al.: Springer.

LORENZ, Oliver; MONAGAN, Gladys (1995). Automatisches Indexieren von Liniengrafiken." In: KUHLEN, Rainer; RITTBERGER, Marc (edd. (1995). Proc. HIM'95 Hypertext – Information Retrieval – Multimedia. Konstanz: UVK Informationswissenschaft, 93-103.

LUHN, H. P. (1957). "A Statistical Approach to Mechanical Encoding and Searching of Literary Information." In: IBM Journal of Research and Development 1(4) 1957, 309-317.

LUHN, H. P. (1958). „The Automatic Creation of Literature Abstracts." In: IBM Journal of Research and Development 2(2) 1958, 159-165.

LYNCH, Patrick J.; HORTON, Sarah (1999). Web Style Guide. Basic Design Principles for Creating Web Sites. New Haven & London: Yale University Press.

MAAß, Susanne (1993). „Software-Ergonomie. Benutzer- und aufgabenorientierte Systemgestaltung." In: Informatik-Spektrum 16(4) (1993), 191-205.

MALER, Eve; ALLEN, Terry (1997A). Overview of the DocBook DTD. [Guide to the DocBook DTD, Teil I]. *http://www.oasis-open.org/docbook/documentation/reference/*.

MALER, Eve; ALLEN, Terry (1997B). User's Guide to the DocBook DTD. [Guide to the DocBook DTD, Teil II]. *http://www.oasis-open.org/docbook/documentation/reference/*.

MALER, Eve; ALLEN, Terry (1997C). Reference for the DocBook DTD. [Guide to the DocBook DTD, Teil III]. *http://www.oasis-open.org/docbook/documentation/reference/*.

MALER, Eve; EL ANDALOUSSI, Jeanne (1996). Developing SGML DTDs. From Text to Model to Markup. Upper Saddle River/NJ et al.: Prentice Hall PTR.

MANDALA, Rila et al. (1999). "Combining Multiple Evidence from Different Types of Thesaurus for Query Expansion." In: SIGIR '99. Proc. 22nd International Conference on Research and Development in Information Retrieval, Berkeley, August 1999. New York: ACM [SIGIR Forum Special Issue], 191-197.

MARCOUX, Yves; SÉVIGNY, Martin (1997). "Why SGML? Why Now?" In: Journal of the American Society for Information Science 48(7) (1997), 584-592.

MARSH, Jonathan (ed.) (1999). XML Base (XBase). *World Wide Web Consortium* Working Draft, Dezember 1999, *http://www.w3.org/TR/xmlbase*.

MARSHALL, Catherine C. (1998). "Toward an Ecology of Hypertext Annotation." In: GRØNBÆK, Kaj; MYLONAS, Elli; SHIPMAN, Frank M., III. (1998). Hypertext 98. Proc. of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh/PA, Juni 1998. New York: ACM, 40-49.

MASON, James David (1997). "SGML and Related Standards: New Directions as the Second Decade Begins." In: Journal of the American Society for Information Science 48(7) (1997), 593-596.

MATENA, Vlada; HAPNER, Mark (1999). Enterprise JavaBeans™ Specification. Version: 1.1. Dezember 1999. Palo Alto/CA: Sun Microsystems, *http://java.sun.com/products/ejb/newspec.html*.

MAURER, Hermann (1996). HyperWave – The Next Generation Web Solution. London et al.: Addison Wesley Longman.

MCCRARY, Victor et al. (edd.) (1999). Open eBook™ Publication Structure 1.0. Draft Version 014, Juli 1999. Open eBook Authoring Group, *http://www.openebook.org/specification.htm*.

MCCREADIE, Maureen; RICE, Ronald E. (1999A). "Trends in Analyzing Access to Information. Part I: Croossdisciplinary Conceptualizations of Access." In: Information Processing & Management 35 (1999), 45-76.

MCCREADIE, Maureen; RICE, Ronald E. (1999B). "Trends in Analyzing Access to Information. Part II: Unique and Integrating Conceptualizations. " In: Information Processing & Management 35 (1999), 77-99.

MCGILL, Michael; KOLL, M.; NOREAULT, T. (1979). An Evaluation of Factors Affecting Document Ranking by Information Retrieval System. Report from the School of Information Science, Syracuse University, Syracuse/NY.

MCGRAW, Gary; FELTEN Edward (1997). Java Security. Hostile Applets, Holes, and Antidotes. New York et al.: John Wiley.

MCLUHAN, Marshall (1995). Die Gutenberg-Galaxis. Das Ende des Buchzeitalters. Bonn et al.: Addison-Wesley.

MERZ, M.; LAMERSDORF, W. (1998). „Agents, Services, and Electronic Markets: How do they Integrate?" In: DAVIS, Nigel; RAYMOND, Kerry, SEITZ, Jochen (edd.) (1998). Middleware '98. Proc. IFIP International Conference on Distributed Systems, Platforms, and Open Distributed Processing, September 1998, *http://www-sor.inria.fr/mirrors/middleware98/porceedings*.

MERZ, Thomas (1997A). Postscript & Acrobat/PDF. Applications, Troubleshooting, and Cross-Platform Publishing. Berlin et al.: Springer.

MERZ, Thomas (1997B). Mit Acrobat ins World Wide Web. München: Thomas Merz Verlag.

MEYER, Bertrand (1999). "On to Components." In: IEEE Computer 32(1) (1999), 139 f.

MEYER, Bertrand; MINGINS, Christine (1999). „Component-Based Development: From Buzz to Spark." In: IEEE Computer 32(7) (1999), 35-37.

MEYER-BOUDNIK, Thomas; EFFELSBERG, Wolfgang (1995). "MHEG Explained." In: IEEE Multimedia 2(1) (1995), 26 ff.

MICHAL, Gerhard (1999). Biochemical Pathways. Biochemie- Atlas. Heidelberg: Spektrum Akademischer Verlag.

MILDE, Jan-Torsten (1999). „Effizientes Document Engineering sprachlicher Daten." In: LOBIN 1999A, 197-220.

MILLARD, D. E.; REICH, Siegfried; DAVIS, H. C. (1999). ”Dynamic Service Discovery and Invocation in OHP.” In: WIIL (1999), 38-42.

MILLER, Eric (1998). “An Introduction to the Resource Description Framework.” In: D-Lib Magazine, Mai 1998, *http://www.dlib.org/dlib/may98/miller/05miller.html*.

MILLER, James R.; BONURA, Thomas (1998). „From Documents to Objects. An Overview of LiveDoc.“ In: SigCHI Bulletin 30(2) (1998), 53-58.

MILLER, Jim; RESNICK, Paul; SINGER, David (1996). Rating Services and Rating Systems (and Their Machine Readable Descriptions). Version 1.1. *World Wide Web Consortium* Recommendation, Oktober 1996, *http://www.w3.org/TR/REC-PICS-services.html*.

MITRA, Mandar; SINGHAL, Amit; BUCKLEY, Chris (1998). “Improving Automatic Query Expansion.” In: SIGIR ’98. Proc. 21st International Conference on Research and Development in Information Retrieval, Melbourne, August 1998, 206-214.

MIYAMOTO, Sadaaki (1998). „Application of Rough Sets to Information Retrieval.“ In: Journal of the American Society for Information Science 49(3) (1998), 195-205.

MÖHR, WIEBKE; Schmidt, Ingrid (edd.) (1999). SGML und XML. Anwendungen und Perspektive. Berlin et al.: Springer.

MORENO, Roxana; MAYER, Richard E. (1999). „Deriving Instructional Design Principles from Multimedia Presentations with Animations.“ In: Proc. ICMCS ’99. 1999 IEEE International Conference on Multimedia Computing and Systems, Florenz, Juni 1999, Bd. 1, 720-725.

MORROW, Terry M.; MUMFORD, Anne, M. (1994). “Publishing on the Network.” In: Computer Networks and ISDN Systems 26 (Suppl. 1) (1994), S15-S28.

MOULTHROP, Stuart (1991). „Beyond the Electronic Book: A Critique of Hypertext Rhetoric. In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1991). Proc. Third ACM Conference on Hypertext, New York. New York: ACM Press, 291-298.

MÜLLER, Günter; PFITZMANN, Andreas (edd.) (1997). Mehrseitige Sicherheit in der Kommunikationstechnik. Verfahren, Komponenten, Integration, Bonn et al.: Addison-Wesley.

MURRAY-RUST, Peter (1998). XML and the Launch of Chemical Markup Language. Nottingham University, Virtual School of Molecular Sciences, *http://www.xml-cml.org/talk/001.html*.

MURUGESAN, San (1999). “Web Engineering.” In: SIGWeb Newsletter 8(3) (1999), 28-32.

MYAENG, Sung Hyon et al. (1998). „A Flexible Model for Retrieval of SGML Documents.“ In: SIGIR ’98. Proc. 21st International Conference on Research and Development in Information Retrieval, Melbourne, August 1998, 138-145.

NANARD, Jocelyne; Nanard, Marc (1995). „Hypertext Design Environments and the Hypertext Design Process.“ In: CACM 38(8) (1995), 49-56.

NELSON, Ted (1999A). Xanadu® Technologies – an Introduction. August 1999, *http://www.xanadu.net/ TECH/xuTech.html*.

NELSON, Ted (1999B). Transcopyright Permissions Page. Juli 1999, *http://www.sfc.keio.ac.jp/~ted/TPUB/TranspubPosterPermish.html*.

NELSON, Theodor Holm (1994). “Xanadu: Document Interconnection Enabling Re-Use with Automatic Author Credit and Royalty Accounting.” In: Information Services and Use 14 (1994), 255-265.

NELSON, Theodor Holm (1995). “The Heart of the Connection: Hypermedia Unified by Transclusion.” In: CACM 38(8) (1995), 31 f.

NETCRAFT (1999). Web Server Survey. Dezember 1999, *http://www.netcraft.com/survey*.

NETSIZER (1999). Internet Growth Reports. Dezember 1999, *http://www.netsizer.com*.

NEUMCKE, Peter (1999). Standardisierte Internet-Kooperationsprotokolle und ihre Anwendung für elektronische Ausschreibungsverfahren. Diplomarbeit, Universität Leipzig, Wirtschaftswissenschaftliche Fakultät, Oktober 1999.

NEWCOMB, Steven R. (1995A). “Multimedia Interchange Using SGML/HyTime. Part I: Structures.” In: IEEE Multimedia 2(2) (1995), 86-89.

NEWCOMB, Steven R. (1995B). “Multimedia Interchange Using SGML/HyTime. Part II: Principles and Applications.” In: IEEE Multimedia 2(3) (1995), 60-64.

NEWCOMB, Steven R. et al. (1991). "'HyTime'; The Hypermedia/Time-Based Structuring Language." In: CACM 34(11) (1991), 67-83.

NEWCOMB, Steven R.; TAUSSING NEWCOMB, Victoria (1992). „Some Background Information about HyTime." In: Journal of the Institute of Image Electronics Engineers of Japan 21(5) (1992), 459-67.

NICHOLAS, Charles K.; WELSCH, Lawrence A. (1992). „On the Interchangeability of SGML and ODA." In: Electronic Publishing: Origination, Dissemination and Design 5(3) (1992), 105-130.

NIELSEN, Jakob (1990). "The art of Navigating Through Hypertext." In: CACM 33(3) (1990), 296-310.

NIELSEN, Jakob (1996). Multimedia, Hypertext und Internet. Grundlagen und Praxis des elektronischen Publizierens. Braunschweig & Wiesbaden: Vieweg.

NIELSEN, Jakob (1999). "User Interface Directions for the Web." In: CACM 42(1) (1999), 65-72.

NIERSTRASZ, Oskar; GIBBS, Simon; TSICHRITZIS, Dennis (1992). „Component-Oriented Software Development." In: CACM 35(9) (1992), 160-165.

NORMAN, Donald A. (1990). The Design of Everyday Things. New York et al.: Doubleday [ursprgl. u. d. T. *The Psychology of Everyday Things*, New York: Basic Books, 1988].

NORMAN, Donald A. (1998). The Invisible Computer. Why Good Products Can Fail, the Personal Computer Is So Complex, and Information Appliances are the Solution. Cambridge/MA & London: The MIT Press.

NOTESS, Greg (1999). Search Engine Showdown. The Users' Guide to Web Searching. Dezember 1999, *http://www.notess.com/search/*.

NÜRNBERG, Peter J.; ASHMAN, Helen (1999). "What was the Question? Reconciling Open Hypermedia and World Wide Web Research." In: Proc. ACM Hypertext '99 Conference on Hypertext and Hypermedia, Darmstadt, Februar 1999, 83-90.

NÜRNBERG, Peter J.; LEGGETT, John J. (1998). "A Vision for Open Hypermedia Systems." In: Journal of Digital Information 1(2) (1998), *http://jodi.ecs.soton.ac.uk/Articles/v01/i02/Nurnberg*.

NÜRNBERG, Peter J.; LEGGETT, John J.; WIIL, Uffe K. (1998). "An Agenda for Open Hypermedia Research." In: GRØNBÆK, Kaj; MYLONAS, Elli; SHIPMAN, Frank M., III. (1998). Hypertext 98. Proc. of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh/PA, Juni 1998. New York: ACM, 198-206.

O'HARA, Kenton et al. (1998). "Student Readers' Use of Library Documents: Implications for Library Technologies." In: Proc. CHI 98: ACM Conference on Human Factors in Computing Systems, Los Angeles, April 1998, 233-240.

OCLC (1999). June 1999 Web Statistics. Online Computer Library Center, Inc, Web Characterization Project, *http://www.oclc.org/oclc/research/projects/webstats/statistics.htm*.

OLSON, Judith Reitman; OLSON, Gary M. (1990). „The Growth of Cognitive Modeling in Human-Computer Interaction since GOMS." In: Human-Computer Interaction 5 (1990), 221-265.

ORFALI, Robert; HARKEY, Dan; EDWARDS, Jeri (1998). Instant CORBA. Bonn et al.: Addison-Wesley.

PAEPCKE, Andreas et al. (1999). "Using Distributed Objects to Build the Stanford Digital Library Infobus." In: IEEE Computer 32(2) 1999, 80-87.

PAPADIAS, Dimitris et al. (1999). "Content-Based Retrieval Using Heuristic Search." In: HEARST, Marti; GEY, Frederic; TONG, Richard (edd.) (1999). SIGIR '99. Proc. 22nd International Conference on Research and Development in Information Retrieval, Berkeley, August 1999. New York: ACM [SIGIR Forum Special Issue], 168-175.

PAPE, Christian; SCHMITT, Peter (1998). Interaktive Visualisierungen in der Lehre der theoretischen Informatik." In: CLAUS (1998), 102-111.

PAPE, Uwe; SANDKUHL, Kurt (1990). „Aspekte und Trends des Elektronischen Publizierens. " In: Informationstechnik – IT 32(4) (1990), 220-230.

PARK, Honseok (1997). "Relevance of Science Information: Origins and Dimensions of Relevance and Their Implications to Information Retrieval." In: Information Processing & Management 33(3) (1997), 339-352.

PEEK, Robin P.; NEWBY, Gregory B. (1996). Scholarly Publishing. The Electronic Frontier. Cambridge/MA & London: The MIT Press.

PEEK, Robin P.; POMERANTZ, Jeffrey; PALING, Stephen (1998). "The Traditional Scholarly Journal Publishers Legitimize the Web." In: Journal of the American Society for Information Science 49(11) (1998), 983-989.

PEMBERTON, Steven et al. (ed.) (1999). XHTML™ 1.0: The Extensible HyperText Markup Language. A Reformulation of HTML 4.0 in XML 1.0. *World Wide Web Consortium* Proposed Recommendation, 24. August 1999, *http://www.w3.org/TR/xhtml1*.

PETERS, Joachim (1998). Bestiarium der Bits ‚n' Bytes. Perspektiven des Electronic Publishing. Berlin et al.: Springer; Hamburg: MacUp.

PHELPS, Thomas A.; WILENSKY, Robert (1996). "Toward Active, Extensible, Networked Documents: Multivalent Architecture and Applications." In: Proc. First ACM Conference on Digital Libraries (DL '96), März 1996, Bethesda/MD, 100-108.

PIEMONT, Claudia (1999). Komponenten in JAVA™. Heidelberg: dpunkt.

PIMIENTA, Daniel et al. (1998). L4. The Fourth Study on Languages and the Internet. The Place of Latin Languages and Cultures on the Internet. September 1998. Santo Domingo: Funredes Networks and Development Foundation, *http://funredes.org/LC/english/L4prologue.html*.

PINGEL, Ulf-Torsten; SCHEMMERT, Ulf; WOLFF, Christian (1998). Entwicklung von Simulationen. Multimediaprojekt Physikalisches Praktikum, Arbeitsbericht L2, Universität Leipzig, April 1998.

PITKOW, James E. (1998). "Summary of WWW Characterizations." In: Proc. Seventh World Wide Web Conference, Brisbane, April 1998, *http://www7.scu.edu.au/programme/fullpapers/1877/com1877.htm*

POPIEN, Claudia (1995). Dienstevermittlung in verteilten Systemen. Stuttgart & Leipzig: Teubner [= Teubner-Texte zur Informatik, Bd. 12].

POPPELIER, Nico; MINER, Robert; ION, Patrick (edd.) (2000). Mathematical Markup Language (MathML). Version 2.0. *World Wide Web Consortium* Working Draft, Februar 2000, *http://www.w3.org/TR/MathML2*.

PRICE, Morgan N.; GOLOVCHINSKY, Gene; SCHILIT, Bill N. (1998). "Linking by Inking: Trailblazing in a Paper-like Hypertext." In: GRØNBÆK, Kaj; MYLONAS, Elli; SHIPMAN, Frank M., III. (1998). Hypertext 98. Proc. of the Ninth ACM Conference on Hypertext and Hypermedia, Pittsburgh/PA, Juni 1998. New York: ACM, 30-39.

PRICE, Roger (1998). "Beyond SGML." In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1998). Proc. Third ACM Conference on Digital Libraries (DL '98), Juni 1998, Pittsburgh/PA, 172-181.

QUASTHOFF, Uwe; WOLFF, Christian (1999). „Korpuslinguistik und große einsprachige Wörterbücher." In: Linguistik Online 3, Heft 2/99 [*http://viadrina.euv-frankfurt-o.de/~wjournal/2_99/*].

QUASTHOFF, Uwe; WOLFF, Christian (2000). „An Infrastructure for Corpus-Based Monolingual Dictionaries." In: Proc. LREC-2000. Second International Conference On Language Resources and Evaluation. Athen, Mai/Juni 2000 [erscheint].

QUINT, Vincent; VATTON, Irène (1997). An Introduction to Amaya. *World Wide Web Consortium* Note, 20. Februar 1997, *http://www.w3.org/pub/WWW/TR/NOTE-amaya*.

RAGGETT, Dave; LE HORS, Arnaud; JACOBS, Ian (1999). HTML 4.01 Specification. W3C Proposed Recommendation, August 1999, *http://www.w3.org/TR/html40*.

RAYWARD, W. Boyd (1999). "H. G. Well's Idea of a World Brain: A Critical Reassessment." In: Journal of the American Society for Information Science 50(7) (1999), 557-573.

REHM, Georg (1999). „Automatische Textannotation. Ein SGML- und DSSSL-basierter Ansatz zur angewandten Textlinguistik." In: LOBIN (1999A), 179-195.

REICH, Siegfried (1999). Definitions and Examples of the Open Hypermedia Protocol. Dezember 1999, *http://www.ifs.uni-linz.ac.at/ifs/staff/reich/ohs/docs/defs.html.*

REICH, Siegfried; Millard, David M.; Davis, Hugh C. (1999). "Naming in OHP." In: WIIL, Uffe Kock (ed.) (1999). Proc. Fifth Workshop on Open Hypermedia Systems, Darmstadt, Februar 1999 [= Aalborg University, Dept. of Computer Science, TR CS-99-01], 43-47.

REICHENBERGER, Klaus; STEINMETZ, Ralf (1999). „Visualisierungen und ihre Rolle in Multimedia-Anwendungen." In: Informatik-Spektrum 22(2) (1999), 88-98.

REINKING, David (1992). „Differences between electronic and printed texts: an agenda for research." In: Journal of Educational Multimedia and Hypermedia 1(1) (1992), 11-24.

RESMER, Mark (1998). „Internet Architectures for Learning." In: IEEE Computer 31(9) 1998, 105f.

RESSLER, Sandy (1993). Perspectives on Electronic Publishing. Standards, Solutions and More. Englewood Cliffs/NJ: PTR Prentice-Hall.

REW, Russ (1998). "Metadata and RDF Survey." *http://www.unidata.ucar.edu/staff/russ/bs/metadata*.

RIEDER, Rufus (1999). „Autorensysteme." In: Screen Business Online, April 1999, 70-74.

RIEGER, Wolfgang (1995). SGML für die Praxis. Berlin et al.: Springer.

RIEHM, Ulrich et al. (1992). Elektronisches Publizieren. Eine kritische Bestandsaufnahme. Berlin et al.: Springer.

ROBERTSON, Scott P.; OLSON, Gary M.; OLSON, Judith S. (edd.) (1991). Reaching through Technology. Proc. CHI '91: Human Factors in Computing Systems. New York: ACM Press.

ROBIE, Jonathan (1998). The Design of XQL. Stockholm: Texcel International AB, *http://www.texcel.no/whitepapers/xql-design.html*.

RÖHRING, Hans-Helmut (1997[6]). Wie ein Buch entsteht. Einführung in den modernen Buchverlag. Darmstadt: Wissenschaftliche Buchgesellschaft.

RÖSCHEISEN, Martin; MOGENSEN, Christian; WINOGRAD, Terry (1995). "Beyond Browsing: Shared Comments, SOAPs, Trails, and On-line Communities." In: Proc. Third World Wide Web Conference, Darmstadt, April 1995, *http://www.igd.fhg.de/www/www95/proceedings/papers/85/TR/WWW95.html*.

ROSENBERG, JONATHAN ET AL. (1991). Multi-media Document Translation. ODA and the EXPRES Project. New York et al.: Springer.

ROSENFELD, Louis; MORVILLE, Peter (1998). Information Architecture for the World Wide Web. Sebastopol/CA: O'Reilly.

RÖSNER, Dietmar (1999). "Desiderata für Autorensysteme für Lehr-/Lernsoftware." In: GIRMES 1999A: 85-92.

ROßBACH, Peter; SCHREIBER, Hendrik (1999). Java Server und Servlets. Portierbare Web-Applikationen effizient entwickeln. Bonn et al.: Addison Wesley Longman.

ROSSI, Gustavo; SCHWABE, Daniel; GARRIDO, Alejandra (1999). "Designing Computational Hypermedia Applications." In: Journal of Digital Information 1(4) (1999), *http://jodi.ecs.soton.ac.uk/Articles/v01/i04/Rossi/*.

ROSTEK, Lothar (1999). „Automatisches Erzeugen von semantischem Markup in Agenturmeldungen." In: MÖHR & SCHMIDT (1999), 307-321.

ROUSH, Wade (1999A). „Sophisticated Interface + Award-winning Design = SoftBook Reader." Rezension, eBookNet.com, November 1999, *http://www.ebooknet.com/story.jsp?id=23*.

ROUSH, Wade (1999B). „Rocket eBook 'Preserves a Book's Bookness'." Rezension, eBookNet.com, Dezember 1999, *http://www.ebooknet.com/story.jsp?id=24*.

ROUSH, Wade (1999C). „Everybook: Facing Screens Designed to 'Emulate a Book Exactly'." Rezension, eBookNet.com, November 1999, *http://www.ebooknet.com/story.jsp?id=41*.

RUTLEDGE, Lloyd et al. (1998). "Practical Application of Existing Hypermedia Standards and Tools." In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1998). Proc. Third ACM Conference on Digital Libraries (DL '98), Juni 1998, Pittsburgh/PA, 191-199.

SALMINEN, Aari; KAUPPINNEN, Katri; LEHTOVAARA, Merja (1997). "Towards a Methodology for Document Analysis." In: Journal of the American Society for Information Science 48(7) (1997), 644-655.

SALTON, Gerard (1975). Dynamic Information and Library Processing. Englewood Cliffs/NJ: Prentice-Hall.

SALTON, Gerard (1989). Automatic Text Processing. The Transformation, Analysis, and Retrieval of Information by Computer. Reading/MA: Addison-Wesley.

SALTON, Gerard (ed.) (1971). The SMART Retrieval System – Experiments in Automatic Document Processing. Englewood Cliffs/NJ: Prentice-Hall.

SALTON, Gerard; ALLAN, James; BUCKLEY, Chris (1990). "Automatic Structuring of Large Text Files." In: Communications of the CACM 37(2) (1994), 97-108.

SAMUELSON, Pamela; GLUSHKO, Robert J. (1991). “Intellectual Property Rights for Digital Library and Hypertext Publishing Systems: An Analysis of Xanadu.” In: Proc. Third Annual ACM Conference on Hypertext (HYPERTEXT '91), 39 – 50.

SANDKUHL, Kurt; KINDT, Andreas (1996). Telepublishing. Die Druckvorstufe auf dem Weg ins Kommunikationszeitalter. Berlin et al.: Springer.

SCHAMBER, Linda; EISENBERG, Michael B.; NILAN, Michael S. (1990). “A Re-Examination of Relevance: Toward a Dynamic, Situational Definition.” In: Information Processing & Management 26(6) (1990), 755-776.

SCHÄUBLE, Peter (1997). Multimedia Information Retrieval. Content-Based Information Retrieval from Large Text and Audio Databases. Boston et al.: Kluwer.

SCHELLER, Martin et al. (1994). Internet: Werkzeuge und Dienste. Berlin et al.: Springer.

SCHICKLER, Matthew A; MAZER, Murray S.; BROOKS, Charles (1996). “Pan-Browser Support for Annotations and other Information on the World Wide Web.” In: Proc. Fifth World Wide Web Conference, Paris, Mai 1996, *http://www5conf.inria.fr/fich_html/papers/P15/Overview.html*.

SCHIFMAN, Richard S.; HEINRICH, Yvonne; HEINRICH, Günther (1999). Multimedia-Projektmanagement. Von der Idee zum Produkt. Berlin et al.: Springer.

SCHILIT, Bill N. et al. (1999). “As We May Read: The Reading Appliance Revolution.” In: IEEE Computer 32(1) (1999), 65-73.

SCHILIT, Bill N.; GOLOVCHINSKY, Gene; PRICE, Morgan N. (1998). “Beyond Paper: Supporting Active Reading with Free Form Digital Ink Annotation.” In: Proc. CHI 98: ACM Conference on Human Factors in Computing Systems, Los Angeles, April 1998, 249-256.

SCHILIT, Bill N.; PRICE, Morgan N.; GOLOVCHINSKY, Gene (1998). “Digital Library Information Appliances.” In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1998). Proc. Third ACM Conference on Digital Libraries (DL ’98), Juni 1998, Pittsburgh/PA, 217-226.

SCHNEIDER, Hans-Jochen (ed.) (1997[4]). Lexikon Informatik und Datenverarbeitung. München: Oldenbourg.

SCHNEIDER, Klaus-Peter; WOLFF, Christian (1998A). Pflichtenheft & Drehbuch. Multimediaprojekt Physikalisches Praktikum, Arbeitsbericht L1, Universität Leipzig, Oktober 1997.

SCHNEIDER, Klaus-Peter; WOLFF, Christian (1998B). Prototypen Viewersoftware auf Browserbasis. Multimediaprojekt Physikalisches Praktikum, Arbeitsbericht L3, Universität Leipzig, April 1998.

SCHNUPP, Peter (1992). Hypertext. München: Oldenbourg [= Handbuch der Informatik Bd. 10.1].

SCHOOP, Eric; STROBEL, Katrin (1999). „SGML als Grundbaustein für das betriebliche Wissensmanagement.“ In: MÖHR & SCHMIDT (1999), 257-277.

SCHRÖDER, Bernhard (1998). „Pro-SGML: Ein Prolog-basiertes System zum Textretrieval.” In: HEYER & WOLFF 1998A, 205-216.

SCHULMEISTER, Rolf (1997[2]). Grundlagen hypermedialer Lernsysteme. Theorie · Didaktik · Design. München & Wien: Oldenbourg.

SCHWABE, Daniel; DE ALMEIDA PONTES, Rita; MOURA, Isabela (1999). „OOHDM-Web: An Environment for Implementation of Hypermedia Applications in the WWW.“ In: SIGWeb Newsletter 8(2) (1999), 18-33.

SCHWARTZ, Candy (1998). „Web Search Engines.“ In: Journal of the American Society for Information Science 49(11) (1998), 973-982.

SCOTT MACKENZIE, I. (1992). „Beating the book: Megachallenges for CD-ROM and hypertext.“ In: Journal of Research on Computing in Education (Summer 1992) 24(4), 486-98.

SEARLE, John (1986[2]). Sprechakte. Ein sprachphilosophischer Essay. Frankfurt am Main: Suhrkamp [= stw Bd. 458].

SEIDT, Markus (1994). „Remis. Zwei Wege zum normierten Dokumentenaustausch.” In: iX 1994 (4), 185-189.

SEILER, Lauren H. (1992). “The concept of book in the age of the digital electronic medium.” In: Library Software Review 11(1) (1992), 19-29.

SENGUPTA, Arijit; Dillon, Andrew (1997). “Extending SGML to Accommodate Database Functions: A Methodological Overview.” In: Journal of the American Society for Information Science 48(7) (1997), 629-637.

SHACKEL, Brian (1997). “Human-Computer Interaction: Whence and Whither?” In: Journal of the American Society for Information Science 48(11) (1997), 970-986.

SHIN, Dongwook; NAM, Sejin; KIM, Munseok (1997). “Hypertext Construction Using Statistical and Semantic Similarity.” In: Proc. Second ACM Conference on Digital Libraries (DL ’97), Juni 1997, Philadelphia/PA, 57-63.

SHNEIDERMAN, Ben (1982). „The Future of Interactive Systems and the Emergence of Direct Manipulation.“ In: Behaviour and Information Technology 1(3) (1982), 237-256.

SHNEIDERMAN, Ben (1983). „Direct Manipulation: A Step beyond Programming Languages.“ In: IEEE Computer 16(8) (1983), 57-69.

SHNEIDERMAN, Ben (1998[3]). Designing the User Interface. Reading/MA et al.: Addison-Wesley.

SILVERSTEIN, Craig et al. (1999). “Analysis of a Very Large Web Search Engine Query Log.” In: SIGIR Forum 33(1) (1999), 6-12.

SIMONS, Gary F. (1998). “Using Architectural Processing to Derive Small, problem-specific XML Applications from Large, widely-Used SGML Applications.” In: Proc. Markup Technologies ’98, Chicago, November 1998 [= SIL Electronic Working Papers 1998-006, Dezember 1998, *http://www.sil.org/silewp/1998/006/SILEWP1998-006.html*].

SINGH, Munindar P. (1998). „Agent Communication Languages: Rethinking the Principles.“ In: IEEE Computer 31(12) (1998), 40-47.

SINGHAL, Amit; PEREIRA, Fernando (1999). ”Document Expansion for Speech Retrieval.” In: HEARST, Marti; GEY, Frederic; TONG, Richard (edd.) (1999). SIGIR ’99. Proc. 22nd International Conference on Research and Development in Information Retrieval, Berkeley, August 1999. New York: ACM [SIGIR Forum Special Issue], 34-41.

SLEIN, J. A. et al. (1998). “Requirements for a Distributed Authoring and Versioning Protocol for the World Wide Web.” Internet Request for Comments N° 2291, Internet Architecture Board, Februar 1998.

SMITH, Sydney L. (1988). “Standards Versus Guidelines for Designing User Interface Software.” In: HELANDER, Martin (ed.) (1988). Handbook of Human-Computer Interaction. Amsterdam: Elsevier (North-Holland), 877-889 [Kap. 40].

SMITH, Terrence R.; GEFFNER, Steven; GOTTSEGEN, Jonathan (1997). “A General Framework for the Meta-Information and Catalogs in Digital Libraries.” In: Proc. First IEEE Metadata Conference, April 1996, Silver Spring/MA, *http://www.computer.org/conferen/meta96/smith/ieee.html*.

SOBOTTA, Johannes (1999). Atlas der Anatomie des Menschen. 2 Bde., München: Urban & Fischer.

SPENGLER, Marion (1999). „Realisierung eines SGML-basierten Publikationsprozesses im Verlag: kritische Anmerkungen.“ In: MÖHR & SCHMIDT (1999), 145-156.

SPERBERG-MCQUEEN, C. M.; BURNARD, Lou (1993). Living with the Guidelines: An Introduction to TEI Tagging. [An Excerpt]. 1993 University of Virginia Rare Books School, *http://www.oasis-open.org/cover/teiu5-uva.html*.

SPERBERG-MCQUEEN; C. M.; BURNARD, Lou (edd.) (1994). Guidelines for Electronic Text Encoding and Interchange. TEI P3. Hrsg. v. der Text Encoding Initiative, Chicago & Oxford für *The Association for Computers and the Humanities* (ACH); *The Association for Computational Linguistics* (ACL); *The Association for Literary and Linguistic Computing* (ALLC) [*http://www-tei.uic.edu/orgs/tei/*].

SRIDHARAN, Prashant (1997). Advanced Java Networking. Upper Saddle River/NJ: Prentice Hall PTR.

STEIN, Dietrich (1999[3]). Instandhaltung von Kanalisationen. Berlin: Ernst & Sohn.

STEINMETZ, Ralf (1999[2]). Multimedia-Technologie. Einführung und Grundlagen. Berlin et al.: Springer.

STEINMÜLLER, Wilhelm (1993). Informationstechnologie und Gesellschaft. Einführung in die angewandte Informatik. Wiesbaden: Wissenschaftliche Buchgesellschaft.

STERN, David (1995). "SGML Documents: A Better System for Communicating Knowledge." In: Special Libraries 86(2) (1995), 117-24.

STÖCKER, Horst (1997). Taschenbuch der Physik. Formeln, Tabellen, Übersichten. Frankfurt/Main: Harri Deutsch [Elektronische Fassung in der Interdoc-Bibliothek erreichbar über *https://sunmedoc1.informatik.tu-muenchen.de:8887/Books/dt_phys/start.htm*].

STOCKWERK2 (1999). Screen Design für das Multimediale Physikalische Praktikum. Oldenburg: Stockwerk2, zugleich Arbeitsmaterialie, BMB+F-Projekt Multimediales Physikalisches Praktikum.

STORRER, Angelika (1998). „Vom Grammatikbuch zur Hypertext-Grammatik. Methodisches Vorgehen bei der Hypertextualisierung nicht-standardisierter Textsorten." In: HEYER & WOLFF (1998A), 33-49.

STORRER, Angelika (1999). „Kohärenz in Text und Hypertext." In: LOBIN (1999A), 33-66.

STORRER, Angelika (2000A). „Schreiben, um besucht zu werden: Textgestaltung fürs World Wide Web." In: BUCHER, Hans-Jürgen; PÜSCHEL, Ulrich (edd.) (2000). Die Zeitung zwischen Print und Digitalisierung. Opladen & Wiesbaden: Westdeutscher Verlag [erscheint, *http://www.ids-mannheim.de/grammis/storrer/trier.pdf*].

STORRER, Angelika (2000B). „Was ist "hyper" am Hypertext." In: KALLMEYER, Werner (ed.) (2000). Sprache und Neue Medien. Berlin u.a.: De Gruyter [= Jahrbuch 1999 des Instituts für deutsche Sprache, erscheint, *http://www.ids-mannheim.de/grammis/storrer/hyper.pdf*]

STOTTS, David P.; FURUTA, Richard; CABARRUS, Cyrano Ruiz (1998). „Hyperdocuments as Automata: Verification of Trace-Based Browsing Properties by Model Checking." In: ACM Transactions on Information Systems 16(1) (1998), 1-30.

STRZALKOWSKI, Tomek et al. (1999). „Natural Language Information Retrieval: TREC-7 Report." In: VOORHEES, Ellen M.; HARMAN, Donna K. (edd.) (1999). The Seventh Text Retrieval Conference (TREC-7). Gaithersburg/MD: National Institute of Standards and Technology [= NIST Special Publications 500-242].

SULLIVAN, Danny (1999). Search Engine Watch. Dezember 1999, *http://www.searchenginewatch.com*.

SUN MICROSYSTEMS (1999). "Secure Computing With Java™: Now And The Future. A White Paper." Sun Microsystems, Mai 1999, *http://www.javasoft.com/marketing/collateral/security.html*.

SUSSMAN, David (1999). "WebDAV: A Panacea for Collaborative Authoring?" In: IEEE Multimedia 6(2) (1999), 76-79.

SWICK, Ralph R.; THOMPSON, Henry S. (edd.) (1999). The Cambridge Communiqué. *World Wide Web Consortium* Note, Oktober 1999, *http://www.w3.org/TR/NOTE-schema-arch.*

TAKAHASHI, Kenji; LIANG, Eugene (1997). „Analysis and Design of Web-based Information Systems." In: Proc. Sixth World Wide Web Conference, Santa Clara/CA, April 1997, *http://www.scope.gmd.de/info/www6/technical/paper245/paper245.html*.

TEN KATE, Warner; WUGOFSKI, Ted; SCHMITZ, Patrick; (2000). Synchronized Multimedia Integration Language (SMIL) Modules. *World Wide Web Consortium* Working Draft, Februar 2000, *http://www.w3.org/TR/smil-boston/Modules/smil-modules.html* [= Teil 2 von AYARS et al. 2000].

THISSEN, Frank (2000). Screen Design Handbuch. Berlin et al.: Springer.

THOMPSON, Henry S. et al. (edd.) (1999). XML Schema Part 1: Structures. *World Wide Web Consortium* Working Draft, Dezember 1999, *http://www.w3.org/TR/xmlscxhema-1/.*

THOMPSON, Michael; WISHBOW, Nina (1992). "Prototyping: Tools and Techniques: Improving Software and Documentation Quality through Rapid Prototyping." In: SIGDOC '92. Proc. 10th Annual International Conference on Systems Documentation, Ottawa, Oktober 1992, 191-199.

THÜRING, Manfred; HANNEMANN, Jörg; HAAKE, Jörg M. (1995). „Hypermedia and Cognition: Designing for Comprehension." In: CACM 38(8) (1995), 57-66.

TOCHTERMANN, Klaus (1995). Ein Modell für Hypermedia. Aachen: Shaker.

TOMEK, I. et al. (1991). „Hypermedia – Introduction and Survey.“ In: Journal of Microcomputer Applications 14(2) (1991), 63-103.

TRAUBOTH, Heinz (1996[2]). Software-Qualiätsicherung. München & Wien: Oldenbourg [Elektronische Fassung in der Interdoc-Bibliothek erreichbar über *https://medoc.rz.uni-leipzig.de:8887/Books/trauboth/index.html*].

TRESCH, Markus (1996). „Middleware: Schlüsseltechnologie zur Entwicklung verteilter Informationssysteme.” In: Informatik-Spektrum 19 (1996), 249-256.

TREVIRANUS, JUTTA et al. (edd.) (1999A). Authoring Tool Accesibility Guidelines 1.0. *World Wide Web Consortium* Working Draft, Oktober 1999, *http://www.w3.org/WAI/AU/WAI-AUTOOLS.*

TREVIRANUS, JUTTA et al. (edd.) (1999B). Techniques for Authoring Tools Accessibility. *World Wide Web Consortium* Working Draft, September 1999, *http://www.w3.org/WAI/AU/WAI-AUTOOLS-TECHS.*

TUCK, Bill et al. (1992). „Online Books.“ In: Electronic Documents 1(10) (1992), 1-32.

TURAU, Volker (1999). “Techniken zur Realisierung Web-basierter Anwendungen.” In: Informatik Spektrum 22(1) (1999), 3-12.

TYMA, Paul „Why are We Using Java Again?“ In: CACM 41(6) (1998), 38-42.

UNESCO (1964). An International Survey of Book Production During the Last Decades. Paris: UNESCO [United Nations Educational, Scientific and Cultural Organization].

UNIVERSITÄT LEIPZIG (ed.) (1998). Tradition und Innovation. Informationen und Bilder einer weltoffenen Universität. Multimedia-CD-ROM. Produziert von Uwe QUASTHOFF & Christian WOLFF, Leipzig 1998.

VALAUSKAS, Edward J. (1994). “Reading and Computers – Paper-based or Digital Text: What' s Best?” In: Computers in Libraries 14(1) (1994), 44-47.

VAN OSSENBRUGGEN, Jacco; HARDMAN, Lynda; RUTLEDGE, Lloyd (1999). ”Interoperability on the World Wide Web.” In: WIIL, Uffe Kock (ed.) (1999). Proc. Fifth Workshop on Open Hypermedia Systems, Darmstadt, Februar 1999 [= Aalborg University, Dept. of Computer Science, TR CS-99-01], 38-42.

VANNINI, Walter (1999). “Teaching Old Code New Clicks, or Xanadu™ Goes Open Source.” In: SIGWeb Newsletter 8(3) (1999), 52f.

VINOSKI, Steve (1998). „New Features for CORBA 3.0.“ In: CACM 41(19) (1998), 44-52.

VOGEL, Andreas; DUDDY, Keith (1997). Java Programming with CORBA. New York et al.: John Wiley.

WALDO, Jim et al.(1994). „A Note on Distributed Computing.“ Sun Microsystems Laboratories Technical Report No. SMLI TR 94-29. [ND in: Arnold et al. (1999), 307-326].

WALSH, Norman (1997). „A Guide to XML“. In: World Wide Web Journal 2(4) (1997), 97-108.

WALSH, Norman (edd.) (1999). XSL Requirements Summary. *World Wide Web Consortium* Working Draft, Mai 1999, *http://www.w3.org/TR/WD-XSL-REQ.*

WANDMACHER, Jens (1993). Software-Ergonomie. Berlin & New York: De Gruyter [= Mensch-Computer-Kommunikation: Grundwissen Bd. 2].

WAPFORUM (1999). WAP™ White Paper. Wireless Application Protocol: The Wireless Internet Today. Mountain View/CA: Wireless Application Forum Ltd., Oktober 1999, *http://www.wapforum.org/what/WAP_white_pages.pdf.*

WEICHEL, Bernhard (1999). „SGML/XML als Verbindungsschicht in Entwicklungsprozessen.“ In: MÖHR & SCHMIDT (1999), 211-239.

WEIDENMANN, Bernd (1997). „Multicodierung und Multimodalität im Wissenserwerb.“ In: ISSING & KLIMSA (1997), 65-84.

WEINER, Scott R.; ASBURY, Stephen (1998). Programming with JFC. New York et al.: John Wiley.

WEINSTEIN, Peter (1998). “Ontology-Based Metadata: Transforming the MARC Legacy.” In: WITTEN, I.; AKSCYN, Robert; SHIPMAN, Frank M. (edd.) (1998). Proc. Third ACM Conference on Digital Libraries (DL ’98), Juni 1998, Pittsburgh/PA, 254-263.

WEINSTEIN, Peter; ALLOWAY, Gene (1997). “Seed Ontologies: Growing Digital Libraries as Distributed, Intelligent Systems.” In: Proc. Second ACM Conference on Digital Libraries (DL ’97), Juni 1997, Philadelphia/PA, 83-90.

WELLS, H. G. (1938). World Brain. London: Methuen.

WHITEHEAD, E. James (1999). WebDAV Interoperability Experience. University of California/Irvine, Dept. of Information and Computer Science, März 1999, *http://www.ics.uci.edu/pub/ietf/webdav/ interop.html*.

WHITEHEAD, E. James; WIGGINS, Meredith (1998). "WebDav: IETF Standard for Collaborative Authoring on the Web." In: IEEE Internet Computing 2(5) (1998), 34-40.

WIIL, Uffe Kock (1998). "Open Hypermedia: Systems, Interoperability, and Standards." In: Journal of Digital Information 1(2) (1998), *http://jodi.ecs.soton.ac.uk/Articles/v01/i02/editorial.shtml*.

WIIL, Uffe Kock (ed.) (1999). Proc. Fifth Workshop on Open Hypermedia Systems, Darmstadt, Februar 1999 [= Aalborg University, Dept. of Computer Science, TR CS-99-01], 38-42.

WIIL, Uffe Kock; NÜRNBERG, Peter J. (1999). „Evolving Hypermedia Middleware Services: Lessons and Observations." In: Proc. 1999 ACM Symposium on Applied Computing, San Antonio/TX, 1999. New York: ACM, 427-436.

WITT, Andreas (1999). „SGML und Linguistik." In: LOBIN (1999A), 121-154.

WOLFF, Christian (1996). Graphisches Retrieval mit Liniendiagrammen. Gestaltung und Evaluierung eines experimentellen Rechercheverfahrens auf der Grundlage kognitiver Theorien der Graphenwahrnehmung. Konstanz: UVK Informationswissenschaft [= Schriften zur Informationswissenschaft Bd. 24].

WOLFF, Christian (1997). „Möglichkeiten multimedialer und elektronischer Bücher für den Fremdsprachenunterricht." In: WOLFF, Armin; BLEI, Dagmar (edd.) (1997). DaF für die Zukunft. Eine Zukunft für DaF?. Proc. DaF-Jahrestagung 1995, TU Dresden, 1995, 67-78 [= Materialien Deutsch als Fremdsprache Heft 44].

WOLFF, Christian (1998). „Elektronische Wahlverfahren als Beispiel angewandter Kryptographie." In: Zimmermann, Harald H.; Schramm, Volker (edd.) (1998). Knowledge Management und Kommunikationssysteme. Proc. 6. Intern. Symposium f. Informationswissenschaft, ISI '98, 373-384 [= Schriften zur Informationswissenschaft, Bd. 36].

WOLFF, Christian (1999A). Einführung in Java. Objektorientiertes Programmieren mit der Java 2-Plattform. Stuttgart & Leipzig: Teubner.

WOLFF, Christian (1999B). Dokumentation des elektronischen Wahlverfahrens zur GLDV-Wahl. Universität Leipzig, Institut für Informatik, Abt. Automatische Sprachverarbeitung, Arbeitsbericht, August 1999.

WOLFF, Christian (2000). Vergleichende Evaluierung von Such- und Metasuchmaschinen im World Wide Web. Arbeitsbericht, Universität Leipzig, Institut für Informatik [in Vorbereitung].

WOLFF, Christian; QUASTHOFF, Uwe (1999). LEMDIC. Leipzig Multilingual Dictionary Design. Draft Technical Report, Version 0.7, Juli 1999. Universität Leipzig, Institut für Informatik, Abt. Automatische Sprachverarbeitung.

WOMSER-HACKER, Christa; ZETTEL, Walter (1998). Experimentelle Ergebnisse zur Verwendung struktureller Texteigenschaften für eine gewichtete Indexierung." In: HEYER & WOLFF 1998A, 217-224.

WOOD, Laren et al. (ed.) (1999). Document Object Model (DOM) Level 2 Specification. Version 1.0. *World Wide Web Consortium* Working Draft, Juli 1999 [WD-DOM-Level-2-19990719], *http://www.w3.org/TR/WD-DOM-Level-2*.

WOOD, Larry E. (ed.) (1997). User Interface Design: Bridging the Gap from User Requirements to Design. Boca Raton/FL et al.: CRC Press.

WOODRUFF, Allison et al. (1995). "An Investigation of Documents from the World Wide Web." In: Proc. Fifth World Wide Web Conference, Paris, Mai 1996, *http://www5conf.inria.fr/fich_html/papers/P7/Overview.html*.

WORLOCK, David R. (1993). „Network publishing issues and opportunities." In: HENDLEY, T. (ed.). Document Management 93. Proc. Document Management Conference. London: Meckler, 482-488

WUGOFSKI, Ted; SCHMITZ, Patrick; MCCARRON, Shane P. (1999). XHTML™ Events Module. An Updated Events Syntax for XM;L-Based Markup Languages. *World Wide Web Consortium* Working Draft, Dezember 1999, *http://www.w3.org/TR/xhtml-events*.

WURMAN, Richard Saul (1996). Information Architects. Hrsg. von Peter Bradford. Zürich: Graphis Press.

ZETIE, Carl (1998). „User Interface Design: Bridging the GAP. Lerry WOOD, Editor.“ Book Review. In: SigCHI Bulletin 30(3) (1998), 45f.

ZHAI, Chengxiang et al. (1997). “Evaluation of Syntactic Phrase Indexing – CLARIT NLP Track Report.” In: VOORHEES, Ellen M.; HARMAN, Donna K. (edd.) (1997). The Fifth Text Retrieval Conference (TREC-5). Gaithersburg/MD: National Institute of Standards and Technology [= NIST Special Publications 500-238].

ZIEMS, Dietrich; NEUMANN, Gaby (1999). „Von der Idee zum Erfolg: Erfahrungen aus der Entwicklung und Nutzung eines Logistik-Lernsystems.“ In: GIRMES 1999A, 117-168.

ZIMMER, Gerhard (1997). „Mit Multimedia von Fernunterricht zum offenen Fernlernen.” In: ISSING & KLIMSA (1997), 337-352.

17.8 Selbständigkeitserklärung

Hiermit erkläre ich, die vorliegende Habilitationsschrift selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht.

Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder erbrachten Dienstleistungen als solche gekennzeichnet.

(Dr. Christian Wolff)