Uwe M. Borghoff · Peter Rödig
Jan Scheffczyk · Lothar Schmitz

# Langzeitarchivierung



## Methoden zur Erhaltung digitaler Dokumente

dpunkt.verlag

dpunkt.verlag im Internet: www.dpunkt.de

**Uwe M. Borghoff • Peter Rödig •**
**Jan Scheffczyk • Lothar Schmitz**

# Langzeitarchivierung

Digitale Dokumente sind heute ein wesentlicher Bestandteil der vernetzten Welt und existieren in vielfältiger Form: als E-Mails und elektronische Bilanzen, Fotos und Satellitenbilder, Konstruktionszeichnungen u.v.a.m.

Die Speicherung von digitalen Dokumenten birgt aber einige Risiken. Dies gilt besonders, wenn sie über längere Zeiträume hinweg zugreifbar und authentisch bleiben sollen – ein Anspruch, dem sich nicht nur öffentliche und wissenschaftliche Bibliotheken stellen müssen, sondern auch viele Unternehmen und Behörden, die über digitale Archive verfügen.

Wie Langzeitarchivierung in diesen Bereichen funktionieren kann und welche Methoden derzeit erprobt und eingesetzt werden ist Thema dieses Buchs.

Die Autoren stellen zunächst das organisatorische Umfeld und verschiedene technische Ansätze in allgemeinverständlicher Form dar. Darauf aufbauend erläutern sie detailliert die technische Umsetzung sowie aktuelle Entwicklungen. Die Beschreibung einiger internationaler Projekte illustriert schließlich die verschiedenen Herangehensweisen und zeigt, in welchem Rahmen organisatorischer Aufwand und Kosten liegen.

Das Buch wendet sich vor allem an Bibliothekare, Archivare und Informatiker, die für die langfristige Nutzbarkeit ihrer digitalen Dokumentenbestände verantwortlich sind. Aber auch Politiker, Juristen und Wirtschaftswissenschaftler profitieren von der verständlichen Einführung in die Thematik.

ISBN 3-89864-245-3

Thema
- **Datenbanken/ Informationssysteme**
- **Information und Dokumentation**

Leser
- **Bibliothekare**
- **Archivare**
- **Dokumentare**
- **Informatiker**

Betriebssystem
- **unabhängig**

€ 45,00 (D)
€ 46,30 (A)

9 783898 642453

Langzeitarchivierung

**Uwe M. Borghoff** ist Professor für Softwaretechnik an der Universität der Bundeswehr in München. Nach mehrjähriger Tätigkeit als Wissenschaftlicher Mitarbeiter und Habilitation an der TU Mün-chen ging er 1994 an das Xerox Research Centre Europe in Grenoble. Dort war er Senior Scientist und Projekt- und Teamleiter der »Coordination Technology Area«. Seit 1998 ist er an der Universität der Bundeswehr in München, seine Arbeitsschwerpunkte sind Computer-Supported Cooperative Work, Digitale Bibliotheken sowie Knowledge Management.

**Peter Rödig** ist seit 1983 Wissenschaftlicher Mitarbeiter an der Universität der Bundeswehr, zuerst am Labor für Ingenieurinformatik, jetzt am Institut für Softwaretechnologie. Er beschäftigt sich hauptsächlich mit Informations-, Entscheidungshilfe- und Dokumentenmanagement-Systemen sowie Langzeitarchivierung digitaler Dokumente in wissenschaftlichen Bibliotheken.

**Jan Scheffczyk** studierte von 1998 bis 2001 Informatik an der Universität der Bundeswehr München, seit 2002 ist er als Wissenschaftlicher Mitarbeiter am Institut für Software-Technologie tätig. Seine Schwerpunktthemen sind Konsistenzhaltung in Repositories sowie Funktionale Programmierung.

**Lothar Schmitz** ist seit Abschluss seines Informatik-Studiums 1975 wissenschaftlicher Mitarbeiter an der Fakultät für Informatik der Universität der Bundeswehr München. 1982 promovierte er, 2001 folgte die Habilitation. Professurvertretungen führten ihn 1992/93 an die Ludwig-Maximili-ans-Universität München und 1996/97 an die TU Dresden. Seine Arbeitsgebiete sind Software Engineering, Programmiermethodik, Compilerbau sowie Lehr- und Lernsoftware.

Uwe M. Borghoff · Peter Rödig · Jan Scheffczyk · Lothar Schmitz

# Langzeitarchivierung

**Methoden zur Erhaltung digitaler Dokumente**

dpunkt.verlag

Uwe M. Borghoff: borghoff@informatik.unibw-muenchen.de
Peter Rödig: roedig@informatik.unibw-muenchen.de
Jan Scheffczyk: jan@informatik.unibw-muenchen.de
Lothar Schmitz: lothar@informatik.unibw-muenchen.de

Lektorat: René Schönfeldt
Copy-Editing: Maria Dehmer, Berlin
Satz: Autoren
Umschlaggestaltung: Helmut Kraus, Düsseldorf
Druck und Bindung: Koninklijke Wöhrmann B.V., Zutphen, Niederlande

Bibliografische Information der Deutschen Bibliothek

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

ISBN 3-89864-245-3

1. Auflage 2003
Copyright © 2003 dpunkt.verlag GmbH
Ringstraße 19 b
69115 Heidelberg

Released by the
publisher for free use

Die vorliegende Publikation ist urheberrechtlich geschützt. Alle Rechte vorbehalten.
Die Verwendung der Texte und Abbildungen, auch auszugsweise, ist ohne die
schriftliche Zustimmung des Verlags urheberrechtswidrig und daher strafbar. Dies gilt
insbesondere für die Vervielfältigung, Übersetzung oder die Verwendung in
elektronischen Systemen. Alle Informationen in diesem Buch wurden mit größter
Sorgfalt kontrolliert.
Weder Autoren noch Verlag können jedoch für Schäden haftbar gemacht werden,
die in Zusammenhang mit der Verwendung dieses Buches stehen. In diesem Buch
werden eingetragene Warenzeichen, Handelsnamen und Gebrauchsnamen verwendet.
Auch wenn diese nicht als solche gekennzeichnet sind, gelten die entsprechenden
Schutzbestimmungen.

5 4 3 2 1 0

# Geleitwort

Eines der wesentlichen Charakteristika des Menschen, seine Fähigkeit zur Kultur, hängt direkt damit zusammen, dass es ihm möglich ist, Informationen nicht nur persönlich und direkt, sondern über den Raum hinweg zu verteilen und über die Zeit hinweg zu bewahren. Beim Kommen und Gehen der Kulturen und während ihrer jeweiligen Epochen gab es Informationsträger aus höchst unterschiedlichen organischen und anorganischen Stoffen wie Stein, Papyrus, Ton, Rinde, Tierhaut, Papier – um nur einige konkret zu nennen.

Die moderne Informationstechnologie hat in mehrfacher Hinsicht Quantensprünge in unsere Welt gebracht. Raum und Zeit hindern die Verbreitung von Informationen nicht mehr, der Anwender hat die freie Wahl der Informationsträger, die Informationsmenge ist nahezu unbegrenzt. Ein Problem bleibt indessen nicht nur ungelöst, sondern es ist schlechter gelöst als in früheren Perioden: die Langzeitarchivierung oder Langzeitverfügbarkeit der elektronischen Dokumente. Über die Jahrhunderte hinweg ist auf diesem Gebiet sogar eine negative Entwicklung zu beobachten. Während sich die mittelalterlichen Dokumente auf Pergament nachweislich über tausend Jahre hinweg und länger erhalten haben, kam mit dem Hadernpapier ein nur noch Jahrhunderte überdauernder Schriftträger ins Spiel, gefolgt von dem »sauren« Industriepapier des 19. und 20. Jahrhunderts, das heute wegen seiner Brüchigkeit so viel Sorgen macht. Im digitalen Zeitalter endlich ordnen wir Informationsträgern lediglich eine Lebenserwartung von maximal Jahrzehnten, meist nur von Jahren zu. Die Langzeitarchivierung elektronischer Dokumente ist zum Problem geworden, das noch dadurch gesteigert wird, dass dieses Faktum im Allgemeinen gleichmütig hingenommen wird. Die moderne Welt wird von der Vorstellung dominiert, dass das Wissen von gestern durch das aktuelle Wissen von heute überschrieben wird. Die Menschheit ist im digitalen Zeitalter im Begriff, ihre historische Dimension zu verlieren. Dies bringt in Besorgnis erregender Weise eine der Säulen unserer Kultur ins Wanken.

Angesichts dieser Dramatik verdient es mehr als Anerkennung, wenn sich Informatiker des Problems der Langzeitarchivierung annehmen. Die vorliegende Publikation vermisst nicht nur das Arbeitsfeld an sich und legt dadurch ein wesentliches Fundament für einen Dialog derer, denen die Aufgabe der Wissensüberlieferung aufgetragen ist. Darüber hinaus werden die grundsätzlich denkbaren und die aktuell versuchten Lösungsmöglichkeiten vorgestellt. So entstand ein wichtiges Kompendium zur Langzeitarchivierung. Dafür sei den Autoren herzlicher Dank gesagt. Das Werk nährt die Erwartung, dass die düsteren Szenarien einer Kultur ohne Gedächtnis nicht Wirklichkeit werden.

München, im Juni 2003

Dr. Hermann Leskien
Generaldirektor der
Bayerischen Staatsbibliothek

# Vorwort

*Nutzen digitaler Bibliotheken*

Digitale Bibliotheken sind aus unserer global vernetzten Welt nicht mehr wegzudenken. Virtuelle Bibliotheksbesuche, spontane Zugriffe auf internationale Dokumentenbestände und verteilte Suchen über Verbundkataloge sind nur einige Beispiele für Aktivitäten, die wir ganz bequem zu Hause oder am Arbeitsplatz ausüben. Die technischen Entwicklungen auf diesem Gebiet sind rasant, die in den Verbunden nachgewiesenen Dokumentenbestände wachsen explosionsartig und die von Forschern angedachten zukünftigen Dienstleistungen sind atemberaubend.

*Probleme bei der Langzeitarchivierung*

Bei aller Freude über die zukünftigen Dienstleistungen bergen rasante technische Entwicklungen und explosionsartiges Wachstum aber große Risiken. Dies gilt in besonderem Maße, wenn digitale Dokumente über einen längeren Zeitraum in gleicher Art und Weise zugreifbar, in ihrem Erscheinungsbild nachweisbar authentisch und als digitale Artefakte der Nachwelt erhalten bleiben sollen. Mit den damit zusammenhängenden Fragen bei der Langzeitarchivierung beschäftigt sich dieses Buch. In den vereinigten Staaten wird dieser Problematik schon große Aufmerksamkeit geschenkt, vgl. etwa (Bergeron 2002).

*»Stakeholders«*

Diese unserer Meinung nach so wichtigen und drängenden Fragen bei der Langzeitarchivierung elektronischer Dokumente erfordern Kommunikation und Zusammenarbeit zwischen Personen unterschiedlichster Berufsgruppen: Politiker, Juristen, Wirtschaftswissenschaftler und vor allem natürlich Bibliothekare, Archivare und Informatiker müssen gemeinsam an einer Lösung arbeiten. Wir stellen zunächst das organisatorische Umfeld und verschiedene technische Lösungsansätze allgemeinverständlich dar. Dadurch soll eine gemeinsame begriffliche Basis für die Diskussion geschaffen werden. Später nähern wir uns den mit der technischen Umsetzung befassten Personen und beschreiben detaillierter einzelne Ansätze und aktuelle Entwicklungen.

*Zielgruppe und Leserschaft*

Das Buch wendet sich folgerichtig an eine breite Leserschaft aus unterschiedlichen Disziplinen, denen die Sorge und die Suche

nach Sicherung und Bewahrung des gemeinsamen digitalen Kulturguts gemeinsam sind. Zu nennen wären hier etwa

- Bibliothekare, Archivare und Historiker zur Schärfung des Problembewusstseins,
- Politiker, Juristen nebst Wirtschaftswissenschaftlern zur Besinnung auf Maßnahmen zur Verbesserung des organisatorischen Umfelds und nicht zuletzt
- Informatiker und andere an den »neuen Medien« Forschende und Entwickelnde zur Bereitstellung der notwendigen technischen Infrastruktur.
- Studierende technischer Disziplinen profitieren beim Lesen dieses Buches von den detaillierten Beschreibungen einschlägiger Projekte.
- Studierende geisteswissenschaftlicher Fächer erweitern ihr Problemverständnis durch den möglichen Brückenschlag zu den eher technischen Problemen der Langzeitarchivierung.
- Wir wollen auch nicht vergessen all diejenigen aufzurütteln, denen die Bewahrung digitaler Daten nicht nur als Kulturgut am Herzen liegt, sondern auch aus rein ökonomischen Überlegungen heraus. Manager in der Industrie, in Banken und Versicherungen müssen einen langfristigen Zugriff auf ihre geschäftsrelevanten Daten verantworten und sind deshalb für viele Aspekte, die in diesem Buch angesprochen werden, wichtige Adressaten.
- Nach Lektüre des Buches sind nach Meinung der Autoren alle interessierten Leser in die Lage versetzt, die Notwendigkeit *jetzt* zu handeln zu spüren, den aktuellen Forschungsstand und die noch zu klärenden Fragenstellungen zumindest zu erahnen sowie realistische Vorgehensweisen zu erkennen.
- Den Experten wird mit diesem Buch ein umfassendes Kompendium zur Langzeitarchivierung geliefert.

*Aufbau des Buches*

Das Buch gliedert sich in zwei Teile. Der erste Teil umfasst sechs Kapitel, in denen wir einen methodischen Überblick geben.

*Kapitel 1*

Kapitel 1 führt in die Problematik der Langzeitarchivierung digitaler Dokumente ein. Wir diskutieren Probleme der Haltbarkeit elektronischer Speichermedien ebenso wie die zahlreichen Abhängigkeiten digitaler Dokumente von spezifischer Interpretationssoftware. Die wichtigsten technischen Ansätze zur Lösung der Probleme werden einführend vorgestellt. Juristische und gesellschaftliche Fragen, die auf dem Weg zu einer tragfähigen Lösung

der gesamten Archivierungsproblematik digitaler Dokumente beantwortet werden müssen, beschließen dieses einführende Kapitel.

*Kapitel 2*

Ein Referenzmodell, das zu einem internationalen Standard für Langzeitarchivierungssysteme führen könnte, wird mit dem so genannten Open Archival Information System Reference Model (OAIS) vorgestellt. Kapitel 2 schafft damit die Grundlage für eine klare und präzise Begrifflichkeit, wenn es gilt, konkrete Lösungsansätze zu bewerten und den unterschiedlichen Erhaltungsstrategien zuzuordnen.

*Kapitel 3*

Kapitel 3 erläutert die Migration als Erhaltungsstrategie und stellt die unterschiedlichen Verfahrensvarianten in der Umsetzung bei der Langzeitarchivierung vor. Diese Varianten umfassen Verfahren zur Auffrischung digitaler Medien ebenso wie die Migration auf andere digitale und nicht-digitale Medien. Wir kategorisieren digitale Dokumente und geben einen kurzen Überblick über archivierungstaugliche Standardformate.

*Kapitel 4*

Kapitel 4 erläutert die Emulation als die zweite wichtige Erhaltungsstrategie. Diese Strategie zielt darauf ab, die Originalabspielumgebungen digitaler Dokumente durch Emulation auf künftigen Computern dauerhaft am Leben zu erhalten. Wir erörtern verschiedene Varianten an Beispielen und diskutieren danach die Chancen und Risiken des Emulationsansatzes.

*Kapitel 5*

In Kapitel 5 gehen wir näher auf die Beschreibung und Auszeichnung von Dokumenten (engl. *Markup*) ein. Nach einer kurzen Klassifikation unterschiedlicher Markup-Varianten stellen wir Anwendungsmöglichkeiten im Bereich der Langzeitarchivierung vor. Abschließend wenden wir uns dem Problem der Persistenz digitaler Daten zu, das sich immer dann ergibt, wenn auf Daten verwiesen wird, die sich mit der Zeit verändern.

*Kapitel 6*

Standardisierte Dokumentenbeschreibungs-Sprachen für die Auszeichnung digitaler Dokumente mit Metadaten sind Gegenstand der Diskussionen in Kapitel 6. Zunächst behandeln wir die Möglichkeiten zur eingebetteten strukturellen Dokumentenbeschreibung, wobei wir genauer auf die Markup-Sprache XML eingehen werden. Danach zeigen wir einige Möglichkeiten der semantischen Dokumentenbeschreibung auf und erörtern damit verbundene technische Schwierigkeiten. Weiterhin stellen wir mit dem so genannten *Semantic Web* eine attraktive und viel diskutierte Erweiterung des heutigen World Wide Web vor.

*Kapitel 7*

Kapitel 7 beschließt den ersten Teil des Buches mit einer Diskussion der vorgestellten Methoden. Wir stellen einen kombinier-

ten Ansatz zur Langzeitarchvierung digitaler Dokumente vor, der uns aus heutiger Sicht vielversprechend erscheint.

Der zweite Teil widmet sich in drei Kapiteln einschlägigen nationalen wie internationalen Projekten. Sie illustrieren beispielhaft diejenigen Techniken, die im ersten Teil tutoriell angesprochen wurden. Die Wahl der Projekte erfolgte dabei nach reinen Überdeckungsgesichtspunkten und erhebt daher keinen Anspruch auf Vollständigkeit. Der interessierte Leser findet eine umfassende aktuelle Übersicht über fast alle laufenden einschlägigen Projekte unter http://www.nla.gov.au/padi/.[1]

*Kapitel 8* In Kapitel 8 stellen wir verschiedene Initiativen und Projekte vor, die sich mit der Auszeichnung von Dokumenten in der Praxis beschäftigen. So wird mit Pandora beispielsweise ein Projekt beschrieben, das sich ganz explizit dem Speichern und Archivieren von Hyperdokumentmengen im World Wide Web widmet.

*Kapitel 9* Kapitel 9 beleuchtet die Migration anhand vier konkreter Projekte: Wir besprechen hierbei die Victorian Electronic Records Strategy (VERS-Projekt) ebenso wie die beiden Projekte »Preserving the Whole« und »Risk Management of Digital Information«. Letzteres sieht als das brennendste Problem nicht die Instabilität von Medien oder überalterte Hardware an, sondern die Veraltung von Datenformaten und zugehöriger Software. Schließlich beschreiben wir ein umfangreiches Projekt zur Problematik der Datenbankmigration. Erste Ergebnisse einer Umsetzbarkeitsuntersuchung werden vorgestellt.

*Kapitel 10* Kapitel 10 beschreibt zunächst ein Emulationsexperiment, wie es im Rahmen des europäischen Verbundprojekts NEDLIB an der Nationalbibliothek der Niederlande (Koninklijke Bibliotheek) durchgeführt wurde. Anschließend stellen wir einen Ansatz auf Basis des so genannten Universal Virtual Computer (UVC) vor. Der UVC-Ansatz gibt eine erste Antwort auf die Risiken des Migrationsansatzes und auf die Bedenken, die gegen den ursprünglichen Emulationsansatz vorgebracht werden.

*Danksagung* Wir bedanken uns bei der Leitenden Direktorin der Universitätsbibliothek der Universität der Bundeswehr in München, Frau Dr. Stefanie Seidel, die uns in steter Bereitschaft kompetent und freundlich unterstützte. Ihr haben wir es auch zu verdanken, dass Herr Thomas Pieruschka Teile des Abschnitts über OAIS beisteuern konnte. Hierfür gebührt ihm unser Dank. Wir bedanken

[1] Preserving Access to Digital Information (PADI).

uns ferner freundschaftlich bei Frau Dr. Marianne Dörr,[2] Leiterin der Hessischen Landesbibliothek Wiesbaden , die uns mehrere Jahre als Projektpartnerin begleitete und immer mit Sachverstand, Ausdauer und Charme für Diskussionen zur Verfügung stand.

Wir bedanken uns bei unserem ehemaligen Mitarbeiter Knut Weißer für seine Vorarbeiten zum Thema Migration und bei Dr. Wolfram Kahl für seine Beiträge und Hinweise zu Dokumentenbeschreibungs-Sprachen.

Die Deutsche Forschungsgemeinschaft (DFG) unterstützte ein Projekt zum Thema *Entwicklung eines organisatorisch-technischen Konzepts für die Langzeitarchivierung digitaler Publikationen in Bibliotheken*, das wir zusammen mit der Bayerischen Staatsbibliothek als Projektpartnerin bearbeiten durften. Ohne diese initiale Projektförderung wären die Ergebnisse des Buches weniger substanziell und in der Breitenwirkung weniger attraktiv ausgefallen. So wurden beispielsweise Projektergebnisse in einem internationalen Workshop diskutiert und Seminare abgehalten. Im Frühjahr 2000 sowie im Winter 2002 fanden an der Universität der Bundeswehr München einschlägige Seminare zu diesem Thema statt. Wir danken den Teilnehmern Dominik Christian Bódi, Thomas Domschke, Jens Ferchland, Matthias Frank, Sebastian Hess, Thomas Kiesling, Alexander Koenen, Stephan Moritz, Danny Poppe, Mirko Pracht, Daniel Prenzel, Patrick Roiss, Daniel Volk sowie Michael Wellhöfer für ihre Vorträge und eingereichten Ausarbeitungen, die eine erste Grobgliederung für dieses Buch erlaubten.[3]

München, im Juni 2003 U.M.B., P.R.
J.S., L.S.

[2] ehemalige Leiterin Digitalisierung und VD17 an der Bayerischen Staatsbibliothek.

[3] Der geneigte Leser findet unter http://www2.informatik.unibw-muenchen.de/Events/Seminare/ sowie http://www2.informatik.unibw-muenchen.de/LZA/index.html detailliertere Informationen.

# Inhaltsverzeichnis


**I Methodischer Überblick 1**

**1 Langzeitarchivierung digitaler Dokumente . . . . . 3**
1.1 Vorzüge und Nachteile digitaler Medien . . . . . 3
1.2 Herausforderungen, Begriffe und Konzepte . . . . . 5
1.3 Konservierung von Zeichenströmen . . . . . 11
1.4 Technische Ansätze zur Langzeitarchivierung . . . . . 14
1.5 Juristische und gesellschaftliche Fragen . . . . . 21

**2 Das OAIS-Referenzmodell und das DSEP-Prozessmodell 25**
2.1 Das OAIS-Referenzmodell . . . . . 25
2.1.1 Hintergrund . . . . . 26
2.1.2 Das Informationsmodell . . . . . 26
2.1.3 Das Umgebungs- und das Prozessmodell . . . . . 29
2.2 Das DSEP-Prozessmodell . . . . . 32

**3 Migration . . . . . 37**
3.1 Migration: Begriffsfeld und Zielsetzung . . . . . 37
3.2 Migration in der Langzeitarchivierung . . . . . 40
3.2.1 Datenformate als Migrationsziele . . . . . 41
3.2.2 Migration durch Wechseln des Datenträgers . . . . . 44
3.2.3 Migration durch Ändern der logischen Struktur . . . . . 47
3.3 Archivierungsprozesse beim Migrationsansatz . . . . . 50
3.4 Chancen und Risiken des Migrationsansatzes . . . . . 55

**4 Emulation . . . . . 59**
4.1 Emulation: Begriffsfeld und Anwendungsbeispiele . . . . . 59
4.2 Emulation in der Langzeitarchivierung . . . . . 66
4.2.1 Gegenstand der Emulation . . . . . 67
4.2.2 Emulationsvarianten . . . . . 69
4.2.3 Verwendung virtueller Maschinen . . . . . 73
4.3 Archivierungsprozesse beim Emulationsansatz . . . . . 76
4.4 Chancen und Risiken des Emulationsansatzes . . . . . 81

**5 Dokumentenbeschreibung . . . . . 85**
5.1 Einführendes Beispiel . . . . . 85
5.2 Klassifikation . . . . . 87
5.2.1 Prozedurales, strukturelles und semantisches Markup . . . . 87
5.2.2 Embedded Markup Considered Harmful . . . . . 91
5.2.3 Markup-Ebenen . . . . . 93
5.3 Anwendungsmöglichkeiten für die Archivierung . . . . . 95
5.3.1 Anforderungen an die Langzeitarchivierung . . . . . 95
5.3.2 Bibliographische Anforderungen . . . . . 98
5.4 Persistence is a Virtue . . . . . 100
5.4.1 Uniform Resource Identifier, -Name und -Locator . . . . . 101
5.4.2 Verweise auf Dokumente . . . . . 102
5.4.3 Handle-System und Digital Object Identifier . . . . . 106
5.4.4 Überblick . . . . . 108

**6 Standardisierte Dokumentenbeschreibungs-Sprachen . . 109**
6.1 Standards für syntaktische Dokumentenbeschreibung . . . . . 109
6.1.1 Tagged Image File Format (TIFF) . . . . . 110
6.1.2 Portable Document Format (PDF) . . . . . 112
6.1.3 HyperText Markup Language (HTML) . . . . . 114
6.1.4 eXtensible Markup Language (XML) . . . . . 115
6.2 Standards für semantische Dokumentenbeschreibung . . . . . 125
6.2.1 Resource Description Framework (RDF) . . . . . 126
6.2.2 Topic Maps . . . . . 129
6.2.3 Ontologien: DAML+OIL . . . . . 132
6.3 Zukunft – Semantic Web . . . . . 134

**7 Diskussion . . . . . 135**
7.1 Von der Notwendigkeit jetzt zu handeln . . . . . 135
7.2 Was wir wissen und was zu klären bleibt . . . . . 136
7.3 Was realistischer Weise anzustreben ist . . . . . 139
7.4 Ein kombinierter Ansatz . . . . . 142

**II Einschlägige Initiativen und Projekte 145**

**8 Markup in der Praxis . . . . . 147**
8.1 Die Dublin Core Metadata Initiative (DCMI) . . . . . 147
8.2 Die Victorian Electronic Records Strategy (VERS) . . . . . 155
8.3 Die Text Encoding Initiative (TEI) . . . . . 161
8.4 Die Research Libraries Group (RLG) . . . . . 171
8.5 Pandora . . . . . 176

**9 Migration in der Praxis . . . . . 187**
9.1 Migrationsaspekte im VERS-Projekt . . . . . 187
9.2 Preserving the Whole . . . . . 193
9.3 Risk Management of Digital Information . . . . . 201
9.4 Datenbankmigration . . . . . 211
9.4.1 Motivation . . . . . 211
9.4.2 Gesamtarchitektur . . . . . 215
9.4.3 Endgültiges Lösen digitaler Objekte vom ursprünglichen Medium . . . . . 216
9.4.4 Dienste eines Datenbank-Managementsystems . . . . . 224
9.4.5 Detailuntersuchungen und praktische Versuche . . . . . 232
9.4.6 Diskussion . . . . . 249

**10 Emulation in der Praxis . . . . . 251**
10.1 Emulationsexperiment von Rothenberg . . . . . 251
10.2 Universal Virtual Computer . . . . . 260

**Abbildungsverzeichnis . . . . . 267**

**Tabellenverzeichnis . . . . . 271**

**Literaturverzeichnis . . . . . 273**

**Index . . . . . 279**

# Teil I

# Methodischer Überblick

# 1 Langzeitarchivierung digitaler Dokumente

*Dieses Kapitel führt in die Problematik der Langzeitarchivierung digitaler Dokumente ein. Abschnitt 1.1 diskutiert Probleme der Haltbarkeit elektronischer Speichermedien sowie die Abhängigkeit digitaler Dokumente von spezifischer Software zu ihrer Interpretation. Abschnitt 1.2 dringt weiter zu den Ursachen dieser Probleme vor und erläutert relevante Begriffe und Konzepte. Grundlage aller Erhaltungsmaßnahmen für digitale Dokumente ist die in Abschnitt 1.3 beschriebene Bewahrung der zugrunde liegenden Zeichenfolgen und Metadaten. Abschnitt 1.4 erläutert die verschiedenen technischen Ansätze zur Archivierung im Zusammenhang. Die wichtigsten dieser Ansätze werden in den folgenden Kapiteln vertieft dargestellt. Juristische und gesellschaftliche Fragen, die auf dem Weg zu einer tragfähigen Lösung der Archivierungsproblematik digitaler Dokumente beantwortet werden müssen, sind Gegenstand von Abschnitt 1.5.*

## 1.1 Vorzüge und Nachteile digitaler Medien

> Digital documents last forever – or five years, whichever comes first. (Rothenberg 1999)[1]

*Dokumente erhalten*

Bibliotheken, Archive und ähnliche Institutionen dienen der Erhaltung von Dokumenten mit bleibendem wissenschaftlichem, künstlerischem oder gesellschaftlichem Wert, sodass künftige Generationen ebenso wie wir heute darauf zugreifen können. Als erhaltenswert betrachten wir neben dem gesamten aktuellen Buchangebot auch wertvolle Originalmanuskripte, Ton- und Bildträger sowie in zunehmendem Umfang digitale Datenträger.

*Vorzüge digitaler Dokumente*

Während alte Folianten teure Konservierungsmaßnahmen erfordern und auf säurehaltiges Papier gedruckte Werke durch Spei-

[1] siehe http://www.clir.org/pubs/reports/rothenberg/introduction.html.

cherung auf Mikrofilm vor dem endgültigen Untergang bewahrt werden müssen, scheinen sich digitale Medien durch ideale Eigenschaften auszuzeichnen: Bitfolgen lassen sich im Prinzip über längere Zeiträume ohne Informationsverlust aufbewahren. Die rasant steigende Speicherkapazität[2] erlaubt es schon heute, umfangreiche Nachschlagewerke auf einer einzigen CD zu speichern. Bei voranschreitender Miniaturisierung der Datenträger scheint sich mit der digitalen Speicherung eine attraktive, universell einsetzbare Basis der Archivierung anzubieten, die noch dazu weitere Vorteile mit sich bringt: Nicht nur die in den elektronischen Katalogen gespeicherten Angaben, sondern auch den gesamten Inhalt digitaler Dokumente können wir in Sekundenschnelle durchsuchen lassen. Ebenso rasch bringt das Internet Dokumente aus aller Welt an unseren Arbeitsplatz oder nach Hause, wo wir sie mit vielfältigen Werkzeugen bearbeiten und zu neuen Dokumenten zusammensetzen können.

Wo liegt dann das Problem?

*nur indirekt lesbar*

Wer über einen längeren Zeitraum hinweg Computer zur Textverarbeitung genutzt hat, kennt die (mehrteilige) Antwort: Zunächst einmal sind elektronische Datenträger für Menschen nicht unmittelbar lesbar. Während wir Jahrtausende alte Höhlenmalereien mit etwa dem gleichen Sinnesapparat wahrnehmen können wie die Zeitgenossen des Künstlers, erschließt sich uns der Inhalt der von uns selbst erzeugten Textdateien nur indirekt mit Hilfe eines geeigneten Computersystems. Für sich genommen ist das kein Problem. Auch der Tatsache, dass Speichermedien wie Disketten, Festplatten und CDs altern und ausfallen können, lässt sich mit geeigneten Maßnahmen (Sicherheitskopien) so begegnen, dass kein Datenverlust eintritt.

*Alterung durch technischen Wandel*

Schneller als die Datenträger altern, veralten aber die Komponenten der Computersysteme: Zu 5,25-Zoll- oder gar 8-Zoll-Disketten finden sich heute kaum mehr geeignete Abspielgeräte und selbst wenn, dann ließen sie sich mangels neuer Treibersoftware nicht an moderne Computer anschließen. Noch größer sind Vielfalt und Wandel bei der Software. Im Laufe der Jahre entstanden tausende verschiedener Editoren für digitale Dokumente (Text-, Graphik-, Tabellenkalkulations-, Folien-, Webseiten-Editoren etc.), die meisten von ihnen mit jeweils eigenen internen Datenformaten. Wegen der Unverträglichkeit dieser Formate ist es häufig gar

[2] Infolge des »Gesetzes von Moore« verdoppeln sich die Verarbeitungsgeschwindigkeit und der Speicherumfang von Computern etwa alle 18 Monate.

nicht oder nur unter erheblichem Qualitätsverlust möglich, Dokumente zwischen solchen Editoren auszutauschen.

*aufwändige Erhaltungsmaßnahmen*

Langfristig überstehen nur relativ wenige dieser Systeme den Wettbewerb im Markt. Nutzer kommerziell unterlegener Produkte stehen am Ende vor dem Problem, wie sie ihren Bestand veralteter Dokumente in eine neue Umgebung retten können. Mit arbeitsaufwändigen, »heroischen Maßnahmen« wie der Neuerstellung der wichtigsten Dokumente oder dem Einscannen von Texten in neue Systeme (mit anschließender manueller Nachbearbeitung) gelingt dies in der Regel. Offensichtlich ist diese Vorgehensweise nicht für die riesigen Dokumentenbestände von Bibliotheken und Archiven geeignet, in denen mit wenig Personal auf Dauer der Fortbestand und der Dokumentenzugriff gesichert werden müssen.

## 1.2 Herausforderungen, Begriffe und Konzepte

*digitale Dokumente sind Bitströme*

Computer speichern Daten als Folgen von Nullen und Einsen, d. h. als so genannte *Bitströme*. Auch digitale Dokumente sind im Kern nichts anderes als solche Bitströme. Wir sprechen etwas allgemeiner von *digitalen Zeichenströmen*, da man statt mit Nullen und Einsen ebensogut mit anderen Basiszeichen arbeiten kann. Auf die Eigenschaften digitaler Zeichenströme und ihre Konservierung werden wir in Abschnitt 1.3 eingehen.

*Zugang über Abspielsysteme*

Für den Menschen ist der reine Datenstrom eines digitalen Dokuments ohne ein passendes *Abspielsystem* nicht zugänglich. Erst in einer geeigneten Umgebung werden aus den Zeichenfolgen wieder die Texte, Graphiken, Bewegungen, Töne und weiteren Bausteine, aus denen sich ein multimediales Dokument zusammensetzt. Häufig ist die einzig geeignete Umgebung das Editorprogramm, mit dem das Dokument ursprünglich erstellt wurde bzw. eine hinreichend ähnliche Version dieses Programms. Wie schon erwähnt, unterscheiden sich die Editoren verschiedener Softwarehersteller und in geringerem Maße auch schon die verschiedenen Versionen des gleichen Editorprogramms in der Art und Weise, wie sie digitale Dokumente strukturieren und als Datenströme speichern.

*Beispiel: unterschiedliche Textformate*

Betrachten wir als typisches Beispiel zwei einfache, hypothetische Texteditoren, T1 und T2: T1 speichert einen Text als Folge von Zeilen und jede Zeile als Folge von Zeichen. Zu jedem Zeichen legt T1 weitere Informationen ab, u. a. den Zeichensatz, die

Größe des Zeichens, ob es fett und/oder kursiv ist und wie viele Bildpunkte oberhalb bzw. unterhalb der Zeilengrundlinie sich das Zeichen befindet. Ganz anders die Speicherung von Texten bei T2: Hier wird der gesamte Text als eine Folge von Textzeichen gespeichert, in die so genannte *Steuerkommandos* eingestreut sind. Beispiele für solche Steuerkommandos sind `Absatzende`, `fett ein`, `kursiv aus`, `verwendeZeichensatz(font)`, `hochstellen`, `tiefstellen`, `auf Zeile` und `verwendeFarbe(color)`. Während die Entwickler von T1 eine möglichst flexible Positionierung der Zeichen auf der Textseite anstrebten, ließen sich die Entwickler von T2 eher von den Eigenschaften eines Druckers leiten. T1-Dokumente sind schwarzweiß und können wegen der punktgenauen Zeichenpositionierung auch komplizierte wissenschaftliche Formeln enthalten. T2-Dokumente dagegen sind weniger für Formeln geeignet, dafür aber bunt und kompakter in der Speicherung.

*Konversion zwischen diesen Formaten*

Die oben skizzierten Interna von T1 und T2 bezeichnet man als ihre *Datenformate*. Wegen der unterschiedlichen Datenformate kann man T1-Dokumente offenbar ebenso wenig mit T2 betrachten wie T2-Dokumente mit T1. Aus kommerziellen Gründen (Abwerbung von Kunden) bieten Hersteller von Texteditoren häufig Konversionsprogramme an, mit denen sich Dokumente eines fremden Datenformats in das eigene Datenformat umwandeln lassen. Man spricht von so genannten *Importfiltern*. Beispielsweise würden die Hersteller von T1 einen Importfilter für T2-Dokumente anbieten, um T2-Nutzern den Übergang zum eigenen System zu erleichtern. Dennoch müssen T2-Nutzer Informationsverlust hinnehmen, wenn sie den Umstieg auf T1 vornehmen: Da T1-Dokumente stets schwarzweiß sind, gehen die in T2-Dokumenten eventuell vorhandenen Farbunterscheidungen verloren. (Auch die umgekehrte Konversion von T1 nach T2 wäre zwangsläufig mit Informationsverlust verbunden, da sich die feine Zeichenpositionierung von T1 nur grob mit T2 annähern ließe.)

*Filter und standardisierte Austauschformate*

Wegen der unbegrenzten Vielfalt der Datenformate werden Importfilter jeweils nur für einige wenige, gängige Dokumentenformate erstellt. Anbieter weniger populärer Editoren stellen manchmal auch *Exportfilter* bereit, mit denen sich Dokumente aus dem eigenen in ein gängigeres Format umwandeln lassen. Damit geben sie den eigenen Kunden die Sicherheit, die erstellten Dokumente auch dann weiter verwenden zu können, wenn der eigene Editor sich wirtschaftlich nicht behaupten sollte. Angesichts dieses Wildwuchses an Datenformaten liegt es nahe, an *Standardisierung* zu denken oder zumindest ein gemeinsames Austauschformat zu schaffen: Theoretisch hätte man die vollständige Kon-

vertierbarkeit zwischen beliebigen Datenformaten erreicht, wenn es ein solches *Austauschformat* AF und zu jedem Editor je einen Import- und einen Exportfilter zwischen AF und dem eigenen Format gäbe. Man macht sich leicht klar, dass dann aber alle Editoren (anders als oben T1 und T2) die gleichen oder wenigstens äquivalente Ausdrucksmöglichkeiten besitzen müssten (vgl. Abb. 1.1).

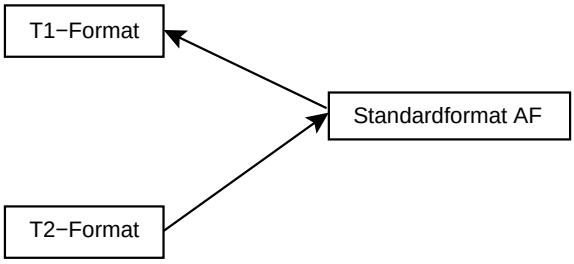



**Abbildung 1.1**
*Austauschbarkeit über Standardformate*

*Fortschritt als Hindernis*

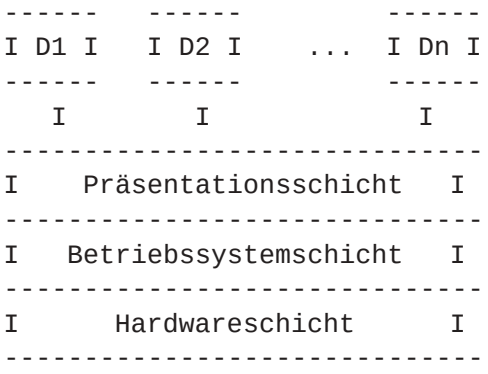
Das Haupthindernis bei einem Standardisierungsansatz ist paradoxerweise der Fortschritt, da er immer neue Möglichkeiten mit sich bringt. Zu reinen Textformaten sind in relativ kurzer Zeit viele neue Formen hinzugekommen: Graphik, Sound, Video, Hypertext, Spreadsheets, jede einzelne dieser Formen wiederum in vielen proprietären Datenformaten. Zusätzlich werden diese Formen in allen denkbaren Kombinationen zu elektronischen Multimediadokumenten verbunden. Eine Konsolidierung dieser Entwicklung oder zumindest eine Verlangsamung der Innovationsrate ist nicht in Sicht. Solchen anhaltenden Veränderungen kann ein einmal gewähltes, festes Datenformat auf Dauer grundsätzlich nicht gerecht werden. Einen Ausweg aus diesem Dilemma bieten Sprachen, in denen Datenformate beschrieben und erweitert werden können. Dazu gehören die in Kapitel 5 behandelten *Auszeichnungssprachen* wie SGML und XML. Auf Basis dieser Standards werden derzeit ergänzende Sprachen und Werkzeuge entwickelt, mit denen man die systematische Transformation von Dokumenten aus einem Format in andere beschreiben und durchführen kann.

*ein möglicher Ausweg*

*Wieso immer neue Formate?*

Die neuen Datenformate unterscheiden sich von ihren Vorgängern nicht nur durch ihre Vielfalt, sondern auch durch eine reichere konzeptionelle Strukturierung: In alten Systemen (wie T1 und T2) hat man beispielsweise Tabellen, mathematische Formeln und sogar Liniengraphiken wie gewöhnlichen Text aus Buchstaben und Symbolen zusammengesetzt und für das menschliche Auge durch Einstreuen von Zwischenräumen notdürftig erkennbar gemacht. Im Unterschied zu neueren Textformaten, die *Konzepte* wie

Tabellen und Formeln als Bausteine bereithalten, wurden früher die gleichen Konzepte nicht explizit dargestellt, sondern wie beschrieben durch eine Ad-hoc-Formatierung nachgebildet. Die Lesbarkeit solcher Nachbildungen hängt übrigens wesentlich von einer scheinbar nebensächlichen Eigenschaft früher Editoren ab: Es gab beim Drucken nämlich nur Zeichensätze fester Breite. Abbildung 1.2 zeigt eine einfache Liniengraphik und ihre Darstellung durch das für dieses Buch verwendete Textsystem LaTeX (in Fließtext fasst LaTeX Zwischenraumfolgen und einzelne Zeilenwechsel zu je einem Zwischenraum zusammen):




—— —— —— I D1 I I D2 I ... I Dn I —— —— —— I I I —— ———————— I Präsentationsschicht I ———————— —— I Betriebssystemschicht I ———————— I Hardwareschicht I ———————— ———

**Abbildung 1.2** Eine Liniengraphik und ... ... was LaTeX daraus macht

*Probleme bei der Konversion von Texten*

Da ein Stillstand bei der Einführung neuer Konzepte nicht zu erwarten ist, ergibt sich also ein weiteres Problem: Die Konversion von Texten in neuere, konzeptionell reichere Formate erfordert das Erkennen von Konzepten, die durch Ad-hoc-Formatierung nur implizit gegeben sind. Das setzt ein gewisses inhaltliches Verständnis des Textes voraus und ist daher nur schwer, wenn überhaupt durch ein Programm automatisch und verlässlich zu erledigen.

*neue Formate neue Probleme*

*Was ist ein »Abspielsystem«?*

Bislang haben wir »Abspielsystem« gleichgesetzt mit dem Editor- oder Anzeigeprogramm, mit dem das Dokument am Bildschirm betrachtet werden kann. Das ist aber nur der Kern einer funktionsfähigen Abspielumgebung, die daneben mindestens auch die Hardware und Betriebssoftware eines passenden Computers, eventuell auch zusätzliche Geräte und zugehörige Treiberprogramme umfasst. Die obige Liniengraphik stellt den Gesamtaufbau einer Abspielumgebung schematisch dar:

*Hardware*

- Zur Hardwareschicht gehört neben dem »nackten« Computer mit Prozessor, Arbeitsspeicher und Busverbindungen ei-

ne Reihe von notwendigen oder wünschenswerten Ergänzungen wie Graphikkarte und Bildschirm, sekundäre, dauerhafte Speicher (Festplatte, Magnetband, optische Medien etc.) und Eingabegeräte (u. a. Tastatur, Maus, Joystick, Mikrofon).

- ❏ Das Betriebssystem und die Treiberprogramme bilden zusammen eine Schicht von Programmen, die es dem menschlichen Nutzer erlauben, von den für ihn weniger relevanten, physikalischen Eigenschaften der Hardware zu abstrahieren und sich auf die so genannten »logischen« Eigenschaften des Rechensystems zu konzentrieren. Beispielsweise sieht der Nutzer die Festplatte nicht als eine in Ebenen, Segmente und Spuren gruppierte Menge von Magnetisierungspunkten, sondern als ein Dateisystem mit ineinander geschachtelten Verzeichnissen, welche Dateien mit elektronischen Dokumenten enthalten.

*Systemsoftware*

- ❏ Die Programme der Präsentationsschicht (u. a. Texteditor, Zeichenprogramm, Web-Browser) setzen auf den darunter liegenden Schichten auf und sorgen für eine geeignete Anzeige der Dokumente D1, D2, ..., Dn. Welches der Präsentationsprogramme geeignet ist, hängt vom Typ des elektronischen Dokuments ab (z. B. Word2000-Datei, TIFF-Graphik, HTML-Datei).

*Präsentationssoftware*

Ein Abspielsystem ist also kein monolithischer Block, sondern wird aus einer Vielzahl von Komponenten zusammengesetzt, die in komplizierter Weise miteinander interagieren.

*Aktualisierungsdruck bei Abspielsystemen*

Ebenso wie die Präsentationsprogramme sind die anderen Komponenten dieser Systeme einem steten Wandel ausgesetzt: Laufend werden neue, bessere Gerätetypen und -varianten entwickelt. Damit diese genutzt werden können, bedarf es neuer Systemsoftware-Komponenten, eventuell sogar Anpassungen der Präsentationsprogramme. Trotz einer gewissen Marktbereinigung bei den Betriebssystemen (hin zu Windows und Unix) gibt es auch davon eine Unmenge von Varianten und Versionen. Aufgrund anhaltender Innovationsschübe – besonders eindrucksvoll darunter die Einführung graphischer Benutzungsoberflächen und das Entstehen des Internets – wird Software auch ohne physikalische Abnutzung regelmäßig nach wenigen Jahren obsolet und ist durch verbesserte Versionen oder neuere Produkte zu ersetzen. Da neue Versionen von Anwendungsprogrammen natürlich nur für aktuelle Betriebssystemversionen entwickelt werden, bedingen sich Erneuerungen von Betriebssystem und Anwendungssoftware häufig

gegenseitig. Die Einbindung von immer mehr Rechnern ins Internet verstärkt diesen Aktualisierungsdruck.

*Folgeprobleme*

Die oben am Beispiel von Texteditoren beschriebenen Herausforderungen bei der Archivierung elektronischer Dokumente potenzieren sich insgesamt, weil

- wie dargestellt Änderungen an einer Komponente der Abspielumgebung Änderungen an den anderen Komponenten nach sich ziehen;
- bei Multimediadokumenten verschiedene Präsentationsprogramme in einer (Browser-)Umgebung interagieren und daher zusammenpassen müssen.

*bibliographische Metadaten*

Bei der elektronischen Archivierung sind neben dem eigentlichen Dokument eine Reihe von Angaben über das Dokument, so genannte *Metadaten*, aufzubewahren. Dabei handelt es sich zum Teil um bibliographische Angaben, wie man sie schon in herkömmlichen Bibliothekskatalogen findet, z. B.:

- Autor, Titel, Verlag, Erscheinungsjahr und evtl. die ISBN-Nummer: Angaben, die das Dokument eindeutig identifizieren.
- Fachgebiet und Schlagwörter: Angaben, die neben Autor und Titel für die Dokumentenrecherche benötigt werden.
- Speicherort (analog zu Aufstellungsort): Angabe, wo man das Dokument tatsächlich findet.

*Metadaten für die Langzeitarchivierung*

Weitere Metadaten sind spezifisch für die elektronische Langzeitarchivierung:

- Codierung und Datenformat: Angaben, die zum Lesen des Dokuments unbedingt erforderlich sind.
- »Migrationshistorie« bzw. Beschreibung der Originalumgebung: Angaben, welche auf den Originalzustand des Dokuments schließen lassen.
- Copyright-Vermerke u. ä.: Angaben, welche die legalen und sonstigen Konditionen des Zugriffs auf das Dokument beschreiben.

Der Hintergrund zu den letzten drei Angaben wird in den Abschnitten 1.3–1.5 erläutert.

*Metadaten beim Dokument speichern?*

Elektronische Dokumente und ihre Metadaten gehören einerseits eng zusammen, da das eine ohne das andere wertlos ist und daher beide unbedingt konsistent gehalten werden müssen. Andererseits gibt es Gründe, beide getrennt voneinander zu speichern:

Metadaten werden u. a. verwendet, um Dokumente aufzufinden. Das geht schneller, wenn Kataloginformationen gesondert aufbewahrt werden. Noch gewichtiger ist das Argument, dass der Dokumenteninhalt überhaupt erst dann gelesen werden kann, wenn man anhand der Metadaten die Codierung und das Datenformat des Dokuments bestimmt hat.

*elektronische Archive*

Die Frage der Speicherung und Kapselung von elektronischen Dokumenten und Metadaten ist grundlegend für die Organisation und den Betrieb eines elektronischen Archivs, ebenso die Struktur der von dem Archiv laufend durchzuführenden Prozesse (wie Aufnahme eines Dokuments, Speicherung und Auffrischung, Ausleihe von Dokumenten). Mit dem vom *Consultative Committee for Space Data Systems* (CCSDS) im Auftrag der ISO entwickelten OAIS-Referenzmodell[3] erhält man hierzu eine allgemein akzeptierte, in den Details aber noch auszugestaltende Vorlage.[4] Das OAIS-Referenzmodell ist Gegenstand von Kapitel 2.

## 1.3 Konservierung von Zeichenströmen

Grundlage einer jeden Archivierung digitaler Dokumente ist die Konservierung von Zeichenströmen. Bevor wir in Abschnitt 1.4 auf die unterschiedlichen technischen Ansätze zur Langzeitarchivierung eingehen, soll hier die gemeinsame Grundlage aller Verfahren dargestellt werden. Ausgangspunkt ist die folgende Erklärung:

*Was ist ein »digitales Dokument«?*

**Erklärung:** Ein *digitales Dokument* besteht aus dem digitalen Dokumenteninhalt und den zum Dokument gehörigen Metadaten. Der *digitale Dokumenteninhalt* ist als ein Strom (Folge) von Zeichen eines endlichen Zeichensatzes A codiert. Zu den Metadaten gehört die Angabe des zur Codierung des Dokumenteninhalts verwendeten Zeichensatzes A sowie weitere Angaben (u. a. Autor, Titel). Die Metadaten sind als Folgen eines (Basis-)Zeichensatzes B codiert. Die Zeichensätze A und B können, müssen aber nicht übereinstimmen. Die Struktur der Metadaten und der zugrunde liegende Zeichensatz B sind archivabhängig festgelegt.

*z. B. PostScript*

Das Adjektiv »digital« weist auf die wesentliche Eigenschaft hin, nämlich die Codierung von Inhalten als Zeichenfolgen. Betrachten wir als Beispiel ein Graphikdokument, das u. a. eine gerade

[3] Das Akronym OAIS steht für *Open Archival Information System*.

[4] siehe http://www.ccsds.org/RP9905/RP9905.html.

Linie vom Punkt mit den Koordinaten $(12.4, 3.1415)$ zum Punkt $(202.4, 47.11)$ enthält. Liegt das Graphikdokument im PostScript-Format vor, dann enthält der digitale Dokumenteninhalt u. a. die Zeichenfolge

```
12.4 3.1415 moveto 202.4 47.11 lineto
```

die textuell ausdrückt, dass sich der Zeichenstift in gehobenem Zustand zum Punkt $(12.4, 3.1415)$ bewegen und von dort auf dem Papier zum Punkt $(202.4, 47.11)$ fahren soll. (Ein »PostScript-fähiger« Laserdrucker ist in der Lage, diese textuelle Beschreibung in eine gerade Linie umzusetzen.)

*Standardcodes wie ASCII und Unicode*

In der Praxis häufig verwendete Zeichensätze sind die von Standardcodes wie ASCII, EBCDIC, Unicode und UCS. Solche Codes enthalten eine feste, endliche Menge von Zeichen und legen eine Codierung dieser Zeichen durch Bitfolgen fest. Die Codes ASCII und EBCDIC verwenden dabei Bitfolgen der Länge 8, Unicode Bitfolgen der Länge 16 und UCS sogar Bitfolgen der Länge 32. Während ASCII an Zeichen nur die Groß- und Kleinbuchstaben, die arabischen Ziffern, die Interpunktionszeichen, mathematische Grundsymbole wie $+$ und $<$ und etwa noch mal so viele Sonderzeichen enthält, deckt Unicode die Schriftzeichen der wichtigsten lebenden Sprachen ab. Mit UCS wird es künftig möglich sein, alle bekannten Schriftzeichen zu codieren (vgl. hierzu Rechenberg und Pomberger (1997), Kapitel B1). Beispiel: Die ASCII-Codierung des Zeichens »m« ist die Bitfolge `01101101`, die des Zeichens »5« ist `00110101`.

*»binäre« Datenformate*

**Erklärung:** Ein *Datenformat* heißt *binär*, wenn die zugehörigen Dokumenteninhalte nur Folgen von Nullen und Einsen sein können. Wenn der Dokumenteninhalt aus Zeichen eines anderen Zeichensatzes (z. B. dem des ASCII-Codes) besteht, dann spricht man von einem *textbasierten* Datenformat.

In diesem Sinne ist das oben verwendete PostScript-Format ein textbasiertes Datenformat. Bei elektronischer Speicherung eines solchen Textes wird entsprechend dem Code die Zeichenfolge in eine Bitfolge umgewandelt und so letztlich auch binär gespeichert.

*Wo werden digitale Dokumente gespeichert?*

Derzeit werden elektronische Dokumente häufig als *Dateien* eines Rechensystems abgelegt. Andere Möglichkeiten sind die Speicherung in einer *Datenbank* oder in einem dedizierten *Dokumentenverwaltungssystem*. Schließlich gibt es so genannte *dynamische Dokumente*, deren Datenströme erst beim Zugriff auf das Dokument entstehen, indem aktuelle Daten bei Bedarf nach

einem vorgegebenen Schema (per Programm) zusammengestellt werden.

*fortgesetztes Kopieren als Erhaltungsstrategie*

Ebenso wie Tontafeln, Pergament, Papier und andere physikalische Träger von Dokumenten sind elektronisch lesbare Datenträger wie Magnetplatten und -bänder und auch optische Speichermedien wie CDs nur von begrenzter Haltbarkeit. Selbst bei regelmäßiger Auffrischung des Inhalts ist der ursprüngliche Datenträger nach wenigen Jahren unbrauchbar. Wie schon dargestellt, liegt ein noch größeres Problem darin, dass der technische Fortschritt laufend bessere Speichermedien hervorbringt und Abspielgeräte für den jeweils aktuellen Standard nur relativ kurze Zeit verfügbar sind. Die in der Datenverarbeitung allgemein bekannte Konsequenz ist, dass sich elektronische Daten und Dokumente nur durch fortgesetztes Kopieren und durch Migration von Speichermedium zu Speichermedium erhalten lassen. Kapitel 3 stellt Migrationsverfahren für elektronische Dokumente ausführlich dar.

*Transformationen wie Datenkompression und Verschlüsselung*

Die für die Codierung von Texten verwendeten Codes (ASCII etc.) haben sich als verhältnismäßig dauerhaft erwiesen. Außerdem kann man auf älteren Codes basierende Datenströme so in Unicode- oder UCS-basierte Datenströme umwandeln, dass die ursprünglichen Datenströme bei Bedarf bit-identisch wiederhergestellt werden können. *Aus heutiger Sicht erlauben Maßnahmen wie fortdauerndes Kopieren, Migration und reversibler Übergang zu neuen Codes eine sichere elektronische Konservierung von Zeichenströmen über lange Zeiträume hinweg.* Eine zusätzliche Speicherung auf haltbaren Medien (Papier, Mikrofilm, Metall o. ä.) kann die Sicherheit des Erhalts von Zeichenströmen weiter erhöhen. Neben dem Übergang zu neueren Codes werden elektronische Datenströme gelegentlich weiteren Transformationen unterworfen, z. B. der Datenkompression zum Zweck kompakter Speicherung oder der Verschlüsselung vertraulicher Dokumenteninhalte. Sofern solche Transformationen eindeutig (und per Programm) umkehrbar sowie in den Metadaten ausreichend dokumentiert sind, beeinträchtigen sie die Konservierung des Zeichenstroms nicht.

## 1.4 Technische Ansätze zur Langzeitarchivierung

*Aufgabe: Formate interpretieren*

Während sich die Konservierung von Zeichenströmen als lösbare technische Aufgabe herausstellt, ist damit für das Problem der elektronischen Langzeitarchivierung nur eine wichtige Voraussetzung geschaffen. Es bleibt die Aufgabe, die ungeheuer vielfältigen, auf binären oder textbasierten Zeichenströmen aufsetzenden Datenformate von Multimediadokumenten zu interpretieren, d. h., auch nach langer Zeit die textuellen, graphischen und akustischen Bestandteile solcher Dokumente wieder in der »gleichen Form« darzustellen, die schon ihre Erzeuger wahrgenommen haben. Bei interaktiven Dokumenten wie Hypertext, Animationen oder Spreadsheets kommt die Forderung nach »gleichartiger Reaktion« auf Benutzereingaben hinzu.

*Wie war das bei Tonträgern?*

Einige *Analogien zur Langzeitarchivierung elektronischer Dokumente* bietet das Feld der Schallkonservierung, das mit Edisons Phonographen 1877 seinen Anfang nahm. Das Prinzip, Schallwellen in wellenförmige Auslenkungen einer Rille auf dem Tonträger umzusetzen, hat sich – unter erheblichen Modifikationen – bis zu heutigen Schallplatten erhalten. Die Verbesserung von Tonqualität, Haltbarkeit und Aufnahmedichte motivierten die Wechsel des Tonträgermaterials von Stanniol über eine Wachsmasse zu Schellack-Harz und Vinyl-Kunststoff. Weitere Technologiesprünge sind verbunden mit elektrischer Wiedergabetechnik, Stereo- und Quadrophonie sowie HiFi-Technik. Daneben entwickelten sich Alternativen, vornehmlich die elektromagnetische Speicherung auf Magnetbändern, die es dem Benutzer erlauben, eigene Aufnahmen zu erstellen und vorhandenes »Tonmaterial« zu kopieren. Mit der Einführung von Digitalaufnahmen hat sich die Audiotechnik an die rasante Entwicklung der digitalen Speichermedien angehängt: Aktuell sind CD-ROMs, Mini-Disks und MPEG3-Player.

*laufend neue Abspielgeräte!*

Wer von diesem Fortschritt profitieren möchte, sieht sich genötigt, jeweils nach einigen Jahren seine Datenträger (und Abspielgeräte!) komplett auszutauschen: Schellacks gegen Langspielplatten, Mono- gegen Stereoplatten, Platten und Bänder nacheinander gegen CD-ROMs und Minidisks. Dies nützt der Industrie und beflügelt den Fortschritt. Wer nicht mitmacht, steht bald ohne Unterstützung da, weil neue Tondokumente nicht in veralteter Technologie angeboten werden und weil für die Abspielgeräte keine Ersatzteile zu bekommen sind. Schellackliebhaber finden Platten und passende Grammophone allenfalls beim Trödler oder im Mu-

seum. Auch wenn die neue Technologie prinzipiell das Kopieren (und dadurch Erhalten) der alten Tondokumente erlaubt, wird wegen Mängeln der alten Träger (Rauschen, fehlende Stereophonie etc.) vielfach darauf verzichtet.

*Digitaltechnik noch schnelllebiger*

Völlig analog, nur wegen der erheblich kürzeren Innovationszyklen wesentlich dramatischer verläuft die Entwicklung in der Digitaltechnik.

*Authentizität einer Aufnahme*

Versetzen wir uns nun in die Rolle einer Musikwissenschaftlerin, die Originalaufnahmen des großen Enrico Caruso studieren möchte. Je nach Zweck der Untersuchung wird sie einen von zwei grundverschiedenen Zugängen wählen. Zum einen kann sie versuchen, zeitgenössische (Schellack-)Aufnahmen auf einem passenden Grammophon abzuspielen. Zum anderen kann sie sich eine mit Rauschfiltern und anderen technischen Transformationen nachträglich auf den heutigen Stand gebrachte Aufnahme beschaffen. Wenn es darum geht, die Interpretationen einer bestimmten Arie durch Caruso und Pavarotti zu vergleichen, ist der zweite Weg vermutlich angemessen. Soll dagegen ein größtmögliches Maß an Authentizität sichergestellt werden, dann wird unsere Musikwissenschaftlerin eher den ersten Weg beschreiten.

*Originale erhalten oder anpassen?*

Technische Maßnahmen zur elektronischen Langzeitarchivierung lassen sich in die gleichen Kategorien einteilen: auf der einen Seite Maßnahmen, die möglichst nah am Originaldokument und der Originalabspielumgebung zu bleiben trachten; auf der anderen Seite Maßnahmen, welche das Dokument fortlaufend dem neuesten Stand der Technik anzupassen versuchen.

### Migration

*Migration*

In der Datenverarbeitung wie in der Audiotechnik werden Maßnahmen der zweiten Kategorie häufig und auf breiter Front eingesetzt. Da hierbei Daten fortlaufend von einer Darstellungsform zur nächsten »wandern«, spricht man von *Migration*. Es bietet sich an, diese bekannten und bewährten Verfahren auch zur Langzeitarchivierung elektronischer Dokumente zu verwenden. Die Migration von Zeichenströmen wurde in Abschnitt 1.3 kurz erwähnt. Die Vorteile der Migration von Dokumenten liegen auf der Hand:

*Vorteile der Migration*

- *allgemein bekannt*

  Da Migration in der Datenverarbeitung allgemein angewandt wird, kann man auf Erfahrungen, Methoden, geschultes Personal und vor allem auf vorhandene Werkzeuge zurückgreifen.

*stets aktuell*

- Migrierte Dokumente sind prinzipiell zu jedem Zeitpunkt auf den gerade aktuellen Abspielsystemen verfügbar. Sofern dem nicht juristische oder organisatorische Gründe entgegenstehen, ist ein weltweiter Zugriff in Sekundenschnelle kein Problem.

*eventuell Qualitätsverbesserung*

- Auch die Qualität migrierter Dokumente entspricht den aktuellen Standards: So wie bei Tonaufzeichnungen heute Stereophonie und HiFi-Qualität selbstverständlich sind, werden bei Textdokumenten moderne Fonts und ein gefälliges Layout erwartet.

*Nachteile der Migration*

Der letzte Punkt deutet aber auch schon die Nachteile der Migration an:

*Verfälschungen des Originals*

- Die bei der Migration notwendigen Anpassungen führen fast immer zwangsläufig zu Verfälschungen des Originals, d. h., die Authentizität des Dokuments ist nicht gewährleistet. Ein anderer Bereich, in dem über lange Zeit hin mit Migration gearbeitet wurde, ist die Bibelübersetzung. Hier wird mühsam auf frühere Fassungen zurückgegriffen, um die Authentizität zu verbessern.

*schwer automatisierbar*

- Die bei der Migration anfallenden Transformationen sind nur in einfachen Fällen automatisch (per Programm) durchführbar. Ein solcher gut beherrschbarer Schritt wäre z. B. die Umwandlung von ASCII-Dokumenten in Unicode-Dokumente (wobei das Erkennen von Ersatzdarstellungen von in Unicode zusätzlichen Zeichen ein manuelles Nachbearbeiten erforderlich machen kann). Wie in Abschnitt 1.2 am Beispiel der Liniengraphik angedeutet, können im alten Layout Informationen implizit enthalten sein, die sich nur dem Auge des verständigen Lesers erschließen.

Die Migration ist Gegenstand von Kapitel 3.

### Hardware-Museen

*Hardware-Museen sind nicht praktikabel*

Die Maßnahme zur elektronischen Langzeitarchivierung, die am stärksten auf den Erhalt des Originalzustands ausgerichtet ist, besteht in der Einrichtung so genannter *Hardware-Museen*. Die Aufgabe von solchen Hardware-Museen ist es, möglichst lückenlos alle bekannten Computer betriebsbereit zu erhalten, damit künftige Generationen unsere elektronischen Dokumente in ihrer ursprünglichen Umgebung betrachten können. An Authentizität ist



**Abbildung 1.3**
*Hardware-Museum an der Universität der Bundeswehr München*

dieser Ansatz nicht zu übertreffen. Leider erweist er sich aus einer Reihe von Gründen als nicht praktikabel:

- Selbst heute schon erscheint es unrealistisch, sämtliche bisher entwickelten Computertypen in einem Museum zu ver- *zu viele Exponate*

sammeln, geschweige denn dies für die Zukunft sicherzustellen.

*unterschiedlichste Software und Hardwarezusätze*

- Wie in Abschnitt 1.2 ausgeführt, besteht eine vollständige Abspielumgebung nicht nur aus Hardware, sondern vor allem auch aus System- und Präsentationssoftware. Neben der Hardware müsste das Museum also auch sämtliche Versionen von Betriebssystemen und Anwendungsprogrammen für ihre Computer vorhalten und bei Bedarf installieren. Gleiches gilt für spezielle Hardwarezusätze (z. B. Joysticks oder Digitalisiertabletts) und ihre Treibersoftware.

*begrenzte Lebensdauer*

- Auch technische Geräte und ihre Bausteine haben nur eine begrenzte Lebensdauer. Da weder die Geräte noch ihre Bausteine weiter produziert werden, müsste man genaue technische Spezifikationen zu Aufbau und Funktionsweise des Gesamtsystems zusammen mit den technischen Leistungsmerkmalen der Komponenten aufbewahren, nach denen die Wartungstechniker bei Bedarf Ersatzteile herstellen könnten (sofern die zur Fertigung benötigten Techniken noch bekannt sind).

Ein solches Projekt wäre wegen seines enormen Bedarfs an materiellen und personellen Ressourcen zum Scheitern verurteilt.

### Emulation

*Emulatoren in der Hardwareentwicklung*

Bei der Entwicklung neuer Hardware werden – um Zeit und Kosten zu sparen – so genannte Emulatoren eingesetzt, mit denen man die Funktionsweise von Prozessoren und anderen Hardwarekomponenten durch Software simulieren und testen kann, bevor man sie tatsächlich baut. Von Rothenberg (1999) stammt der Vorschlag, *Emulation* zur elektronischen Langzeitarchivierung zu verwenden.

*Vorgehen beim Emulationsansatz*

Beim Emulationsansatz werden von jedem elektronischen Dokument nicht nur der Zeichenstrom und die Metadaten konserviert (z. B. durch Migration), sondern zusätzlich eine Emulatorgeeignete Spezifikation der Hardware und binäre Zeichenströme, welche die komplette Software der Abspielumgebung enthalten. Will jemand in beispielsweise 100 Jahren ein so konserviertes Dokument betrachten, dann ist in folgenden Schritten vorzugehen:

*Emulator herstellen, ...*

1. Verwende die Spezifikation der Hardware, um zusammen mit dem Emulator eine Software-Implementierung der Originalhardware herzustellen, die sich funktional identisch verhält. (Dies ist der entscheidende Schritt und setzt voraus, dass die

Hardwarespezifikation in einem vom Emulator verarbeitbaren Format vorliegt.)

2. Installiere auf dem emulierten Computer die Systemsoftware und die benötigten Präsentationsprogramme. (Die Informationen zur Handhabung dieses 100 Jahre alten Computertyps, insbesondere zur Installation von Software, sollten daher zusammen mit dem Dokument abgelegt werden.)

*. . . darauf Software installieren . . .*

3. Lade den Zeichenstrom des elektronischen Dokuments in den emulierten Computer und starte das passende Präsentationsprogramm. (Hierzu empfiehlt sich die Speicherung des Handbuchs zum Präsentationsprogramm mit dem Dokument.)

*. . . und Dokumente »abspielen«*

Der Emulationsansatz wird detailliert in Kapitel 4 beschrieben.

Trotz seiner offenkundigen Komplexität ist die Langzeitarchivierung mit Emulation aus einer Reihe von Gründen attraktiv:

*Vorteile der Emulation*

- ❏ Ist ein elektronisches Dokument erst einmal in die für die Emulation benötigte Form gebracht, dann sind später keine weiteren Verarbeitungsschritte mehr nötig (wenn man von den Maßnahmen zur Konservierung des zugrunde liegenden Zeichenstroms und der Migration des Emulators einmal absieht). Im Unterschied dazu erfordert Migration in regelmäßigen Abständen Anpassungen an einen immer anwachsenden Dokumentenbestand, wobei die Anpassungen auch nur teilweise zu automatisieren sind.

*Aufwand je Dokument gering*

- ❏ Da wie beim Museumsansatz mit den Datenströmen der Originaldokumente gearbeitet wird, beschränkt sich der bei der Konservierung anfallende Aufwand im Wesentlichen auf das Erstellen der Hardwarespezifikationen, das aber nicht bei jedem Dokument, sondern »nur« bei jedem neuen Computertyp zu leisten ist. Wünschenswert wäre eine Übernahme der Emulatorspezifikationen der Hardwarehersteller aus der Entwicklungsphase.

*»nur« ein Emulator für viele Dokumente*

- ❏ Die Authentizität der konservierten Dokumente ist ähnlich hoch wie beim Museumsansatz, da das Dokument praktisch nicht verändert wird.

*hohe Authentizität*

- ❏ Der mit der Emulation verbundene laufende Aufwand besteht hauptsächlich aus den – allerdings hohen – Zugriffskosten, ist also proportional zur Verwendungshäufigkeit.

*Aufwand proportional zur Verwendungshäufigkeit*

*Probleme bei der Emulation*

Die Emulation befindet sich noch in der Erprobungsphase. Während man bei der Migration jeweils kleine, überschaubare Schritte

macht und Lösungen immer dann gefunden werden müssen, wenn sich die Computertechnologie ein Stück weiterentwickelt, hängt die Durchführbarkeit des Emulationsansatzes von positiven Antworten auf folgende Fragen ab:

*präzise Beschreibung der Hardware*

- Findet man eine Hardware-Beschreibungssprache, in der man die Spezifikationen eines breiten Spektrums von Hardware mit ausreichender Genauigkeit fassen kann? Damit künftige Entwicklungen berücksichtigt werden können, sollte die Sprache erweiterbar sein. Alle Eigenschaften, welche an der Benutzerschnittstelle sichtbar sind (Bildschirmgröße, -auflösung und -farben, Taktrate des Prozessors etc.) müssen mit aufgeführt werden.

*Generierung von Emulationsprogrammen*

- Lassen sich aus solchen Beschreibungen automatisch Emulationsprogramme generieren, die laufzeiteffizient genug sind, um die Realzeit-Eigenschaften der nachzubildenden Hardware (z. B. wenn Musik oder Filmsequenzen wiedergegeben werden sollen) zu erfüllen?

*portable Emulatoren?*

- Kann der Emulator ausreichend portabel gestaltet werden? Theoretisch könnte man ein dezidiertes Emulatorsystem (als Hardware) bauen. Sinnvoller erscheint es aber, eine Emulatorsoftware herzustellen und jeweils auf alle wichtigen Plattformen zu portieren, damit der Emulator von den technologischen Fortschritten profitieren kann und auch überall verfügbar ist.

*Speicherungsredundanz vermeiden*

- Lässt sich eine praktikable Organisation finden, welche der oben beschriebenen Speichertechnik inhaltlich gleichwertig ist, aber deren Redundanz (Computerspezifikation und Software bei jedem Dokument) vermeidet, ohne die Integrität der Dokumente zu gefährden?

*Aufwand wegen Vielzahl der Formate*

Ganz besonders bei der Migration, aber auch bei der Emulation gilt: der für die Langzeitspeicherung zu treibende Aufwand steigt bzw. fällt mit der Anzahl der zugelassenen Datenformate. Es liegt daher nahe, Vielfalt und Wildwuchs durch Einführung von Standards zu beschneiden, um den Aufwand in einem handhabbaren Rahmen zu halten.

*Standards für einfache Zeichenfolgen*

Die in Abschnitt 1.3 erwähnten Standards (ASCII, Unicode) für einfache Zeichenfolgen sind wohletabliert und können auf absehbare Zeit als stabil gelten. Das macht sie besonders attraktiv als Grundlage für die Speicherung der Metadaten, die den Inhalt von elektronischen Dokumenten und deren Handhabung beschrei-

ben. Es ist wichtig, diese Zugriffs- und Zugangsschlüssel in einem einfachen und verlässlichen Format aufzubewahren.

*Standards für komplexe Dokumentenarten*

Je komplexer und gebräuchlicher die Dokumentenarten sind, desto größer die Palette an unterschiedlichen, konkurrierenden Formaten: Für Textdokumente mit eingebetteten Graphiken reicht das Spektrum von internen, proprietären Formaten (z. B. Word) über prozedurale Beschreibungen (PostScript) und Anzeigeformate (DVI und PDF) bis hin zu Auszeichnungssprachen (LaTeX und XSL). Auf der Basis von XML entsteht derzeit eine Reihe von »selbstbeschreibenden« Formaten, die für künftige Standardisierungsbemühungen möglicherweise richtungweisend sein werden. Dazu gehören RDF und Dublin Core für Metadaten, XSL für Desktop Publishing, SVG für Vektorgraphiken, SMIL für Multimediadokumente und MathML für mathematische Formeln.

*Auszeichnungssprachen*

Auszeichnungssprachen auf der Basis von SGML und XML werden daher den Schwerpunkt des Kapitels 6 über Standardformate bilden.

## 1.5 Juristische und gesellschaftliche Fragen

*sieben Thesen*

Mit den von uns behandelten technischen Aspekten der elektronischen Langzeitarchivierung sind juristische und gesellschaftliche Fragestellungen verknüpft, die in den folgenden sieben Thesen nur kurz angerissen werden.

1. *»Digital Rights Management« und Kopierschutz*

   Bevor elektronische Dokumente archiviert werden können, sind *Copyright*-Fragen zu klären: Da insbesondere bei der Migration von Dokumenten Kopien erstellt werden und evtl. Originaldokumente verändert werden müssen, ist das Einverständnis des Urhebers mit solchen Maßnahmen vorab einzuholen. Weitere Kopien, die an Leser von Dokumenten ausgehändigt werden, sind angemessen zu honorieren und müssen gegebenenfalls mit Sperrvermerken verbunden werden, wenn eine unentgeltliche Weitergabe nicht erlaubt ist. Der Unterschied zwischen herkömmlichen und elektronischen Dokumenten ergibt sich daraus, dass bei letzteren Kopie und Original praktisch nicht zu unterscheiden sind. Basis einer für alle Seiten handhabbaren Lösung könnte eine entsprechende Ausweitung der GEMA-Regelungen sein.

2. *Rechte an der Software*

   Beim Emulationsansatz sind (mittelfristig) zusätzlich die Rechte an den Software-Anteilen der Originalabspielumge-

bung zu wahren. Auch hier wäre eine pauschale, zweckgebundene Regelung nützlich.

*Emulatoren von den Hardwareherstellern?*

3. Soll der Emulationsansatz Anwendung finden, wäre es sinnvoll, die Hardwarehersteller gesetzlich zu verpflichten, nach einer angemessenen Frist vollständige, für Emulationszwecke geeignete Spezifikationen ihrer Geräte der Allgemeinheit zur Verfügung zu stellen. Da aktuelle Geräte verfügbar sind, müsste eine solche Verpflichtung erst dann greifen, wenn die Entwickler den vollen wirtschaftlichen Nutzen aus dem eher kurzlebigen Innovationsgehalt ihrer Produkte gezogen hätten.

*Kriterien zur Selektion*

4. Arbeitsgrundlage für Bibliotheken und Archive ist eine klare Aussage darüber, was eigentlich als bewahrenswert zu betrachten ist. Für Druckerzeugnisse gibt es Kriterien (z. B. die Existenz einer zugehörigen ISBN-Nummer) und gesetzlich abgesicherte Verfahrensweisen (Verlage müssen Belegexemplare liefern). Es wäre zu kurz gegriffen, nur solche elektronischen Dokumente als relevant zu betrachten, die an greifbare Datenträger wie CD-ROMs gebunden und mit einer ISBN-Nummer versehen sind. Wissenschaftliche elektronische Zeitschriften und weitere, »nur« im Internet existierende Publikationen gehören sicherlich auch zum bewahrenswerten kulturellen Erbe. Neue Kriterien müssen aufgestellt und vom Gesetzgeber durch geeignete Verfahrensweisen unterstützt werden. Wegen der Fülle des Materials und der hohen Kosten sind insbesondere *Kriterien zur Selektion* wichtig und sollten sorgfältig überdacht werden. Andererseits drängt die Zeit, da jetzt schon Abspielumgebungen laufend veralten. Anders als bei Druckerzeugnissen kann man nicht gut abwarten, was langfristig nachgefragt wird und dadurch seinen Wert belegt.

*Kosten als begrenzender Faktor*

5. *Kosten* sind ein entscheidender begrenzender Faktor. Auf nationaler und internationaler Ebene sind langfristig tragende Vereinbarungen erforderlich, welche die Träger der elektronischen Langzeitarchivierung sowie ihre Zuständigkeiten und finanzielle Ausstattung verlässlich festlegen. Natürlich sind solche Vereinbarungen laufend fortzuschreiben und damit nicht endgültig fixiert; ein tragfähiges Rahmenprogramm ist aber Grundvoraussetzung für eine dauerhafte Kulturarbeit. Sinnvoll ist es sicher auch, durch Kooperation und Aufgabenteilung global Kosten zu senken: Gemeinsame Standards sind ein Aspekt, Verteilung von einander ergänzenden Schwerpunkten ein anderer.

6. Kosten können allein von staatlicher Seite getragen oder zum Teil von anderen übernommen werden, z. B. die Zugriffskosten in Form von Lesegebühren und die Kosten der erstmaligen Archivierung (nebst entsprechender Datenaufbereitung) von Autoren, die ihre Produkte erhalten sehen möchten. Mit Vorschriften über die Form der eingereichten Dokumente (zulässige Standardformate, erforderliche Metadaten) ließen sich die Kosten der Archivierung ebenfalls senken. *Kosten tragbar machen*
7. Alle Regelungen müssen die berechtigten Interessen der beteiligten Parteien (Autoren und Verlage, Bibliotheken und Archive, Wissenschaftler und Öffentlichkeit) ausgewogen berücksichtigen. Die verschiedenen Standpunkte werden von Haynes *et al.* (1997) ausführlich einander gegenübergestellt. *Interessenausgleich zwischen allen Beteiligten*

Einige dieser Fragen werden wir in der Diskussion in Kapitel 7 nochmals aufgreifen.

# 2 Das OAIS-Referenzmodell und das DSEP-Prozessmodell

*Der erste Teil dieses Kapitels enthält eine Einführung in das OAIS-Referenzmodell, das einen prinzipiellen Rahmen für elektronische Archive vorgibt. Es werden sowohl das Informations- als auch das Prozessmodell vorgestellt. Der zweite Teil erläutert das DSEP-Prozessmodell, das auf dem OAIS-Referenzmodell aufbaut, aber mehr auf die speziellen Bedürfnisse elektronischer Bibliotheken ausgerichtet ist.*

## 2.1 Das OAIS-Referenzmodell

Im Zusammenhang mit digitalen Bibliotheken und elektronischem Publizieren stößt man häufig auf die Akronyme OAIS und OAI. Wegen ihrer Ähnlichkeit können diese leicht verwechselt werden. Um daher klarzumachen, worum es in diesem Kapitel geht, sind ihre Bedeutungen deutlich voneinander zu unterscheiden (weiter beschäftigen werden wir uns nur mit OAIS):

*OAI – Open Archives Initiative*

Die *Open Archives Initiative* (OAI) hat sich zum Ziel gesetzt, Standards für den Austausch von Informationen zwischen elektronischen Archiven zu entwickeln und deren Anwendung zu fördern. Unter Archiv wird hierbei einfach eine Sammlung elektronischer Medien auf einem Server verstanden, etwa einem Hochschulschriftenserver. Die Dokumente, die auf dem Server liegen, sollen frei zugänglich sein und die dazugehörenden Metadaten nach einem einheitlichem Schema abgerufen werden können. Hierzu wurde von der OAI das *Protocol for Metadata Harvesting* (OAI-PMH) entworfen. Nach einer mehr experimentellen Phase liegt dieses Protokoll seit Juni 2002 in der stabilen Version 2.0 vor und wird bereits von zahlreichen Institutionen eingesetzt.

*OAIS – Open Archival Information System*

Dagegen beschäftigt sich das *Open Archival Information System Reference Model* (OAIS-Referenzmodell) mit der Langzeitarchivierung elektronischer Publikationen. »Archiv« wird hier in sei-

nem eigentlichen Sinne gebraucht: Es bezeichnet eine Organisation, deren Aufgabe es ist, Dokumente langfristig zu bewahren. Das OAIS-Referenzmodell dient vorwiegend dazu, grundlegende Begriffe der Archivierung zu klären und die entsprechenden Schlüsselprozesse zu identifizieren. Es soll so eine gemeinsame Basis für den internationalen Erfahrungsaustausch auf diesem Gebiet liefern.

### 2.1.1 Hintergrund

*ISO-Norm aus der Weltraumforschung*

Das OAIS-Referenzmodell ist ein ISO-Standard, der die Langzeitarchivierung elektronischer Dokumente zum Inhalt hat. Mit seiner Ausarbeitung hat die *International Organization for Standardization* (ISO) im Jahr 1995 das *Consultative Committee for Space Data Systems* (CCSDS) beauftragt. Das CCSDS, das formal der NASA untersteht, wurde 1982 von mehreren Raumfahrtorganisationen mit dem Ziel gegründet, den Datenaustausch in der Weltraumforschung zu vereinfachen. Nach mehrjährigen Forschungen präsentierte die CCSDS im Mai 1999 eine erste Fassung des Referenzmodells, die noch mehrmals überarbeitet wurde. Die Version von Februar 2003 wurde als ISO-Norm Nr. 14721:2003 (ISO 2003) akzeptiert.

*ein umfassendes Modell für Archive*

Obwohl es seinen Ursprung in der Weltraumforschung hat, liefert das OAIS-Referenzmodell ein allgemeines, theoretisches Modell für das Funktionieren eines Archivs. Da im OAIS-Referenzmodell keine Annahmen über die Art der Daten gemacht werden, ist es auf konventionelle und digitale Archive gleichermaßen anwendbar. Sein Schwerpunkt liegt aber auf der Archivierung elektronischer Dokumente.

Das Problem der Langzeitarchivierung wird im OAIS-Referenzmodell unter zwei sich ergänzenden Perspektiven betrachtet: Es werden sowohl ein Informations- als auch ein Prozessmodell entwickelt.

### 2.1.2 Das Informationsmodell

*Daten und ihre Interpretation*

Grundlegend für das Informationsmodell des OAIS-Referenzmodells (Abb. 2.1) ist die Unterscheidung zwischen Daten (engl. *Data Object*) und Informationen (engl. *Information Object*). Daten können alle möglichen Arten von nichtdigitalen und digitalen Objekten sein. Wie werden aus Daten dann Informationen? Damit Daten Bedeutung erhalten, müssen sie interpretiert werden. Dies setzt ein bestimmtes Wissen voraus, z. B. kann man ein Java-

Programm nur dann verstehen, wenn man die Programmiersprache Java kennt. Dieses zum Verstehen von Daten nötige Wissen wird im OAIS-Referenzmodell als *Knowledge Base* bezeichnet. Wer nicht über die entsprechende Knowledge Base verfügt, kann mit den Daten erstmal nichts anfangen. Zum Verständnis der Daten benötigt man noch weitere Informationen. Neben dem Quellcode des Java-Programms beispielsweise muss man noch ein Lehrbuch über die Programmiersprache Java zur Verfügung stellen. Diese zusätzlichen Informationen werden im OAIS-Referenzmodell unter der Bezeichnung *Representation Information* zusammengefasst. Die eingangs gestellte Frage wird vom OAIS-Referenzmodell daher folgendermaßen beantwortet: Daten werden erst aufgrund einer entsprechenden Knowledge Base und zusätzlicher Representation Information zu Informationen. Natürlich ist dieser Prozess damit noch nicht abgeschlossen: Denn auch die Representation Information ist nur aufgrund einer geeigneten Knowledge Base verständlich. Wo diese fehlt, benötigt die Representation Information selbst weitere Representation Information zu ihrer Interpretation. Theoretisch könnte dieser Vorgang bis ins Unendliche so weitergehen, praktisch wird er jedoch durch die kulturelle Überlieferung begrenzt.

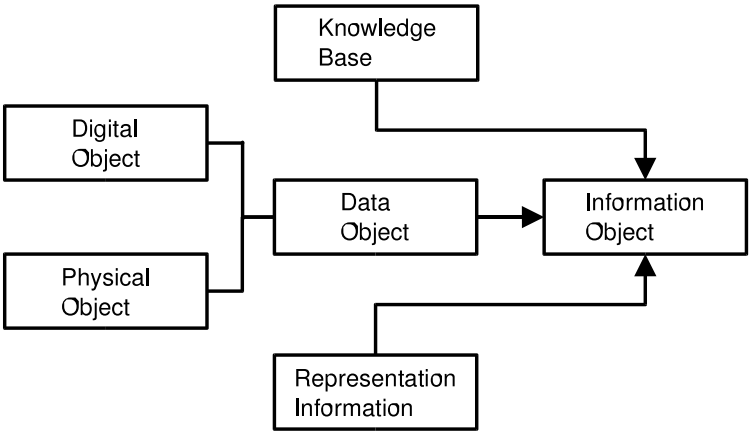
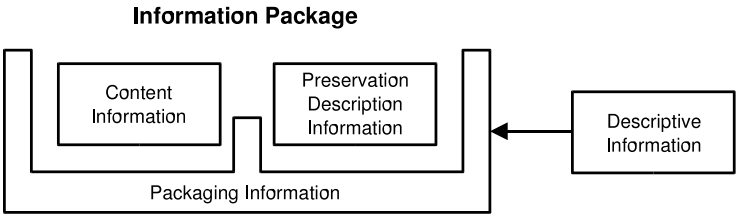



**Abbildung 2.1** *Das Informationsmodell des OAIS-Referenzmodells*

*Information Packages enthalten ...*

Nach dieser Abgrenzung modelliert das OAIS-Referenzmodell im nächsten Schritt die Informationseinheiten innerhalb eines Archivs. Der zentrale Begriff hierfür ist der des *Information Package* (Abb. 2.2). Ein Information Package setzt sich aus den Komponenten *Content Information* und *Preservation Description Information* zusammen:

**Abbildung 2.2**
*Bestandteile eines Information Package*




*. . . Content Information und . . .*

Die *Content Information* ist das eigentliche Informationsobjekt, dessen Bewahrung Aufgabe des Archivs ist. Sie umfasst alle Informationen, die nach dem ersten Teil dieses Kapitels zum Verständnis von Daten nötig sind: neben dem Data Object also auch die Representation Information.

*. . . Preservation Description Information*

Als *Preservation Description Information* (PDI) hingegen werden im OAIS-Referenzmodell alle die Informationen bezeichnet, die notwendig sind, um die zugehörige Content Information angemessen zu bewahren. Hierzu gehören beispielsweise Informationen, die die Integrität der Content Information garantieren oder ihre Herkunft und ihre Beziehung zu anderen Objekten im Archiv angeben.

*logische Einheit*

Die Content Information und die PDI werden im Information Package zu einer nur logischen Einheit zusammengefasst. Dies ist ein generelles Merkmal des OAIS-Referenzmodells: Alle seine Definitionen und Beschreibungen sind logisch-konzeptionell zu verstehen und lassen ihre tatsächliche Realisierung völlig offen.

*Packaging Information*

Da die beiden Komponenten eines Information Package aber auch auf einem physikalischen Speichermedium miteinander verknüpft sein müssen, sieht das OAIS-Referenzmodell noch die *Packaging Information* vor. Sie gibt an, wie sich Content Information und PDI tatsächlich aufeinander beziehen, etwa indem sie die Verzeichnisstruktur einer CD-ROM beschreibt. Da sie aber nichts zum Inhalt eines Information Package hinzufügt, schreibt das OAIS-Referenzmodell die Bewahrung der Packaging Information nicht verbindlich vor, sondern überlässt diese Entscheidung jedem einzelnen Archiv.

*Descriptive Information*

Schließlich benötigt ein Archiv noch die Metadaten über die gespeicherten Information Packages. Das OAIS-Referenzmodell sieht hierfür die *Descriptive Information* vor. Sie liefert Informationen über den Inhalt eines Information Package und ermöglicht so dessen Auffinden im Archiv.

### 2.1.3 Das Umgebungs- und das Prozessmodell

*Umgebung der Prozesse*

Um die Prozesse, die in einem Archiv ablaufen, genauer zu definieren, betrachtet das OAIS-Referenzmodell zunächst die Umgebung eines Archivs (Abb. 2.3). Aufgabe eines Archivs ist es, Informationen zu bewahren, die es von außen erhält, und diese zukünftigen Nutzern zur Verfügung zu stellen. Deshalb muss es auf der einen Seite mit dem Erzeuger (bei OAIS »Producer« genannt), also Autoren, Institutionen u. ä., zusammenarbeiten, die die Informationen für das Archiv liefern. Auf der anderen Seite muss es die bewahrten Dokumente interessierten Verbrauchern (bei OAIS »Consumers« genannt) zugänglich machen. Schließlich bedarf es noch einer Einrichtung, die die konkrete Bestimmung des Archivs festlegt, also z. B. welche Informationen gesammelt werden sollen und welche nicht. Hierfür ist nach dem OAIS-Referenzmodell das Management des Archivs zuständig.

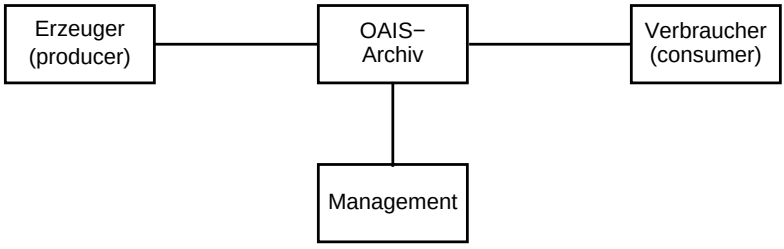



**Abbildung 2.3**
*Umgebung eines OAIS-Archivs*

*Information Packages: SIPs, AIPs und DIPs*

Nach ihrem Bezug zur Umgebung eines Archivs differenziert das OAIS-Referenzmodell drei Arten von Information Packages: das *Submission Information Package* (SIP), das vom Erzeuger an das Archiv gesandt wird, das *Archive Information Package* (AIP), das im Archiv gespeichert wird, und das *Dissemination Information Package* (DIP), das der Verbraucher vom Archiv erhält.

*Archivprozesse*

Diese verschiedenen Information Packages sind die Eingabedaten der Prozesse, die in einem Archiv ablaufen. Das OAIS-Referenzmodell modelliert diese anhand folgender Funktionen: Ingest, Archival Storage, Data Management, Administration, Preservation Planning und Access (Abb. 2.4).

*Ingest*

Der Prozess *Ingest* empfängt ein SIP von einem Erzeuger und bereitet es für die Speicherung und Verwaltung innerhalb des Archivs vor. Zu seinen Aufgaben gehören u. a. das Aushandeln der Verträge mit den Erzeugern, die Kontrolle der Vollständigkeit und Authentizität des SIP, die Umwandlung des SIP in ein AIP und das Erzeugen der Descriptive Information zu den AIPs. Anschlie-

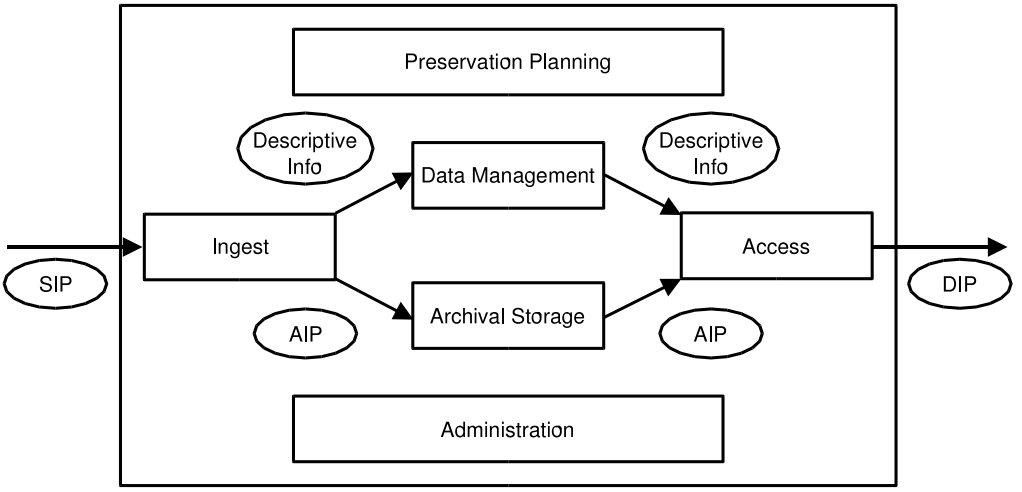



**Abbildung 2.4**
*Das Prozessmodell des OAIS-Referenzmodells*

ßend gibt er das AIP an den Archival-Storage-Prozess weiter und die dazugehörende Descriptive Information an das Data Management.

*Archival Storage*

Für die Speicherung, die Erhaltung der Bitfolge und das Retrieval der AIPs ist der Prozess *Archival Storage* zuständig. Er übernimmt die AIPs vom Ingest-Prozess und verteilt sie auf die verschiedenen Speichermedien. Der Prozess ist auch verantwortlich für die Speicherverwaltung, für die Unversehrtheit der AIPs gegenüber Speicherfehlern und das periodische Wiederauffrischen der Speichermedien. Er garantiert auch, dass bei einem Systemausfall die AIPs wieder rekonstruiert werden können. Auf Anfrage übergibt er die AIPs an den Access-Prozess.

*Data Management*

Der Prozess *Data Management* verwaltet die Descriptive Information und zusätzlich die Daten, die für das Funktionieren des Systems erforderlich sind. Diese Daten werden in einer Datenbank gehalten, für deren Administration und Fortschreiben das Data Management verantwortlich ist. Auch für die Durchführung von Anfragen an die Archivdatenbank und das Aufbereiten der gelieferten Daten ist dieser Prozess zuständig.

*Administration*

Das routinemäßige Funktionieren des gesamten Archivs ist Aufgabe des Prozesses *Administration*. Er handelt mit den Erzeugern die Bedingungen aus, unter denen sie ihre Informationen an das Archivs senden, und kontrolliert, ob die gelieferten SIPs mit

den Standards des Archivs übereinstimmen. Weiter ist der Prozess für die Hardware- und Softwarearchitektur des Archivs verantwortlich. Hierzu überwacht er die Systemfunktionen und untersucht Möglichkeiten ihrer Optimierung. Er entscheidet auch über ein Fortschreiben bzw. eine Migration des Archivinhalts. Schließlich entwickelt und überwacht er die Standards, die für das Archiv verbindlich sind.

*Preservation Planning*

Die Funktion *Preservation Planning* war im ursprünglichen Entwurf des OAIS vom Mai 1999 noch nicht vorgesehen. Als die *Networked European Deposit Library* (NEDLIB) (siehe unten) und andere Organisationen begannen, das OAIS-Referenzmodell praktisch umzusetzen, wurde die Notwendigkeit klar, auch für die Aufgabe der Langzeitarchivierung eine eigene Funktion einzuführen. Auf Vorschlag der NEDLIB wurde daher das Preservation Planning in das OAIS-Referenzmodell aufgenommen. Es umschreibt alle Aufgaben, die notwendig sind, um auch in ferner Zukunft den technischen Zugriff auf die im Archiv gespeicherten Informationen zu garantieren. Dazu gehört es u. a., die Entwicklungen auf dem Hardware- und Softwaremarkt zu überwachen und zu prüfen, ob die gespeicherten Informationen auch auf neuentwickelten Systemen noch ablauffähig sind. Falls nicht, muss entschieden werden, welche Strategie (etwa Migration oder Emulation) die Zugänglichkeit der Dokumente auch unter den geänderten Bedingungen garantiert.

*Access*

Als Schnittstelle zwischen den gespeicherten Dokumenten und den Verbrauchern ist im OAIS-Referenzmodell der Prozess *Access* vorgesehen. Er ermöglicht es den Nutzern, den Inhalt des Archivs zu durchsuchen, etwa mittels eines Online-Katalogs, erzeugt auf eine Benutzeranfrage hin ein DIP und überwacht dessen Auslieferung an den Verbraucher.

Diese sechs Prozesse beschreiben im OAIS-Referenzmodell den Informationsfluss vom Erzeuger in das Archiv und vom Archiv zum Verbraucher. Sie liefern ein Modell für die Schlüsselprozesse eines Archivs, ohne jedoch auf die Details einer konkreten Implementierung einzugehen. Wie eine solche aussieht, wird sich von Archiv zu Archiv unterscheiden.

## 2.2 Das DSEP-Prozessmodell

*DSEP – OAIS-Adaption der NEDLIB*

Zu den ersten Organisationen, die das OAIS-Referenzmodell anwandten, gehörte die NEDLIB,[1] die zum 1. Januar 1998 vom *Congress of European National Libraries* gegründet wurde. NEDLIB war ein Gemeinschaftsprojekt mehrerer europäischer Nationalbibliotheken und -archive, ICT-Organisationen und Verlage, das von der Koninklijke Bibliotheek, der Nationalbibliothek der Niederlande, geleitet wurde. Zu den Teilnehmern gehörte auch Die Deutsche Bibliothek. Ziel des Projekts war es, im europäischen Rahmen die neuen Anforderungen, die elektronische Publikationen an nationale Archivbibliotheken stellen, zu untersuchen und Wege aufzuzeigen, wie diesen begegnet werden kann. Zur Klärung der Begriffe und als Grundlage für alle praktischen Versuche entwickelten die beteiligten Partner das Konzept eines *Deposit System for Electronic Publications* (DSEP). Darin werden die Prozesse allgemein modelliert, die Bibliotheken zur langfristigen Archivierung elektronischer Dokumente anwenden müssen. Seinen Abschluss fand NEDLIB durch einen Workshop[2] im Dezember 2000.

*Langzeitarchivierung im Geschäftsgang*

Im NEDLIB-Entwurf wird das Problem der Langzeitarchivierung elektronischer Publikationen nicht isoliert betrachtet, sondern in den Geschäftsgang einer digitalen Bibliothek eingeordnet. Nach Hacker (2000) unterteilt man den Geschäftsgang einer Bibliothek in:

- Bestandsaufbau
- Bestandserschließung
- Bestandsaufbewahrung und Bestandserhaltung
- Bestandsvermittlung

*Aufgaben eines DSEP*

Abbildung 2.5 zeigt, wie im NEDLIB-Entwurf die Aufgaben, die ein DSEP übernehmen soll, bestimmt werden (EP ist die Abkürzung für »Elektronische Publikation«). Die zusätzliche Aufgabe »überwachen« dient dazu, das Funktionieren des Systems zu garantieren. Das DSEP ist dabei als ein eigenständiges Modul eines typischen Bibliothekssystems konzipiert, wie es heute in Bibliotheken benutzt wird. Der Vorteil dieses Ansatzes liegt darin, das DSEP nicht von Grund auf neu entwickeln zu müssen, sondern die Funktionen nutzen zu können, die heutige Bibliothekssysteme schon bieten.

[1] siehe http://www.kb.nl/coop/nedlib/index.html.

[2] Der Workshop wurde in der Zeitschrift für Bibliothekswesen und Bibliographie 48 (2001), Heft 3-4 dokumentiert.

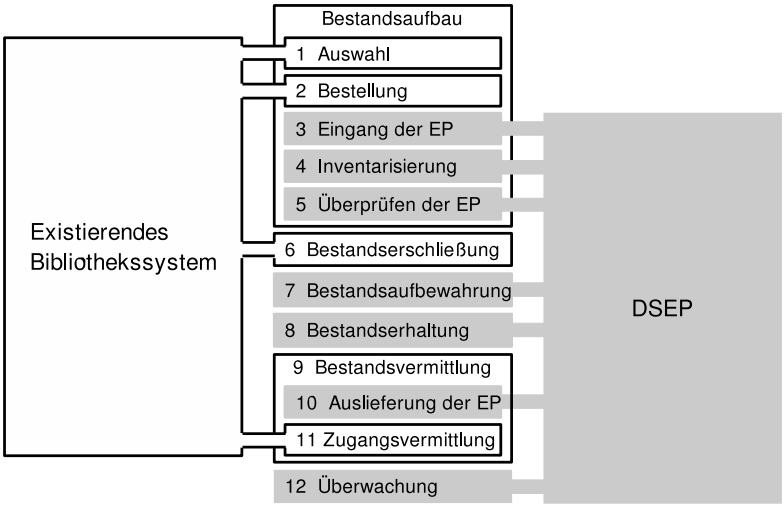



**Abbildung 2.5**
*Geschäftsgang einer digitalen Bibliothek mit einem DSEP*

*Schnittstellen zum Bibliothekssystem*

Bereits im Dezember 1998 entschloss man sich, sich bei der genaueren Bestimmung der Prozesse, mit denen ein DSEP die ihm zugeordneten Aufgaben erfüllen kann, auf das OAIS-Referenzmodell zu stützen. Von Beginn des Projekts an sollten so die Vorteile internationaler Standardisierungsbemühungen genutzt werden. Im Unterschied zum OAIS-Referenzmodell verfügt ein DSEP bereits über eine bestimmte Umgebung: An die Stelle der allgemeinen Erzeuger und Verbraucher aus Abb. 2.3 tritt das bereits vorhandene Bibliothekssystem. Im DSEP werden deshalb zwei neue Prozesse als Schnittstellen zum Bibliothekssystem definiert. Abbildung 2.6 zeigt das sich dadurch ergebende DSEP-Prozessmodell. Da die Prozesse innerhalb des DSEP die gleichen sind wie beim OAIS-Referenzmodell aus Abb. 2.4, wurden sie nicht noch einmal graphisch dargestellt[3].

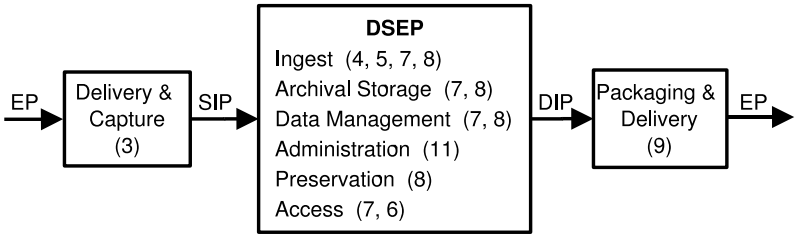



**Abbildung 2.6**
*Das Prozessmodell eines DSEP*

[3]Ausnahme hiervon ist der Prozess *Preservation* s. u.

*Arbeitsabläufe des Geschäftsgangs*

Die eingeklammerten Ziffern nach den einzelnen Prozessen beziehen sich auf den Geschäftsgang aus Abb. 2.5. Da die Arbeitsabläufe des Geschäftsgangs im DSEP-Modell unter logisch-funktionalen Gesichtspunkten gegliedert werden, können einzelne Abläufe auf mehrere DSEP-Prozesse verteilt sein. Ein Beispiel aus der *Bestandserhaltung* (8) soll dies verdeutlichen: Für die Migration eines bereits gespeicherten Dokuments ist zunächst der Prozess *Preservation* zuständig, der das Dokument in ein neues Format konvertiert. Da das so erzeugte neue Dokument wieder im DSEP verwaltet werden soll, wird es an den Prozess *Ingest* weitergegeben. Dieser überprüft das Dokument und fügt den Metadaten die die Migration betreffenden Informationen hinzu. Die Metadaten werden dann an das *Data Management* weitergeleitet, während das Dokument selbst dem *Archival Storage* übergeben wird.

*neue Prozesse*

Die beiden neu hinzugekommenen Prozesse *Delivery & Capture* und *Packaging & Delivery* fungieren als Schnittstellen zum umgebenden Bibliothekssystem. Im Einzelnen übernehmen sie dabei folgende Aufgaben:

*Delivery & Capture*

Der Prozess *Delivery & Capture* erhält von der Erwerbungskomponente des Bibliothekssystems alle neuen elektronischen Publikationen, die ins DSEP aufgenommen werden sollen. Da eine Bibliothek in der Regel das Datenformat einer Publikation nicht beeinflussen kann, liegen die eingegangenen Dokumente in unterschiedlichen Formaten vor. Der Prozess *Delivery & Capture* sorgt nun für die Transformation der Dokumente in ein SIP, das den Standards des DSEP genügt. Anschließend leitet er das SIP an den Prozess *Ingest* weiter.

*Packaging & Delivery*

Auf der anderen Seite vermittelt der Prozess *Packaging & Delivery* die im DSEP enthaltenen Informationen an die Benutzer. Dazu fordert er vom *Access*-Prozess ein DIP an und verarbeitet es weiter. Ein DIP kann neben dem eigentlichen Dokument noch die Metadaten und Software enthalten, die zum Installieren und Anzeigen des Dokuments benötigt werden. Der Prozess *Packaging & Delivery* muss das DIP dann auspacken und in eine solche Form transformieren, in der es vom Bibliothekssystem abgespielt werden kann.

*Abgrenzung zum Bibliothekssystem*

Da die übrigen DSEP-Prozesse *Ingest*, *Archival Storage*, *Data Management*, *Administration* und *Access* denjenigen des OAIS-Referenzmodells entsprechen, brauchen sie hier nicht noch einmal aufgeführt werden. Folgende Einschränkung ist allerding zu machen: Viele der Aufgaben, die im OAIS-Referenzmodell den Prozessen zukommen, werden im DSEP vom umgebenden Bibliothekssystem erledigt. Beispielsweise wird das Aushandeln der Verträge

mit den Autoren bzw. Verlagen – im OAIS-Referenzmodell eine Aufgabe von *Ingest* – von der Erwerbungsabteilung einer digitalen Bibliothek übernommen. Dadurch können die Schlüsselprozesse im DSEP enger gefasst werden.

*Zentraler Prozess: Preservation*

Die einzige Ausnahme hiervon ist der *Preservation*-Prozess. Zum Zeitpunkt der Konzeption des DSEP sah das OAIS-Referenzmodell keinen eigenen Prozess für die Langzeitarchivierung vor. Da eines der Hauptanliegen der NEDLIB aber darin bestand, die Langzeitarchivierung elektronischer Publikationen sowohl theoretisch wie experimentell zu untersuchen, sollte die entsprechende Funktion auch im Modell abgebildet sein. Die NEDLIB entschloss sich daher, das OAIS-Referenzmodell um den Prozess *Preservation* zu erweitern. Dieser Prozess besteht im DSEP-Modell aus den beiden Subprozessen *Preservation Planning*, der für die Erforschung und Entwicklung von Konzepten zur Langzeitarchivierung verantwortlich ist, und *Preservation Activities*, der die Konzepte umsetzt und ihre Funktion überwacht. Bei der Fortentwicklung des OAIS-Referenzmodells wurde nur der Prozess *Preservation Planning* neu ins Modell aufgenommen. Hier kommt ein Unterschied in den Konzeptionen von DSEP und OAIS zum Ausdruck: Während im DSEP-Modell konkret Migration und Emulation als Strategien der Langzeitarchivierung betrachtet werden und deshalb auch eine Instanz zu deren Überwachung nötig ist, legt sich das OAIS-Referenzmodell auf keine Strategie fest.

*Subprozesse*

Wegen seiner zentralen Bedeutung wird der *Preservation*-Prozess im Folgenden noch detaillierter beschrieben: Er ist dafür verantwortlich, den Zugang zu den archivierten Dokumenten langfristig zu garantieren. Wie bereits erwähnt, gliedert er sich in die beiden Subprozesse *Preservation Planning* und *Preservation Activities*.

*Preservation Planning*

Der Subprozess *Preservation Planning* ist für die Erforschung der Probleme der Langzeitarchivierung und für die Entwicklung von Ansätzen zu deren Lösung zuständig. Er umfasst mehrere Teilprozesse:

- *Develop Preservation Standards & Strategies:* Dieser Prozess entwirft die Standards und Konzepte, die es dem Archiv ermöglichen, auf die zukünftige Entwicklung des elektronischen Publizierens flexibel und angemessen zu reagieren.
- *Develop Packaging Designs:* Dieser Prozess bestimmt das Design, nach dem SIPs, AIPs und DIPs gebildet werden, und entwickelt dieses fort.

- *Define Reference Platforms:* Eine *Reference Platform* umfasst diejenigen Hardware- und Softwarekomponenten, die zum Zugriff auf eine elektronische Publikation notwendig sind. Dazu gehören u. a. Rechnertyp, Betriebssystem, Editoren. Der Prozess stellt sicher, dass für alle im Archiv vorhandenen Publikationen Referenzsysteme definiert sind.
- *Monitor Technology:* Dieser Prozess dient der Beobachtung der aktuellen Entwicklungen auf dem IT-Sektor, soweit sie das DSEP betreffen.

*Preservation Activities*

Der zweite Subprozess *Preservation Activities* hat die Aufgabe, die im Archiv tatsächlich laufenden Aktivitäten zur Langzeitarchivierung zu koordinieren und zu überwachen. Er enthält zwei Teilprozesse:

- *Create New Reference Platform:* Dieser Prozess erzeugt die neu definierten Referenz-Plattformen und wandelt sie zur Speicherung im Archiv in SIPs um. Ein möglicher Ansatz, ein neues Referenzsystem hervorzubringen, ist die Emulation.
- *Archival Information Update:* Dieser Prozess ist für die Aktualisierung der im Archiv gespeicherten AIPs verantwortlich. Ein Fortschreiben ist beispielsweise bei einer Migration der Dokumente erforderlich.

*funktionales Modell vs. Implementierung*

Zum Schluss sei nochmals darauf hingewiesen, dass die OAIS- und DSEP-Referenzmodelle nur die Anforderungen an elektronische Archivierungs- bzw. Bibliothekssysteme beschreiben. Trotz des teilweise großen Detaillierungsgrads stellt der definierende NEDLIB-Bericht Nr. 6 (van der Werf 2000) immer wieder heraus, dass es sich um ein funktionales Modell handelt, das die Leistungen eines solchen Systems festlegt, nicht aber die Struktur der Implementierung. In diesem Sinn ist es beispielsweise nicht als Implementierungsentscheidung anzusehen, wenn im Bericht der Ablageort der elektronischen Dokumente als »media«, also Datenträger, und der Ablageort der Metadaten als »data base«, also Datenbank, bezeichnet wird. Aus guten Gründen darf (und wird) die konkrete Implementierung ganz anders strukturiert sein als die Referenzmodelle. Wichtig ist nur, dass die Implementierung die durch die Referenzmodelle festgelegten Leistungen erbringt.

# 3 Migration

*Wie in Kapitel 1 dargestellt, zielt der Migrationsansatz darauf ab, die Originaldokumente durch fortgesetzte Übertragung auf künftige Computer dauerhaft zugreifbar und nutzbar zu erhalten. Abschnitt 3.1 beschreibt anhand eines Beispiels »aus dem täglichen Leben« verschiedene Migrationsoptionen, ordnet den Migrationsansatz begrifflich ein und umreißt die angestrebten Ziele. Abschnitt 3.2 handelt von der Anwendung des Migrationsansatzes auf die Langzeitarchivierung elektronischer Dokumente und diskutiert mögliche Varianten. Abschnitt 3.3 beschreibt die organisatorische Umsetzung innerhalb des in Kapitel 2 beschriebenen Rahmens. Abschnitt 3.4 fasst die Chancen und Risiken des Migrationsansatzes kurz zusammen.*

## 3.1 Migration: Begriffsfeld und Zielsetzung

*Migration überall*

Migration hat jeder Computernutzer, der seine Dokumente aufbewahrt, selber schon durchgeführt, z. B. um alte oder archivierte Dokumente mit einer neuen Software oder auf einem neuen Computer weiter zu benutzen. Aber nicht erst dann wird Migration notwendig: Durch das Internet sind verschiedene Generationen von Computern unterschiedlicher Herkunft sowie völlig heterogener Hard- und Softwareausstattung miteinander verbunden. Wollen Personen aus unterschiedlich ausgestatteten Umfeldern Dokumente austauschen, dann müssen sie Mittel und Wege finden, einander durch Migration ihre verschiedenen Formate zugänglich zu machen.

*Probleme am Beispiel von Textdokumenten*

Handelt es sich beispielsweise auf der einen Seite um Microsoft-Word- und auf der anderen Seite um LaTeX-Dokumente, dann bestünde ein pragmatisches Vorgehen darin, beide Dokumentenarten in ein gemeinsames Drucker- oder Seitenbeschreibungsformat (derzeit wohl PostScript bzw. PDF) zu migrieren. Beide Seiten könnten dann problemlos alle Dokumente lesen. Da Dokumente in diesen Austauschformaten aber nur mühsam weiter-

bearbeitet werden können, wird man bei engerer Zusammenarbeit andere Lösungen suchen. Sich auf das Format einer der beiden Seiten zu einigen, ist oftmals schwierig, wenn dazu die benötigte Software beschafft, installiert und ihr sachgemäßer Gebrauch erst erlernt werden muss. Häufig behilft man sich in solchen Fällen damit, die Textanteile bei Bedarf aus dem Dokument zu extrahieren und in einem überall verwendbaren Format (ASCII) auszutauschen. Leider gehen bei diesem Migrationsschritt sämtliche Formatierungseigenschaften (wie Fettdruck, Zeichensatz, Farbe und Größe) verloren und müssen später manuell wieder eingearbeitet werden. Und selbst bei Verwendung von ASCII als »kleinstem gemeinsamen Nenner« ist man vor Verfälschung und Nachbearbeitungsaufwand nicht sicher, weil verschiedene Betriebssysteme (Unix, Microsoft-DOS, MacOS) die Zeilenwechsel in Textdateien verschieden codieren.

*Handhabung der Migration*

Das Beispiel zeigt, dass Migration sehr unterschiedlich gehandhabt (u. a. nach PDF oder ASCII) und auf sehr unterschiedlichen Ebenen (u. a. die des Textverarbeitungsprogramms oder die der Textdateien von Betriebssystemen) durchgeführt werden kann. Wegen der großen und laufend wachsenden Vielfalt an Formaten für Textdokumente, Graphiken, Videos, Datenbanken etc. ist es nicht möglich, eine universelle Migrationsstrategie für alle vorstellbaren Daten zu entwickeln. Tatsächlich versteht man unter Migration auch nicht ein klar umrissenes Verfahren, sondern eine Klasse von Vorgehensweisen. Der Begriff »Migration« wird in der Literatur in verschiedenen Bedeutungen verwendet, die nun gegeneinander abgegrenzt werden sollen.

*Definition von »Migration«*

Eine häufig zitierte Definition von Migration stammt aus dem Abschlussbericht der *Task Force on Archiving of Digital Information* (TFADI 1996):

> **Erklärung:** *Migration* ist der periodische Transfer digitalen Materials von einer Hard-/Softwarekonfiguration zu einer anderen Konfiguration, von einer Generation der Computertechnologie zur nachfolgenden Generation. Das Ziel der Migration ist es, die Integrität von digitalen Objekten zu erhalten. Auf Anwenderseite soll stets gewährleistet sein, dass Daten trotz sich ständig ändernder Technologien empfangen, angezeigt oder anders genutzt werden können.

*Migration ist regelmäßig durchzuführen!*

Ein wichtiger Aspekt der Definition der TFADI ist die Feststellung, dass die Migration keinen einmaligen Ad-hoc-Vorgang darstellt, sondern regelmäßig abhängig von den technologischen

Veränderungen durchgeführt werden muss. Daraus lässt sich ableiten, dass die Institutionen, die digitale Informationen archivieren und die Migration als Erhaltungsstrategie einsetzen, geeignete Organisationsstrukturen schaffen müssen, die regelmäßig die Notwendigkeit von Migrationen überprüfen, diese planen und geeignet durchführen können. Im Abschnitt 3.3 werden wir einige solche Maßnahmen betrachten.

*Migration im weiteren und im engeren Sinn*

Die Definition der TFADI beschreibt »Migration im weiteren Sinn« und damit die wesentlichen Ziele jeder digitalen Langzeitarchivierungsstrategie: Es geht darum, digitales Material in möglichst unverfälschter Form für künftige Nutzer zugänglich zu erhalten. Da es – wie in Kapitel 1 zum »Museumsansatz« ausgeführt – nicht möglich ist, die digitalen Originaldokumente und ihre Originalabspielumgebungen beliebig lange zu erhalten, müssen entweder die Dokumente in Formate übertragen werden, die künftig darstellbar sind oder Nachbildungen der Abspielumgebungen für die die digitalen Originaldokumente hergestellt werden. Beide Strategien entsprechen der »Migration im weiteren Sinn«. Die erste Strategie heißt »Migration im engeren Sinn« und ist Gegenstand dieses Kapitels; die zweite Strategie ist die »Emulation«, die in Kapitel 4 behandelt wird.

*Anforderungen an die Migration*

Bevor wir uns den verschiedenen Migrationsverfahren zuwenden, rekapitulieren wir kurz einige unterschiedliche Zielsetzungen und Anforderungen an die Migration als Form der Langzeitarchivierung. Lievesley (1995) nennt folgende Ziele zur Erhaltung digitaler Daten:

- physikalische Zuverlässigkeit der Daten, d. h. Erhaltung der Datenmedien sowie Verhinderung von Datenverlusten,
- Bewahrung der Daten vor unautorisiertem Zugriff,
- Sicherstellung der kontinuierlichen Nutzbarkeit der Daten und
- Integration in Informationsvermittlungssysteme.

*Planung von Migrationsstrategien*

Es ist sehr schwierig, eine Migrationsstrategie zu planen, da die Wünsche und Anforderungen an die Daten in der Zukunft nicht vorhersehbar sind. Wenn eine Migration notwendig wird, stellen sich folgende Fragen: Wieviel Formatierungsverluste oder Umformungen sind akzeptabel? Wieviel wird der ganze Prozess kosten? Der Migrationsprozess kann die Integrität der Daten (TASI 2000) beeinträchtigen. Die Integrität bezieht sich dabei sowohl auf Form und Struktur (z. B. die Auflösung, Genauigkeit oder Farbtiefe bei Bildern) als auch auf den intellektuellen Inhalt, wie er ursprüng-

lich erzeugt wurde. Der Nutzer von Archiven will selbstverständlich die Gewissheit, dass der Inhalt sowohl in Qualität als auch in Echtheit erhalten bleibt. Im Internet werden heutzutage bei Graphik- und Videoformaten verlustbehaftete Komprimierungstechniken eingesetzt, um die Übertragungsbandbreiten niedrig zu halten. Solche Formate sind für die Migrationen ungeeignet (TASI 2000). Aus der Sicht der Archive und Bibliotheken müssen die Erhaltungskosten natürlich so gering wie möglich gehalten werden. Auch die Frage, welche Daten für die Zukunft erhaltenswürdig sind, spielt hier eine große Rolle. Die Entscheidung, aus Kostengründen bei der Migration nur ein Surrogat wie eine Zusammenfassung zu erzeugen, kann später in der Regel nicht mehr revidiert werden.

*Dokumente stets lesbar*

Michelson und Rothenberg (1992) nehmen an, dass digitale Bibliotheken vor allem von Lehr- und Forschungseinrichtungen genutzt werden. Daher ist es wichtig, dass Dokumente jederzeit gefunden, analysiert und weiterverarbeitet werden können (vgl. auch Endres und Fellner (2000)). Die digitalen Informationen müssen also künftig immer auf aktuellen Systemen lesbar und interpretierbar bleiben. Ein bloßes Erhalten des digitalen Bitstroms reicht dazu offenbar nicht aus.

*Verbesserung der Daten bei Migration*

Migration kann auch zur Verbesserung der Daten eingesetzt werden. Oft hat sich der Stand der technischen Möglichkeiten zwischen der Erstellung eines Dokuments und dem Migrationszeitpunkt erheblich verbessert, sodass bei einer Migration neben einer Umwandlung z. B. in ein aktuelleres Format Nachbearbeitungen am Datenmaterial durchgeführt werden können. Das Entfernen von Hintergrundrauschen in eingescannten Bildern oder Texten oder eine Indexerstellung für Textdokumente sind nur zwei Beispiele dafür. Das Hinzufügen von Metadaten gehört im weitesten Sinne auch hierzu, berücksichtigt jedoch mehr den Bibliotheks- bzw. Archivierungsaspekt und wird daher in Abschnitt 3.3 angesprochen.

## 3.2 Migration in der Langzeitarchivierung

*Einteilung der Migrationsverfahren*

Bei der »Migration im engeren Sinn« wird also das digitale Dokument auf neue Hard-/Softwarekombinationen übertragen. Die Unterscheidung zwischen Hardware- und Softwaretransfers teilt die Migrationsverfahren in natürlicher Weise weiter auf in Verfahren zum Wechseln der physischen Träger der digitalen Daten und in solche Verfahren, die die logische Struktur oder das For-

mat der digitalen Informationen ändern. Als erstes betrachten wir noch die vielen möglichen *Zielformate* einer Migration, insbesondere langlebigere Standards. Unabhängig von der für die Erhaltung des Dokumenteninhalts eingesetzten Strategie müssen die Metadaten durch Migration laufend in einem lesbaren Format erhalten werden; auch darauf gehen wir in Abschnitt 3.3 ein.

### 3.2.1 Datenformate als Migrationsziele

*proprietäre und offene Formate*

Die Programmvielfalt in vielen Bereichen beschert uns eine unüberschaubare Vielzahl verschiedener Datenformate. Viele von ihnen sind proprietär an Programmpakete und Computersysteme gebunden (beispielsweise Microsoft-Office-Datenformate oder WAV-Audiodateien). Andere Datenformate jedoch, deren Beschreibung offengelegt wurde und die in der Folge in Export- und Importfilter von Fremdsystemen Eingang gefunden haben, werden gerne für den Austausch zwischen Systemen genutzt. RTF und PDF sind solche Datenformate.

*standardisierte Formate ...*

Die zunehmende Globalisierung und die fortschreitende Vernetzung fördern einheitliche Datenformate und Kommunikationsstandards. Die große Anzahl an Standardisierungsvorhaben im Vergleich zu den 70er oder 80er Jahren belegt dies. Angetrieben von Geschäftsinteresse beteiligt sich auch die Wirtschaft an der Entwicklung und Anpassung von Datenstandards. Ein Beispiel dafür ist das Umfeld der *Geographic Information System* (GIS). Dort versucht man, Datenkonvertierungen auf ein Minimum zu beschränken und Wartungskosten durch die Nutzung von weitverbreiteten Standards zu minimieren. Auch im Bereich e-Commerce ist man bestrebt, aus den bestehenden (unterschiedlichen) Verfahren der sicheren Datenübertragung ein gemeinsames und standardisiertes Verfahren, verbunden mit standardisierten Datenaustauschformaten, zu entwickeln.

*und ihre Vorteile*

Durch Einführung von Standardformaten verringert man nicht nur wie gewünscht die Abhängigkeit von Hardwareumgebungen und Abspielsystemen. Es wird ferner die Anzahl der insgesamt zu verwaltenden Datenformate reduziert, weil es deutlich weniger Hardware-/Software-unabhängige Formate als abhängige gibt. Von diesen wenigen Formaten werden Bibliotheken sinnvollerweise für jeden Datentyp ein Format bzw. nur sehr wenige Formate als Migrationsziele auswählen. Dadurch behalten Bibliotheken einen besseren Überblick über die verschiedenen Datenformate ihrer Einrichtung und müssen weniger teures, weil personalintensives Know-How vorhalten. Gleichzeitig verlängern sie künfti-

ge Migrationsintervalle, da Standardformate im Regelfall länger unterstützt werden.

*formale Kriterien*

Um die angesprochenen Vorteile aber auch ausnutzen zu können, sollte man Datenformate wählen, die folgenden formalen Kriterien genügen:

- Das Datenformat sollte öffentlich sein, d. h., die vollständige Syntax und Semantik des Formats sind veröffentlicht.
- Es sollte für das Datenformat einen Standard einer anerkannten internationalen Standardisierungsinstitution wie die *International Organization for Standardization* (ISO), *American National Standard for Information Sciences* (ANSI) oder das *World Wide Web Consortium* (W3C) geben. Obwohl PDF nicht standardisiert ist, hat es sich quasi als Defacto-Austauschformat für druck- und anzeigbare Dokumente entwickelt. Standardisierung ist also kein Muss, gewährleistet aber einen gewissen Grundkonsens, den Abspielprodukte einhalten müssen, und verhindert ein Ausufern von herstellerspezifischen Veränderungen.
- Das Datenformat ist anerkannt und wird häufig genutzt. Hierdurch wird sichergestellt, dass es für viele Hardwareumgebungen Abspielsysteme gibt und dass auch für künftige Plattformen Abspielsysteme entwickelt werden.
- Darüber hinaus sollte das Datenformat an sich frei von patent- und lizenzrechtlichen Gebühren sein. Vorsicht ist vor allem bei Formaten geboten, die patentiert sind, deren Hersteller aber bis auf weiteres vorbehaltlich von Nutzungsgebühren absehen, um ihr Format mit ihren Produkten besser vermarkten zu können. Ist das Format einmal allgemein eingeführt, dann kann es – wie bei JPEG 15 Jahre nach Erteilung des Patents geschehen – vorkommen, dass Patenthalter plötzlich Lizenzgebühren erheben und dadurch Nutzer in unerwartete Schwierigkeiten bringen.

*zielgerechte Auswahl*

Erfüllen mehrere Formate obige Anforderungen, hängt es von den Aufgaben und Zielen der Bibliothek und den verwalteten Inhalten ab, welches von ihnen bevorzugt werden sollte. Die Bandbreite reicht von der Wahl der einfachsten Form, die für die Archivierung der Inhalte zweckmäßig ist und meistens weniger Anforderungen an das Retrieval-System stellt (Hedstrom 2000), bis hin zu Formaten, die das alte Dokument originalgetreu darstellen können. Letzteres ist für Bibliotheken oder Museen notwendig, die Inhalte nichtdigitaler Werke durch Digitalisierung konservieren.

| Kategorie | Datentypen | Standardformate |
|---|---|---|
| Daten | alphanumerische Daten | PDF, PostScript, ASCII, CSV, SQL |
| Strukturierter Text | alphanumerisch, Bildreferenzen, Markup | PostScript, PDF, TeX, DSSSL, SGML, HTML, XML |
| Office-Dokument | alphanumerisch, Raster- und Vektorgraphik, animierte Graphik | PostScript, PDF, DSSSL, RTF, ASCII, SGML, TIFF, CGM |
| Design-Dokument | Raster- und Vektorgraphik, alphanumerische Daten | HPGL, PostScript, EPS, DXF/DWG, IGES, CGM, TIFF, ASCII/RTC |
| Präsentations-graphik | Raster- und Vektorgraphik, alphanumerische Daten, animierte Graphik | PostScript, PDF |
| Bild | Rastergraphik | PostScript, PDF, TIFF, GIF, JPEG |
| Tonaufnahme | Audiodaten | MPEG-1 audio layers 1/2/3, MP3, MIDI |
| Videoaufnahme | Videodaten | MPEG-1, MPEG-2, MPEG-4 |
| Geographische Daten (GIS) | Raster- und Vektorgraphik, alphanumerische Daten | PostScript, EPS, HPGL, TIFF, ASCII, CGM |
| Interaktive Multimedia-Veröffentlichung | alle | MPEG-1, MPEG-2 |

**Tabelle 3.1** *Dokumentenkategorien und Standardformate nach Hendley (1998)*

*aktuelle Dokumenten-kategorien und Datenformate*

Hendley (1998) führte eine umfangreiche Analyse aktueller digitaler Dokumentenkategorien und zugehöriger Datenformate durch. Tabelle 3.1 listet die wesentlichen Resultate mit aktuellen Standards (sowohl für bearbeitbare als auch nur darstellende Datenformate) auf. Für Dokumente, die nur Daten eines einzigen Datentyps enthalten, gibt es einige wenige etablierte Standards. Insbesondere haben sich PDF und PostScript für die visuelle Repräsentation von Text und Graphik durchgesetzt. Für Dokumente, die aus mehreren Datentypen zusammengesetzt sind, gibt es allerdings noch keine Standardformate, sondern nur die proprietären Formate der Anwendungshersteller, so z. B. im Bereich der Virtuellen Realität, der Office-Anwendungen und der Geographischen

Informationssysteme (GIS). Möglicherweise besitzen die Anwendungen Exportfunktionen, die aus diesen proprietären Umgebungen herausführen können (Jones und Beagrie 2001). Solange es keine Standards für die Dokumentenkategorie gibt, kann sie sich nur auf den Export in die einzelnen Datentypstandards beschränken. Die in der Tabelle aufgelisteten Standards für diese Kategorien beziehen sich deshalb auf die enthaltenen Datentypen.

*Problematik der Hypertext-Dokumente*

Problematisch sind auch Hypertext-Dokumente, insbesondere deren charakteristische Verweise (engl. *links*) auf andere Dokumente: Die bisher verwendeten Referenzierungsformate (Uniform Resource Locator (URL) im World Wide Web) basieren auf dem Ablageplatz des zu referenzierenden Dokuments im Hyperspace. Notwendig ist jedoch ein dauerhafter eindeutiger Bezeichner (engl. *permanent identifier*) für das Dokument selber, da sich der Ablageplatz aus organisatorischen Gründen leicht ändern kann. Welche der im Forschungsumfeld angebotenen, möglichen Realisierungen (*Persistent Uniform Resource Locator* (PURL), *Digital Object Identifier* (DOI) oder *Uniform Resource Name* (URN)) sich künftig durchsetzen wird, ist noch nicht abzusehen. Wir gehen auf dieses Thema später genauer ein.

### 3.2.2 Migration durch Wechseln des Datenträgers

*Fallstudie: Migration sozialwissenschaftlicher Studiendaten*

Das Wechseln des Datenträgers (engl. *media migration*) von digitalen Daten wird notwendig, weil zum einen die Lebensdauer der bisher genutzten Datenträger im Vergleich zum Papier oder zu Mikrofilmen relativ kurz ist und weil zum anderen der technologische Fortschritt in sehr kurzen Zeitabständen neue Geräte auf den nach größeren Speicherkapazitäten und schnelleren Zugriffsverfahren suchenden Markt bringt. Die Produktion und Unterstützung veralteter Datenträgertechnologien und die Pflege sowohl der Lesegeräte als auch deren Anbindung an Computersysteme werden oft nach nur wenigen Jahren eingestellt. Wie schwer es ist, Datenbestände von nicht mehr unterstützten Systemen wieder nutzbar zu machen, zeigt eine detaillierte Fallstudie von Green *et al.* (1999), in der Lochkartenstapel von sozialwissenschaftlichen Studiendaten in ein Hardware- und Software-unabhängiges Format migriert wurden.

*einfach und kostengünstig*

Beim Wechseln des Datenträgers handelt es sich in der Regel um einfache, kostengünstige Verfahren (TFADI 1996), wobei allerdings das Übernehmen von komplexen Strukturen, Sicherheitsmechanismen oder Echtheitsmerkmalen Probleme bereiten kann.

Bei der folgenden Betrachtung konkreter Strategien folgen wir der Klassifikation aus CCSDS (1999).

### Auffrischung digitaler Medien

*Auffrischung auf Original oder Kopie*

Das einfachste Verfahren ist das Auffrischen (engl. *Refreshment*) digitaler Medien, d. h. das Kopieren von Informationen von einem Medium zu einem anderen Medium desselben Medientyps ohne Veränderung des Datenformats. Dadurch will man den Verlust digitaler Informationen verhindern, der durch physikalische Effekte bei der Lagerung (z. B. Entmagnetisierung von Magnetbändern) oder durch die Alterung der Speichermaterialen (z. B. Abnutzung oder Ablösung der magnetischen Beschichtung von Magnetbändern) entsteht. Das Auffrischen im eigentlichen Wortsinn ist bei Speichermedien, die nur einmal beschreibbar sind (wie z. B. CD-ROMs), nicht möglich, das Herstellen eines Duplikats des Datenträgers in den meisten Fällen jedoch schon. Für das nutzende Rechensystem sind Duplikat und Original nicht zu unterscheiden, die duplizierten Daten daher ohne Anpassung zu verwenden.

### Migration auf andere digitale Medien

*Probleme trotz Beibehaltung der logischen Struktur*

Wenn man dagegen die Informationen auf ein Medium eines anderen Medientyps als den des Originals kopiert, kann man das neue Medium im Allgemeinen nicht ohne Änderung des nutzenden Systems einsetzen. Diese Änderungen betreffen oftmals nur Treiber in der Rechnersystemsoftware. Die Anwendungen selbst müssen nur dann angepasst werden, wenn im Betriebssystem der neue Medientyp die von der Anwendung genutzte Zugriffsform oder Schnittstelle nicht unterstützt. Bei dem Übergang von Magnetbändern auf optische Datenträger wird die logische Struktur der Daten zwar weitestgehend beibehalten, aber die unterschiedlichen Zugriffsformen und Strukturen der Daten (sequenzieller Zugriff auf Byte-Ströme mit Endemarkierung bei Magnetbändern, blockorientierter Direktzugriff auf die Blöcke und Sektoren optischer Datenträger) erfordern erhebliche Anpassungen der Anwendungssoftware. Die Migrationsform, bei der ohne Änderung der logischen Struktur Informationen von einem Medium zu einem anderen Medium eines anderen Medientyps kopiert werden, bezeichnet man im Englischen als *Replication*.

*Echtheitsmerkmale und Kopierschutz*

Zugriffsbeschränkungen können beim Kopieren auf ein neues Medium oftmals problemlos übernommen werden, da diese Informationen meist unabhängig vom verwendeten Medium gespei-

chert sind. Gleiches gilt für Echtheitsmerkmale wie etwa Prüfsummen. Wurden die Daten mit einem Kopierschutz versehen oder wurden Echtheitsmerkmale direkt auf dem ursprünglichen Datenträger angebracht, ist es nicht ohne weiteres möglich, diese Daten auf neue Medien zu migrieren. Heutzutage wird oftmals bei Daten- und Audio-CDs diese Art von Kopierschutz verwendet. Um die Migration solcher Daten möglich zu machen, muss in Zusammenarbeit mit dem Hersteller eine Möglichkeit gefunden werden, den Kopierschutz entweder auszuschalten oder auf das neue Medium zu übertragen.

*wachsende Speicherkapazitäten*

Als klassische Medien für die Archivierung wurden bisher meist langsame, aber kapazitätsstarke Datenträger wie Magnetbänder und optische Laufwerke eingesetzt. Möglicherweise erreichen die Preise von Festplatten schon bald ein Niveau, bei dem sie als Alternative zu Magnetbändern oder ähnlichen Medien in Frage kommen. Dadurch würden ganz neue Möglichkeiten in der Archivierung geschaffen, denn Festplatten haben im Gegensatz zu Magnetbändern und anderen Wechselmedien den Vorteil, dass die Daten ständig zugriffsbereit vorliegen. Dies ist dann besonders wichtig, wenn die Informationen online über das Internet zur Verfügung gestellt werden sollen. Als weitere Alternative bieten sich DVD-Systeme an, die voraussichtlich bald die CD-Technologie im IT-Bereich ablösen werden. In Verbindung mit Wechsler- oder Robotersystemen erreichen sie enorme Speicherkapazitäten, und angebunden über Standardbussysteme (z. B. SCSI) erlauben sie ebenso direkte Datenzugriffe wie Festplattensysteme.

### Migration auf nichtdigitale Medien

*nichtdigitale Sicherheitskopien*

Archivierung digitaler Informationen auf nichtdigitale Medien ist nach Hedstrom (2000) wahrscheinlich die häufigste Strategie zur Langzeiterhaltung. Dies ist insbesondere deswegen bemerkenswert, da viele Bibliothekare gerade die Digitalisierung historischer Dokumente zur Erhaltung der Informationen verwenden. Nun zeigt sich, dass die digitalen Daten wohl noch vor der Zerstörung der historischen Originale nutzlos werden.

*Papier und Mikrofilm*

Wie in Abbildung 3.1 angedeutet stehen zwei Datenträger bei diesem Migrationsverfahren im Vordergrund: Zum einen Papier, auf dem die digitalen Daten ausgedruckt werden und zum anderen Mikrofilm. Die Speicherung von Daten auf diesen Medien hat den Vorteil, dass Papier bzw. Mikrofilm wesentlich weniger anfällig gegen Umwelteinflüsse und Zeitfaktoren ist als viele digitale Speichermedien. Darüber hinaus ist keine teure Hard- und

Software nötig, um eine Datenspeicherung auf Papier oder Mikrofilm vornehmen zu können. Doch der größte Vorteil ist, dass sie zur Rückgewinnung der Informationen ohne aufwändige Lese- und Interpretationssysteme direkt mit dem Auge oder durch eine Lupe betrachtet werden können.[1]

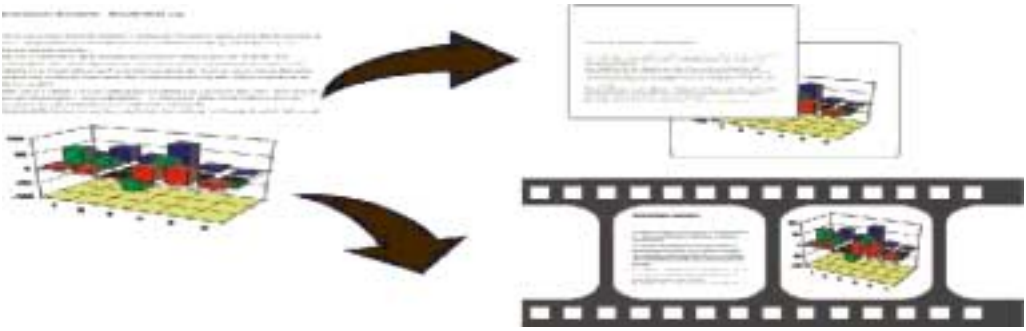


**Abbildung 3.1**
*Migration auf nichtdigitale Medien*

Diese Art der Datenspeicherung ist vor allem bei einfachen Texten und unstrukturierten Dokumenten sehr gut anwendbar. Die Speicherung von komplexen Daten stellt sich dagegen als sehr problematisch dar, da es nicht ohne weiteres möglich ist, strukturierte Dokumente und komplexe Daten – wie Datenbanken, Videos oder Hyperlinksysteme – adäquat auf nichtdigitalen Medien abzubilden. Auch ist es beispielsweise nicht möglich, die Formel, die hinter den Daten einer Tabelle steht, mit aufs Papier zu speichern, wenn diese Tabelle mit ihren Werten gedruckt wird. Die Dokumente werden sozusagen »geglättet« (TFADI 1996). Sicherheitsprotokolle und Authentisierungsverfahren, die möglicherweise in das Dokument integriert waren, gehen bei der Speicherung auf Papier oder Mikrofilm verloren, ebenso Echtheitsmerkmale wie digitale Wasserzeichen.

*nicht geeignet für komplexe Dokumentenarten*

### 3.2.3 Migration durch Ändern der logischen Struktur

*transformierende Migration*

Eine Reihe weiterer Migrationsstrategien beruht auf der Änderung des Datenformats (engl. *transformation* bzw. *content migration*). Diese Verfahren sind besonders wichtig, wenn die digitalen Daten laufend bearbeitbar oder manipulierbar bleiben müssen.

[1] vgl. dazu den Vortrag von Karl-Ernst Lupprian »Archiv für 1000 Jahre?« auf den vierzehnten EDV-Tagen Theuern 2001;
siehe http://www.museumtheuern.de/edvtage/.

Aber auch zur Kostenreduzierung werden sie eingesetzt. Die Daten werden bei diesem Migrationsprozess in Formate portiert, die (momentane) Standards darstellen und somit von aktuellen Anwendungsprogrammen verarbeitet werden können oder die weniger Platz verbrauchen bzw. leichter zu verarbeiten sind. Transformationen zum Zweck der Datenkompression (z. B. mit Hilfe von zip-Programmen) müssen natürlich verlustfrei umkehrbar sein (z. B. durch Anwendung des entsprechenden unzip-Programms). Komprimierte Daten sind nur so lange zugänglich, wie die benötigten Dekomprimierungsprogramme auf einer aktuellen Plattform ausgeführt werden können: Auch diese Programme sind zu migrieren (oder die Daten rechtzeitig wieder zu dekomprimieren).

*Upgrading*

Durch das sich schnell ändernde Softwareangebot, aber auch durch die Verabschiedung von Standards sind immer wieder Datenkonvertierungen in andere Formate erforderlich. In vielen aktuellen Softwareprodukten ist bereits die Möglichkeit enthalten, Daten früherer Softwareversionen in das Datenformat der aktuellen Version (so genanntes »Upgrading«) zu übertragen.

*Verluste bei Migration*

Bei der Konvertierung in das neue Format wird darauf geachtet, dass logische Strukturen, Zugriffsbeschränkungen, Formatierungen oder Ähnliches erhalten bleiben. Da allerdings grundsätzlich Unterschiede zwischen altem und neuem Format bestehen, ist es nahezu unmöglich, alle Informationen verfälschungsfrei in das neue Format zu übertragen. Selbst bei ASCII-Dokumenten kann eine Migration Probleme bereiten: Bei einem einfachen Textdokument, in dem es ausschließlich auf den reinen Informationsgehalt ankommt, ist die Anzahl der Silben pro Zeile völlig irrelevant. Ganz anders verhält sich dies bei lyrischen Texten wie Gedichten, in denen mit dem Versmaß auch die äußere Form wichtig ist. Wie beim Beispiel der Liniengraphik in Abschnitt 1.2 werden hier zusätzliche Strukturinformationen über Eigenschaften eingebracht, die sonst als unwesentlich gelten: die Position der Zeilenwechsel bzw. die Breite der Einzelzeichen.

Praktisch kann die Migration werkzeugunterstützt auf verschiedene Arten durchgeführt werden, die wir im Folgenden kurz erläutern.

#### Aktualisierung innerhalb der Produktfamilien

*Kompatibilität in Produktfamilien*

Es gehört in der Softwareentwicklungsgemeinde zum guten Ton, in neuen Versionen der Software zusätzliche Eigenschaften und Funktionen einzubauen. Dies erfordert häufig auch eine Erweiterung der Datenformate. Ein bekanntes Beispiel dafür sind die

Microsoft-Office-Produkte. Um früher erstellte Dokumente nach einem Software-Update weiterhin verwenden zu können, bieten die neuen Anwendungsversionen grundsätzlich die Möglichkeit, eines oder mehrere Vorgängerformate einzulesen und im neuen Format wieder zu speichern. Hierbei treten dann keine Informationsverluste auf, wenn das neue Datenformat *abwärtskompatibel* ist, d. h. eine echte Erweiterung des alten Formats. Da die Softwareentwickler auf Dauer weder die Abwärtskompatibilität der Datenformate für zukünftige Versionen noch überhaupt die Weiterentwicklung der Anwendungen garantieren können, reicht auf längere Sicht diese Migrationstechnik allein nicht aus. Weitere, vor allem finanzielle Nachteile ergeben sich aus der Notwendigkeit, für jedes Datenformat mindestens ein aktuelles Produkt vorzuhalten und regelmäßig zu aktualisieren. Zum Beispiel gab es beim Word-Format von Microsoft innerhalb von zehn Jahren mehr als sechs Formatwechsel von Word 6 bis WinWord 2000.

#### Exportieren in Fremdformate

*keine Unterstützung von Konkurrenzformaten*

Viele der heute verfügbaren Anwendungssysteme erlauben es, die bearbeiteten Dokumente auch in Fremdformaten, d. h. in Formaten von Konkurrenzprodukten, zu speichern. Es werden meistens nur gängige Formate unterstützt. Das ist an sich kein Nachteil, weil unbedeutende oder proprietäre Formate bei der Langzeiterhaltung als Ziel einer Migration keine Rolle spielen. Leider kann man nicht erwarten, dass die Softwareentwickler sich bei der Entwicklung dieser Exportfunktionen besondere Mühe geben, denn schließlich besteht ja die Gefahr, dass der Kunde zum Konkurrenzprodukt abwandert. Ausnahmen bilden hier die allgemein anerkannten produktunabhängigen Formate wie RTF, PDF oder ganz aktuell die *Markup*-Sprachen SGML sowie XML. Auch Produkte, die nicht Marktführer sind, versuchen, durch Kompatibilität mehr Marktanteile zu gewinnen. Oftmals kommt als Fehlerquelle noch hinzu, dass die Firmen die Spezifikationen ihrer Datenformate nicht oder nur unvollständig offenlegen. Bei dieser Migrationsart muss man im Allgemeinen davon ausgehen, dass die exportierbaren Formate nicht die gleiche Darstellungsmöglichkeit bieten wie das Originalformat, sodass auch hierdurch mit Informationsverlusten zu rechnen ist. Ob diese Verluste als relevant erachtet werden, hängt entscheidend von den Nutzern der Daten bzw. den Aufgaben und Archivierungsstrategien der Bibliotheken ab.

#### Importieren von Fremdformaten

*Importe direkt oder über Zwischenformate*

Auch das Laden oder Einfügen von Dokumenten, die mit Konkurrenzprodukten erstellt wurden, wird in den meisten Fällen unterstützt. Abgesehen davon, dass für Softwareanbieter Importfilter attraktiver sind, weil sie den Übergang zum eigenen Produkt unterstützen, gelten hierbei ähnliche Aussagen wie beim Exportieren in Fremdformate. Sind zwei Anwendungssysteme nicht *interoperabel*, d. h., die beiden Systeme haben keine Export- oder Importfilter des anderen Systems, dann gibt es immer noch die Möglichkeit, eine Migration über die erwähnten produktunabhängigen Datenformate als Zwischenformate durchzuführen. Allerdings besteht hierbei die Gefahr eines doppelten Informationsverlusts.

#### Einsatz von Konvertierungssoftware

*kommerzielle Konversionstools*

Es werden reichlich vor allem kommerzielle Produkte angeboten, die allgemein oder aufgabenspezifisch dafür eingesetzt werden können, Daten von einem Format in ein anderes zu migrieren. Sie vereinen jeweils in einem Programm die Funktionalität der oben beschriebenen Import- und Exportfunktionen und dies meist für viele Quell- und Zielformate. Um bei $m$ Quellformaten und $n$ Zielformaten nicht jede der $m$ mal $n$ Transformationen einzeln ausprogrammieren zu müssen, liegt es nahe, ein leistungsfähiges Zwischenformat zu verwenden. Dann genügt es nämlich, $m$ plus $n$ Transformationsschritte zu programmieren: $m$ Transformationen von den Quellformaten in das Zwischenformat und $n$ Transformationen vom Zwischenformat in die Zielformate. Wie schon erwähnt, birgt dieses Vorgehen die Gefahr, dass man zweimal Informationen verliert.

## 3.3 Archivierungsprozesse beim Migrationsansatz

Wir betrachten nun die Einbettung des Migrationsansatzes in den in Kapitel 2 beschriebenen organisatorischen Rahmen. Es geht dabei um die Zuordnung von Migrationsaktivitäten zu den einzelnen Archivierungsprozessen des DSEP-Modells bzw. des zugrunde liegenden OAIS-Modells.

*das Ziel*

Ziel aller Archivierungsaktivitäten beim Migrationsansatz ist es, dass jedes im Archiv befindliche elektronische Dokument zu

jedem Zeitpunkt auf mindestens einer aktuellen Hardware- oder Softwareplattform ausgeführt werden kann. Diese Plattform muss nicht mit der Plattform übereinstimmen, auf der das Archivierungssystem selber läuft.

*Alte Fassungen aufheben?*

Bei transformativer Migration ist die Gefahr von Verfälschungen groß: Fehler können während der automatischen Transformation großer Dokumentenmengen leicht von der Qualitätskontrolle unbemerkt bleiben. Sollte man deshalb »sicherheitshalber« die alte Fassung aufheben? Aber was hat das für einen Sinn, wenn sie nicht mehr lesbar ist? Alte Fassungen sollte man also nur dann aufheben, wenn es sich um wichtige Dokumente handelt und die künftige Rekonstruierbarkeit zumindest plausibel erscheint. Wir wollen hier nicht den Einfallsreichtum und die Leistungsfähigkeit zukünftiger »Datenarchäologen« unterschätzen.

*benötigte Software*

Die von einem Dokument benötigte Ausführungsumgebung ist meist durch zwei Softwareprodukte charakterisiert: die Betriebssoftware der Rechnerhardware einerseits und das zur Anzeige verwendete Anwendungsprogramm andererseits. Wesentlich für die Ausführbarkeit des Anwendungsprogramms ist das benötigte Betriebsprogramm (z. B. Windows 2000). Zweitrangig ist dagegen, auf welchem Rechnertyp (PC von IBM, Siemens oder anderen) das Betriebsprogramm läuft.

*Plattform und Umgebung*

**Erklärung:** Eine verträgliche Kombination aus Rechner und Betriebssoftware bezeichnen wir im Folgenden als *Rechnerplattform*, eine verträgliche Kombination aus Rechnerplattform und Anwendungsprogramm als *Umgebung* (für eine Klasse von Dokumenten).

Zu den wichtigsten, laufend zu aktualisierenden Entscheidungen des Bibliotheksmanagements gehört die Festlegung einer Menge von *Standardumgebungen*.

*abwärtskompatible Produktfamilien*

Im Laufe der Entwicklung entstehen von den drei Produkttypen (Rechner, Betriebs- und Anwendungssoftware) häufig immer leistungsfähigere Versionen. Eine Folge von Produktversionen, bei der neuere Versionen stets alle Funktionen der früheren Versionen anbieten (und eventuell weitere), bezeichnet man als *abwärtskompatible Produktfamilien*. Bei der Langzeitarchivierung kann man ältere Mitglieder einer solchen Familie durch neuere ersetzen.

*Behandlung der Metadaten*

Die zu einem elektronischen Dokument gehörigen Metadaten beschreiben verschiedene Aspekte des Dokuments, darunter Informationen, die zur Auswahl aus dem Gesamtbestand herangezogen werden, wie bibliographische Angaben, Schlagwörter und die Ar-

chivierungsnummer. Metadaten spielen also eine doppelte Rolle: Einerseits gehören sie als wichtige beschreibende Daten zur Datenkapsel des elektronischen Dokuments und sind daher vom Archivierungssystem als *Inhalt* in einem geeigneten Format zu speichern. Andererseits werden sie vom Archivierungssystem selbst als für den Zugriff wesentliche *Katalogdaten* benötigt; dabei müssen sie aus Effizienzgründen in einem für den Zugriff optimierten Format getrennt vom eigentlichen Dokument gehalten werden. Bei einer doppelten Speicherung der Metadaten in zwei möglicherweise unterschiedlichen Formaten ergibt sich das Folgeproblem, die Konsistenz der beiden Darstellungen zu jedem Zeitpunkt zu garantieren. Die Entscheidung darüber, ob man die Metadaten in zwei Formaten speichert oder nur in einem (und das andere bei Bedarf daraus ableitet), können wir glücklicherweise den Herstellern des Archivierungssystems überlassen.

*Formate für Metadaten*

Metadaten sind in der Regel strukturierte, reine Textdaten. Zur Speicherung bietet sich also eines der Basistextformate (etwa ASCII, Unicode) oder eines der in Abschnitt 6.1 beschriebenen, strukturierten Textformate (SGML oder XML) an. Da die strukturierten Formate ihrerseits als Texte der Basisformate gespeichert werden, dürfte auch mittelfristig die Migration von Metadaten keine großen Probleme bereiten. Bezüglich des Erhalts gelten die gleichen Bedingungen wie für den Rest der Datenkapsel.

Bezogen auf das in Kapitel 2 beschriebene DSEP-Prozessmodell fallen beim Migrationsansatz folgende Aufgaben an:

**Aufnahme eines neuen elektronischen Dokuments in den DSEP-Prozessen Delivery & Capture und Ingest**

*Umgebung verfügbar?*

In der Mehrzahl der Fälle wird das Dokument in einer der von der Bibliotheksleitung festgelegten Standardumgebungen abspielbar sein. Es genügt dann,

- dies (für alle in Frage kommenden Umgebungen) zu prüfen und
- die benötigten Metadaten zu erfassen und zu standardisieren. Dazu gehören neben den üblichen bibliographischen Angaben insbesondere die Verweise auf die verwendbaren Umgebungen (getrennt danach, ob sie in der Bibliothek als Standardumgebung anerkannt sind oder nicht).

*neue Umgebung erforderlich*

Ist das Dokument in keiner der von der Bibliotheksleitung festgelegten Standardumgebungen abspielbar, dann kann das Dokument vorerst nicht aufgenommen werden. Der Sachverhalt wird

gegebenenfalls der Bibliotheksleitung zur Entscheidung vorgelegt. Bei entsprechendem Interesse wird die Bibliotheksleitung eine oder mehrere der für das Dokument benötigten Umgebungen zur Menge der Standardumgebungen hinzufügen; dann kann das Dokument wie oben beschrieben aufgenommen werden. Andernfalls wird der Aufnahmevorgang erfolglos beendet.

### Erhaltung des Dokumentenbestands in den DSEP-Prozessen Archival Storage und Preservation

*automatische Auffrischung*

Wie bei allen Strategien muss der gesamte Inhalt der Datenkapsel des elektronischen Dokuments von Zeit zu Zeit »aufgefrischt« werden, bevor die Magnetisierung nachlässt. Schon im OAIS-Modell sind hierfür die Formen Refreshment und Replication der Migration vorgesehen, die wir im Abschnitt 3.2 behandelt haben. Diese Migrationsschritte sollten vom Archivierungssystem automatisch in festgelegten Zeitintervallen vorgenommen werden, ebenso andere Maßnahmen (u. a. Backups und redundante Speicherung), die den Erhalt des Bestands sicherstellen.

*Pakete bleiben intern unverändert*

Wenn sich die Zusammensetzung, der Inhalt oder der Archivierungsort der AIP-Paketinhalte verändert haben, ist dem durch eine weitere Form der Migration, das Repackaging, Rechnung zu tragen. Allerdings sehen weder OAIS noch DSEP eine Veränderung einmal eingebrachter Pakete vor. Stattdessen muss man die Pakete auf dem vorgesehenen Weg als DIPs dem Archiv entnehmen und nach der gewünschten Modifikation als SIPs (d. h. genauso wie ganz neue Pakete) wieder einbringen. Für Trivialänderungen wie das Repackaging werden Implementierungen der Archivierungssysteme vermutlich pragmatische Lösungen anbieten.

Die folgenden Maßnahmen sind sehr aufwändig, weil sie zum Teil für jedes einzelne Dokument durchzuführen sind.

*laufende Beobachtung der Technologie*

Im DSEP-Modell sieht ein Teilprozess von *Preservation* die Aktivität *Monitor Technology* vor, welche die Entwicklung einschlägiger Technologie (Hardware, Software, Speichermedien) laufend beobachtet und der Bibliotheksleitung geeignete Maßnahmen vorschlägt. Für den Migrationsansatz sind folgende Ereignisarten relevant:

- *Eine neue Umgebung erweist sich als relevant:* Die neue Umgebung wird zur Menge der Standardumgebungen hinzugenommen. Für jedes Dokument ist zu prüfen, ob es sich auf der neuen Umgebung abstützen kann. Ist das der Fall, dann sind die Metadaten der betroffenen Dokumente ent-

*Einbringen einer neuen Umgebung*

sprechend zu aktualisieren. Wird zu einer Standardumgebung eine neue Version der Betriebs- oder der Anwendungssoftware erworben, dann ist diese als AIP im Archiv zu speichern. Es kann geprüft werden, ob aufgrund von Abwärtskompatibilität frühere Versionen entbehrlich werden. Auch wenn die Hersteller Abwärtskompatibilität zusichern, empfiehlt es sich, dies am eigenen Dokumentenbestand zumindest stichprobenartig nachzuprüfen.

*Maßnahmen beim Auslaufen einer Umgebung*

- *Das Auslaufen einer der aktuellen Standardumgebungen.* Für jedes Dokument ist zu prüfen, ob es sich auf die aktuelle Standardumgebung abstützt. Ist das der Fall, dann sind die Metadaten der betroffenen Dokumente zu aktualisieren. Droht ein Dokument durch das Auslaufen der Umgebung unzugänglich zu werden, weil es künftig in keiner Umgebung (oder nur noch in demnächst auslaufenden Umgebungen) ausführbar ist, dann ist der »Dokumentenverlust« durch eine der folgenden Maßnahmen zu vermeiden:

*Ersatzumgebung*

  - Man findet eine neue Umgebung, in der das Dokument ausführbar ist, und nimmt diese zur Menge der Standardumgebungen hinzu. Das geschieht typischerweise, wenn die Betriebssoftware durch eine neuere, abwärtskompatible Version ersetzt wird.

*eine Migration durchführen*

  - Ist die zur Dokumentenbetrachtung notwendige Anwendungssoftware bald auf keiner Rechnerplattform mehr verfügbar, dann muss das Dokument in ein Format transformiert werden, das mit der zugehörigen Anwendungssoftware einer aktuelleren Standardumgebung betrachtet werden kann. Günstig dafür ist es, wenn das alte Dokumentenformat eines der Importformate der neuen Anwendungssoftware ist oder wenn die Konversion mittels eines verfügbaren Tools vorgenommen werden kann. Wenn die Formate der neuen und alten Anwendungssoftware sehr unterschiedlich sind, sind dabei Informationsverluste kaum zu vermeiden. Zu dieser Situation sollte es daher möglichst selten kommen. Um spätere Rückschlüsse auf den Originalzustand zu ermöglichen, sollte man solche Schritte in den Metadaten (bei der Migrationshistorie) gut dokumentieren und eventuell sogar die alte Fassung zusätzlich aufbewahren.

#### Zugriff auf gespeicherte Dokumente im Prozess Access

*Zugriff einfach*

Der Zugriff auf ein elektronisches Dokument ist beim Migrationsansatz problemlos, weil das Ziel ja gerade darin besteht, alle Dokumente jederzeit in einer aktuellen Standardumgebung lauffähig zu erhalten.

*Authentizität des Dokuments?*

Wenn aber sehr hoher Wert auf die Authentizität des Dokuments gelegt wird und dieses ein- oder mehrmals strukturverändernden Transformationen unterworfen wurde, dann wird man versuchen, mit Hilfe der Metadaten den Originalzustand zu erschließen. Das ist allerdings nicht nur sehr aufwändig, sondern auch der Erfolg ist nicht immer garantiert.

## 3.4 Chancen und Risiken des Migrationsansatzes

*Chancen*

Die wesentlichen Chancen wurden zum Teil schon mehrfach angesprochen:

- *Dokumente immer aktuell*

  Migration erlaubt es, jederzeit alle gespeicherten Dokumente mit aktuell verfügbaren Anwendungsprogrammen zu betrachten. Anders als beim Emulationsansatz verwendet der Betrachter also Software, die zu seiner Zeit *aktuell* ist und muss sich nicht in die Bedienelemente früherer Umgebungen einarbeiten. Angesichts der raschen Entwicklung der Benutzungsoberflächen von Computern ist kaum vorstellbar, dass in 100 Jahren Benutzer ohne Spezialwissen mit unseren heutigen Dokumentenbearbeitungs-Programmen umgehen könnten.

- *und bearbeitbar*

  Da Dokumente in einem aktuellen Format vorliegen, können sie mit aktuellen Werkzeugen bearbeitet werden (z. B. Suchwerkzeuge oder Copy-Cut-Paste-Mechanismen). Ebenso problemlos kann man sie im Internet publizieren oder an andere weitergeben. Diese Möglichkeiten sind heute so geläufig, dass man kaum auf diesen Mehrwert verzichten möchte.

- *Verbesserung der Darstellung*

  Migration in ein aktuelles Format bedeutet in vielen Fällen eine *Verbesserung* gegenüber dem Originalformat. Seit der Zeit der Schreibmaschinendokumente und ihrer elektronischen Gegenstücke sind reichlich Darstellungsmöglichkeiten hinzugekommen, nicht nur bei der Textgestaltung, sondern auch durch Einbettung von Graphiken oder Anima-

tionen, außerdem »aktive Textelemente« wie Spreadsheet-Zellen oder Hypertext-Links.

*wenig Schulungsaufwand*

- Durch laufenden Gebrauch sind die Mechanismen und Werkzeuge für die Migration Allgemeingut und erfordern wenig zusätzlichen Schulungsaufwand.

*Werkzeuge verfügbar*

- Wegen des großen Bedarfs werden Konversionswerkzeuge für wichtige Formate wahrscheinlich auch in Zukunft stets erhältlich sein.

*»lebendige« Dokumente*

- Die fortgesetzte Migration in immer neue Formate erfordert eine andauernde, qualitätssichernde Auseinandersetzung mit den Dokumenten des gesamten Bestands. Das erhält Dokumente in dauernder Erinnerung und damit »lebendig« .

*Risiken*

Der letzte Punkt weist bereits auf die mit dem Migrationsansatz verbundenen Risiken hin:

*keine einheitliche Methode, Ad-hoc-Werkzeuge*

- Migration ist *keine einheitliche Methode*, sondern ein Oberbegriff für eine Menge recht unterschiedlicher Transformationstechniken. Was ein solcher Schritt und das zu seiner Durchführung benötigte Werkzeug leisten müssen, hängt im Wesentlichen von den beteiligten Quell- und Zielformaten und deren Strukturverträglichkeit ab. Da immer wieder neue Dokumentenformate und -elemente hinzukommen, ist eine allgemeine Lösung nicht in Sicht: Konversionswerkzeuge werden vermutlich auch in absehbarer Zukunft immer wieder neu ad hoc zu entwickeln sein. Für das automatische Analysieren und Erzeugen von Dokumenten bestimmter Formate lassen sich bestehende Informatikwerkzeuge nutzen. Das Problem liegt darin, jeweils festzulegen, wann Dokumente verschiedener Formate als einander entsprechend betrachtet werden können.

*Gefahr von Verfälschungen*

- Wie mehrfach ausgeführt, ist bei transformativer Migration die *Gefahr von Verfälschungen* groß. Im Beispiel der Textgraphiken aus Abschnitt 1.2 könnte man etwa annehmen, dass der Autor nur mangels verfügbarer Technologie Liniengraphiken durch Textgraphiken angenähert hat und vermutlich ein echtes graphisches Format vorgezogen hätte. Wie er die Zeichnung aber tatsächlich gestaltet hätte, bleibt der Spekulation desjenigen überlassen, der den Migrationsschritt durchführt. Ohne direktes Befragen des Autors – eine bei der Langzeitarchivierung selten gegebene Option – ist dessen Absicht nicht zweifelsfrei zu rekonstruieren. Hier wäre es

sinnvoll, das Migrationsergebnis *zusätzlich* zu der ursprünglichen Fassung aufzubewahren und den Zusammenhang zu dokumentieren. Da bei Auslaufen ihrer Formate Dokumente immer wieder migriert werden müssen, akkumulieren sich im Laufe der Zeit Verfälschungen (wie beim Text der Bibel), wenn man nicht »in irgendeiner Weise« auf frühere Fassungen zurückgreifen kann.

- Da bei jedem Migrationsschritt alle Dokumente eines bestimmten Formats bearbeitet werden müssen und da die Zahl der archivierten Dokumente im Lauf der Zeit immer weiter ansteigt, ist die Migration ein *sehr aufwändiges Verfahren*. Das gilt schon, wenn alle Transformationen automatisch durchgeführt werden können. Qualitätssichernde oder restaurative Maßnahmen, die von Menschen durchgeführt werden, sind angesichts der enormen Datenvolumina auf lange Sicht nur für einige besonders wertvolle Dokumente bezahlbar.

*sehr hoher Aufwand*

# 4 Emulation

*Wie in Kapitel 1 dargestellt zielt der Emulationsansatz darauf ab, die Originalabspielumgebungen von Dokumenten durch Emulation auf künftigen Computern dauerhaft am Leben zu erhalten. Abschnitt 4.1 stellt Informatikbegriffe und -techniken vor, die beim Emulationsansatz eine Rolle spielen, und beschreibt an Beispielen, wie dort Emulation bislang eingesetzt wird. Abschnitt 4.2 handelt von der Anwendung des Emulationsansatzes auf die Langzeitarchivierung elektronischer Dokumente und diskutiert mögliche Varianten. Abschnitt 4.3 beschreibt die organisatorische Umsetzung innerhalb des in Kapitel 2 beschriebenen Rahmens. Abschnitt 4.4 fasst die Chancen und Risiken des Emulationsansatzes kurz zusammen.*

## 4.1 Emulation: Begriffsfeld und Anwendungsbeispiele

*Arten elektronischer Dokumente*

Elektronische Dokumente sind Daten, die von einem Computerprogramm interpretiert werden müssen, damit eine mit den menschlichen Sinnen wahrnehmbare Form entsteht. Überwiegend handelt es sich um visuelle und akustische Darstellungen. Weitere Sinneseindrücke wie Düfte sind noch »in Arbeit«. Zum Teil bilden elektronische Dokumente konventionelle Originaldokumente nach: Bücher, Bilder, Schallaufnahmen, 3D-Modelle von Bauwerken und Plastiken etc. In zunehmendem Maß treten daneben elektronische Originaldokumente wie Hypertexte und Spreadsheets sowie interaktive Dokumente auf, die sich während der Verwendung nach den Bedürfnissen der Konsumenten verändern. Dazu gehören z. B. elektronische Produktkataloge oder die Lektionen »intelligenter« Lerntexte. Insbesondere die elektronischen Originaldokumente sind stark mit ihrer Abspielumgebung verwoben. Einige enthalten sogar Programmstücke und werden damit selbst zum Bestandteil der Abspielumgebung.

*Originalabspielumgebungen emulieren*

Diese Beobachtungen motivieren die Grundannahme des Emulationsansatzes, nach der elektronische Dokumente am sichersten in authentischer Form für die Nachwelt erhalten bleiben, wenn man sie in Originalform speichert und außerdem dafür sorgt, dass in Zukunft stets ein Exemplar der Originalabspielumgebung verfügbar ist.[1] Wie in Abschnitt 1.3 ausgeführt, ist es nicht praktikabel, die Originalhardware dauerhaft funktionsfähig zu erhalten. Stattdessen – und daher der Name des Ansatzes – soll die Originalabspielumgebung auf künftigen Computern in ihrer Funktionsweise nachgebildet, also *emuliert* werden. Man kann das auch so ausdrücken: Emulation migriert anstelle der Dokumente die Abspielumgebungen. Eine wesentliche Voraussetzung für die Langzeitarchivierung im Allgemeinen und für den Emulationsansatz im Besonderen erscheint angesichts der enormen Leistungssteigerungen von einer Hardwaregeneration zur nächsten selbstverständlich erfüllt zu sein: Künftige Rechner, auf denen heutige Systeme emuliert werden sollen, müssen wesentlich leistungsstärker sein als unsere, da durch die Emulation Leistung verloren geht.

*Portierung von Software*

In der Informatik bezeichnet man die Migration von Software als *Portierung*. Aufgrund der bekannt kurzen Innovationszyklen bei der Hardware wird erhaltenswerte Software laufend von einer Rechnergeneration zur nächsten und von einer Systemfamile zur anderen portiert. Für die Portierung gibt es daher seit langem bewährte Vorgehensweisen und Werkzeuge. Wir stellen im Folgenden die einschlägigen Begriffe und Techniken vor, die beim Emulationsansatz von Nutzen sein können.

*Modellvorstellung von Computern*

Für unsere Betrachtung genügt eine relativ grobe *Modellvorstellung von Computern:* Neben Eingabegeräten (Tastatur, Maus, Mikrofon etc.) und Anzeigevorrichtungen (Bildschirm, Drucker, Lautsprecher etc.) besteht ein Computer im Wesentlichen aus dem Prozessor, dem Arbeitsspeicher und verschiedenen Hintergrundspeichern. Alle diese Komponenten sind durch Datenleitungen (»Busse« ) verbunden, auf denen Daten und Befehle transportiert werden können. Im *Arbeitsspeicher* befinden sich die Befehle und Daten der Programme, die gerade ausgeführt werden. Der schnelle Arbeitsspeicher ist teuer (daher eher klein) und vor allem auf laufende Stromversorgung angewiesen. Gerade nicht bearbeitete Programme und Daten werden deswegen auf erheblich langsameren, dafür aber auch wesentlich größeren und dauerhafteren *Hin-*

[1] In diesem Kapitel folgen wir weitgehend der Argumentation von Rothenberg (1995, 1999, 2000a, 2000b) sowie Rothenberg und Bikson (1999).

*tergrundspeichern* (Festplatten, Magnetbänder, optische Speichermedien etc.) aufbewahrt. Die eigentliche Verarbeitung geschieht im *Prozessor* (engl. *Central Processing Unit* – CPU), der im Wesentlichen aus Steuer- und Rechenwerk besteht. Aufgabe des *Steuerwerks* ist es, jeweils den nächsten Befehl nebst den benötigten Daten aus dem Arbeitsspeicher zu holen und dem Rechenwerk zur Ausführung zu übergeben. Das *Rechenwerk* ähnelt einem besonders leistungsfähigen Taschenrechner, der einfache Befehle wie Addition und Multiplikation sehr schnell ausführen kann. Dazu kommen Vergleichs- und Sprungbefehle, die über das Steuerwerk die Reihenfolge der Befehlsausführung beeinflussen. Typisch ist ein Vorrat von einigen zehn bis wenigen hundert solcher Befehlsarten.

*Programmiersprachen und Übersetzer*

Die Hardware eines Computers ist also eine Maschine, die relativ wenige, einfache Befehlsarten ausführen kann, diese aber enorm schnell. Andererseits stellt ein Computersystem dem Nutzer sehr vielfältige und komplexe Operationen zur Verfügung, in der Textverarbeitung etwa das Kopieren eines Dokuments oder das Umwandeln ganzer Textpassagen von Flattersatz in Blocksatz. Realisiert werden diese komplexen Operationen durch Software, d. h. letztlich durch das Ausführen langer Folgen einfacher Maschinenbefehle. Das Erstellen solcher Maschinenbefehlsketten ist sehr mühsam und fehleranfällig. Daher kam man schon früh[2] auf den Gedanken, bei der Programmentwicklung weniger detailbehaftete, »maschinenunabhängige« Befehlssprachen zu verwenden und darin formulierte Programme durch Übersetzungsprogramme (engl. *Compiler*) automatisch in Folgen von Maschinenbefehlen umsetzen zu lassen. Das ist möglich, weil die Programmbefehle ebenso wie Daten im Hauptspeicher liegen und so nach Bedarf entweder zur Maschinensteuerung oder als zu bearbeitende Daten verwendet werden können. Das in Abb. 4.1 dargestellte *T-Diagramm* repräsentiert graphisch einen Compiler, der Programme einer Befehlssprache P (wie Pascal, C oder Java) in eine andere Sprache Q überträgt. Der Buchstabe M verdeutlicht, dass der Compiler selbst als Programm in einer (maschinennahen) Befehlssprache abgefasst ist. Wie wir gleich sehen werden, kann M durchaus mit einer der beiden anderen Sprachen übereinstimmen.

[2] FORTRAN, die erste maschinenunabhängige Programmiersprache, entstand 1956. Sie ist (in mehrfach überarbeiteter Form) noch heute im Einsatz und wird u. a. von Ingenieuren zur Programmierung numerischer Verfahren genutzt.

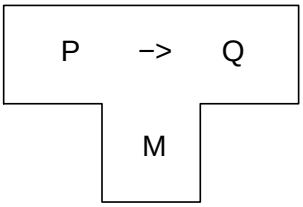



**Abbildung 4.1**
*T-Diagramm eines Übersetzers*

*T-Diagramme beschreiben Übersetzungsschritte*

Maschinenunabhängigkeit und automatische Übersetzung erleichtern die Portierung von Programmen. Um in einer Programmiersprache P abgefasste Programme von einer Plattform M (z. B. Motorola 68000) auf eine neue Plattform N (z. B. Intel Pentium) zu übertragen, genügt es nämlich, *einen* neuen Compiler zu entwickeln, der für N geeignete Befehlssequenzen erzeugt, und anschließend sämtliche P-Programme von diesem Compiler in N-Maschinenbefehle übertragen zu lassen. Mit den T-Diagrammen aus Abb. 4.2 können wir diese Aufgabe präziser[3] beschreiben: Gegeben ist ein Compiler, der auf der Maschine M läuft und P-Programme in die Maschinensprache von M übersetzt. Gewünscht ist ein entsprechender P-Compiler für die neue Plattform N.

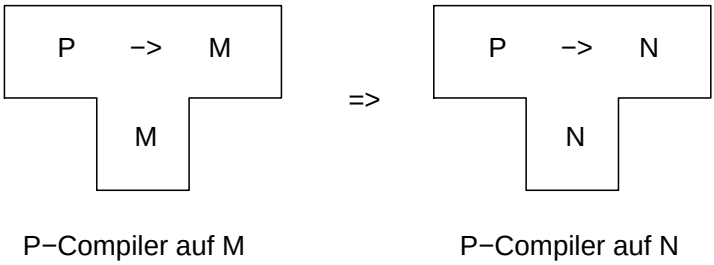



**Abbildung 4.2**
*Die Portierungsaufgabe*

*Analyse und Synthese*

Wie man weiß, lässt sich dieser Aufwand bei geschickter Organisation noch weiter reduzieren. Übersetzer für Programmiersprachen arbeiten ähnlich wie Menschen, die einen Text von einer Quell- in eine Zielsprache übertragen: Zuerst wird der Quelltext analysiert und dann (in der Synthese) entlang der gefundenen Struktur des Quelltextes systematisch der Zieltext erzeugt. Das Vorgehen bei der Analyse ist offenbar unabhängig von der zu erzeugenden Zielsprache. Wenn man also im P-Compiler auf M den Syntheseteil für M durch einen Syntheseteil für N ersetzt, dann entsteht der Compiler aus Abb. 4.3.

[3] Wem die nun folgende Diskussion zu technisch wird, der möge den Rest überspringen und nach dem letzten T-Diagramm (bei der Randnotiz »Emulatoren«) wieder einsetzen.

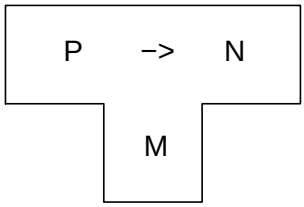



**Abbildung 4.3**
*Übersetzer mit neuem Syntheseteil*

*nur den Syntheseteil variieren*

Wie man sieht, ist damit die Aufgabe nicht ganz gelöst, weil der neue Übersetzer noch in der Maschinensprache M der alten Plattform abgefasst ist. Außerdem sind maschinennahe Programme (in M) detailbehafteter und damit unangenehmer zu modifizieren als Programme in einer problemnahen Programmiersprache (wie P).[4] Um letztlich Arbeit zu sparen, treibt man zunächst zusätzlichen Aufwand: Für eine Programmiersprache P, die voraussichtlich mehrfach zu portieren ist, erstellt man ein einziges Mal einen in P geschriebenen Übersetzer ohne Syntheseteil. Bei jedem Portierungsschritt wird dieser Übersetzer um einen Syntheseteil für die neue Zielplattform (hier N) ergänzt. Das ist alles, was man bei einem Portierungsschritt leisten muss. Der Rest erledigt sich in zwei Übersetzungsschritten »von selbst« .

*Cross-Compiler*

Im ersten Schritt wird der von Hand in P ergänzte Übersetzer in den P-Übersetzer auf M eingegeben, wodurch ein P-Übersetzer entsteht, der auf M läuft und Programme für die neue Plattform N erzeugt (ein so genannter Cross-Compiler; vgl. hierzu Abb. 4.4).

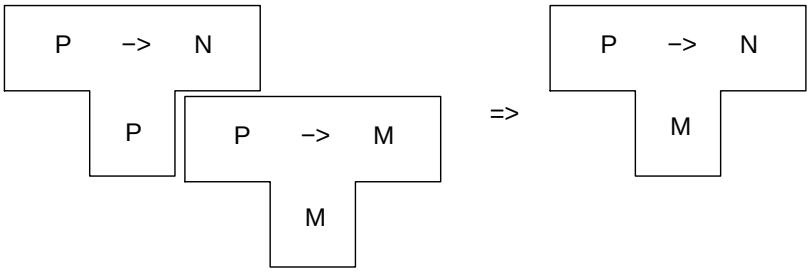



**Abbildung 4.4**
*Erzeugung des Cross-Compilers*

*Ziel-Compiler*

Der zweite Schritt sieht ähnlich aus wie der erste, nur wird diesmal der in P geschriebene Übersetzer nicht vom P-Übersetzer auf M verarbeitet, sondern von dem im ersten Schritt erzeugten Cross-Compiler. Das Ergebnis ist der gewünschte Ziel-Compiler aus Abb. 4.5.

[4] In den Abbildungen lassen sich T-Diagramme von Übersetzern wie Legosteine kombinieren, indem man passende Enden nebeneinanderlegt.

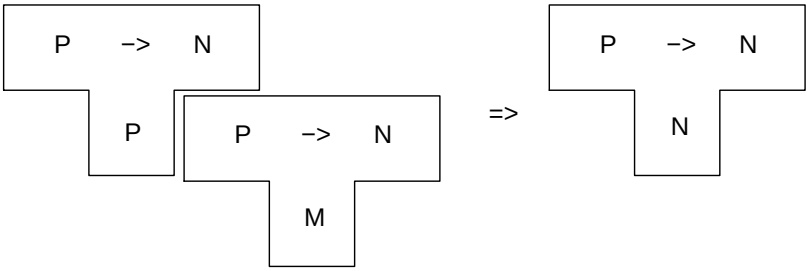



**Abbildung 4.5** *Erzeugung des Ziel-Compilers*

*Bootstrapping*

Dieses kompliziert erscheinende Vorgehen wird als *Bootstrapping* bezeichnet und hilft in der Praxis, Übersetzer mit möglichst geringem Aufwand von Plattform zu Plattform zu portieren. Mit Hilfe der dabei entstehenden Cross-Compiler und Ziel-Compiler kann man automatisch beliebige P-Programme auf die neue Zielplattform N portieren.

*Emulatoren*

Portierung per Übersetzung ist *eine* Methode, mit der man P-Programme, die auf einer Maschine M lauffähig waren, auf einer neuen Maschine N ausführbar machen kann. Eine andere Methode, um das gleiche Ziel zu erreichen, besteht darin, die Maschine M auf der neuen Maschine N zu emulieren. Ein *Emulator* ist ein Programm E, das auf N läuft und die Wirkung der einzelnen Befehle von M jeweils durch Befehlssequenzen auf N nachbildet (»interpretiert« ). Die Maschine N und der Emulator E zusammen verhalten sich so wie die ursprüngliche Maschine M. Insbesondere können also auf M lauffähige Programme unverändert auf der (N+E)-Kombination ausgeführt werden. Gegenüber der Portierung mit Übersetzern spart man sich die Übersetzungsvorgänge für die zu portierenden Programme. Dafür ist die Ausführungszeit eines M-Programms auf (N+E) in der Regel erheblich höher als die eines nach N übersetzten Programms. Erfahrungsgemäß sind übersetzte Programme laufzeiteffizienter als interpretierte.

*virtuelle Maschinen*

Speziell für Portierungszwecke verwendet man so genannte *virtuelle Maschinen*. Eine virtuelle Maschine wird nicht als Hardware gebaut, sondern so definiert, dass sie möglichst günstig auf konkreten Hardwareplattformen emuliert werden kann. Dieser Ansatz hilft beispielsweise, die Internetprogrammiersprache Java überall verfügbar zu machen:[5] Dazu wird ein fertiges Java-Programm mit Hilfe eines Übersetzers in Maschinenbefehle der standardisierten *Java Virtual Machine* (JVM) übersetzt. Da es für

[5] Das gleiche Konzept wurde schon Jahrzehnte früher bei den Programmiersprachen Pascal (P-Code) und Smalltalk erfolgreich eingesetzt, um deren Portierbarkeit zu erhöhen.

die meisten der im Internet zusammengeschlossenen Computer JVM-Emulatoren gibt, können die übersetzten Java-Programme praktisch im ganzen World Wide Web ausgeführt werden.[6]

*historische Beispiele*

Eines der Java-Programme, das auf diesem Weg jedermann zugänglich ist, emuliert den allerersten Computer, die Analytical Engine von Charles Babbage aus dem Jahre 1842.[7] Ein anderes historisches Beispiel: Im Jahr 1949 schrieb der Computerpionier Maurice Wilkes für den EDSAC Computer in Cambridge ein Programm zur Lösung von Differentialgleichungen. Ein Lochstreifen mit diesem Programm wurde 1979 wiederentdeckt und Jahre später auf einem Apple Macintosh Computer mit Hilfe eines EDSAC-Emulators »zum Leben erweckt«.

*Entwicklung von Computern mit Emulatoren*

Besonders wichtig sind Emulatoren bei der Entwicklung neuer Computer: Neue elektronische Komponenten werden per Software entworfen und getestet, um Kosten und Entwicklungszeit zu sparen. Durch Emulation lässt sich dabei die Funktionalität künftiger Prozessoren erproben und die Entwicklung zugehöriger Systemsoftware vorantreiben, ehe der erste Prozessor tatsächlich gefertigt wird. Mit feingranularer Emulation kann man Details wie Zeitabläufe in Schaltwerken schon am Softwaremodell untersuchen. Von den Hardwareherstellern selbst verwendete Emulatoren sind – weil authentisch – für den in diesem Kapitel vorgeschlagenen Emulationsansatz von zentraler Bedeutung.

*Emulatoren für Systemsoftware*

Um die Investitionen ihrer Kunden in Software zu schützen, bauen Hardwarehersteller manchmal in neue Rechner Emulatoren ein, die den Befehlssatz von Vorgängermodellen nachbilden und dadurch erlauben, Software abgelöster Rechnermodelle weiterzuverwenden. Ein Beispiel dafür ist die IBM 360, die Emulationen der älteren Modelle IBM 7090/7094 enthielt. Ein anderes Beispiel lieferte die Firma Apple, die beim Übergang zum PowerPC-Prozessor einen Emulator für die abgelösten Motorola 68000er Prozessoren bereitstellte, auf dem nicht nur die alten Benutzerprogramme, sondern während der Umstellung auch weite Teile der alten Systemsoftware liefen. Von eher untergeordneter Bedeutung ist es, ob die Emulatoren im Microcode (wie bei IBM) oder als normale Programme (wie bei Apple) abgelegt sind.

*Emulatoren in der Ausbildung*

Populär sind Emulatoren auch bei der Ausbildung von Informatikern in maschinennaher Programmierung:

[6] Um die Geschwindigkeit zu erhöhen, werden manchmal die Befehlsfolgen nicht auf einem JVM-Emulator interpretiert, sondern weiter in das Befehlsformat der Hardwareplattform übersetzt und dann direkt auf der Hardware ausgeführt.

[7] siehe http://www.fourmilab.ch/babbage/applet.html.

- Emulatoren von aufwändigen Prozessoren können den Studenten plattformunabhängig (z. B. auch auf ihren heimischen PCs) zur Verfügung gestellt werden.
- Emulatoren haben meist eine bessere Testunterstützung als reale Prozessoren.
- Interessante hypothetische Maschinen, die so niemals hergestellt worden sind, lassen sich emulieren.

*Emulatoren für Computerspiele*

Eine Programmkategorie, die bekannt hohe Anforderungen an Computer – und damit auch an Emulatoren – stellt, bilden die Computerspiele. Diese Anforderungen ergeben sich besonders bei der Programmierung graphischer Animationen, die aus Prozessor und Graphikkarte das Äußerste herauszuholen versucht. Verfechter des Emulationsansatzes betrachten es deswegen als ermutigend, dass Liebhabern veralteter Videospielplattformen und Spielecomputer im Internet hunderte von Emulatoren angeboten werden, mit denen sich die alten Spiele auf heutigen Rechnern ausführen lassen.

*Emulation als erprobte Technik*

Die genannten Beispiele weisen die Emulation von Maschinen als eine seit langem in vielen Bereichen erprobte Technik aus. Der Einsatz von Emulation bei der Entwicklung neuer Prozessoren und insbesondere auch bei der Nachbildung früherer Hardwareplattformen nährt die Hoffnung, dass man Rechner auf neueren Plattformen detailgetreu emulieren und auf diese Weise Software langfristig ausführbar halten kann.

## 4.2 Emulation in der Langzeitarchivierung

*Komponenten einer Abspielumgebung*

Gemäß Kapitel 1 besteht die Originalabspielumgebung für ein elektronisches Dokument aus unterschiedlichen Komponenten: Basis ist die Hardwareplattform, bestehend aus dem Computer sowie gegebenenfalls benötigter Zusatzausstattung; darauf aufsetzend die Systemsoftwareschicht mit dem Betriebssystem und erforderlichen Treibern; schließlich in der Präsentationsschicht die Anwendungsprogramme, die zum »Abspielen« des jeweiligen Dokumentenformats geeignet sind. Bei Textdokumenten werden hierfür neben reinen Anzeigeprogrammen (für Formate wie PDF, PostScript und DVI) häufig die Editoren verwendet (z. B. Word), mit denen das Dokument erstellt wurde.

### 4.2.1 Gegenstand der Emulation

*Emulationsalternativen*

Auf die Frage, was denn eigentlich zu emulieren sei, gibt es demnach drei plausible Antworten:

- nur die Hardwareplattform, also der Computer;
- der Computer mit Betriebssystem als »gebrauchsfertige« Einheit;
- die komplette Abspielumgebung mit Computer, Betriebssystem und Anzeigeprogramm.

*Speicherbedarf und Komplexität*

Entscheidet man sich für eine der beiden ersten Alternativen, dann ist mit jedem Dokument auch ein passendes Abspielprogramm, bei der ersten Alternative zusätzlich das passende Betriebssystem mitzuarchivieren, damit bei Bedarf wieder die vollständige Abspielumgebung hergestellt werden kann. Das spricht für Alternative drei. Dagegen spricht, dass es erheblich mehr Kombinationen aus Computer, Betriebssystem und Anzeigeprogramm gibt als Computer oder Computer mit passendem Betriebssystem, also auch erheblich mehr auf allen künftigen Plattformen bereitzustellende Emulatoren. Der geringere Umfang und die daraus folgende geringere Komplexität der Emulatoren sprechen für die erste Alternative.

*die »einfachste« Alternative*

Den Ausschlag gibt ein technisches Argument: Trotz langjährigen, intensiven Forschens und Bemühens ist es bis heute noch nicht Stand der Kunst, die Wirkung komplexer Programme (wie es z. B. Betriebssysteme und Texteditoren sind) unabhängig von ihrem Programmtext präzise und vollständig zu beschreiben. Die einzige Hoffnung, Systeme mit größeren Softwareanteilen auf künftigen Plattformen zu emulieren, bestünde deshalb darin, die Programmtexte aufzuheben und auf alle Zielplattformen zu portieren. Voraussetzung dafür wäre, die stets weiter wachsende Zahl der dabei benötigten Übersetzer und Programmbibliotheken auf allen künftigen Plattformen verfügbar zu halten – ein offenkundig aussichtsloses Unterfangen! Im Unterschied dazu kann Hardware erst dann gefertigt werden (teils von verschiedenen Herstellern), wenn ein präziser Bauplan vorliegt. Wie im letzten Abschnitt beschrieben, werden Emulatoren häufig schon beim Entwurf von Hardware eingesetzt. Man kann also davon ausgehen, dass sich Hardware hinreichend präzise beschreiben und durch Emulatoren nachbilden lässt. Von den drei betrachteten Alternativen bleibt damit nur die erste als gangbar übrig.

*Konfigurationsvarianten*

Computer eines bestimmten Typs sind oft in vielen unterschiedlichen Varianten und Ausbaustufen erhältlich. Typisch sind

Unterschiede in der Größe des Arbeitsspeichers, in der Anzahl und Art von Anschlussmöglichkeiten und angeschlossenen Geräten. Das betrifft auch Aspekte, die an der Benutzerschnittstelle zu wahrnehmbaren Unterschieden führen: Leistungsfähigkeit der Graphikkarte (evtl. vorhandener Graphikbeschleuniger) und Auflösung und Größe des Monitors (wie viele Punkte, wie viele Farben, wie groß die Bildschirmdiagonale). Es erhebt sich also die Frage, welche Konfigurationsunterschiede bei der Emulation als relevant und welche als nicht so wesentlich angesehen werden sollen. Ziel der Emulation ist es, das ursprüngliche Erscheinungsbild des elektronischen Dokuments zu reproduzieren. Daher kann die volle Bandbreite an Computertypen und -konfigurationen, auf denen das Dokument zur Entstehungszeit abspielbar war, als zulässig betrachtet werden: Es reicht aus, einen Emulator für *eines* dieser Systeme zu erhalten. Natürlich muss die Konfiguration das Emulators (Arbeitsspeicherausstattung etc.) mindestens den Anforderungen genügen, die in den Installationsvoraussetzungen des Abspielprogramms genannt wurden. Beispielsweise muss eine der als zulässig genannten Bildschirmauflösungen einstellbar sein. Bei Software – und das sind Emulatoren ja in der Regel – ist es aber kein Problem, quantitative Parameter wie Speichergröße und Bildschirmauflösung einstellbar zu halten.

*Abwärtskompatibilität reduziert Aufwand*

Um die Investitionen ihrer Kunden in Software zu sichern und dadurch die Kunden an sich zu binden, achten Computerhersteller auf *Abwärtskompatibilität* ihrer Produkte: Ein neuer Prozessor wird in der Regel erheblich leistungsstärker sein als das Vorgängermodell. Dieser Schritt ist »abwärtskompatibel«, wenn der Befehlssatz des neuen Prozessors den des alten Prozessors umfasst, d. h., wenn alle vom alten Prozessor ausführbaren Programme unverändert auf dem neuen Prozessor laufen. Mit anderen Worten: Der neue Prozessor emuliert den alten. Man ist bestrebt, Prozessorfamilien möglichst kontinuierlich und abwärtskompatibel auszubauen. Man riskiert erst dann einen Bruch, wenn die alten Strukturen die Weiterentwicklung zu stark behindern. Für eine ganze Familie abwärtskompatibler Prozessoren genügt ein einziger Emulator, nämlich der des leistungsstärksten Modells, das seinerseits alle seine Vorgänger emuliert. Aus den genannten Gründen reicht eine verhältnismäßig kleine Anzahl an Emulatoren aus, um ein großes Spektrum von Computern verschiedener Ausbaustufen und unterschiedlicher Ausstattungsvarianten zu emulieren.

*Softwareschichten*

Wie kann man sich nun die Verwendung von Emulation bei der Langzeitarchivierung konkret vorstellen? Hier geht es um die

technischen Aspekte; die organisatorische Abwicklung ist Gegenstand von Abschnitt 4.3. Neben dem Emulator für die originale Hardwareplattform werden zwei *Softwareschichten* benötigt:

- das Betriebssystem und gegebenenfalls erforderliche Treibersoftware;
- das Anwendungsprogramm zum »Abspielen« des Dokuments.

*Speicherabzug der Systemsoftware*

Um die Installation solcher Softwareschichten zu vereinfachen, schlägt Rothenberg vor, von der fertig installierten Software (die sich dann als Bitfolge im Arbeitsspeicher befindet) einen Speicherabzug zu machen. Diese Bitfolge kann dann später in den Arbeitsspeicher eines Emulators geladen werden, wodurch die Software sofort laufbereit zur Verfügung steht. Ein Problem bei diesem Ansatz ist, dass die beiden Softwareschichten nicht unabhängig voneinander gespeichert und geladen werden können, weil das Betriebssystem ja die Anwendungsprogramme verwaltet und daher intern Informationen über die geladenen Programme halten muss. Denkbar wäre es, nach Installation beider Softwareschichten einen gemeinsamen Speicherabzug zu bilden, wobei nach dem Laden in den Emulator die gesamte Abspielumgebung sofort betriebsbereit wäre. Dieser Ansatz verbietet sich wegen der damit verbundenen, enormen Ressourcenverschwendung: Mit jedem Anwendungsprogramm würde eine (jeweils fast identische Kopie) der umfangreichen Systemsoftware mit gespeichert.

*Installationsskript für Anwendungssoftware*

Sinnvoll erscheint folgender Weg: Man behandelt die Systemsoftwareschicht wie oben beschrieben, verwendet zur Installation der Anwendungssoftware aber ein automatisiertes Installationsskript, wie sie heute schon üblich sind. Angaben, die dabei typischerweise vom Benutzer erfragt werden (Auf welcher Festplatte wohin legen? Welche Programmteile installieren?), spielen in unserem Kontext keine Rolle bzw. können vorab entschieden werden, sodass in der Regel eine vollautomatische Installation möglich sein sollte.

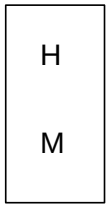
### 4.2.2 Emulationsvarianten

*Emulator als I-Diagramm*

Ein Emulator wird benötigt, sobald die verwendete Hardwareplattform H ausläuft und keine abwärtskompatible Nachfolgerversion mehr entwickelt wird. Spätestens dann ist eine zukunftsträchtige, neue Plattform M auszuwählen und ein Emulatorprogramm zu entwickeln, welches auf M den Befehlssatz von H emuliert. Abbildung 4.6 stellt den Emulator durch ein »I-Diagramm« dar, was betont, dass der Emulator sich nach außen (oben) hin

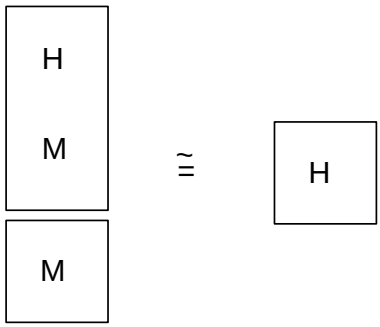
verhält wie Maschine H und selbst zu seiner Ausführung einen Prozessor benötigt, der den Befehlssatz von M realisiert.[8]

**Abbildung 4.6** *Emulator von H auf M*




Der Emulator, aufgesetzt auf die Maschine M, verhält sich wie die Maschine H (vgl. Abb. 4.7). Angenommen H wäre der Prozessor des Commodore C64 und M der Motorola 68000 Prozessor, dann verhält sich der auf dem Motorola 68000 ablaufende Emulator wie ein Commodore C64.

**Abbildung 4.7** *Verwendung des Emulators*




*zweistufige Emulation*

Wenn die neue Plattform M ihrerseits obsolet wird, kann man auch sie in gleicher Weise auf einer dann aktuellen Plattform N emulieren. Die für die Plattform H entwickelten Programme bleiben durch *zweistufige Emulation* ausführbar, wie es in Abb. 4.8 skizziert ist. Nehmen wir an, die Plattform N wäre konkret ein Intel Pentium Prozessor und H und M wie oben der Commodore C64 und der Motorola 68000 Prozessor, dann verhält sich jede der drei Säulen wie ein Commodore C64.

*mehrstufige Emulation*

Dieser Schritt lässt sich beliebig oft wiederholen und führt zu *mehrstufiger* (engl. *layered*) *Emulation*. Der Vorteil dieses Vorgehens ist, dass es sich im Prinzip beliebig oft iterieren lässt, ohne dass dadurch wesentlicher Zusatzaufwand entsteht: Für alle neuen Plattformen (wie oben M und N) braucht man ohnehin Emulatoren, um die für sie entwickelte Abspielsoftware lauffähig zu er-

[8] Wem die folgende Diskussion zu technisch wird, der möge den Rest überspringen und beim Fazit am Ende von Abschnitt 4.2 wieder einsetzen.

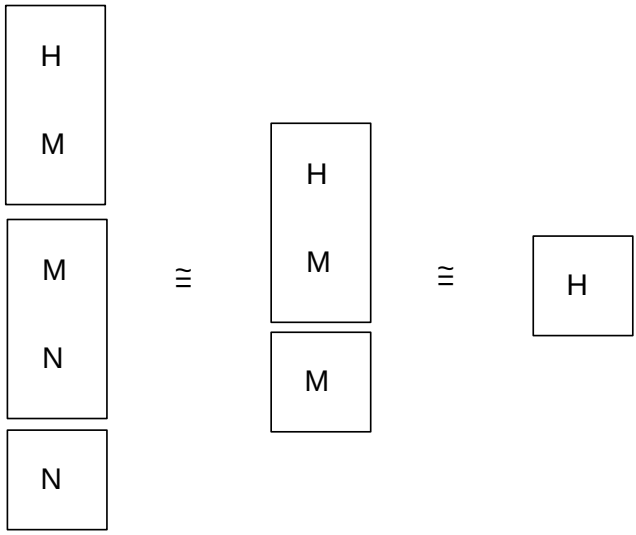



**Abbildung 4.8**
*Zweistufige Emulation*

halten. Der Nachteil der mehrstufigen Emulation ist der hohe Effizienzverlust: Jeder Emulationsschritt kostet einen Laufzeitfaktor, und entlang der Emulationskette multiplizieren sich diese Faktoren auf. So entstehen rasch sehr hohe Verlustfaktoren.

*kürzere Emulationsketten durch Übersetzung*

Geschickteres Vorgehen hilft, die Emulationsketten wesentlich kürzer zu halten. Ein Emulatorprogramm wird nämlich kaum direkt in der Maschinensprache der Maschine M abgefasst, auf der es laufen soll, sondern meist in einer höheren, maschinenunabhängigen Programmiersprache P. Der eigentliche Emulator entsteht durch Übersetzung des Emulatorprogramms mit Hilfe eines Übersetzers für P (in Abb. 4.9 von links nach rechts).

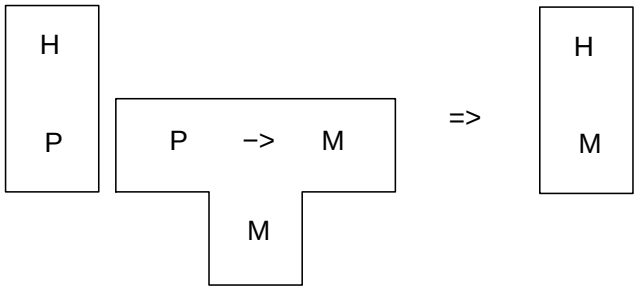


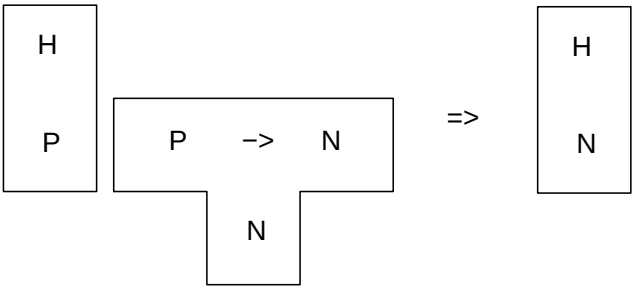

**Abbildung 4.9**
*Erzeugung eines Emulators*

*langlebige Programmiersprache nutzen*

Wählt man für P nun eine Programmiersprache, die weit verbreitet und relativ langlebig ist – wie heute z. B. die Sprache C – dann kann man mit Hilfe eines anderen P-Übersetzers den

Emulator ohne Aufwand auf eine andere Plattform bringen. Abbildung 4.10 verdeutlicht dieses Vorgehen.

**Abbildung 4.10**
*Erzeugung auf einer anderen Plattform*




*einstufig statt mehrstufig*

Auf diese Weise ersetzt man zweistufige (allg.: mehrstufige) Emulation durch einstufige. Diese Einsparmöglichkeit gibt es solange, wie für die jeweils neuen Plattformen P-Übersetzer entwickelt werden. Erst danach ist eine neue Stufe der mehrstufigen Emulation erforderlich.

*Wechsel der Programmiersprache*

Es gibt einige Variationen dieses Vorgehens. Wenn eine vielfach verwendete Programmiersprache P nicht mehr unterstützt wird, dann gibt es ein starkes kommerzielles Interesse daran, die große Menge vorhandener P-Programme in Programme einer neuen Programmiersprache Q zu konvertieren, am besten automatisch mit Hilfe eines Konversionsprogramms. Auch einen in P abgefassten Emulator kann man auf diese Weise in die neue Sprache Q übertragen und damit neue Emulationsstufen vermeiden. Sicherheitshalber sollte man aber den in P geschriebenen Emulator archivieren.

*Emulatorspezifikation*

Eine andere Variante ist der Ansatz der *Emulatorspezifikation*, bei dem der Emulator nicht als Programm einer gängigen Programmiersprache abgefasst, sondern in einer Spezifikationssprache S (für Emulatorspezifikationen) beschrieben wird. Ein der Spezifikation entsprechendes Emulatorprogramm kann zu einem späteren Zeitpunkt entwickelt werden. Dieses Vorgehen ist aus folgenden Gründen attraktiv:

*Spezifikation langlebiger als Programm*

- Es ist einfacher, die Spezifikation eines Prozessors in einer dafür geeigneten Spezifikationssprache abzufassen als ein Emulatorprogramm in einer Allzweckprogrammiersprache. Eine Spezifikation ist abstrakter als ein Programm, d. h. weniger mit unwesentlichen Details behaftet. Eine Spezifikation ist daher tendenziell kürzer und langlebiger als ein Programm.

- Da Hardwarehersteller ihre Produkte während der Entwicklung ohnehin spezifizieren müssen, kann man die Spezifikation eventuell sogar von ihnen beziehen (frühzeitig vereinbart im Interesse der archivierenden Institution, aber erst freigegeben, sobald der Hersteller den vollen Nutzen aus seinem Produkt gezogen hat).

*Spezifikation vom Hersteller*

- Beim Spezifikationsansatz ist man weitgehend von den Zufälligkeiten der Programmiersprachenentwicklung und -verbreitung entkoppelt. Ist die Spezifikationssprache sogar hinreichend formal, dann lassen sich wie bei einer Programmiersprache Programme per Übersetzer aus der Spezifikation generieren. Anderenfalls dient die Spezifikation als Arbeitsgrundlage der manuellen Erstellung eines Emulatorprogramms in nicht allzu ferner Zukunft – solange die Spezifikation noch von Fachleuten verstanden wird.

*Generierung von Emulatoren*

Da Emulatorspezifikationen für Menschen lesbare (elektronische) Dokumente sind, liegt es nahe, auch sie zu archivieren.

### 4.2.3 Verwendung virtueller Maschinen

Bei Verwendung *virtueller Maschinen*, wie sie in Abschnitt 4.1 eingeführt wurden, reduziert sich die Anzahl der benötigten Emulatorportierungen drastisch: Anstatt Emulatoren aller benötigten Plattformen auf alle künftigen Plattformen zu portieren, verwendet man *eine* virtuelle Maschine als gemeinsame Plattform für alle Emulatoren und emuliert die virtuelle Maschine ihrerseits auf allen künftigen Plattformen. Für die Aufgabe, die $m$ alten Plattformen $\mathtt{M}_1, \mathtt{M}_2, \ldots, \mathtt{M}_m$ auf $n$ neuen Plattformen $\mathtt{N}_1, \mathtt{N}_2, \ldots, \mathtt{N}_n$ verfügbar zu machen, braucht man ohne virtuelle Maschine einen Emulator wie aus Abb. 4.11 für jedes Paar $\mathtt{M}_\mathtt{i}$, $\mathtt{N}_\mathtt{j}$, d. h. insgesamt $m \cdot n$ Emulatoren.

*weniger Portierungen bei virtuellen Maschinen*



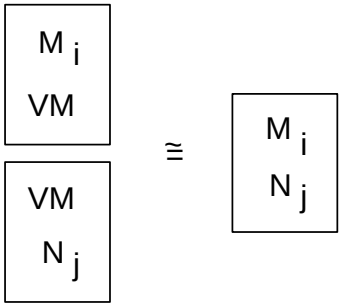

**Abbildung 4.11** *Kombinatorische Explosion der Emulatoren*

Bei Verwendung einer virtuellen Maschine VM ersetzt eine Kombination den obigen Emulator (vgl. Abb. 4.12):

Um alle $\mathtt{M}_i$ auf der virtuellen Maschine zu emulieren, braucht man $m$ Emulatoren und um die virtuelle Maschine auf allen $\mathtt{N}_j$

*statt $m \cdot n$ nur $m + n$ Emulatoren*

**Abbildung 4.12**
*Verwendung einer virtuellen Maschine*

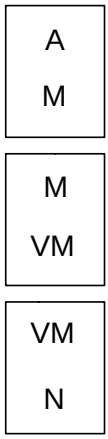



verfügbar zu machen, weitere $n$ Emulatoren, insgesamt also nur noch $m + n$ Emulatoren.

*verschlechterte Effizienz ...*

Andererseits verlängert die Verwendung einer virtuellen Maschine die Emulationskette um eins und verschlechtert dadurch die Ausführungszeiten. Wenn man auf diese Weise ein Abspielprogramm A, das für die alte Plattform M entwickelt wurde, auf der neuen Plattform N ausführen möchte, ergibt sich die in Abb. 4.13 dargestellte Ausführungs- und Emulationskette:

**Abbildung 4.13**
*Beispielhafte Ausführungs- und Emulationskette*




*durch Übersetzungsschritte verbessern*

Mit Hilfe geeigneter Übersetzer kann man diese Kette je Übersetzungsschritt um eins verringern (siehe Abb. 4.14). Der erste Übersetzer übersetzt das in P geschriebene Abspielprogramm nicht wie ursprünglich in die Maschinensprache von M, sondern in die der virtuellen Maschine:

Der nächste Übersetzungsschritt eliminiert die virtuelle Maschine als Zwischenstufe (siehe Abb. 4.15).

Bei $m$ alten und $n$ neuen Plattformen bräuchte man $m + n$ Übersetzer, um diese Emulationsketten vollständig zu reduzieren.

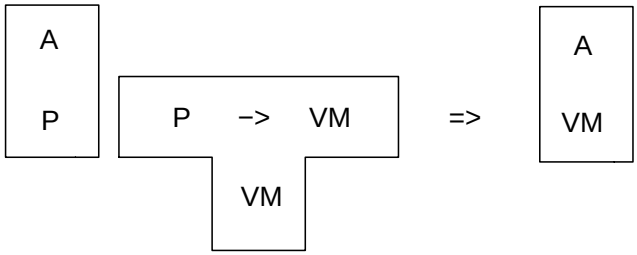



**Abbildung 4.14**
*Ausführungs- und Emulationskette mit virtueller Maschine*

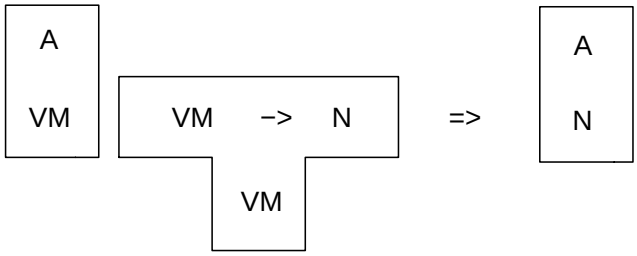



**Abbildung 4.15**
*Ausführungs- und Emulationskette ohne virtuelle Maschine*

## Fazit

*Rückgriff auf bewährte Techniken*

Bewährte Techniken aus Emulation und Übersetzerbau erlauben es prinzipiell, Programme obsoleter Computer dauerhaft auf aktuellen Plattformen verfügbar zu halten. Bei günstiger Organisation entspricht die Anzahl der benötigten Emulatoren und Übersetzer der Anzahl der alten und neuen Hardwareplattformen.

*günstige zeitliche Verteilung*

Günstig ist auch die zeitliche Verteilung der anfallenden Arbeiten: Es bietet sich an, Emulatoren (Übersetzer) für die virtuelle Maschine zu alten Plattformen dann zu erstellen, wenn sie auslaufen. Ebenso entwickelt man Emulatoren (Übersetzer) für die virtuelle Maschine auf neuen Plattformen immer dann, wenn diese Plattformen entstehen. Auf diese Weise sind nicht nur stets alle benötigten Emulationsbausteine verfügbar; es ist auch der Arbeitsaufwand kontinuierlich über die Zeit verteilt.

*Problem des Dokumentenkontextes*

Ein grundsätzliches Problem, das nicht nur beim Emulationsansatz besteht, aber dort besonders deutlich sichtbar wird, ist das der Abgrenzung und Konservierung eines geeigneten *Dokumentenkontextes*. Bei der Diskussion sind wir bisher immer von einer abgeschlossenen Umgebung – bestehend aus Hardware und Systemsoftware – ausgegangen. Zunehmend greifen Abspielprogramme aber auch auf Daten außerhalb dieser Ausführungsumgebung im engeren Sinn zu:

- Bilanzen und Statistiken ebenso wie eBusiness-Kataloge beziehen Daten dynamisch aus Datenbanken, die auf dem gleichen oder einem anderen Computer liegen.
- E-Mail-Attachments binden Dokumente ein, die eigene Abspielprogramme erfordern.
- Hypertexte beziehen sich wie selbstverständlich auf andere Dokumente, die zurzeit irgendwo im Internet verfügbar sind.

*Was gehört zum Dokument?*

Bei der Archivierung solcher Dokumente ist zu klären, welche Daten als zum Dokument gehörig zu betrachten sind und welche zusätzlichen Anzeigeprogramme sie erfordern. Solche Daten müssen statisch verfügbar gemacht und als Kontext zusammen mit dem elektronischen Dokument gespeichert werden.

## 4.3 Archivierungsprozesse beim Emulationsansatz

*organisatorische Einbettung bei Emulation mit virtueller Maschine*

Wir betrachten nun die Einbettung des Emulationsansatzes in den in Kapitel 2 beschriebenen organisatorischen Rahmen. Es geht dabei um die Zuordnung von Emulationsaktivitäten zu den einzelnen Archivierungsprozessen des OAIS-Modells bzw. des durch Verfeinerung daraus hervorgegangenen DSEP-Modells.[9] Um die Diskussion nicht unnötig aufzublähen, wählen wir aus den im Abschnitt 4.2 dargestellten Varianten die Emulation mit virtueller Maschine aus. Dies ist die nach unserer Einschätzung wohl zukunftsträchtigste Variante. Abbildung 4.16 zeigt die Bestandteile dieses Ansatzes im Zusammenhang: Die linke Säule entspricht der Originalabspielumgebung. Alle Komponenten des Bildes außer der Originalplattform bilden zusammen die Abspielumgebung unter Emulation mit virtueller Maschine. Andere Emulationsvarianten wären analog zu behandeln.

*Inhalt der Datenkapsel*

Nun zum Inhalt der Datenkapsel: Um ein digitales Dokument auf einer aktuellen Plattform abspielen zu können, müssen neben dem eigentlichen Dokument die Softwareumgebung (Anwendungsprogramm und Systemsoftware) sowie die Emulation der Originalplattform (Emulatorprogramm und virtuelle Maschine) aufbewahrt werden. Für die tatsächliche Verwendung des Dokuments benötigt man zusätzlich Beschreibungen, die erläutern, aus welchen Bestandteilen das elektronische Dokument zusammengesetzt ist und wie man die vier Programme der Softwareumgebung

[9] Hier werden ausschließlich die Prozesse betrachtet, auf die sich Emulation auswirkt, und auch diese Prozesse werden nur kurz skizziert.

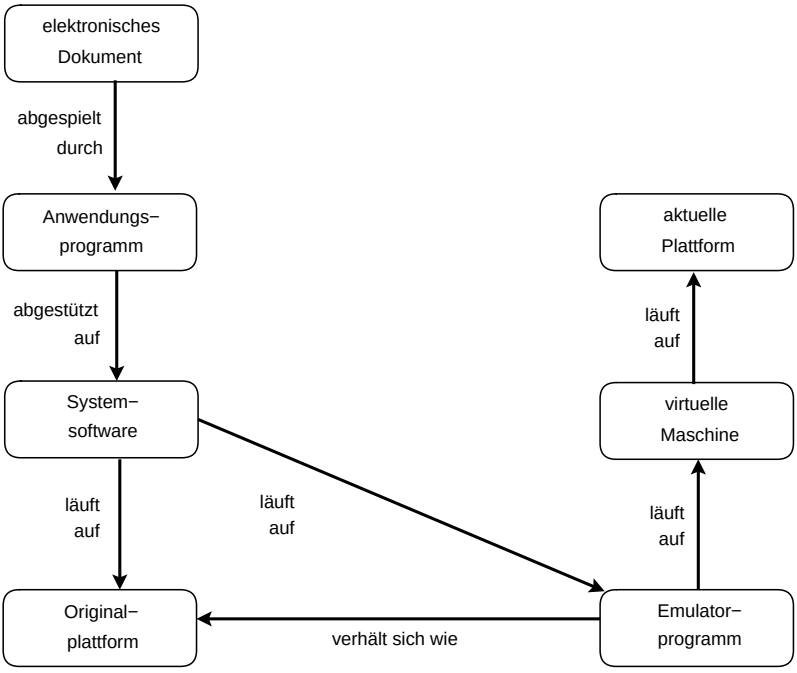



**Abbildung 4.16**
*Emulation mit virtueller Maschine*

und des Emulators installieren und betreiben kann. Mit weiteren Angaben wie Name des Autors, Verlag, Erscheinungsort und -datum, ISBN bilden sie die *Metadaten*, die sagen, *was* im Dokument enthalten ist und *wie* man darauf zugreift.

*Speicherung der Softwareumgebung*

Abbildung 4.17 zeigt die vollständige, ein elektronisches Dokument und seine Emulationsumgebung betreffende Informationsmenge. Man spricht von einer *Speicherkapsel*, da diese Bestandteile logisch zusammengehören. Damit ist nicht festgelegt, dass die Teile der Speicherkapsel auch physisch zusammen abgelegt werden. Da sich in der Regel viele Dokumente die gleiche Softwareumgebung und Emulationssoftware teilen, liegt es nahe, die gemeinsamen Teile herauszulösen und nur einmal zu speichern (bzw. aus Sicherheitsgründen redundant in einigen wenigen Kopien).

*Metadaten in »lesbarer« Form speichern*

Diese Aufteilung (und das spätere Zusammenführen der Bestandteile) ist Aufgabe des Archivierungssystems; dieses enthält zur Unterstützung von Recherchen und Zugriffen zumindest auch einen Teil der Metadaten direkt. Bezüglich der Speicherungsform folgt daraus ein wichtiger Unterschied zwischen den Metadaten und dem Rest der Datenkapsel: Metadaten müssen in einer für das Archivierungssystem (und indirekt auch die menschlichen Benutzer) »lesbaren« Form gespeichert werden. Wie schon in Kapi-

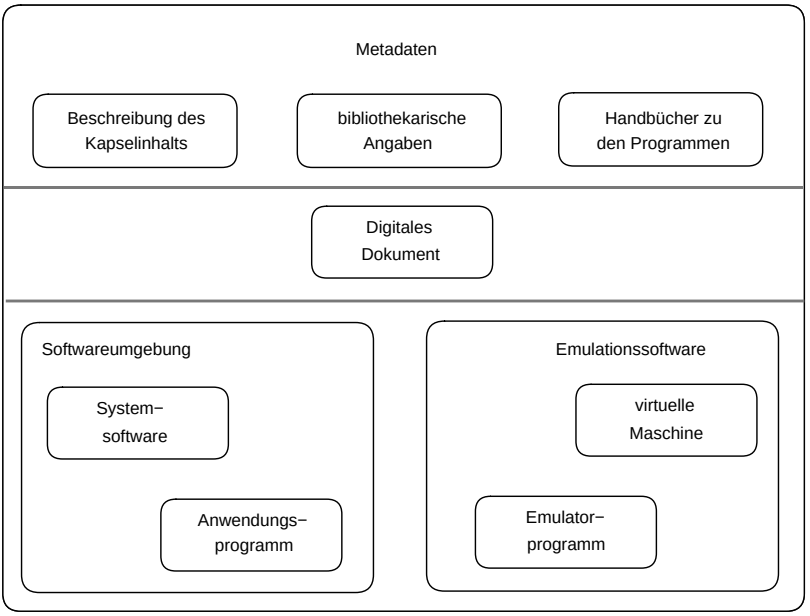



**Abbildung 4.17** *Aufbau der vollständigen Speicherkapsel*

tel 1 erwähnt, bieten sich hierfür einfacher Text (in ASCII, Unicode o. ä.) bzw. ein geeignetes XML-Format an. Die laufende Lesbarkeit der Metadaten kann durch einfache (möglichst umkehrbare) Migrationsschritte sichergestellt werden.

*Emulationssoftware aktualisieren*

Vom Rest der Datenkapsel können das eigentliche Dokument und die zu seiner Interpretation erforderliche Softwareumgebung als langfristig unveränderte Bitströme abgelegt werden; die Emulationssoftware ist schritthaltend mit der technischen Entwicklung zu aktualisieren.

*DSEP-Prozesse*

Bezogen auf das in Kapitel 2 beschriebene DSEP-Prozessmodell fallen beim Emulationsansatz folgende Aufgaben an:

#### Aufnahme eines neuen elektronischen Dokuments in den DSEP-Prozessen Delivery & Capture und Ingest

*Abspielumgebung liegt vor*

In der Mehrzahl der Fälle wird die für das Dokument benötigte Abspielumgebung bereits im DSEP-Archiv gespeichert vorliegen. Es genügt dann,

- zu prüfen, ob das Dokument tatsächlich in der angegebenen Abspielumgebung ausführbar ist, und
- die benötigten Metadaten zu erfassen und zu standardisieren. Dazu gehören neben den üblichen bibliographischen An-

gaben insbesondere die Verweise auf die (Ablagestelle der) Softwareumgebung und die passende Emulationssoftware.

*Beteiligung des Produzenten*

Rothenberg schlägt vor, die Überprüfung der Ausführbarkeit über geeignete Internet-Schnittstellen bereits vom Produzenten durchführen zu lassen, bevor das Dokument zur Archivierung eingereicht wird. Bibliotheksseitig könnte man sich dann auf qualitätssichernde Stichproben und die Bearbeitung der Metadaten beschränken. Eingereichte elektronische Dokumente werden in keinem Fall verändert, sondern entweder zurückgewiesen oder als Originalbitfolge gespeichert.

*Abspielsoftware noch nicht im Archiv*

Ist die vom Dokument benötigte Softwareumgebung (oder die erforderliche Version davon) noch nicht im Archiv vorhanden, wohl aber eine passende Emulationssoftware, dann sollte *vor* der Aufnahme des Dokuments in Zusammenarbeit mit der Archivverwaltung die Softwareumgebung selbst als eigenes Paket ins Archiv eingebracht werden. Dies löst möglicherweise eine Reihe weiterer organisatorischer Schritte aus, wenn die neue Softwareumgebung eine früher gespeicherte vollständig ersetzen kann.

*gesamte Abspielumgebung fehlt*

Als letzter Fall bleibt die Situation zu betrachten, dass weder die Softwareumgebung noch eine passende Emulationssoftware im Archiv gespeichert sind. Wenn die Archivverwaltung entscheidet, dass die Abspielumgebung vom Archiv künftig unterstützt werden soll, dann muss zunächst die passende Emulationssoftware beschafft und gespeichert werden, danach die Softwareumgebung und zum Schluss das elektronische Dokument. Der aufwändigste Schritt ist die Beschaffung der Emulationssoftware. Als Anbieter kommen u. a. der Hardwarehersteller und spezialisierte Softwarehäuser in Betracht. Wie in Abschnitt 4.1 dargestellt wird Emulationssoftware auch in anderem Zusammenhang produziert und verwendet.

*Emulationssoftware rechtzeitig beschaffen*

Solange die ursprüngliche Hardwareplattform verfügbar ist, wird streng genommen die Emulationssoftware noch nicht benötigt. Um in einem Vergleich die Authentizität und Güte der Emulationssoftware prüfen zu können, sollte man sie aber rechtzeitig vor dem Auslaufen der Originalhardware erwerben. Erst danach kann man von einem lückenlosen und dauerhaften Erhalt der zugehörigen Dokumente ausgehen.

### Erhaltung des Dokumentenbestands in den beiden DSEP-Prozessen Archival Storage und Preservation

*Bitstrom erhalten*

Wie bei anderen Strategien muss der Bitstrom des elektronischen Dokuments von Zeit zu Zeit »aufgefrischt« werden, bevor die Magnetisierung nachlässt. Mit gängigen Techniken der Datenspeicherung (wie etwa Prüfsummen, »Error Correcting Codes«, redundanten Kopien o. ä.) lassen sich Datenverluste mit sehr hoher Wahrscheinlichkeit vermeiden. Dieser Vorgang betrifft nicht nur den Bitstrom des Dokuments, sondern ebenso den gesamten Inhalt der Datenkapsel.

*Metadaten migrieren*

Die Migration der »unmittelbar lesbaren« Metadaten in ein aktuelleres Format ist jeweils nach relativ langen Zeitabständen erforderlich, z. B. anlässlich einer Erneuerung des Archivierungssystems (Versions- oder Systemwechsel). Um in Zweifelsfällen auf frühere Versionen zurückgreifen zu können, sollte man umkehrbare Transformationen verwenden, etwa von ASCII in einen Teilbereich von Unicode, oder die alten Versionen archivieren.

*für Preservation relevante Ereignisse*

Die übrigen, aufwändigeren Maßnahmen beziehen sich auf den Erhalt der Emulationssoftware, sind also nicht für jedes einzelne Dokument, sondern für die (deshalb am besten in eigenen AIPs gespeicherten) Programme durchzuführen. Im DSEP-Modell sieht ein Teilprozess von *Preservation* die Aktivität *Monitor Technology* vor, welche die Entwicklung einschlägiger Technologie (Hardware, Software, Speichermedien) laufend beobachtet. Für den Emulationsansatz sind folgende Ereignisarten besonders relevant:

- *Das Auslaufen einer aktuellen Plattform steht an*:
  Spätestens zu diesem Zeitpunkt sollte die Emulatorentwicklung für diese Plattform abgeschlossen sein.
- *Eine neue Hardwareplattform erweist sich als relevant*:
  Dann kann mit der Emulatorentwicklung für die neue Hardwareplattform begonnen werden. Außerdem sollte – unabhängig davon – die virtuelle Maschine umgehend auf die neue Hardwareplattform portiert werden.
- *Zu einer der relevanten Hardwareplattformen erscheint eine neue (Version der) Systemsoftware*:
  Diese ist als AIP im Archiv zu speichern. Es kann geprüft werden, ob aufgrund von Abwärtskompatibilität frühere Versionen entbehrlich werden.

**Zugriff auf gespeicherte Dokumente im Prozess Access**

Zugriff auf ein elektronisches Dokument bedeutet beim Emulationsansatz:

- Wenn die Originalplattform noch verfügbar ist, dann ist gegebenenfalls eine passende Systemsoftware- und Anwendungssoftwareversion zu installieren (sofern nicht schon vorhanden), das Dokument zu laden und zu betrachten.
- Ist die Originalplattform nicht mehr verfügbar, dann müssen auf einer aktuellen Plattform die virtuelle Maschine und der passende Emulator geladen werden. Danach geht es weiter wie im ersten Fall.

*Gebrauchskopie ableiten*

Um den Aufwand beim Zugriff zu verringern, wird man häufig benötigte Plattformen und Softwareumgebungen fertig installiert bereithalten. Wird ein einzelnes Dokument häufig benötigt, dann lohnt es sich, durch Migration aus dem Original eine auf aktuellen Plattformen abspielbare Kopie abzuleiten und zusätzlich im Archiv zu speichern.

*Zugang zum Original*

Legt ein Benutzer hohen Wert auf die Authentizität des Dokuments, dann wird er den aufwändigeren Weg beschreiten und eine Emulation der Originalplattform installieren. Dazu kann auch das Studium der mit den Metadaten gespeicherten (und im Laufe der Zeit migrierten) Handbücher notwendig sein.

## 4.4 Chancen und Risiken des Emulationsansatzes

Die wesentlichen Chancen wurden schon mehrfach angesprochen:

*Authentizität*

- Emulation bietet ein größtmögliches Maß an *Authentizität*, weil das Originaldokument über den gesamten Zeitraum bitweise unverändert bleibt und weil man bestrebt ist, die ursprüngliche Hardware durch Emulationsprogramme funktionsgetreu nachzubilden.

*alle Dokumententypen*

- Bezüglich der unterstützten *Dokumententypen* gibt es keine Einschränkungen. Auch das dynamische Verhalten von Dokumenten bleibt – innerhalb eines mit zu speichernden Kontextes – dauerhaft erhalten.

*Aufwand unabhängig von der Dokumentenanzahl*

- Im Unterschied zur Migration ist der für den Erhalt der Dokumente erforderliche Aufwand (mit Ausnahme der von allen Strategien benötigten Auffrischphasen) *nicht* von der An-

zahl der im Archiv gespeicherten Dokumente abhängig, sondern nur von der Anzahl der unterstützten Quell- und Zielplattformen: Für jede Quellplattform ist ein Emulationsprogramm zu entwickeln, auf jede Zielplattform die virtuelle Maschine zu portieren.

Ambivalent ist der folgende Punkt:

*Emulation leistet mehr als benötigt*

- Die Emulationstechnik *leistet mehr* als nur elektronische Dokumente dauerhaft zugreifbar zu erhalten. Sozusagen »nebenbei« werden praktisch beliebige Programmsysteme für künftige Generationen erhalten. Das eröffnet Chancen in ganz anderen Anwendungsbereichen: Beispielsweise sind Finanzämter gesetzlich gehalten, Bescheide auch nach zehn Jahren noch originalgetreu nachvollziehen zu können. Wenn deren Software durch ein neues System ersetzt wird, weil die alte Plattform ausläuft, dann wäre eine Emulationslösung vermutlich ein günstiger Weg, um Einsprüche auch nach Jahren noch sicher behandeln zu können.

*eine sparsamere Variante*

Auf der anderen Seite liegt die Vermutung nahe, dass ein weniger leistungsfähiger Ansatz auch mit weniger Aufwand zu realisieren sein müsste. In diese Richtung gehen die Vorschläge von Lorie (2000), der Emulation auf eine Maschine anwenden möchte, die ein noch zu entwickelndes, universelles Dokumentenformat (»XML-ähnlich«) interpretieren kann.

Großen Chancen stehen beim Emulationsansatz erhebliche Risiken entgegen:

*Ansatz wirklich tragfähig?*

- Trotz der in Abschnitt 4.1 genannten, Hoffnung machenden Beispiele bestehen grundsätzliche Zweifel, ob ausreichend detailgetreue und effiziente Emulatoren mit vertretbarem Aufwand erstellt werden können. Um die *Tragfähigkeit des Ansatzes* zu bestätigen, wären systematische Experimente erforderlich.

*urheberrechtliche Probleme*

- Der langfristige Einsatz kommerzieller Anwendungs- und Systemsoftware wirft *urheberrechtliche Probleme* auf, die noch zu lösen sind. In noch stärkerem Maß gilt das für die Rechte an den durch Emulatoren nachgebildeten Rechensystemen. Entsprechende Lizenzen müssten im Einzelfall erst von den Herstellern erworben werden.

*Zugang im Detail problematisch*

- Noch nicht ausreichend geklärt ist auch die Frage nach der konkreten Benutzung elektronischer Dokumente in ferner

Zukunft, wenn Tastatur und Maus einmal nicht mehr Standardbedienelemente sein werden und vermutlich die heutigen Interaktionsformen an der *Mensch-Maschine-Schnittstelle* durch ganz andere Mechanismen abgelöst sein werden. Ohne Kontextwissen werden in Zukunft nicht einmal die migrierten Handbücher verständlich sein. Für Nichtspezialisten kommt dann wahrscheinlich nur die Nutzung von Dokumentenkopien in Frage, die jeweils durch Migration in die aktuelle Technologie entstehen müssen. Die Verwendung der digitalen Originale selbst wird künftigen »Computer-Archäologen« vorbehalten sein.

# 5 Dokumentenbeschreibung

*In diesem Kapitel werden wir näher auf die Beschreibung von Dokumenten, d. h. ihre Auszeichnung mit Metadaten, eingehen. Dokumentenbeschreibung wird häufig auch als Markup bezeichnet. Es geht also um das Markieren von bestimmten Dokumentteilen, um diese Teile gegenüber anderen auszuzeichnen. Beispiele hierfür sind Hervorhebungen für das menschliche Auge, Korrekturmarken oder aber die Bedeutung (Semantik). Wie schon erwähnt, handelt es sich bei Metadaten um Daten über Daten, nicht um die eigentlichen Inhalte selbst. Wie wir später sehen werden, ist es jedoch durchaus üblich und auch sinnvoll, Metadaten mit den Dokumenteninhalten zu verschränken.*

*Dieses Kapitel ist wie folgt aufgebaut: Nach einem einführenden Beispiel in Abschnitt 5.1 werden wir in Abschnitt 5.2 die verschiedenen Möglichkeiten der Dokumentenbeschreibung klassifizieren. Auf die Anwendungsmöglichkeiten im Bereich der Langzeitarchivierung gehen wir in Abschnitt 5.3 ein. Ein wenig technischer wird es in Abschnitt 5.4, in dem wir das Problem der Konsistenz zwischen einem Dokument und seiner Beschreibung behandeln. Zu den eigentlichen Standards für die Dokumentenbeschreibung kommen wir in Kapitel 6.*

## 5.1 Einführendes Beispiel

*Aufsatzkorrektur*

Eine spezielle Form von Metadaten, das Auszeichnen von Texten, ist uns aus der Schule bekannt: Lehrer versehen Aufsätze (Texte) mit gewissen Zeichen (Metadaten), die uns davon überzeugen sollen, dass mit bestimmten Textteilen etwas stimmt oder eben nicht. Dazu benutzen Lehrer eine spezielle Syntax, beispielsweise die Unterstreichung in Rot und diverse Zeichen am Seitenrand wie senkrechte oder waagerechte Striche. Am Schluss steht dann zumindest ein Kommentar oder eine Note. Alles was Lehrer in unsere Aufsätze schreiben, sind Metadaten.

*Sprache für Metadaten*

Bei obigem Beispiel wird noch ein weiterer Punkt deutlich, dem wir unbedingt unsere Aufmerksamkeit schenken sollten: Wir müssen die *Sprache*, die unser Lehrer zum Auszeichnen verwendet hat, zumindest verstehen und interpretieren können. Sollte dies nicht der Fall sein, so können wir mit den Zeichen unseres Lehrers im »günstigsten« Fall nichts anfangen und sie verwerfen, in ungünstigen Fällen jedoch fehlinterpretieren.

*Standards*

Im Allgemeinen versuchen standardisierte Dokumentenbeschreibungs-Sprachen solche Fehlinterpretationen zu vermeiden, indem sie einen gewissen Zeichenvorrat (Sprache) und dessen Interpretationsmöglichkeiten (Bedeutung) in geeigneter Weise festlegen. Für unser obiges Beispiel bietet sich der Duden an, der festlegt, wie Korrekturzeichen syntaktisch aussehen und was sie bedeuten. Abbildung 5.1 zeigt einen kurzen Ausschnitt der Korrekturvorschriften des Duden (2000).

**Fehlende Buchstaben** werden angezeichnet, indem der vorangehende oder folgende Buchstabe durchgestrichen und zusammen mit dem fehlenden wiederholt wird. Es kann auch das ganze ~~Wri~~ oder die Silbe durchgestrichen und auf dem Rand berichtigt werden.

**Fehlende Wörter (Leichen)** werden in der Lücke durch Winkelzeichen gemacht und auf dem Rand angegeben.
**Bei größeren Auslassungen** wird auf die Manuskriptseite verwiesen. Die Stelle ist auf der Manuskriptseite zu kennzeichnen.

Diese Presse bestand aus befestigt war.

**Zu tilgende Buchstaben** oder **Wörter** werden durchgestrichen und auf dem Rand durch ℘ (für: deleatur, d. h. »es werde getilgt«) angezeichnet.

**Fehlende** oder **zu tilgende Satzzeichen** werden wie fehlende oder zu tilgende Buchstaben angezeichnet.

**Verstellte Buchstaben** werden durchgestrichen und auf dem Rand in der richtigen Reihenfolge angegeben.
**Verstellte Wörter** durch werden das Umstellungszeichen gekennzeichnet.

**Abbildung 5.1**
*Korrekturvorschriften (mit freundlicher Genehmigung des Duden-Verlags)*

## 5.2 Klassifikation

In diesem Abschnitt klassifizieren wir die verschiedenen Arten des Markup, zunächst nach dem Verwendungszweck und anschließend nach dem Ort und der Tiefe (Genauigkeit) der Dokumentenbeschreibung.

### 5.2.1 Prozedurales, strukturelles und semantisches Markup

*prozedurales Markup*

Prozedurale Markup-Sprachen ähneln sehr Programmiersprachen, die die visuelle *Darstellung* eines Dokuments beeinflussen können. Sie können aus Schaltern bestehen, die von der Stelle aus, an der sie gesetzt wurden, ein bestimmtes Verhalten bei einem Interpreter des Dokuments hervorrufen. Diese Schalter können z. B. **fett** oder *kursiv* heißen, um den Schriftstil zu beeinflussen.

*Beispiel PostScript*

Eine weitere Möglichkeit ist ein virtueller »Zeichenstift«, den wir absetzen, bewegen und hochnehmen können, wie es beispielsweise bei der Markup-Sprache PostScript der Fall ist. Diese Sprache kann dann von einem PostScript-fähigen Drucker oder Programm interpretiert werden. Hier noch einmal das Beispiel aus dem Einführungskapitel:

```
12.4 3.1415 moveto 202.4 47.11 lineto
```

Diese Anweisung veranlasst einen PostScript-Interpreter, eine Linie zwischen den Punkten (12.4, 3.1415) und (202.4, 47.11) zu zeichnen. Sollte ein Ausgabegerät jedoch nicht in der Lage sein, obige Zeichenkette zu interpretieren, wird es im günstigsten Fall eben nur die Zeichenkette `12.4 3.1415 moveto 202.4 47.11 lineto` ausgeben. Es »versteht« unsere Anweisungen nicht, und sein Verhalten ist demzufolge undefiniert. Prozedurales Markup ist ausschließlich im zu verarbeitenden Dokument eingebettet.

*strukturelles Markup*

Während es beim prozeduralen Markup um die reine Darstellung eines Dokuments geht, versucht man beim strukturellen Markup, die *Dokumentenstruktur* nachzubilden. So werden beispielsweise Überschriften, Absätze und Kapitel gekennzeichnet, wobei natürlich eine gewisse Schachtelung gemäß der Dokumentenstruktur zu erwarten ist. Die Verwendung von strukturellem Markup ist heute sehr gebräuchlich; Beispiele für Markup-Sprachen sind u. a.: LaTeX (Kopka 2002), HTML (W3C 1999a, Tolksdorf 2002) sowie XML (W3C 2000a, Turowski und Fellner 2001). Die Standards HTML und XML werden wir in Abschnitt 6.1 genauer besprechen.

*Beispiel HTML* Hier ein kurzer Auszug aus einem HTML-Dokument:

```
<h2>
Eine Tabelle
</h2>
<table>
 <tbody>
  <tr>
   <td>Zeile 1, Spalte 1</td>
   <td>Zeile 1, Spalte 2</td>
  </tr>
  <tr>
   <td>Zeile 2, Spalte 1</td>
   <td>Zeile 2, Spalte 2</td>
  </tr>
 </tbody>
</table>
```

In obigem Beispiel wird zunächst eine Überschrift und danach eine Tabelle mit zwei Zeilen und zwei Spalten ausgezeichnet. In den jeweiligen Tabellenzellen stehen die Koordinaten. Wir erkennen hierbei eine Dokumentenstruktur: Eine Tabelle besteht aus Zeilen, die ihrerseits wieder aus Spalten bestehen. In den Spalten kann dann einfacher Text stehen. Wir sehen auch, dass das Markup mit dem eigentlichen Dokumenteninhalt verschränkt ist.

*Regeln* Charakteristisch für die strukturierte Dokumentenbeschreibung sind *Regeln*, denen die Struktur eines Dokuments genügen muss. Eine Tabelle besteht aus einem optionalen Titel, einem optionalen Tabellenkopf und danach aus Zeilen, die ihrerseits Spalten enthalten. Diese Regeln können entweder intern durch die Markup-Sprache fest vorgegeben sein (LaTeX, HTML) oder den eigenen Bedürfnissen angepasst werden (SGML, XML).

*Inhalt vs. Darstellung* Die Annotation der Dokumentenstruktur birgt noch einen weiteren Vorteil. Das gleiche Dokument kann auf verschiedene Weise präsentiert werden, indem man die Strukturelemente unterschiedlich darstellt. Es wird also eine Trennung zwischen dem Inhalt eines Dokuments und seiner Darstellung möglich. So kann ein und dasselbe Dokument, welches mit einer strukturierten Markup-Sprache annotiert wurde, leicht in verschiedene Ausgabeformate überführt werden, beispielsweise eine Internetansicht, ein Druckformat (beispielsweise PostScript oder PDF) oder sogar eine gesprochene Sprache. PDF nimmt hierbei eine Sonderrolle ein, da es prozedurale Markup-Elemente à la PostScript und strukturelle Markup-Elemente à la HTML oder XML miteinander

vereint. Wir werden in Abschnitt 6.1.2 noch ein wenig genauer auf dieses weit verbreitete Format eingehen.

*semantisches Markup*

Vielfach wird jedoch der Begriff des strukturellen Markup mit dem *semantischen* Markup verwechselt, falls »sprechende« Namen zur Kennzeichnung der Dokumentenstruktur verwendet wurden – dazu später mehr. Der Leser sei jedoch an dieser Stelle schon einmal vorgewarnt. Beim semantischen Markup wird versucht, die Bedeutung von Dokumenten zu erfassen, sodass eine Maschine diese Dokumente interpretieren kann. Wir sprechen im Fachjargon von *machine-readability* und *machine-understandability*.

*Wozu zusätzliches Markup?*

Warum sollten wir unsere Dokumente eigentlich maschinenverständlich machen? Dafür sprechen mehrere Gründe: Suchmaschinen können wesentlich exaktere Ergebnisse abliefern und ihre Suche geschickt eingrenzen. Im Informationszeitalter, wo es nicht mehr darum geht, eine Informationsquelle zu finden, sondern die Fundstellen auf das *Wesentliche* zu begrenzen, ist dies unabdingbar. Aber auch für konkrete Ansätze der Langzeitarchivierung (z. B. Emulation und Migration) ist eine maschinenverständliche Darstellung von Dokumenten oder Instruktionen notwendig. Warum brauchen wir dazu eine neue *zusätzliche* Art Markup? Computer tun sich nach wie vor schwer, die Bedeutung von Zeichenketten, also von Dokumenteninhalten, zu erfassen. Auf dem Gebiet der künstlichen Intelligenz wurden zwar in den letzten 30 Jahren erhebliche Fortschritte erzielt, jedoch sind Maschinen noch sehr weit von dem entfernt, was wir Menschen Textverständnis nennen. Man stelle sich vor, einem Rechner Ironie beizubringen, ein von Menschen gern genutztes Stilmittel.

*allgemeine Anwendung*

Wie funktioniert semantisches Markup im Allgemeinen? Zunächst einmal setzen wir voraus, dass in einem Dokument die Teile, aus denen es besteht, gekennzeichnet sind. Dies kann durch eine geeignete Markup-Sprache geschehen, die es erlaubt, die Struktur eines Dokuments zu erfassen (siehe oben). Strukturierte Dokumentenbeschreibung stellt eine essenzielle Voraussetzung für jegliche semantische Dokumentenbeschreibung dar. Die Bedeutung eines Dokuments wird dabei auf die Bedeutung der einzelnen Dokumentteile zurückgeführt. Die Bedeutung der atomaren Dokumentbestandteile, wie des Titels, wird an möglichst zentraler Stelle definiert. Dies kann etwa über persistente Verweise (engl. *persistent uniform resource locator*) geschehen, wie wir sie in Abschnitt 5.4.2 noch genauer kennen lernen werden. Über Assoziationen zwischen atomaren Dokumentbestandteilen kann dann die Bedeutung des Dokuments nach und nach hergeleitet werden.

*Atome und ...*

Die Definition atomarer Dokumentbestandteile wie Autor oder Titel, obliegt Standards, beispielsweise Dublin Core (vgl. Abschnitt 8.1). Allerdings geschieht diese Beschreibung letztendlich als recht gut verständlicher englischer Text. Die Formulierung des letzten Satzes lässt allerdings aufhorchen – ging es nicht um eine maschinenverständliche Art der Semantikbeschreibung? Dem lässt sich entgegenhalten, dass wir ja irgendwo beginnen müssen, Dokumentbestandteile zu definieren. Eine mögliche Basis wären die Wahrheitswerte wahr und falsch. Allerdings ist eine derart formale Definition zum einen sehr aufwändig, und zum anderen ist ihr Nutzen vermutlich recht beschränkt. Weiterhin ist ein solch formaler Ansatz wohl nur sehr schwer verständlich und wenigen Menschen eingänglich, sodass wir davon ausgehen können, dass nur sehr wenige Dokumente derart formal beschrieben würden.

*... Assoziationen*

Sind die atomaren Dokumentbestandteile einmal definiert, können wir semantische Verknüpfungen zwischen ihnen aufbauen. Hierfür wurden bislang zwei Standards entwickelt – RDF (W3C 1999c) und Topic Maps (ISO 2000b). In den Abschnitt 6.2.1 und 6.2.2 werden wir genauer auf diese Standards eingehen.

*Vokabular für semantisches Markup*

Doch halt, keiner dieser beiden Ansätze definiert ein überall gültiges zentrales Vokabular, vielmehr obliegt es dem Anwender, das Vokabular für die Beschreibung seiner Dokumente selbst zu definieren. Es wird demnach eine zusätzliche Instanz benötigt, die Beziehungen zwischen den für semantisches Markup verwendeten Begriffen herstellt. Für diesen Zweck benötigen wir so genannte Ontologien. Abschnitt 6.2.3 gibt hierfür einen kurzen Überblick.

*»sprechende« Tags*

Wir sollten mit dem Begriff Semantik allerdings sehr vorsichtig sein. Oftmals wird das strukturierte Kennzeichnen von Dokumentteilen (beispielsweise durch XML) schon als semantisches Markup bezeichnet, da ja »sprechende« Tags wie `<title>` verwendet wurden. Wir Menschen assoziieren mit dem Begriff title sicher die Bedeutung des Titels eines Subjekts, sofern wir der englischen Sprache mächtig sind. Ein Computer kann mit obiger Zeichenkette nichts anfangen. Wir können uns selbst in seine Lage versetzen, indem wir die einzelnen Zeichen des `<title>` Tags einfach durch andere Zeichen ersetzen, beispielsweise so: `<scshl>`.[1] Aber selbst wir Menschen sind bei der Interpretation sprechender Tags nicht unfehlbar, da Tag-Namen im Allgemeinen nicht eindeutig sind. So kann das Tag `<table>` sowohl den Anfang einer Tabelle kennzeichnen, oder die Beschreibung eines Tisches einleiten.

[1] Im Tag `<title>` wurden folgende Buchstaben ersetzt: `t` durch `s`, `i` durch `c`, `l` durch `h` und `e` durch `l`.

Schlimmer wird es noch, wenn wir uns nicht über die verwendete natürliche Sprache (u. a. deutsch, englisch, französisch) bei Tag-Namen sicher sein können.

### 5.2.2 Embedded Markup Considered Harmful

*eingebettetes und externes Markup*

Die Beschreibung eines Dokuments kann entweder im Dokument selbst erfolgen – sie ist dann mit dem Dokumenteninhalt verschränkt – oder an externer Stelle abgelegt sein. Im ersten Fall sprechen wir von eingebettetem Markup, während wir im zweiten Fall von externem Markup sprechen. Ein Beispiel für eingebettetes Markup haben wir bereits in der Einführung mit der Textauszeichnung unseres Lehrers kennen gelernt. Dazu passend stellt das Abschlusszeugnis eine Art externes Markup (Bewertung) aller unserer Leistungen in einem Schuljahr dar. Abbildung 5.2 verdeutlicht noch einmal die Unterschiede.

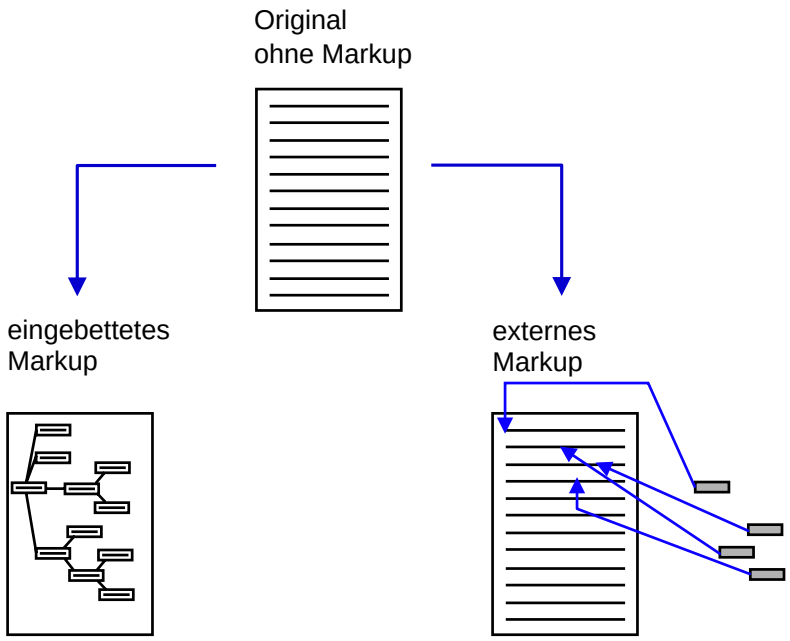



**Abbildung 5.2**
*Eingebettetes und externes Markup*

Ein weiteres Beispiel für externes Markup stellen die Formatinformationen in digitalen Bibliotheken dar. Eine Datenbank speichert zu jedem Dokument das Format, in dem es gespeichert wurde. Aus dieser Datenbank wird dann auf das beschriebene Dokument verwiesen.

*Eingebettet oder extern?*

Ist eine in das betreffende Dokument eingebettete Beschreibung einer externen Beschreibung vorzuziehen, oder umgekehrt? Wir werden hier keine endgültige Antwort geben können, da es sie pauschal nicht geben kann. Sowohl die interne als auch die externe Dokumentenbeschreibung haben Vor- und Nachteile, die wir je nach Verwendungszweck abwägen sollten.

*Anwendung von eingebettetem Markup*

In den vorhergehenden Abschnitten wurde erwähnt, dass für prozedurales Markup eine möglichst enge Bindung zwischen Dokument und Beschreibung vorteilhaft ist. Weiterhin dient es natürlich der maschinellen Lesbarkeit und Austauschbarkeit von Dokumentenformaten, wenn die Dokumentenstruktur schon aus dem Dokument selbst hervorgeht. Dies kommt vor allem zum Tragen, falls wir Dokumentenbeschreibungs-Sprachen verwenden, die sich selbst beschreiben können.[2] Wir werden also versuchen, Dokumentenformate zu verwenden, die schon eine Art strukturiertes Markup enthalten. Leider ist dies nicht immer möglich, beispielsweise dann, wenn die zu archivierenden Dokumente schon in einem anderen Format vorhanden sind.

*Anwendung von externem Markup und ...*

Der größte Vorteil der externen Dokumentenbeschreibung ist vermutlich, dass das zu beschreibende Dokument nicht verändert zu werden braucht, wenn wir es beschreiben, seine Beschreibung ändern oder eine neue Beschreibung hinzufügen. Oftmals ist dies auch der einzig gangbare Weg, da verschiedene Dokumentenformate keine internen Beschreibungen zulassen, die Dokumente auf einem nicht veränderbaren Datenträger (z. B. CD-ROM) gespeichert sind oder eine Änderung aus rechtlichen oder Authentizitäts-Gründen nicht möglich ist. Für semantisches Markup verwenden wir meist externe Beschreibungen, da sich die Dokumentenstruktur im Allgemeinen nicht mit den semantischen Verknüpfungen der Dokumentteile deckt. Bei der Verwendung von externem Markup werden die Metadaten selbst meist in eigenen Metadaten-Datenbanken gehalten.

*Voraussetzungen*

Allerdings setzt eine externe Dokumentenbeschreibung leistungsfähige Verweis- und Vernetzungsmöglichkeiten (Hyperlinks) voraus – mehr dazu in Abschnitt 5.4. Weiterhin stellt sich die Frage, wie wir vom Dokument zu seiner Beschreibung gelangen. Eine gekapselte Speicherung von Dokument und Beschreibung, wie bereits bei der Emulation vorgestellt, wäre eine Möglichkeit. Diese Vorgehensweise schränkt aber wiederum die Flexibilität ein. Weiterhin müssen wir uns um die Haltbarkeit der Meta-

[2] Was zunächst ein wenig nach Magie klingen mag, wird in Abschnitt 6.1.4 näher erläutert.

daten(-Datenbank) Gedanken machen. In Abschnitt 9.4 gehen wir näher auf dieses Thema ein.

Einen ausführlichen Artikel, der die Vor- und Nachteile von eingebettetem bzw. externem Markup gegenüberstellt, hat Nelson (1997) veröffentlicht.

### 5.2.3 Markup-Ebenen

Markup lässt sich auf verschiedenen Ebenen implementieren. Es ist sowohl möglich, nur einen Kopfteil für Dokumente zu definieren, als auch ein Dokument komplett oder nur bis zu einer bestimmten Tiefe mit semantischen Informationen zu strukturieren. Es sind natürlich auch Mischformen möglich. Abbildung 5.3 verdeutlicht noch einmal die verschiedenen Möglichkeiten.

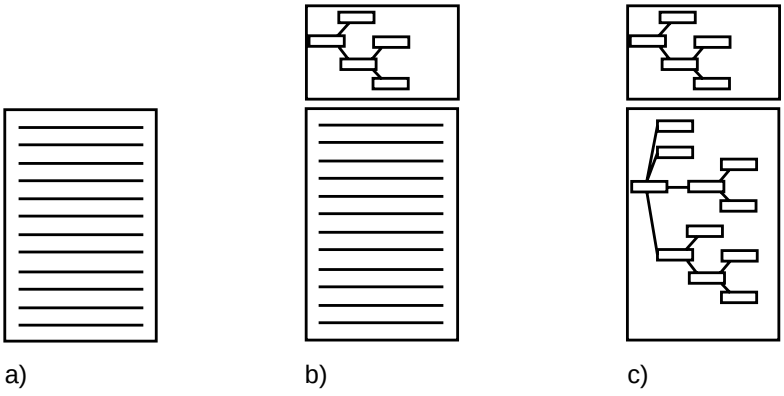



**Abbildung 5.3**
*Markup-Ebenen*

a) nur Text, aber kein Markup;
b) mit Kopfteil für Metadaten, aber kein Markup für Text;
c) mit Kopfteil für Metadaten und mit Markup für Text.

*Detaillierungsstufen*

Selbst innerhalb eines Dokuments oder Dokumentkopfes kann es sinnvoll sein, verschiedene Detaillierungsstufen einer strukturierten Dokumentenbeschreibung zuzulassen. Machen wir uns dies am Beispiel des Namens eines Buchautors klar: Eine einfache Möglichkeit, den Autor mit seinem Namen zu markieren, könnte so aussehen:

```
<autor>Uwe M. Borghoff</autor>
```

Ein wenig genauer wird es bei der Unterscheidung nach Vor-, Nachname und eventuellen Zusätzen.

```
<autor>
 <vorname>Uwe</vorname>
 <nachname>Borghoff</nachname>
 <zusatz>M.</zusatz>
</autor>
```

Eine ganz genaue Beschreibung könnte letztlich noch Aufschluss geben über die Initialen, Titel, die Adresse oder andere von Herrn Borghoff geschriebene Werke:

```
<autor>
 <vorname>Uwe</vorname>
 <nachname>Borghoff</nachname>
 <zusatz>M.</zusatz>
 <initialen>UB</initialen>
 <titel>Prof. Dr.</titel>
 <adresse>Universität der Bundeswehr, München
          </adresse>
 <werke>
  <buch>
    <titel>Langzeitarchivierung: ...</titel>
    ...
  </buch>
  ...
 </werke>
</autor>
```

Mit ein wenig Phantasie können wir uns leicht eine noch feinere Gliederung vorstellen, beispielsweise eine Aufteilung der Adresse. Obiges Beispiel könnte Teil einer Buchbeschreibung sein, die neben dem Autor noch detaillierte Angaben zum Buch selbst enthält, beispielsweise die ISBN oder Schlagwörter.

*feine vs. grobe Detaillierung*

Inwieweit eine sehr feine Untergliederung des Markup sinnvoll ist, hängt von der jeweiligen Anwendung ab. Sicher ist eine sehr feine Detaillierung arbeitsintensiver als ein nur sehr einfaches Markup – man stelle sich ein Buch wie das, was Sie gerade lesen, mit mehreren Autoren vor, die ihrerseits wieder eine Reihe von Büchern geschrieben haben. Andererseits bietet eine genaue Dokumentenbeschreibung auch mehr Möglichkeiten zur gezielten Bezugnahme, beispielsweise für die Suche nach Dokumenteninhalten.

## 5.3 Anwendungsmöglichkeiten für die Archivierung

*Suche und mehr*

Wir können die oben vorgestellten Markup-Techniken zunächst »klassisch« zur Suche anwenden. Markup-Techniken können jedoch auch helfen, *langlebige Dokumentenformate* zu entwickeln. Um für Zwecke der Langzeitarchivierung sinnvoll einsetzbar zu sein, sollte eine Markup-Sprache einige Anforderungen erfüllen, mit denen wir uns im Folgenden auseinandersetzen. Anschließend beleuchten wir Aspekte, die sich aus dem Bibliotheksumfeld ergeben.

### 5.3.1 Anforderungen an die Langzeitarchivierung

*konkurrierende Eigenschaften*

Schon vorweg sei gesagt, dass es sich bei den folgenden Eigenschaften oft um *konkurrierende* Eigenschaften handelt und mit Sicherheit nicht alle Eigenschaften von einer einzelnen Markup-Sprache erfüllt werden können. Vielmehr sind Auswahlen bzw. Gewichtungen der folgenden gewünschten Eigenschaften nötig, die sich natürlich je nach Einsatzfeld unterscheiden.

*Plattformunabhängigkeit*

Das Dokument selbst und die Darstellung seiner Bedeutung sollten auf möglichst vielen Plattformen lesbar sein. Es bietet sich demzufolge ein möglichst einfaches Format (z. B. nur Text) an. Allerdings treten hierbei wiederum Schwierigkeiten bzgl. der Codierung auf: Codierungen in Unicode-Varianten (UniCode 2002) oder ASCII (ISO 1991) sind durchaus üblich. Wir sollten also unbedingt das verwendete Format näher kennzeichnen. Dies kann z. B. in den Metadaten der digitalen Bibliothek geschehen.

*Lesbarkeit für Maschinen*

Für die maschinelle Verarbeitung unserer Dokumente, ist es notwendig, sie in einer Sprache zu beschreiben, die vom Computer verarbeitet werden kann. Hierbei spielen Standards für die Dokumentenbeschreibung eine große Rolle. Um beispielsweise eine semantische Suche in digitalen Dokumenten zu ermöglichen, sollte jedoch die eingesetzte Markierungsmethode für die Maschine nicht nur lesbar, sondern auch *interpretierbar* sein. Das bedeutet, es muss eine für die Maschine interpretierbare Abbildung des verwendeten Markup auf seine Bedeutung existieren. Diese Vision konnte jedoch bis heute noch nicht umgesetzt werden.

*Lesbarkeit für Menschen*

Die Lesbarkeit für Menschen erfordert ein Markieren in natürlicher Sprache. Dies ist sozusagen der letzte Ausweg, falls sämtliche Versuche fehlschlagen, die Dokumente maschinell zu interpretieren. Gerade die Metadaten der Datenkapsel bei Emulation

digitaler Dokumente sind auf *human readability* angewiesen. Insbesondere tritt hierbei das Problem auf, dass sich von Menschen gesprochene Sprachen ebenfalls im Laufe der Zeit ändern und so einzelne Worte an Bedeutung verlieren oder eine neue (u. U. völlig andere) Bedeutung bekommen. Es ist demnach notwendig, eine Datierung eines archivierten Dokuments *und* der für das Markup verwendeten natürlichen Sprache vorzunehmen.

*Eindeutigkeit und Vollständigkeit*

Es ist notwendig, eindeutige Markierungen zu verwenden, um Fehlinterpretation durch Interpretationsspielraum zu vermeiden. Insbesondere gilt es, das Stille-Post-Problem bei fortschreitender Migrierung zu reduzieren. Es soll möglichst der vollständige Inhalt eines Dokuments durch seine Beschreibung erfasst werden, um Fehlinterpretation durch fehlende Informationen zu vermeiden. So kann es beispielsweise bei Gedichten wichtig sein, ob ein Querstrich am Zeilenende nun ein Gedankenstrich, Bindestrich oder einfach nur ein einfacher Trennstrich ist. Weiterhin können Darstellungsaspekte wie die Anzahl der Wörter pro Zeile bedeutungsrelevant sein.

*leichte Emulierbarkeit und Migrierbarkeit*

Um weiterhin lesbar zu bleiben, werden Dokumente im Laufe ihrer Archivierung in andere Formate umgewandelt (Migration) oder es werden die Erstellungs-/Leseplattformen der Dokumente emuliert. Bezüglich der Dokumentenbeschreibung ergeben sich hier insbesondere die folgenden Aspekte. Die Beschreibung eines Dokuments muss ebenfalls migriert bzw. emuliert werden. Da externe Metadaten oft in eigenen Datenbanken gehalten werden, eröffnet sich hierbei das gesamte Feld der *Datenbankmigration*. Hierbei werden die in Kapitel 3 und Kapitel 4 besprochenen Migrations- und Emulationstechniken auf Datenbanken angewendet. Für ein eventuell semantisches Markup muss natürlich auch die Darstellung der Abbildung von verwendetem Markup auf seine Bedeutung migriert bzw. emuliert werden. Bei der Verwendung natürlicher Sprache zur Dokumentenbeschreibung müssen eventuell Anpassungen aufgrund von Veränderungen der natürlichen Sprache vorgenommen werden (z. B. bei einer Rechtschreibreform).

*leichte Edierbarkeit vs. Schreibschutz*

Auch nach der Erstellung von (semantischem) Markup für ein Dokument müssen einige Informationen edierbar bleiben, um gegebenenfalls Anpassungen vorzunehmen. Andere Informationen (beispielsweise zur Authentifizierung) sollten selbstverständlich nicht mehr änderbar sein. Eventuell gibt es auch rechtliche Aspekte, die Änderungen an einem vorliegenden elektronischen Dokument generell verbieten.

*Einfachheit*

Der eigentliche Vorgang des Markierens sollte einfach gehalten sein, um den notwendigen Mehraufwand so gering wie möglich

zu halten. Man halte sich in diesem Zusammenhang vor Augen, dass Bibliotheken eine Vielzahl von Dokumenten besitzen und das Markieren oftmals noch von Hand geschehen muss. Weiterhin verursacht der Prozess der Katalogisierung Kosten, die dem Nutzen der dabei entstehenden Dokumentenbeschreibung gegenübergestellt werden müssen. Eine möglichst einfache Art, Dokumente zu beschreiben, könnte sich zudem positiv auf die Bereitschaft von Autoren auswirken, ihre erstellten Dokumente schon selbst mit einem entsprechenden Markup zu versehen. Damit reduziert sich für die Bibliothek der Aufwand der Katalogisierung auf die Prüfung des zur Verfügung gestellten Materials.

*Modularität und Erweiterbarkeit*

Nicht immer werden sämtliche Möglichkeiten der Dokumentenbeschreibung benötigt. Man stelle sich medizinische Dokumente zur Krebsvorsorge vor, die ganz andere Formen der Auszeichnung benötigen als beispielsweise Lyrik aus dem 19. Jahrhundert. Ein vollkommen universeller Ansatz würde also zu einem Datenaufkommen führen, das nicht gerechtfertigt ist. Weiterhin sind die Anforderungen an und der Umfang der nötigen Daten zur Dokument- und Informationsbeschreibung für die Zukunft nicht abschätzbar. Demzufolge muss das verwendete System der Dokumentenbeschreibung leicht erweiterbar sein.

*Verhältnis Markup zu Dokumenteninhalt*

Wir sollten versuchen, ein »vernünftiges« Verhältnis (das sich je nach Anwendung unterscheiden wird) zwischen dem eigentlichen Dokumenteninhalt und seiner Beschreibung zu finden. Es handelt sich hierbei um ein reines Mengenkriterium. In Extremfällen kann es durchaus vorkommen, dass die Dokumentenbeschreibung wesentlich mehr Platz einnimmt als die eigentlichen Dokumenteninhalte – insbesondere dann, wenn eine Markierung in natürlicher Sprache und in einem Textformat gewählt wird. Ein Beispiel dafür bieten die Dokumentenbeschreibungs-Sprachen HTML und XML. Mehr dazu in Abschnitt 6.1.

*Kompromiss*

Es ist klar, dass die oben definierten Anforderungen zum Teil miteinander in Konkurrenz stehen. Allein die Punkte *Vollständigkeit*, *Einfachheit* und *Modularität* zwingen zu einem Kompromiss. Aus der human-readability-Forderung ergeben sich neue Problemfelder: Ist die natürliche Sprache überhaupt eindeutig genug, um komplexe Zusammenhänge zu erläutern? Entstehen durch den Wandel der natürlichen Sprache zusätzliche Risiken für die Langzeitarchivierung?

#### Langlebige Dokumentenformate

*Restauration digitaler Dokumente*

Ein Problem der Langzeitarchivierung liegt darin, dass wir es beim »Auspacken« im Allgemeinen mit Dokumenten zu tun haben, deren Format wir nicht kennen. Stellen wir uns folgendes Zukunftsszenario vor: Wir haben das Problem der physischen Haltbarkeit digitaler Daten gelöst, allerdings stellt sich ein Dokument von damals nur noch als Strom von Bits dar. Die verwendeten Dokumentenformate existieren nicht mehr, längst sind wir dazu übergegangen, uns historische Dokumente vorlesen oder vorspielen zu lassen.

*haltbare Metadaten*

Ein möglicher Ansatz wäre nun, Metadaten zu einem Dokument zu speichern, aus denen ersichtlich wird, in welchem Format ein bestimmtes Dokument gegeben ist und wie wir dieses Format in eine *zum gegenwärtigen Zeitpunkt* maschinen- oder menschenlesbare Form bringen können. Offenbar helfen uns Metadaten aber nur weiter, wenn wir sie entschlüsseln können. Der Ansatz, Metadaten über Metadaten anzulegen hilft hier auch nicht weiter. Was wir benötigen sind Metadaten in einem Format, das sich gewissermaßen selbst beschreibt. Der XML-Schema-Standard bietet einen Ansatz, das oben beschriebene Dilemma zu lösen. Grob gesagt werden Regeln, denen ein XML-Dokument genügen muss, wieder in XML codiert. Diesen Trick werden wir in Abschnitt 6.1.4 näher erläutern.

### 5.3.2 Bibliographische Anforderungen

*Verschlagwortung*

Die Verschlagwortung von Dokumenten wird Bibliothekaren sehr bekannt vorkommen. Früher gab es Karteikärtchen, auf denen Angaben über Bücher enthalten waren, wie Autor, Titel, Sachgebiet und Stichworte. Diese Karteikärtchen waren meist nach einem bestimmten Kriterium (z. B. Autor) geordnet, was die Suche nach bestimmten Büchern erleichterte.

*elektronische Karteikarten*

Das Karteikärtchenprinzip lässt sich problemlos auf die elektronische »Welt« übertragen. Zu einem elektronischen (oder physischen) Dokument speichern wir einen Satz von Metadaten, die das Dokument näher beschreiben. Hierbei kann es sich wie schon erwähnt um den Autor, den Titel, eine Kurzbeschreibung oder auch Fundstellen (z. B. per Hyperlink) handeln. Elektronische Karteikarten haben gegenüber physischen Karteikarten den Vorteil, dass sie wesentlich schneller durchsucht werden können und es natürlich mehr Such- und Sortierkriterien als nur den Autor geben kann.

*detaillierte Metadaten vs. Suche*

Wir können eine flexible Suche durch eine hinreichend genaue Darstellung der Metadaten unterstützen. Nehmen wir nochmals das Autorenbeispiel aus Abschnitt 5.2.3. Wenn wir von der sehr einfachen Beschreibung

```
<autor>Uwe M. Borghoff</autor>
```

ausgehen, ist nicht klar, wie Vor- bzw. Nachnahme des Autors lauten. Verschiedene Möglichkeiten der Namensschreibweise verdeutlichen diese Tatsache:

```
<autor>Uwe M. Borghoff</autor>
<autor>Borghoff, Uwe M.</autor>
<autor>M. Borghoff, Uwe</autor>
<autor>Uwe Borghoff, M.</autor>
<autor>Borghoff, Uwe</autor>
<autor>Uwe Borghoff</autor>
```

Eine gezielte Suche nach Vor- bzw. Nachname scheint hier unmöglich. Eine genauere Beschreibung des Autoren schafft Abhilfe. Die Reihenfolge der Namensbestandteile ist nun ohne Belang, da sie ja eine eindeutige Bezeichnung haben:

```
<autor>
 <vorname>Uwe</vorname>
 <nachname>Borghoff</nachname>
 <zusatz>M.</zusatz>
</autor>
```

*Auswahl zu archivierender Dokumente*

Eine der Hauptaufgaben digitaler Bibliotheken ist, wie bei klassischen Bibliotheken auch, die Erfassung bzw. Auswahl zu archivierender Dokumente. Sicher können nicht alle entstehenden Dokumente für die Nachwelt aufgehoben werden. Es gilt also, zwischen wesentlichen und unwesentlichen Dokumenten zu unterscheiden. Angesichts des enormen Datenaufkommens, das aufgrund der Nutzung elektronischer Medien eher zu- als abnimmt, entstehen dabei folgende Hauptschwierigkeiten:

1. die Auswahl zu archivierender Dokumente,
2. die Katalogisierung und Klassifizierung zu archivierender Dokumente und
3. die fortlaufende Evaluierung der archivierten Dokumente hinsichtlich ihres Verbleibs im Archivs.

*Metadaten für Auswahl, Katalogisierung, Klassifizierung*

Sollten unsere Autoren ein strukturiertes Dokumentenformat benutzen, welches außerdem noch Metadaten zum Dokument selbst enthält, so gibt dies wertvolle Hinweise bzgl. obiger Punkte. Nützliche Metadaten im Dokument selbst sind beispielsweise Autor, Sachgebiet oder Titelstichworte. Da die Metadaten ja schon im Dokument enthalten sind, müssen sie nur noch aus dem Dokument herausgefiltert werden. Dieser Vorgang kann zumindest teilautomatisch geschehen und verschlingt keine wertvolle Zeit mehr. Sollte sich ein Dokument einmal ändern, ändern sich die Metadaten dabei automatisch mit. Natürlich passiert dies nicht von selbst, weiterhin sind qualitätssichernde Stichproben erforderlich.

## 5.4 Persistence is a Virtue

Die konsequente Beschreibung von Dokumenten mittels Metadaten birgt einige Risiken und fördert diverse technische Schwierigkeiten zu Tage. Im Folgenden werden wir einige technische Schwierigkeiten und Herausforderungen ansprechen und Lösungsmöglichkeiten diskutieren.

*Konsistenz*

Metadaten beschreiben generell Dinge, die schon einmal vorhanden sind. Sie stellen also eine Art doppelte Datenhaltung dar. Aus diesem Grunde müssen wir uns um die *Konsistenz* zwischen Metadaten und dem durch sie beschriebenen Material (Dokumente) Gedanken machen.

*Änderungen*

Das Hauptproblem stellen Materialien dar, die verändert werden können. Solche Veränderungen könnten beispielsweise der Ort der Speicherung, die Art der Speicherung (Format) oder Änderungen im Material selbst sein. Metadaten, die einmal mühsam angelegt worden sind, können so sehr leicht ungültig werden. Sollte sich unser archiviertes Material ändern, könnte eine Änderung der Metadaten ebenfalls nötig sein.

*Verweise*

In Abschnitt 5.3.2 war die Rede von Metadaten, die schon in uns zur Verfügung gestellten Dokumenten vorhanden sind. Auf diese Metadaten müssen wir dann nur noch verweisen. Ergeben sich dann etwa Änderungen hinsichtlich des Autors, es könnte z. B. ein Co-Autor hinzukommen, so sind unsere Metadaten weiterhin aktuell, da sie ja keine physische Kopie darstellen, sondern nur einen Verweis. Allerdings werden auch unsere Verweise ungültig, sollte sich beispielsweise der Speicherort des referenzierten Dokuments ändern.

*Persistenz*

Ein *persistenter* Zugriff auf digitale Dokumente hat zwei Aspekte: zum einen die dauerhafte Existenz des Dokuments, zum

anderen die dauerhafte Identifizierbarkeit und Wiederauffindbarkeit. Dem ersten Aspekt haben wir uns schon ausgiebig in den Diskussionen zu Emulation und Migration gewidmet. Dieser Abschnitt beschäftigt sich ausführlicher mit dem Aspekt der Identifizierbarkeit und der Wiederauffindbarkeit und folgt dabei in weiten Teilen den Ausführungen von Dack (2001).

### 5.4.1 Identifikation durch Uniform Resource Identifier (URI), -Name (URN) und -Locator (URL)

*Uniform Resource Identifier (URI)*

Der Begriff *Uniform Resource Identifier* (URI) umfasst sowohl die Namensaspekte eines digitalen Dokuments (Wie kann das Dokument in einem u. U. weltweit verteilten System identifiziert werden?) als auch dessen (Speicher-) Ortaspekte (Wo im verteilten System kann das Dokument wieder aufgefunden werden?). Der URI ist also ein Oberbegriff für die beiden viel häufiger verwendeten und seit Anbeginn des World Wide Web reichlich umworbenen Begriffe URN und URL.

*Uniform Resource Name (URN)*

Der so genannte *Uniform Resource Name* (URN) bezieht sich genau auf denjenigen Teil eines URI, der sich um die Namensgebung kümmert. Hierbei müssen zwei Problemfelder gelöst werden. Zum einen benötigt man Absprachen bezüglich der zu vergebenden Namen, zu anderen muss ein robustes Resolutionsverfahren entworfen werden. Die Resolution ist derjenige Prozess, der einen Namen interpretiert und das zugehörige Dokument lokalisiert. Um beiden Problemfeldern gleichermaßen gerecht zu werden, muss ein URN eine Reihe wichtiger Anforderungen erfüllen, um als Identifikationsmechanismus in einem verteilten System effektiv eingesetzt werden zu können.

*Anforderungen*

- Ein URN ist *global eindeutig*. Der gleiche URN wird niemals zwei verschiedenen Dokumenten zugeordnet.
- Man fordert die Unabhängigkeit der Namen gebenden Organisationen (*naming authority*). Diese Organisationen sollen sich hierarchisch organisieren können. Es ist explizit beabsichtigt, dass ein URN eine dauerhafte Gültigkeit hat, selbst dann, wenn das referenzierte Dokument schon lange aufgehört hat zu existieren. In einem anderen Extrem mag ein URN noch Gültigkeit haben, obwohl es keine *naming authority* mehr gibt, die mit diesem Namen noch etwas Sinnvolles anfangen kann.
- URNs müssen »skalieren«. Es muss also auch in vielen hundert Jahren noch möglich sein, einem neu hinzukommen-

den Dokument einen URN zuzuordnen. Jeder URN-Mechanismus muss zukünftige Erweiterungen erdulden können.

- Die Namensvergabe soll weitgehend unabhängig vom verwendeten Resolutionsverfahren sein. Nachträgliche Änderungen am Prozess der Resolution dürfen die bereits vergebenen Namen hinsichtlich globaler Identifizierbarkeit und Eindeutigkeit nicht gefährden.
- Gibt es einen korrespondierenden URL zu einem URN (und umgekehrt), so wird weiterhin gefordert, dass umkehrbar eindeutige Übersetzungsschritte geschaffen werden.

*erfolgloser URN*

URNs sind bislang nicht sehr weit verbreitet. Der mangelnde Erfolg hängt ganz wesentlich mit dem fehlenden Standard eines passenden Resolutionsverfahrens zusammen. Daran konnte bisher auch die XML-Community nichts ändern, die ja mit ihren *name spaces* eine eng verwandte Problematik attackiert. Ganz anders ist es hier bei den URLs.

*erfolgreicher Uniform Resource Locator (URL)*

Der so genannte *Uniform Resource Locator* (URL) identifiziert ein Dokument über dessen Zugriffscharakteristik, in der die Information über den Speicherort mit codiert wird. Im Vergleich zu den URNs sind URLs unglaublich erfolgreich und populär. Der Grund dafür ist sehr einfach. Zum einen ist der Bekanntheitsgrad der URLs dank der breiten Verwendung im World Wide Web sehr hoch (alle gängigen Browser »verstehen« URLs), zum anderen ist das Resolutionsverfahren standardisiert.

*Domain Name System (DNS)*

Das so genannte *Domain Name System* (DNS) bildet den Domainnamen eines URL auf die IP-Adresse eines Servers ab. Über diesen Server kann die Anfrage nach dem zugehörigen Dokument dann demjenigen Dienst zugeordnet werden, der letztlich den Zugriff auf das Dokument ermöglicht.

Wie sieht es aber aus, wenn wir auf Dokumentteile oder gar andere Metadatensätze verweisen müssen? Speziell die Variante der externen Dokumentenbeschreibung macht reichlich Gebrauch von solchen Verweisen. Im nächsten Abschnitt werden wir Verweise auf ganze Dokumente und Stellen innerhalb von Dokumenten näher betrachten.

### 5.4.2 Verweise auf Dokumente

*Verweise im Internet*

Verweise auf Dokumente sind uns allen aus dem Internet bekannt. Abbildung 5.4 stellt im Wesentlichen das derzeitige Vorgehen dar. Die Verweise sind *unidirektional* und es gibt kein globales Ver-

weisverzeichnis. Es ist also für ein Dokument nicht erkennbar, wie oft und von wem auf es verwiesen wird.

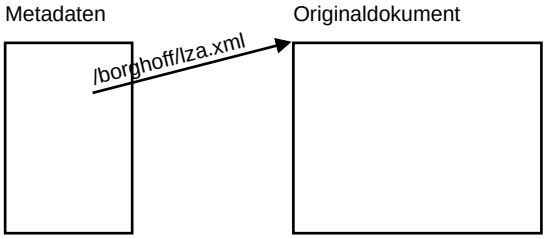



**Abbildung 5.4** *Verweis auf ein Dokument*

*Broken Links*

Dies führt immer dann zu Problemen, wenn sich Veränderungen hinsichtlich des Ortes des referenzierten Dokuments ergeben. Sollte sich beispielsweise der Speicherort unseres referenzierten Dokuments ändern, so sprechen wir von »kaputten« Verweisen (engl. *broken links*). Abbildung 5.5 zeigt dies ganz plastisch.

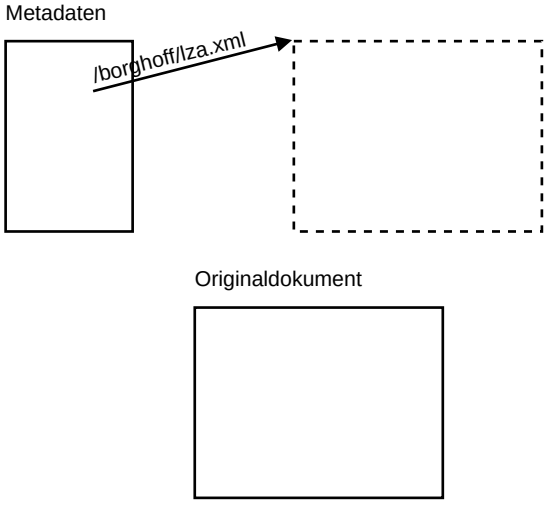



**Abbildung 5.5** *Broken-Links-Problem*

*Lösung des Broken-Links-Problems*

Eine mögliche Lösung des Broken-Links-Problems liegt zum einen in einer Zweiteilung der Verweise in einen Verweis auf einen eindeutigen Namen und einen Verweis auf den Speicherort. Weiterhin benötigen wir eine gewisse Bidirektionalität des Verweises auf den Speicherort. Dieses Verfahren ist uns auch aus dem Internet bekannt, z. B. aus der IP-Adressierung: Es wird lediglich auf eine Rechnernummer (seine IP-Adresse) verwiesen. Der eigentliche Standort des Rechners ist dabei ohne Belang.

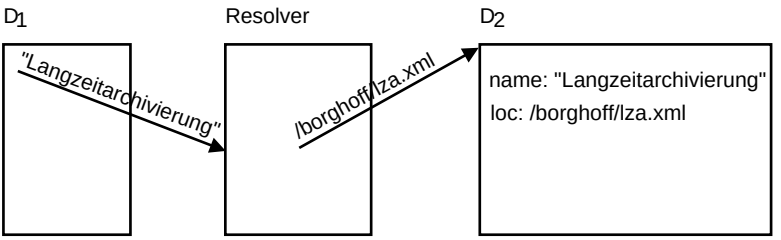



**Abbildung 5.6** *Zweiteilung von Verweisen*

*zweigeteilte Verweise*

Konkret könnte die eben abstrakt beschriebene Zweiteilung der Verweise wie folgt aussehen (vgl. Abb. 5.6). Wir sehen uns mit der Aufgabe konfrontiert, von einem Dokument $D_1$ auf ein Dokument $D_2$ zu verweisen. Wir verweisen nicht mehr auf den Speicherort $loc(D_2)$, sondern nur noch auf den Namen $name(D_2)$, der an möglichst zentraler Stelle gespeichert ist. Diese zentrale Instanz, nennen wir sie *Resolver*, übernimmt die Übersetzung des Namens $name(D_2)$ in seine eigentliche Speicheradresse $loc(D_2)$. Sollte sich die Speicheradresse $loc(D_2)$ einmal ändern, muss der Resolver darüber informiert werden. Da der Resolver aber eine zentrale Instanz ist, die als einzige auf das Dokument $D_2$ verweist, ist dies ohne weiteres für den Bearbeiter von $D_2$ möglich.

*Persistent URL (PURL)*

Ein derartiges Resolutionsverfahren wird übrigens auch schon im Internet angewandt, nämlich bei den *Persistent URLs* oder kurz PURLs. PURLs wurden vor knapp zehn Jahren von OCLC (*Online Computer Library Center, Inc.*) als Antwort auf die Frage entwickelt, wie man die Mühsal der regelmäßigen URL-Anpassungen in Zeiten der Katalogisierung von Internetdokumenten verringern könnte. OCLC war aktiver Teilnehmer in den IETF-Arbeitsgruppen, die sich ausgiebig mit der URI-Problematik beschäftigten. OCLC sah wohl zum damaligen Zeitpunkt keine Konsensfähigkeit, was eine Standardlösung für ein dauerhaftes Identifikationsverfahren anbelangte. PURLs wurden als Zwischenlösung auf dem Weg zu einem Standardverfahren entworfen.

*Namensauflösung*

Rein funktional ist ein PURL nichts anderes als ein URL. Aber statt direkt den Ort des Dokuments zu codieren (dies führte ja zur Mühsal der regelmäßigen URL-Anpassungen bei wechselnder Lokalität der Dokumente, die in einem Katalog nachgewiesen werden sollten), codiert ein PURL den Ort eines zwischengeschalteten Resolutionsdienstes. Dieser Dienst läuft auf einem PURL-Server. Der Resolutionsdienst des PURL-Servers löst nun folgende Aufgabe: Die Anfrage nach einem PURL führt zu einem Abbildungsschritt, bei dem die nachgefragte PURL in die aktuelle URL

des angefragten Dokuments umgesetzt wird. Dieser Abbildungsschritt findet heute standardmäßig als http-redirect statt, was die Nutzerakzeptanz extrem erhöht, da hierbei weder Protokollerweiterungen noch Modifikationen an der Client-Software notwendig werden. PURLs können folgerichtig mit Hilfe von Standard-Webbrowsern erzeugt, zugeordnet und verarbeitet werden. Hierzu nutzen die Browser die entsprechenden Resolutionsdienste auf dem PURL-Server.

*Anwendung*

OCLC hat den Quelltext der Software des Resolutionsdienstes offen gelegt. Seither haben weitere Institutionen eigene PURL-Server erstellt und mit Erfolg betrieben. Eine erfolgreiche Installation befindet sich in der *National Library of Australia* (NLA). Im Rahmen der Diskussion des Pandora-Projekts kommen wir darauf noch zu sprechen (vgl. Abschnitt 8.5). Weitere Informationen zu PURLs sind unter http://purl.oclc.org/ zu finden.

*Verweise auf Dokumentstellen*

Die bisherigen Probleme verschärfen sich, wenn Markup-Techniken nicht nur auf Dokumente als Ganzes, sondern auch auf Stellen innerhalb von Dokumenten verweisen müssen. Generell gibt es zwei Möglichkeiten, auf Dokumentstellen zu verweisen. Zum einen können wir uns auf Byte-Positionen innerhalb von Dokumenten beziehen, was im Wesentlichen die gerade beschriebenen Probleme wieder aufwirft. Eine flexiblere und robustere Möglichkeit der Referenzierung stellen Bezüge auf gekennzeichnete Dokumentteile dar. Eine wesentliche Voraussetzung dafür ist allerdings eine strukturelle Dokumentenbeschreibung (vgl. Abschnitt 5.2.1).

*Marken*

Die Kennzeichnung von Dokumentstellen geschieht hierbei über so genannte Marken, die fest mit einem Dokumentteil verknüpft sind. Hier noch einmal unser Tabellenbeispiel, wobei wir die Tabelle mit der Marke `Tab1` versehen haben.

```
<h2>
Eine Tabelle
</h2>
<table id="Tab1">
 <tbody>
  ...
 </tbody>
</table>
```

Eine Referenz auf diese Tabelle könnte nun wie folgt aussehen, wobei die Byte-Position des physikalischen Vorkommens der Tabelle im Dokument ohne Belang ist:

```
<p>
Wie in Tabelle <a href="#Tab1">1</a> dargestellt
</p>
```

Sollten wir das Dokument bearbeiten, also Text hinzufügen oder löschen, bleibt obiger Verweis weiterhin gültig, solange eine eindeutige Marke `Tab1` existiert. Letztere Einschränkung sollte uns allerdings nachdenklich stimmen: Niemand hindert uns daran, eine zweite Marke `Tab1` im gleichen Dokument zu schaffen, oder einfach einen anderen Dokumentteil mit `Tab1` zu identifizieren. Das macht den obigen Verweis sofort unbrauchbar.

*Probleme komplexer Verweise ...*

Weitere Probleme treten bei komplizierteren Verweisen auf, die sich auf »echte« Positionen beziehen, beispielsweise Verweise der Art: Der dritte Absatz vom Abschnitt Verweise. Während »Abschnitt Verweise« noch auf eine Marke verweist, bezeichnet »dritter Absatz« eine absolute Position innerhalb der Dokumentabschnitte. Ein absoluter Positionsverweis geht natürlich kaputt, falls wir vorher Abschnitte einfügen oder entfernen. Generell können wir gegen solche kaputten Links technisch wenig tun. Es gibt jedoch Möglichkeiten, kaputte Verweise innerhalb von Dokumenten zu »reparieren«.

*... und Lösungsansätze*

Man könnte beispielsweise Positionen nicht nur auf einen Anfangspunkt, sondern auch auf einen Endpunkt beziehen. Unser Beispiel von oben könnte dann etwa so aussehen: Der dritte Absatz von vorn (= zweiter Absatz von hinten) von Abschnitt Verweise. Sollten wir jetzt nur Teile vor bzw. hinter dem referenzierten Absatz edieren, so kann obiger Verweis repariert werden. Eine weitere Möglichkeit besteht darin, jeweils einen kleinen Ausschnitt des referenzierten Dokumentteils mit an den Verweis zu heften. Dies macht insbesondere im Zusammenspiel mit semantischem Markup Sinn.

### 5.4.3 Identifikation durch Handle-System und Digital Object Identifier (DOI)

Das so genannte *Handle-System* löst das Problem der Namensgebung digitaler Dokumente in verteilten Bibliotheksanwendungen. Es ist eine Entwicklung der *Corporation for National Research Initiatives* (CNRI) im Rahmen des NCSTRL-Projekts.[3]

*Namensauflösung*

CNRI betreibt die oberste der Namen gebenden Organisationen. Dies sichert die Eindeutigkeit der Namensstrukturen auf

[3] NCSTRL steht für *Networked Computer Science Technical Reports Library*; gefördert von der amerikanischen DARPA.

oberster Ebene zu. Hier läuft auch der oberste Resolutionsdienst. Lokale Handle-Systeme (das sind die Namen gebenden Organisationen auf tieferen Hierarchieebenen) können sich die offen gelegte Handle-Software zunutze machen. Solange es noch keine allgemein akzeptierten Standardprotokolle für das Handle-System und die zugehörigen Resolutionsdienste gibt, muss man auf http-Proxyserver zurückgreifen. Nur so können Standard-Webbrowser Verwendung finden, was die notwendige Akzeptanz erst schafft. Einer der Nutzer des Handle-Systems ist die Library of Congress.

*Digital Object Identifier (DOI)*

Das Handle-System wird ebenfalls im so genannten DOI-System zur Namensresolution benutzt. Die *Digital Object Identifier* (DOI) sind eine Entwicklung der AAP (engl. *American Association of Publishers*). Auf der Frankfurter Buchmesse 1997 wurden die DOI offiziell vorgestellt als Antwort auf die drängende Frage nach Rechteschutz in offenen verteilten Netzen, speziell zum Schutz von elektronischen Zeitschriftenartikeln.[4] Der Schutz der Rechte sollte dadurch erreicht werden, dass man einem Zeitschriftenartikel bereits im Entwurfsstadium einen festen, unveränderlichen und standardisierten Namen zuordnet. Dieser Name sollte dabei folgende Aufgaben erfüllen helfen:

*Aufgaben eines DOI*

- Kontrolle der Transaktionen beim Arbeiten mit dem Zeitschriftenartikel
- Kommunikationserleichterung zwischen Autoren, Editoren und Verlagen
- Ansatzpunkt für das Rechtemanagement

*Voraussetzungen*

Um dies zu erreichen, benötigt man einerseits Absprachen bezüglich der Syntax dieser festen, unveränderlichen und standardisierten Namen, andererseits ein stabiles, anerkanntes und industrieweit akzeptiertes Resolutionssystem (es wurde wie gesagt auf das Handle-System zurückgegriffen) sowie eine abgesprochene Metadatenstruktur. Für letztere wurde auf den so genannten INDECS-Metadatensatz zurückgegriffen.[5] Um die Absprachen bezüglich der Namensgebung kümmert sich die *International Digital Object Identifier Foundation* (kurz IDF).

[4] Registriert unter ANSI/ISO Z39.84-2000.

[5] INDECS steht für *Interoperability of Data in E-Commerce Systems*.

### 5.4.4 Überblick

Die besprochenen Varianten zur dauerhaften Identifizierbarkeit und Wiederauffindbarkeit von Dokumenten folgen einem gemeinsamen Muster. Es fällt auf, dass immer eine Zweiteilung des Namens vorliegt. Der erste Teil beschreibt häufig den Namen einer Namen gebenden Organisation (*naming authority*). Dieser Name ist dann global eindeutig. Der zweite Teil beschreibt immer einen Namen, der eindeutig innerhalb der Namen gebenden Organisation ist.

Dieses Muster erfüllt die notwendigen Eigenschaften einer URN-Spezifikation, wie sie im RFC 1737 festgelegt wurde. Tabelle 5.1 fasst die besprochenen Varianten kurz zusammen.

| Variante | Beispiele |
|---|---|
| URI | `<Protokoll>://<Authority>:<Authority-spezifischer Name>` |
| URN | `urn:<Namespace-Id>:<Namespace-spezifischer Name>` |
| URL | `<Protokoll>://<DNS-Domainname>/<Rechner-spezifischer Pfadname>` |
| PURL | `<Protokoll>://<PURL-Server-Name>/<Rechner-spezifischer Pfadname>` |
| Handle | `<Authority>:<Authority-spezifischer Name>` |
| DOI | `<Directory-Code.Registrant-Code>:<DOI-Suffix Name>` |

**Tabelle 5.1** *Beispiele prominenter Varianten zur dauerhaften Identifizierbarkeit und Wiederauffindbarkeit von Dokumenten*

# 6 Standardisierte Dokumentenbeschreibungs-Sprachen

*In Kapitel 5 haben wir uns mit dem generellen Problem und der Notwendigkeit der Auszeichnung von Dokumenten auseinandergesetzt. Doch wie geschieht dies in der Praxis? Diese Frage beantworten wir in diesem Kapitel. Wir stellen verschiedene Dokumentenbeschreibungs-Sprachen vor, die häufig verwendet werden und inzwischen zu Standards geworden sind. Dabei beleuchten wir auch immer die jeweiligen Vor- und Nachteile in Hinblick auf den Einsatz bei der Langzeitarchivierung in digitalen Bibliotheken.*

*Dieses Kapitel ist wie folgt aufgebaut: In Abschnitt 6.1 stellen wir jeweils zwei Vertreter prozeduraler und struktureller Dokumentenbeschreibungs-Sprachen vor. Konkret geht es um die Formate TIFF, PDF, HTML und XML. Standardformate zur semantischen Beschreibung von Metadaten und Dokumenten diskutieren wir in Abschnitt 6.2. Wir konzentrieren uns dabei auf die Formate RDF und Topic Maps sowie DAML+OIL für die Beschreibung von Ontologien. Abschließend beleuchten wir noch die Vision des Semantic Web in Abschnitt 6.3.*

## 6.1 Standards für eingebettete syntaktische Dokumentenbeschreibung – Markup-Sprachen

In diesem Abschnitt werden wir verschiedene Markup-Sprachen vorstellen, die es erlauben, die Dokumentenstruktur *innerhalb* eines Dokuments nachzubilden. Es handelt sich hierbei wohlgemerkt nicht nur um Dokumentenbeschreibungen, sondern gleichzeitig auch um Dokumentenformate.

*binäre Formate: PDF, TIFF*

Zunächst werden wir die Standardformate TIFF und PDF vorstellen. Beide Formate haben es beim elektronischen Publizieren und in Bibliotheken zu einer beachtlichen Verbreitung gebracht. Während es bei TIFF ausschließlich um die prozedurale Darstellung einer Graphik mittels Pixeln geht, erlaubt PDF schon gewisse Annotationen bzgl. der Dokumentenstruktur – es handelt sich also gewissermaßen um einen Hybriden zwischen prozeduraler und struktureller Dokumentenbeschreibung. TIFF und PDF sind *binäre* Dokumentenformate, die wir Menschen nicht ohne weiteres entziffern können. Es werden also spezielle Programme benötigt, um TIFF bzw. PDF-Dateien zu betrachten und zu edieren, mit starken Wechselwirkungen für die Migrier- und Emulierbarkeit.

*Textformate: HTML, XML*

Anschließend stellen wir mit den Standardformaten HTML und XML zwei Vertreter zur strukturellen Dokumentenbeschreibung vor. Ein Vorteil von HTML und XML besteht darin, dass keine komplexen Werkzeuge benötigt werden, um HTML- bzw. XML-Dokumente zu erstellen. Prinzipiell reicht ein einfacher Texteditor aus. Die Dokumenteninhalte sind als Klartext gespeichert, ebenso wie ihre Auszeichnungen. Diese Einfachheit wirkt sich positiv auf die Aspekte der Plattformunabhängigkeit, der Lesbarkeit für Mensch und Maschine und einer leichten Emulier- und Migrierbarkeit. Allerdings ist das Verhältnis von Dokumentenbeschreibung zu Dokumenteninhalt ungünstiger als bei den Binärformaten PDF und TIFF.

### 6.1.1 Tagged Image File Format (TIFF)

*Rastergraphiken*

Rastergraphiken sind eine wichtige Darstellungsform in digitalen Publikationen und in fast allen Anwendungsbereichen zu finden. Dabei hat eine Pixel-Matrix als mehr oder weniger genaues Abbild selbst keine explizit dargestellte logische Struktur, die die Semantik der Inhalte in einer für den Computer verstehbaren Form beschreibt. Typischerweise sind Rastergraphiken das Ausgabeformat von Scanprozessen oder von Mal- und Bildbearbeitungsprogrammen. Mit Programmen der Bildverarbeitung kann versucht werden, die Inhalte zu erschließen, wobei OCR-Programme bis jetzt die wichtigste Rolle spielen.

*TIFF als ein flexibles Format für Rastergraphiken*

Wegen seiner herausragenden Bedeutung und seiner Flexibilität soll das *Tagged Image File Format* (TIFF) näher betrachtet werden. Hierbei handelt es sich um ein Format zur Speicherung und zum Austausch von Rastergraphiken, aber weder um eine Druckersprache noch um eine Seitenbeschreibungs-Sprache. Die Spezifikation zu TIFF wurde 1986 von der Firma Aldus publi-

ziert, die jetzt zur Firma Adobe gehört. Die aktuelle Spezifikation 6.0[1] ist hauptsächlich mit neuen Bildformaten angereichert worden, ohne die früheren Versionen obsolet zu machen. Umgekehrt können aktuelle Versionen im Allgemeinen auch von älteren Programmen gelesen werden, sofern nur eine als *baseline* gekennzeichnete Untermenge der Spezifikation verwendet wird.

*Flexibilität durch dynamische Strukturen und private Felder*

Um die Möglichkeiten von TIFF besser zu verstehen, hilft ein kurzer Blick auf die internen Strukturen dieses Formats. Das wesentliche Element der Strukturierung ist das *Bilddateiverzeichnis* (image file directory – kurz IFD), welches die *TIFF-Felder* (field) enthält, die wiederum den Inhalt des Bildes beschreiben und Zeiger auf die eigentlichen Bilddaten bzw. auf Feldwerte, die eine festgesetzte Größe überschreiten, beinhalten. Diese Felder werden durch numerische *Marken* (tag) identifiziert. Das durch die Marke 256 identifizierte Feld beschreibt beispielsweise die *Bildbreite* (ImageWidth) in Pixel des als Pixel-Matrix gespeicherten Bildes. Ein IFD, dessen Länge nicht beschränkt ist, kann auch einen Zeiger auf einen weiteren IFD enthalten. Dadurch können innerhalb einer TIFF-Datei auch mehrere Bilder gespeichert werden. Schon durch die bereits erwähnten Mechanismen ergibt sich eine große Flexibilität. Die Definition *privater Felder* erhöht diese Flexibilität zusätzlich. Um Konflikte zu vermeiden, sollte eine Registrierung bei Aldus bzw. jetzt bei Adobe vorgenommen werden. Es besteht jedoch leider keine Pflicht, diese Definitionen zu veröffentlichen. Dieser Punkt wird auch im Projekt *Risk Management of Digital Information* angesprochen, das in Abschnitt 9.3 vorgestellt wird. Zu den veröffentlichten Marken gehört u. a. 330, die als *SubIFDs* bezeichnet wird. Diese erlaubt die Definition von beliebig vielen *Kinder-IFDs*, sodass Bilder baumartig strukturiert werden können. Somit kann beispielsweise bei einer zusammengehörigen Reihe von Bildern, z. B. die Seiten eines gescannten Buches, zu jedem Bild direkt ein Bild mit geringerer Auflösung oder ein *thumbnail* (Minibild) gespeichert werden.

*Bildformate und Kompressionsverfahren*

TIFF 6.0 unterstützt standardmäßig eine Reihe von Bildformaten und Kompressionsverfahren, die an dieser Stelle nicht weiter erläutert werden sollen. Die Spezifikation enthält eine anschauliche und knappe Darstellung, sowie Hinweise auf weitere Standards. Hier eine Übersicht der unterstützten Bildformate:

- Bilevel (baseline)
- Grayscale (baseline)
- Palette-color (baseline)

[1] siehe http://www.adobe.com/Support/TechNotes.html.

- RGB Full Color (baseline)
- CMYK
- YCbCr
- CIE L*a*b

Und eine Übersicht der Kompressionsverfahren:

- PackBits (baseline)
- Modified Huffman (baseline)
- LZW
- JPEG

Beim Einsatz von Kompressionsverfahren muss beachtet werden, dass Verfahren bzw. bestimmte Ausprägungen durch Patente geschützt sein können. Die aufgelisteten Bildformate und Kompressionsverfahren werden alle durch spezifizierte Felder und Werte identifiziert und beschrieben. Weitere Felder beschreiben die Eigenschaften der einzelnen Bilder sowie die interne Struktur der Datei genauer. Im Wesentlichen sind technische Belange berücksichtigt, aber der Erweiterungsmechanismus der privaten Felder würde es erlauben, auch beliebige deskriptive Metadaten aufzunehmen. Ob dies für Zwecke der Langzeitarchivierung sinnvoll ist, muss zumindest bei einer Sammlung mit unterschiedlichen Dateitypen in Frage gestellt werden.

### 6.1.2 Portable Document Format (PDF)

*Standard PDF*

Das *Portable Document Format* (PDF 2001) wurde von der Firma Adobe 1993 in seiner ersten Version als freier Standard verabschiedet. Die Idee hinter PDF war es, einen würdigen Nachfolger für das zu dieser Zeit sehr weit verbreitete PostScript (PostScript 1999) zu etablieren. So war es wichtig, Dokumente originalgetreu und plattformunabhängig darzustellen sowie Navigations- und Ediermöglichkeiten zu schaffen.

*Eigenschaften*

PDF-Dokumente können neben formatiertem Text auch Bilder oder sogar interaktive Elemente (z. B. JavaScript-Code) enthalten. Verwendete Schriftarten können in ein PDF-Dokument eingebettet werden, falls es sich um exotische Schriftarten handelt, die nicht allgemein verfügbar sind. Weiterhin können Textstellen über Verweise referenziert oder auch Kommentare zu Textstellen abgegeben werden. Wir können Text sowohl in seiner Reinform als edierbare Sequenz von Buchstaben als auch als Bild in einem PDF-Dokument unterbringen.

*Anwendung: hybride Dokumente*

Diese Möglichkeiten machen PDF insbesondere für die Archivierung von Dokumenten interessant, deren äußere Erscheinungsform von Relevanz ist. Nur auf dem Papier vorhandene Dokumente können zunächst gescannt und somit in ein Bild umgewandelt werden. Eine Schrifterkennung versucht danach, Text aus dem gescannten Bild zu erkennen (engl. *Optical Character Recognition*). Der erkannte Text kann dann *zusammen* mit den Bildern der Seiten des Papierdokuments in einem PDF-Dokument abgelegt werden. So verlieren wir einerseits nicht die Vorteile einer textbasierten Darstellung, andererseits ist das vorgestellte Verfahren aber auch recht robust gegenüber Fehlern bei der automatischen Texterkennung. Wir sprechen in diesem Zusammenhang auch von hybriden PDF-Dokumenten, da sie sowohl den Text als auch die Bilder des Textes »wild gemischt« enthalten.

*Aufbau eines PDF-Dokuments*

Das zentrale Element eines PDF-Dokuments ist wie bei PostScript die *Seite*. Ein PDF-Dokument besteht wie ein PostScript-Dokument aus einer geordneten Menge von Seiten, auf denen sich Daten befinden. Gegenüber PostScript ist es in PDF jedoch auch möglich, ein Inhaltsverzeichnis mit Verweisen auf die einzelnen Kapitel oder Abschnitte einzubetten. Das Inhaltsverzeichnis gibt also zumindest eine grobe Struktur des Dokuments wieder. Weiterhin ist es möglich, Metadaten in einem Dokument zu speichern wie Seitennummerierungen, Seitenformat oder auch (unstrukturierte) Informationen zum Dokumentinhalt selbst in Form von einfachen Kommentaren.

*Verarbeitung und Erstellung*

Es gibt eine Reihe von Programmen am Markt, mit denen man PDF-Dokumente erstellen und verarbeiten kann. Viele von ihnen werden von Adobe selbst vertrieben. Der Acrobat Reader, ein Betrachter für PDF-Dokumente, steht für eine Reihe von Betriebssystemen zur Verfügung.[2] Es handelt sich bei PDF zwar um einen freien Standard, dieser wird jedoch ausschließlich von Adobe weiterentwickelt. Die Werkzeugpalette für PDF kann indes nicht an die (frei) verfügbaren Werkzeuge für HTML und XML heranreichen, da PDF ein so genanntes Binärformat ist. Das heißt, die Daten sind nicht in Klartext gespeichert, was es unmöglich macht, PDF-Dokumente mit einem einfachen Texteditor zu lesen oder zu bearbeiten.

*elektronisches Publizieren*

PDF hat sich aufgrund seiner Vorteile gegenüber früheren Dokumentenformaten zu einem Standard für elektronisches Publizieren entwickelt. Insbesondere, wenn es um die originalgetreue

[2] Verfügbar unter http://www.adobe.de/products/acrobat/readstep.html.

Darstellung von Dokumenten geht, ist PDF heute das Format der Wahl.

*Vor- und Nachteile*

Die Konzentration auf den Darstellungsaspekt bringt sowohl Vor- als auch Nachteile. Ein Vorteil liegt sicherlich in der weitgehend originalgetreuen *Darstellung* einer Vielzahl von Dokumenten, was bei HTML und XML in dieser Form nicht möglich ist. Die Nachteile von PDF sollen jedoch auch nicht verschwiegen werden. Es handelt sich bei PDF um eine *Seiten*beschreibungs-Sprache und nicht um eine *Dokumenten*beschreibungs-Sprache. Die logische Struktur eines Dokuments kann eben nur durch ein relativ grobes Inhaltsverzeichnis wiedergegeben werden. Weiterhin sind PDF-Dokumente nur relativ ineffizient verarbeitbar, zumindest gegenüber HTML- oder XML-Dokumenten. Machen Sie einmal die Probe aufs Exempel, indem Sie in einem umfangreichen PDF-Dokument nach einem Text suchen und das gleiche in einem ebenso umfangreichen XML-Dokument wiederholen. Auch sind die Suchmöglichkeiten in PDF-Dokumenten aufgrund fehlender Strukturinformationen weit weniger mächtig als in XML-Dokumenten (vgl. auch Abschnitt 6.1.4). Hinzu kommt, dass das Angebot an Werkzeugen zur Erzeugung und Weiterverarbeitung im Gegensatz zu HTML oder XML recht beschränkt ist und die Werkzeugentwickler relativ abhängig von einer Firma sind. Jedoch sei noch einmal betont, dass es bis heute kein mit PDF direkt vergleichbares Dokumentenformat gibt, das den Fokus auf die Darstellung von Dokumenten legt und es gleichzeitig erlaubt, strukturelle Elemente in die Dokumentenbeschreibung mit aufzunehmen.

### 6.1.3 HyperText Markup Language (HTML)

*HTML: Sprache des WWW*

Die *HyperText Markup Language* (HTML) (W3C 1999a) bildet das Rückgrat des World Wide Web, dem wohl meistgenutzten Teil des Internets. Sie stellt eine einfache Möglichkeit dar, strukturelles Markup zu verwenden.

*Aufbau eines HTML-Dokuments*

Ein HTML-Dokument besteht aus mehreren Dokumentteilen, die Elemente genannt werden (z. B. `table`). Durch *Tags* werden der Beginn und das Ende eines Elements näher gekennzeichnet. Wir unterscheiden *BeginTags* (z. B. `<table>`) und *EndTags* (z. B. `</table>`). Es existiert ein fester Satz von Elementen, die aber nur nach bestimmten Regeln verwendet werden dürfen. So ist beispielsweise festgelegt, dass eine Tabelle aus Zeilen und Spalten besteht und optional einen Titel enthalten kann (vgl. Abschnitt 5.2.1).

*Darstellung*

Die Darstellung eines so strukturierten HTML-Dokuments ist zunächst einmal ausschließlich durch seine Struktur festgelegt. Es gibt auch Elemente, die nicht der Dokumentenstruktur dienen, sondern lediglich prozeduralen Charakter haben, etwa das Element HR für eine waagerechte Linie. Oft haben solche Elemente keinen Inhalt – sie sind leer. Wir können für leere Elemente entweder Begin- und EndTag direkt aufeinander folgen lassen (<HR></HR>) oder aber das BeginTag am Ende mit einem Schrägstrich versehen (<HR/>). Um eine flexiblere Darstellung zu erreichen, wurden Sprachen für so genannte Stylesheets entwickelt, die es erlauben, auf die Darstellung der einzelnen Elemente gezielt Einfluss zu nehmen. Ein Beispiel hierfür sind die *Cascading Stylesheets* (W3C 1998).

*Attribute*

Jedes Element kann durch Attribute näher spezifiziert werden, die einfache Zeichenketten als Werte haben können. Beispielsweise hat das HR Element das Attribut `width` zur Bestimmung der Linienlänge und das Attribut `align` zur Bestimmung der Ausrichtung:

```
<HR width="50%" align="center"/>
```

*Problem: fester Element-Satz*

Die von vornherein festgelegte Menge an Elementen stellte sich als ein gravierender Nachteil von HTML heraus: Der ursprüngliche HTML-1.0 Standard wurde im Laufe der Zeit weiterentwickelt und es kamen immer neue Elemente hinzu. Schnell wurde jedoch klar, dass es unmöglich ist, alle Eventualitäten der Dokumentenstruktur mit einem festen Element-Satz zu berücksichtigen. Hinzu kommt, dass durch die festgelegte Namensgebung der Elemente das Ziel der »sprechenden« (also für Menschen selbsterklärenden) Tags in den Hintergrund rückt. Obiges Beispiel verdeutlicht diesen Umstand recht gut. Wer vermutet hinter dem Element-Namen `td` schon ein Tabellendaten-Element. Auch die feste Bindung der Dokumentenstruktur an die Darstellung wäre hinderlich, wenn man versuchen würde, HTML als universelles Dokumentenformat zu etablieren.

### 6.1.4 eXtensible Markup Language (XML)

*Standards: XML vs. SGML*

Die *eXtensible Markup Language* (XML) (W3C 2000a) wurde im Februar 1998 vom World Wide Web Consortium (W3C)[3] verabschiedet und im Jahre 2000 einer Revision unterzogen (W3C

[3]Das W3C ist eine der größten Organisationen im Internet, die sich mit der Entwicklung von Standards beschäftigt. Unter anderem wurden die hier besprochenen Standards HTML und XML vom W3C entwickelt.

2000a). XML gilt als Nachfolger und Einschränkung der *Standard Generalized Markup Language* (SGML) (ISO 1986). Wir werden hier nicht weiter auf SGML eingehen, da diese Sprache aufgrund ihrer ungeheuren Komplexität lediglich von wenigen Spezialisten vernünftig eingesetzt werden kann und keine breite Anwendung fand. Die grundlegenden Ideen von SGML werden ebenso in XML umgesetzt und verfeinert. Der Leser sei an dieser Stelle darauf hingewiesen, dass wir nur einige Eigenschaften und Anwendungsgebiete von XML vorstellen können. Nähere Einzelheiten findet man beispielsweise bei Goldfarb (2002).

*Metasprache XML*

Mit XML vermeidet man die oben genannten Nachteile von HTML: Es wird generell zwischen dem Inhalt eines Dokuments und seiner Darstellung unterschieden. Dies macht es möglich, sowohl für ein Dokument mehrere Darstellungen (Sichten) zu etablieren, als auch für mehrere Dokumente ein einheitliches Erscheinungsbild zu definieren. Weiterhin wurde XML nicht mit einem festen Element-Satz ausgestattet. Vielmehr kann jeder Nutzer sich einen eigenen Element-Satz und dazu passende Struktur-Regeln selbst definieren. Es handelt sich bei XML also nicht um eine Markup-Sprache, sondern um eine *Meta*sprache, die es uns erlaubt, eine für den jeweiligen Anwendungszweck passende Markup-Sprache zu definieren. Dieser Sachverhalt steht ganz im Gegensatz zu den vorher vorgestellten Dokumentenformaten.

*wohlgeformte XML-Dokumente*

Die Sprache XML selbst legt nur einige grundlegende Regeln fest. Beispielsweise wie die ein Element begrenzenden Tags syntaktisch auszusehen haben. Weiterhin ist die Groß- bzw. Kleinschreibung von Tags relevant, und es dürfen auch keine End-Tags weggelassen werden. Überlappende Tags sind ebenfalls verboten. Insgesamt gleicht ein XML-Dokument logisch gesehen einem Baum – das oberste Element wird dabei auch Wurzel genannt. Dokumente, die diesen Regeln genügen, heißen wohlgeformt (engl. *well-formed*). Die obigen Einschränkungen vereinfachen die automatische Verarbeitung von XML-Dokumenten erheblich, was der Verbreitung von XML sehr zu Gute kommt.

### Definition der Dokumentenstruktur: DTD, XML-Schema, Namensräume

*Document Type Definition (DTD)*

Die erlaubten Elemente selbst können in einer *Document Type Definition* (DTD) definiert werden. Weiterhin werden Regeln bestimmt, denen die syntaktische Struktur eines Dokuments genügen muss, das auf dieser DTD basiert. Hierzu ein Beispiel:

```
<!ELEMENT table  (title, row*)>
<!ELEMENT row    column+>
<!ELEMENT column (#PCDATA)>
<!ELEMENT title  (#PCDATA)>
```

Nach dieser Strukturbeschreibung besteht eine Tabelle (`table`) aus einem Titel (`title`), gefolgt von beliebig vielen Tabellenzeilen (`row`). Eine Folge wird dabei durch das Komma (`,`) und eine beliebige Anzahl durch einen Stern (`*`) dargestellt. Eine Tabellenzeile besteht aus mindestens einer Tabellenspalte (`column`), dargestellt durch das Plus-Zeichen (`+`). Eine Tabellenspalte besteht ebenso wie der Titel der Tabelle aus beliebig vielen Zeichen, was durch das Synonym `#PCDATA` (engl. *parsable character data*) gekennzeichnet wird.

*gültige XML-Dokumente*

Ein Dokument, das gültig (engl. *valid*) bezüglich obigem DTD-Ausschnitt ist, könnte so aussehen:[4]

```
<?xml version='1.0'?>
<!DOCTYPE table "table.dtd">

<table>
 <title>Eine Tabelle</title>

 <row>
  <column>Zeile 1, Spalte 1</column>
  <column>Zeile 1, Spalte 2</column>
 </row>

 <row>
  <column>Zeile 2, Spalte 1</column>
  <column>Zeile 2, Spalte 2</column>
 </row>
</table>
```

Im Gegensatz zu obigem Dokument ist das folgende Dokument nicht gültig, da die Tabelle zumindest einen Titel enthalten muss:

```
<?xml version='1.0'?>
<!DOCTYPE table "table.dtd">

<table></table>
```

[4] Der Verweis auf die verwendete obige DTD table.dtd steht in der Anweisung `DOCTYPE`, wobei wir als Wurzel-Element `table` verwenden. Zusätzlich wird bei XML-Dokumenten die XML-Sprachversion, hier `1.0`, angegeben.

Allerdings genügt es den Grundregeln von XML und ist daher wohlgeformt.

*Darstellung*

Wir können nun Dokumente erstellen, die einer DTD genügen und auch wohlgeformt sind. Dies hat allerdings nichts mit ihrer Darstellung zu tun. Die Darstellung der einzelnen Elemente wird in Stylesheets festgelegt, ähnlich wie es auch schon bei HTML möglich ist. Allerdings ist die Idee hier wesentlich konsequenter umgesetzt. Ein Element hat von sich aus *keine* feste Darstellung, sondern benötigt dafür zwingend ein Stylesheet. Diese Konsequenz ermöglicht es im Endeffekt, verschiedene Stylesheets zu programmieren, die jeweils eine mögliche Sicht auf das Dokument bieten oder es sogar in verschiedene Formate umwandeln, beispielsweise PostScript für den Druck oder auch HTML für den Internetbrowser.

*zusätzliche Strukturinformationen durch leere Elemente*

Ebenso wie in HTML können wir in XML leere Elemente in einer DTD definieren, die dann bei einer Strukturprüfung nicht mitgeprüft werden, da sie ja nichts zur Dokumentenstruktur beitragen. Leere Elemente können u. a. dazu dienen, eine zusätzliche Gliederung auszuzeichnen, neben der eigentlichen logischen Struktur. Wir werden in Abschnitt 8.3 sehen, wie dies in der Praxis genutzt wird, um beispielsweise Seitenumbrüche zu markieren.

*DTD-Syntax vs. XML-Syntax*

Beim Vergleich der Syntax von DTD und XML-Dokument wird ein Nachteil der Document Type Definition deutlich. Eine DTD folgt offensichtlich einer anderen Syntax als ein XML-Dokument. Ein Parser muss also zwei Sprachen beherrschen: die XML-Syntax und die DTD-Syntax. Die Auswirkungen auf die Langzeitarchivierung sind ebenso klar: Es müssen eigentlich zwei Formate migriert (bzw. emuliert) werden.

*XML-Schema*

Der Schlüssel zum erfolgreichen Lösen dieses Problems liegt darin, sich bei der Definition der zulässigen Elemente wieder der XML-Syntax zu bedienen. XML-Schemata (W3C 2001c) genügen der XML-Syntax und definieren (ebenso wie DTDs) neue Elemente und Regeln für die Dokumentenstruktur.

*Beispiel*

Ein Ausschnitt eines XML-Schema, das genau die gleichen Eigenschaften einer Tabelle definiert wie der obige DTD-Ausschnitt, könnte wie folgt aussehen:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">

 <element name="table">
  <complexType>
   <sequence>
    <element ref="title" minOccurs="1"
                         maxOccurs="1"/>
    <element ref="row" minOccurs="0"
                       maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>

 <element name="row">
  <complexType>
   <sequence>
    <element ref="column" minOccurs="1"
                          maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>

 <element name="title" type="string"/>
 <element name="column" type="string"/>
</schema>
```

XML-Schemata unterscheiden zwischen einfachen Elementen mit atomaren Typen, wie `string`, und komplexen Elementen, die wiederum andere Elemente enthalten können (definiert durch `complexType`. Eine Folge von einem oder mehreren Elementen wird dabei durch `sequence` ausgedrückt. Anstatt der bei der DTD üblichen Kennzeichnungen + und * verwendet XML-Schema explizite Angaben darüber, wie oft ein Element vorkommen darf. Dabei bezeichnet `minOccurs` die untere Grenze und `maxOccurs` die obere Grenze, wobei `unbounded` für »unendlich oft« steht.

*XML-Schema und die Langzeitarchivierung*

Zugegeben, der obige Abschnitt sieht ein wenig komplizierter aus als der ursprüngliche DTD-Ausschnitt. Diese Komplexität liegt hauptsächlich daran, dass die XML-Schema-Sprache gegenüber den Möglichkeiten einer DTD wesentlich erweitert wurde. Maschinen haben hiermit aber weit weniger Probleme als wir Menschen; für sie ist die Syntax-Konformität zu XML wesentlich wichtiger. Nun muss nur noch ein Format migriert bzw. emuliert

werden. Zusätzlich benötigen wir noch eine Art Mutterschema,[5] das die Struktur eines XML-Schema beschreibt, also beispielsweise die XML-Schema-Elemente `complexType` oder `element`. Da das Mutterschema auch wieder ein XML-Schema ist, beschreibt es seine eigene Struktur.

*XML-Namensräume*

Eine weitere Spezialität von XML sind *Namensräume* (W3C 1999b). Sie eröffnen uns die Möglichkeit, Elemente aus verschiedenen XML-Schemata oder DTDs in einem XML-Dokument zu verwenden, ohne Konflikte bei gleichen Element-Namen befürchten zu müssen. Oder anders ausgedrückt definiert jedes XML-Schema und jede DTD einen Namensraum. Der Namensraum wird einfach dem eigentlichen Element-Namen vorangestellt und durch einen Doppelpunkt von ihm getrennt.

*Beispiel*

Hierzu wieder ein kleines Beispiel: Nehmen wir an, wir wollen die Eigenschaften verschiedener Tische in einer Tabelle vergleichen. Dazu müssen wir obiges XML-Schema (`tabellen-schema`) erweitern um die Elemente eines XML-Schema `tisch-schema`, in dem die Elemente für eine Tischbeschreibung (u. a. `table`, `name`, `height`) definiert sind.

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:tisch="tisch-schema">

 <element name="table">...</element>
 <element name="row">...</element>
 <element name="title" type="string"/>

 <element name="column">
  <complexType mixed="true">
   <sequence>
    <element ref="tisch:table" minOccurs="1"
             maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
</schema>
```

Das Element `table` markiert eine Tabelle, während das Element `tisch:table` einen Tisch markiert. Wir können nun beide Schemata als Grundlage von XML-Dokumenten verwenden, beispielsweise so:

[5] Das »Mutterschema« ist unter http://www.w3.org/2001/XMLSchema zu finden.

```
<?xml version='1.0'?>
<table xmlns="tabellen-schema"
       xmlns:tisch="tisch-schema">
 <title>Eine Tabelle mit Tischen</title>
 <row>
  <column>
    <tisch:table>
     <name>Bartisch</name>
     <height>80 cm</height>
    </tisch:table>
    <tisch:table>
     <name>Couchtisch</name>
     <height>40 cm</height>
    </tisch:table>
  </column>
  <column>Zeile 1, Spalte 2</column>
 </row>
</table>
```

### Universalformat XML

Was können wir mit unseren XML-Dokumenten anfangen?

*Navigation in XML-Dokumenten: XPath*

Beispielsweise können wir Elemente mit der Sprache XPath (W3C 1999d) *adressieren*, um sie maschinell zu verarbeiten. Logisch gesehen ist ein XML-Dokument ein Baum, dessen Knoten die Elemente sind und dessen Kanten ausschließlich vom Eltern-Element zu den Kind-Elementen zeigen. Wenn wir uns ein XML-Dokument als Baum vorstellen, können wir mittels XPath einen *Pfad* in diesem Baum definieren. Wir können uns dies ganz ähnlich einem (Unix) Dateisystem vorstellen. Näheres zu XPath findet man u. a. bei Wilde und Lowe (2002).

*Abfrage von XML-Dokumenten: XQuery*

Mit der Metasprache XML können wir Dokumente strukturiert beschreiben, was eine gewisse Ähnlichkeit zur strukturierten Ablage von Daten in einer Datenbank hat. Nur ist die Struktur der Daten in einem XML-Dokument nicht ganz so streng wie in einer Datenbank. Abfragesprachen für Datenbanken sind weit verbreitet und bekannt. Ein weit verbreiteter Standard ist beispielsweise SQL (engl. *Structured Query Language*). Was liegt also näher, als auch für XML-Dokumente eine SQL-ähnliche Abfragesprache zu definieren? Die Vorarbeit dazu wurde mit XPath bereits getan – es ist uns bereits möglich, innerhalb von XML-Dokumenten zu navigieren. Um eine benutzbare Abfragesprache zu entwerfen, bedarf

es zusätzlich noch der Möglichkeit, die Abfrageergebnisse entsprechend aufzubereiten. Das W3C treibt momentan die Abfragesprache XQuery voran,[6] die auf der Navigationssprache XPath 2.0 basiert. XPath 2.0 ist der Nachfolger des eben vorgestellten XPath und ebenfalls erst in der Entwurfsphase.[7] Näheres zu XQuery findet man u. a. bei Nagarajan *et al.* (2003).

*Transformation und Präsentation von XML-Dokumenten*

Wie schon im vorherigen Abschnitt angedeutet, benötigen wir zwingend ein Stylesheet, um XML-Dokumente darzustellen. Hierzu wurde vom World Wide Web Consortium (W3C) die *eXtensible Stylesheet Language* (XSL) entwickelt. Diese besteht aus den beiden Teilen XSLT (engl. *XSL Transformations*) und XSL-FO (engl. *XSL Formatting Objects*). Die Sprache XSLT ähnelt einer Programmiersprache, mit der man die logische Struktur von XML-Dokumenten verändern (transformieren) kann. Das Ergebnis einer Transformation können wieder XML-Dokumente oder Dokumente anderer Formate sein. Die zweite XSL-Komponente XSL-FO bietet Möglichkeiten, die wir aus modernen Desktop-Publishing-Systemen kennen. Wir können hiermit sehr präzise die Darstellung eines XML-Dokuments beeinflussen. Nehmen wir an, wir hätten das vorliegende Buch als XML-Dokument vorliegen. Dann können wir mittels XSLT eine HTML-Version erzeugen, die aus mehreren, untereinander verlinkten HTML-Seiten besteht. Zusätzlich wäre denkbar, die Literaturangaben unseres Buches in eine Datenbank umzuwandeln. Mittels XSL-FO würden wir schließlich eine qualitativ hochwertige Druckvorlage des Buches in PDF erzeugen.

*XML-Dialekte*

Mit zunehmender Beliebtheit der Sprache XML wurden eine Reihe von XML-DTDs und XML-Schemata entwickelt. Wir sprechen in diesem Zusammenhang auch von XML-*Dialekten*, da ja eine DTD oder ein Schema prinzipiell eine eigene Sprache definiert.

Interessante Sprachen sind beispielsweise

- Scalable Vector Graphics (SVG) für die strukturierte Beschreibung von Vektorgraphiken (W3C 2001b),
- Mathematical Markup Language (MathML) für die strukturierte Beschreibung mathematischer Formeln (W3C 2001a),
- VoiceXML für die strukturierte Beschreibung gesprochener Sprache.[8]

*XHTML*

Wir sehen, dass es eine Reihe von XML-Dialekten für verschiedenste Anwendungsbereiche gibt. Es wurde inzwischen selbst für

[6] Für eine Vorabversion siehe http://www.w3.org/TR/xquery/.

[7] Für eine Vorabversion siehe http://www.w3.org/TR/xpath20/.

[8] siehe hierfür http://www.w3.org/TR/voicexml20/.

den HTML-Standard eine XML-DTD (XHTML) erstellt, sodass XML-Werkzeuge auch mit HTML-Dokumenten umgehen können (W3C 2002). Im Wesentlichen umfasst XHTML zwei Einschränkungen für HTML-Dokumente, die den well-formed Regeln von XML zu verdanken sind: Geöffnete Tags müssen auch wieder geschlossen werden, und die Tagnamen sind ausschließlich klein zu schreiben.

*XML als Datenaustauschformat*

Aufgrund seiner Selbstbeschreibungsfähigkeit eignet sich XML hervorragend als Datenaustauschformat. Dies haben auch viele Softwarehersteller bemerkt und bieten an, ihre Dokumente in einem XML-konformen Format zu speichern bzw. XML-Dokumente zu bearbeiten. Einige wenige Beispiele kommerzieller Software sind:

- WordPerfect Office 10 (WordPerfect-Erstellung),
- Lotus Domino (Lotusscript-Erstellung und -Bearbeitung),
- Quark Express (Export) und
- Microsoft-Office XP (Access- und Excel-Exportfunktionen).

Andere Softwarehersteller gehen sogar soweit, XML zu ihrem nativen Dokumentenformat zu erklären, z. B. StarOffice. Weitere Beispiele hierfür finden wir im OpenSource-Bereich;[9] zu nennen wären etwa OpenOffice oder KOffice.

*Verhältnis von Markup zu Dokumenteninhalt*

Bei der Verwendung von XML als Datenformat der Wahl für die Langzeitarchivierung elektronischer Dokumente sollten wir uns jedoch trotz aller Vorteile des relativ großen Overhead dieses Formats bewusst sein. Das Verhältnis von Daten und eigentlichem Dokumenteninhalt ist im Vergleich zu Binärformaten alles andere als gut. Hier hilft nur die Komprimierung der XML-Dateien, wie sie z. B. OpenOffice anbietet. Allerdings beeinträchtigt eine Komprimierung der Daten wieder die Austauschbarkeit und die Möglichkeiten zur langfristigen Archivierung – schließlich muss der Kompressionsalgorithmus bekannt sein, um auf derart komprimierte Dokumente später wieder zurückgreifen zu können. Eine Beschreibung bzw. Spezifikation des Kompressionsalgorithmus in XML würde hier sicher helfen, allerdings ist die formal korrekte Spezifikation von Algorithmen äußerst komplex und zumindest momentan nicht mit vernünftigem Aufwand in XML zu realisieren.

*Beschreibung von Nicht-Dokument-Daten*

Abschließend wollen wir einen kleinen Ausblick geben, wofür sich XML noch eignet. Grob gesagt, können wir mit XML jedwede Art strukturierter Daten darstellen, also beispielsweise Daten-

[9] Es handelt sich hierbei um frei verfügbare Software.

banken, Datenmodelle für die Softwareerstellung oder auszutauschende Daten von Internetdiensten. So können wir die Vorteile des Dokumentenformats XML auch in anderen Anwendungsgebieten sinnvoll nutzen. Für die letzten beiden Anwendungsgebiete wollen wir zwei XML-basierte Beschreibungsansätze vorstellen: *XML Metadata Interchange* (XMI) und *Web Service Description Language* (WSDL). Beide Ansätze sind noch relativ neu; weiterhin würde eine detaillierte Beschreibung sicher den Rahmen dieses Buches sprengen. Wir werden uns daher auf die grundlegenden Ideen hinter XMI und WSDL beschränken.

*Beschreibung von Datenmodellen: XMI*

Der XMI-Standard (OMG 2002) wurde von der *Object Management Group* (OMG) 1999 verabschiedet. Das Ziel von XMI besteht darin, Metadaten zwischen Werkzeugen zur Softwareerstellung über das Internet auszutauschen. Es gibt eine Vielzahl solcher Werkzeuge, die sich dem gleichen Programmierparadigma verschrieben haben und die gleiche Modellierungssprache benutzen. In der Softwareerstellung werden heute oft das objektorientierte Programmierparadigma und die Modellierungssprache UML (OMG 2001) verwendet. Trotzdem speichern die Werkzeuge ihre Daten und Metadaten jeweils in einem eigenen proprietären Format. Mit XMI können wir diese Daten zwischen verschiedenen Werkzeugen austauschen. So ist es beispielsweise möglich, komplette UML-Modelle mittels XMI darzustellen. Das jeweilige Werkzeug muss die Daten und Metadaten seiner Modelle nur noch nach XMI exportieren.

*Internetdienste und Agenten*

Das Internet bietet nicht nur statische HTML-Seiten, sondern kann für uns auch *Dienste* erbringen. Diese Dienste können wir einfach selbst abrufen, beispielsweise einen Flug buchen, ein Taxi bestellen, online einkaufen oder Börsenkurse abrufen. Die Materie wird ein wenig komplizierter, wenn wir ein Programm beauftragen, verschiedene Dinge für uns im Internet »zu erledigen«. Nennen wir ein solches Programm *Agent*. Wir können einen Agenten beispielsweise beauftragen, einen Flug für uns zu buchen, am Ankunftsort ein Hotel zu reservieren und dafür zu sorgen, dass ein Taxi auf uns wartet. Der Agent nutzt dabei die von verschiedenen Internetseiten angebotenen Dienste. Dabei ist er auf eine präzise Definition der *Schnittstellen* zu diesen Diensten angewiesen.

*Beschreibung von Internetdiensten: WSDL*

Welche Parameter erwartet ein Dienst und in welcher Reihenfolge? Wie sehen die Ergebnisse eines Dienstes aus? Die *Web Service Description Language* (WSDL) ist ein XML-basierter Ansatz, um Internetdienste zu beschreiben. Die Sprache WSDL ist jedoch noch längst nicht soweit gediehen wie XMI – sie steckt gewissermaßen noch in den Kinderschuhen. Eine Vorabversion befindet

sich auf den Internetseiten des W3C.[10] Ein Internetdienst wird in WSDL mit einem XML-Dokument beschrieben, das den Dienst und seine angebotenen Operationen beschreibt. Jede Operation tauscht dabei mit dem Auftraggeber (meist ein Agent) Nachrichten in einer genau definierten Reihenfolge aus – Eingabe- und Ausgabe-Parameter. Die Parameter sind jeweils durch ihren Typ beschrieben. Die Typbeschreibung selbst geschieht dabei durch XML-Schema.

*WSDL vs. Semantik*

Leider ist es in WSDL in seiner gegenwärtigen Form nur möglich, Internetdienste syntaktisch anhand der Typisierung der ausgetauschten Nachrichten zu beschreiben. Eine semantische Beschreibung von Internetdiensten findet in WSDL zumindest gegenwärtig nicht statt. Es ist zwar klar, wie wir den Dienst ansprechen können, jedoch nicht, was er eigentlich leistet!

Wir hoffen, mit den Beispielen XMI und WSDL einen kleinen Ausblick in die XML-Welt gegeben zu haben. XML ist zwar ein Dokumentenbeschreibungsformat, wir können jedoch nahezu beliebige Informationen mit XML strukturiert darstellen und einfach über das Internet austauschen.

Für die Langzeitarchivierung eröffnet sich hier ein gewaltiges Potenzial, aber es entstehen auch schier unüberwindliche Problemfelder. Soll etwa die Art und Weise der Diensterbringung für die Nachwelt erhalten und nachvollziehbar bleiben?

## 6.2 Standards zur semantischen Beschreibung von Metadaten und Dokumenten

In Abschnitt 5.2.1 haben wir die prinzipielle Möglichkeit erörtert, wie wir Dokumente mit ihrer Bedeutung versehen können. Eine strukturelle Beschreibung von Dokumenten reicht für die maschinelle Erfassung ihrer Bedeutung bei weitem nicht aus, ist jedoch eine wichtige Grundlage.

*Prinzip der Semantikbeschreibung*

Rufen wir uns noch einmal das prinzipielle Vorgehen ins Gedächtnis zurück. Die Bedeutung eines Dokuments wird auf die Bedeutung der einzelnen Dokumentteile zurückgeführt. Die Bedeutung der atomaren Dokumentbestandteile, etwa des Titels, wird an möglichst zentraler Stelle definiert. Über Assoziationen zwischen Dokumentteilen kann dann die Bedeutung des Dokuments nach und nach hergeleitet werden.

[10] Abrufbar unter http://www.w3.org/TR/wsdl/.

*RDF und Topic Maps*

In den folgenden beiden Abschnitten stellen wir die Standards RDF und Topic Maps vor, die der Definition von Assoziationen zwischen (atomaren) Dokumentteilen dienen. Im letzten Abschnitt beleuchten wir das Thema der Definition von Beziehungen zwischen in RDF oder Topic Maps verwendeten Vokabularen näher.

Eine Möglichkeit der Definition atomarer Dokumentbestandteile werden wir am Beispiel der *Dublin Core Metadata Initiative* im zweiten Teil dieses Buches (Abschnitt 8.1) vorstellen.

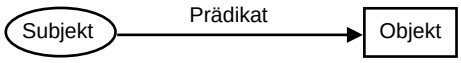
### 6.2.1 Resource Description Framework (RDF)

Das *Resource Description Framework* (RDF) (W3C 1999c) wurde vom W3C im Februar 1999 als Empfehlung verabschiedet. Wie der Name schon sagt, handelt es sich bei RDF lediglich um ein Rahmenwerk (engl. *framework*), das es erlaubt, Dinge (engl. *resources*) zu beschreiben. Diese Dinge können zunächst einmal alles sein, was sich eindeutig identifizieren lässt. Die Identifikation geschieht über so genannte URI (vgl. Abschnitt 5.4.1).

*RDF-Modell*

RDF modelliert die Beziehungen zwischen den beschriebenen Dingen mittels Sätzen, die sich graphisch darstellen lassen. Jeder Satz (engl. *statement*) besteht aus einem Subjekt (engl. *subject*), einer Eigenschaft (engl. *predicate* für Prädikat) und einem Objekt (engl. *object*). Wir können RDF-Sätze also mit einfachen Aussagen vergleichen. Abbildung 6.1 zeigt die allgemeine graphische Darstellung eines Satzes im RDF-Modell.

**Abbildung 6.1**
*RDF-Modell*




*Eigenschaften und Objekte*

Jede Eigenschaft ist mit einer ihr eigenen, fest verbundenen Bedeutung verknüpft. Atomare Eigenschaften können beispielsweise via Dublin Core (vgl. Abschnitt 8.1) definiert werden. Komplexere Eigenschaften werden zusammen mit ihren Charakteristika in so genannten RDF-Schemata (W3C 2000b) definiert – dazu später mehr. Ein Objekt kann neben einem atomaren Wert wieder eine Aussage (also ein RDF-Satz) sein. Wir sind somit in der Lage, Aussagen über Aussagen zu machen.

*Beispiel*

Dieses eigentlich bestechend einfache Modell lässt uns nun unsere Dokumente mit einer Semantik versehen. Dazu ein kleines Beispiel: Nehmen wir an, wir sollen in RDF die Tatsache modellieren, dass der Name des vorliegenden Buches »Langzeitarchivierung« ist. Dazu ist es notwendig, die Eigenschaft Name

zu definieren und das Buch eindeutig mit einer URI zu identifizieren. Wir nehmen weiterhin an, dass das Buch unter der URL http://www.xyz.de/lza.pdf veröffentlicht ist. Abbildung 6.2 enthält den zugehörigen RDF-Graphen. Es ist leicht einzusehen, dass eine Fülle (grammatikalisch korrekter) deutscher Sätze existiert, die dem RDF-Graphen aus Abbildung 6.2 entsprechen. Diese Vielfalt wird unüberschaubar, wenn wir auch andere natürliche Sprachen wie Englisch oder Französisch in Betracht ziehen.

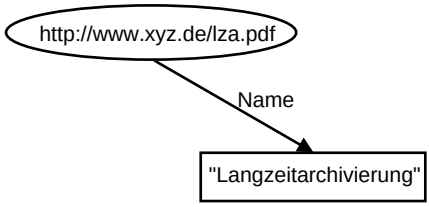



**Abbildung 6.2**
*RDF: Direkte Aussagen*

*indirekte Aussagen*

Ein Beispiel für eine indirekte Aussage ist der folgende deutsche Satz: Ein Autor unseres Buches sagt, es hätte den Titel »Langzeitarchivierung«. Abbildung 6.3 zeigt den zugehörigen RDF-Graphen.[11] Hierbei ist der Autor zunächst anonym, es ist jedoch leicht möglich, ihn mittels geeigneter RDF-Teilgraphen näher zu beschreiben. Weiterhin müssen indirekte Aussagen reifiziert werden. Unter *Reifikation* versteht man die explizite Angabe der indirekt vorhandenen Tatsache, dass ein RDF-Satz (Teilgraph) ein RDF-Objekt ist.

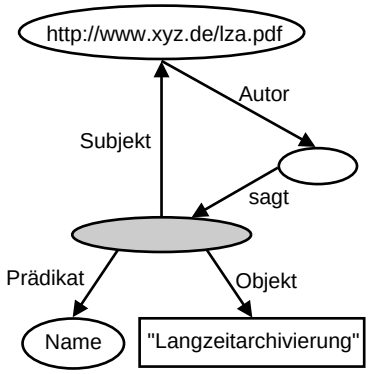



**Abbildung 6.3**
*Indirekte Aussagen in RDF*

*RDF-Schema*

Das verwendete Vokabular eines RDF-Graphen wird mittels so genannter RDF-Schemata definiert. Hierbei handelt es sich um

[11] Wir haben die Typinformationen innerhalb des Graphen der Übersicht halber stark vereinfacht und ins Deutsche übersetzt.

einen sehr mächtigen Mechanismus, der den Möglichkeiten der Klassendefinitionen objektorientierter Programmiersprachen ähnelt. Der Rest dieses Abschnitts ist sehr technisch und kann beim ersten Durchlesen übergangen werden.

*Eigenschaften und Klassen*

Wir können sowohl neue Eigenschaften (Unterklasse von `rdfs:Property`[12]) als auch neue Klassen (Unterklasse von `rdfs:Class`) definieren und zumindest Teilmengenbeziehungen (via Unterklassenbildung `rdfs:subClassOf`) zwischen ihnen ausdrücken. Klassen können wir hierbei als Mengen von Objekten verstehen. Sowohl `rdfs:Class` als auch `rdfs:Property` sind Unterklassen der globalen Kernklasse `rdfs:Resource`.

*Beispiel*

Ein mögliches Beispielschema für den RDF-Graphen in Abb. 6.2 sehen wir in Abb. 6.4. Das Schema definiert die neue Eigenschaft (`rdfs:Property`) *Name*.[13] Ihr Definitionsbereich (`rdfs:domain`) ist ein beliebiges »Ding« (`rdfs:Resource`). Der zugehörige Wertebereich (`rdfs:range`) ist ein atomarer Wert (`rdfs:Literal`).

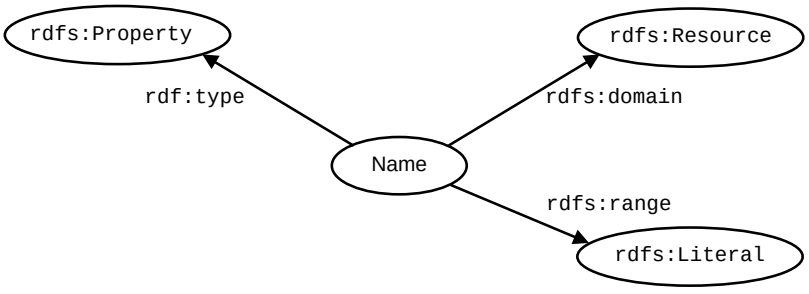



**Abbildung 6.4** *RDF-Schema für Abb. 6.2*

*RDF codiert als XML*

Natürlich können wir einer Maschine keine Graphen zur semantischen Beschreibung von Dokumenten anbieten. Hierfür gibt es eine serialisierte Syntax in XML. Das erste Beispiel (Abb. 6.2) lässt sich folgendermaßen in XML-Syntax darstellen:[14]

```
<rdf:Description
     about="http://www.xyz.de/lza.pdf">
 <s:Name>Langzeitarchivierung</s:Name>
</rdf:Description>
```

[12] Wir benutzen den für RDF-Schemata üblicherweise verwendeten Namensraum `rdfs`.

[13] Die Eigenschaft `type` ist im Namensraum für RDF definiert und trägt daher das Präfix `rdf`. Die anderen Eigenschaften sind im Namensraum für RDF-Schemata definiert und tragen daher das Präfix `rdfs`.

[14] Wir nehmen an, dass wir das obige RDF-Schema eingebunden und mit dem Namensraum `s` verknüpft haben.

Für RDF-Schemata gibt es ebenfalls eine serialisierte Syntax. Wir wollen jedoch hier nur das Beispielschema von oben angeben und nicht detaillierter darauf eingehen, da eine genaue Behandlung den Rahmen dieses Buches sprengen würde.

```
<rdf:Description ID="Name">
 <rdf:type
    resource="http://...-rdf-syntax-ns#Property"/>
 <rdfs:domain
    resource="http://...-rdf-syntax-ns#Resource"/>
 <rdfs:range
    resource="http://...-rdf-syntax-ns#Literal"/>
</rdf:Description>
```

Mehr zu RDF und RDF-Schema findet man u. a. bei Powers (2003). Wir werden konkrete Anwendungsmöglichkeiten von RDF am Ende des folgenden Abschnittes vorstellen, da Topic Maps im Wesentlichen das gleiche Ziel wie RDF haben.

### 6.2.2 Topic Maps

Der *Topic Map*-Standard wurde 1999 von der International Organization for Standardization (ISO) verabschiedet (ISO 2000b). Es handelt sich um eine Beschreibungsform der Semantik unter Zuhilfenahme von Netzstrukturen, wobei eine gewisse Ähnlichkeit zu RDF auffällt.[15]

*Ansatz: Buchindex*

Wir können die Grundidee hinter Topic Maps gut nachvollziehen, wenn wir den Index eines Buches betrachten, der den Inhalt des Buches näher beschreibt. Er vermittelt einen Gesamteindruck und stellt die Struktur des Buches auf eine andere Art und Weise dar. Die im Index enthaltenen Begriffe sind sowohl untereinander (via »siehe auch«) als auch mit dem Inhalt des Buches selbst verknüpft (Seiten, in denen der Begriff vorkommt).

*Sprache der Topic Maps*

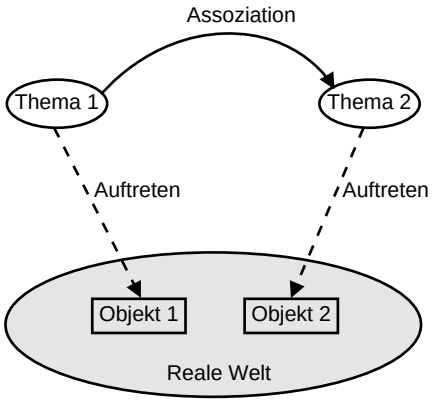
In der Sprache der Topic Maps bezeichnet man einen Begriff als Thema (engl. *topic*), die Beziehungen zwischen Begriffen als Assoziationen (engl. *associations*) und das Auftreten eines Begriffs an einer konkreten Stelle als *occurence*. Abbildung 6.5 zeigt eine schematische Darstellung einer Topic Map.

*Thema*

Ein Thema kann prinzipiell alles sein, über das wir reden können, es handelt sich also um einen sehr abstrakten Begriff ähnlich einer RDF-Ressource. Themen können durch Thematypen (engl.

[15] Historisch ähneln Topic Maps sehr den konzeptuellen Graphen (engl. *conceptual graphs*), bei denen jedoch die *occurences* fehlen.

**Abbildung 6.5**
*Topic Map-Modell*




*topic types*) gruppiert werden. Ein Thematyp ist dabei wieder ein Thema. Ein Thema hat prinzipiell drei Arten von Eigenschaften: einen oder mehrere Namen, sein Auftreten (occurences) und Rollen bei Assoziationen.

*Auftreten*

Ein Auftreten (occurence) verknüpft ein Thema mit einer oder mehreren Ressourcen, für die es relevant ist. Diese Ressourcen können beispielsweise Dokumente oder Dokumentteile sein, die sich (semantisch) mit dem verknüpften Thema beschäftigen. Occurences verknüpfen also eine Topic Map mit der realen Welt. Ein Auftreten selbst hat wieder einen Typ – ein Thema in der Topic Map, ähnlich wie bei den Themen selbst.

*Assoziation*

Assoziationen verknüpfen verschiedene Themen untereinander, sie drücken also Beziehungen zwischen den einzelnen Themen aus. Die in einer Assoziation vorkommenden Themen spielen jeweils eine bestimmte Rolle. Assoziationen können ähnlich wie Themen und Auftreten wieder durch so genannte Assoziationstypen gruppiert werden.

*Beispiel*

An dieser Stelle wollen wir die Tatsache, dass unser Buch den Namen »Langzeitarchivierung« hat, einmal mit Hilfe von Topic Maps modellieren. Wir benötigen dazu zwei Themen *Buch* und *Name*, eine Assoziation *heißt* und die Verknüpfungen der beiden Themen mit ihrem konkreten Auftreten (vgl. Abb. 6.6).

*Synonyme*

Nun kann es durchaus vorkommen, dass verschiedene Themen synonym sind, also den gleichen Sachverhalt beschreiben. Um die Identität zwischen verschiedenen Themen auszudrücken, gibt es das Identitätsattribut, das im Wesentlichen eine spezielle Assoziation darstellt (semantische Gleichheit oder auch Äquivalenz). Es wird aus einer Menge identischer Themen ein Repräsentant gesucht, der den Sachverhalt am besten beschreibt. Die anderen

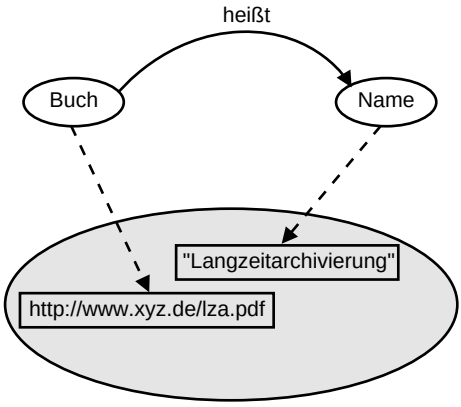



**Abbildung 6.6**
*Topic Map-Beispiel*

Themen erhalten ein Identitätsattribut, das ihre Äquivalenz mit dem Repräsentanten ausdrückt. Somit bieten Topic Maps bereits einfache ontologische Möglichkeiten – wir werden auf Ontologien im nächsten Abschnitt näher eingehen.

*XML Topic Maps*

Auch für Topic Maps gibt es eine serialisierte Syntax (XTM 2001). Der aufmerksame Leser wird den hierfür verwendeten Standard unschwer erraten können – es handelt sich um XML. Wir wollen hier jedoch nicht näher auf diese Syntax eingehen, da eine eingehende Behandlung den Rahmen dieses Buches sprengen würde.

*Topic Maps-Schema?*

Angesichts der RDF-Schemata drängt sich auch bei Topic Maps die Frage nach Regeln für das Erstellen von semantisch korrekten Topic Maps auf. Ein Schema-Standard für Topic Maps ist jedoch bis zur Drucklegung dieses Buches noch nicht verabschiedet worden, was wohl auch daran liegen dürfte, dass es sich bei Topic Maps um einen noch recht jungen Standard handelt. Es wird jedoch an einer Schema-Sprache gearbeitet, die sich in den Grundzügen an den RDF-Schemata orientiert.

*Topic Maps vs. RDF*

Wenn wir die Standards RDF und Topic Maps vergleichen, fällt auf, dass sie sich sowohl in der Zielsetzung als auch in der technischen Umsetzung sehr ähneln. RDF ist weitaus allgemeiner, es stellt ja auch »nur« ein Rahmenwerk dar. Tatsächlich lassen sich die wesentlichen Topic Map-Elemente auch in RDF ausdrücken, jedoch bleibt die in Topic Map »fest eingebaute« Semantik bei einer naiven Übersetzung auf der Strecke. Die semantischen Unterschiede beispielsweise zwischen Themen und deren Auftreten müssen in einer RDF-Umsetzung von Topic Map klar herausgestellt werden. Weiterhin schränkt die feste semantische Bindung der Begriffe Thema, Auftreten und Assoziation Topic Maps ge-

genüber RDF ein. Umgekehrt, haben Topic Maps aber einige Spezialitäten, die RDF fremd sind – eine eingehende Behandlung finden wir z. B. bei Widhalm und Mück (2002).

*Anwendung: Metadaten, semantische Suche*

Wir können Topic Maps ebenso wie RDF dazu verwenden, Dokumente mit Metadaten zu versehen oder die Semantik von Metadaten zu definieren. Weiterhin ergeben sich zahlreiche Nutzungsmöglichkeiten auf dem Gebiet der semantischen Suche. Die Relevanz einer semantischen Suche für die Langzeitarchivierung ergibt sich schon aus der einfachen Anforderung, dass wir archivierte Dokumente irgendwann einmal *wiederfinden* möchten – und das anhand ihres Inhalts, also semantischer Kriterien. Ein globaler Archiv-RDF-Graph oder eine globale Archiv-Topic Map könnte beispielsweise helfen, ähnliche Dokumente zu einem einmal archivierten oder neuen Dokument zu finden. In diesem Sinne könnten RDF und Topic Maps auch bei der Auswahl und Kategorisierung zu archivierender Dokumente helfen.

### 6.2.3 Ontologien: DAML+OIL

*Beziehungen zwischen Begriffen*

Der Begriff der *Ontologie* stammt ursprünglich aus der Philosophie und bezeichnet die Lehre vom Seienden. In der Informationstechnik bezeichnet eine Ontologie die Spezifikation einer Konzeptualisierung – was zunächst nicht weniger abstrakt klingt. Tatsächlich können wir Ontologien benutzen, um Beziehungen zwischen Begriffen zu definieren, die über die Möglichkeiten von RDF und Topic Maps hinausgehen. So werden z. B. äquivalente Begriffe verschiedener natürlicher Sprachen definiert oder Beziehungen näher charakterisiert.

*DAML+OIL*

Für Ontologien gibt es wieder eine Vielzahl von Modellierungssprachen. Hier sei nur auf DAML+OIL,[16] eine Erweiterung von RDF, eingegangen. DAML+OIL erweitert RDF und RDF-Schema um einige Modellierungsarten, die es erlauben, Beziehungen zwischen Klassen zu definieren, die über einfache Teilmengenbeziehungen hinausgehen.

*Beispiel*

Ein Beispiel hierfür kommt aus der Ahnenforschung: Ein Mensch kann entweder ein Mann oder eine Frau sein. Männer sind keine Frauen und umgekehrt. In der englischen Sprache werden statt Mensch, Frau und Mann die Begriffe human, man und woman verwendet, die die gleiche semantische Bedeutung haben

[16] Es handelt sich hierbei um den Zusammenschluss der beiden Sprachen *Darpa Agent Markup Language* (DAML) und *Ontology Inference Layer* (OIL).

wie ihre deutschen Äquivalente. In DAML+OIL codieren wir diese Sachverhalte wie folgt:[17]

```
<rdf:RDF
  xmlns:rdf="http://.../rdf-syntax-ns#">
  xmlns:rdfs="http://.../rdf-schema#">
  xmlns:daml="http://.../daml+oil#">

 <daml:Class rdf:ID="Mensch">
  <rdfs:comment>Ein Mensch ist entweder ein Mann
                oder eine Frau.</rdfs:comment>
  <daml:disjointUnionOf
        rdf:parseType="daml:collection">
   <daml:Class rdf:about="#Mann"/>
   <daml:Class rdf:about="#Frau"/>
  </daml:disjointUnionOf>
  <daml:sameClassAs rdf:resource="#Human"/>
 </daml:Class>

 <daml:Class rdf:ID="Mann">
  <rdfs:comment>Ein Mann ist keine Frau.
                </rdfs:comment>
  <daml:complementOf rdf:resource="#Frau"/>
  <daml:sameClassAs rdf:resource="#Man"/>
 </daml:Class>

 <daml:Class rdf:ID="Frau">
  <rdfs:comment>Eine Frau ist kein Mann.
                </rdfs:comment>
  <daml:complementOf rdf:resource="#Mann"/>
  <daml:sameClassAs rdf:resource="#Woman"/>
 </daml:Class>
</rdf:RDF>
```

Wir könnten diese Beispielontologie um Beziehungen erweitern, etwa um festzulegen, dass ein Mensch genau ein Elternpaar hat, bestehend aus einem Mann und einer Frau.

[17] Den Namensraum `daml` haben wir der derzeitigen Definition von DAML+OIL zugewiesen. Eine Vorabversion findet man unter http://www.w3.org/TR/daml+oil-reference. Analog verfahren wir mit den Namensräumen `rdf` und `rdfs`.

## 6.3 Zukunft – Semantic Web

*Vision nützliches Internet*

Beim *Semantic Web* handelt es sich (noch) um eine Vision des Begründers des Internets und Chefs des W3C, Tim Berners Lee. Unter der Nutzung der in Abschnitt 6.2 beschriebenen Möglichkeiten, Dokumente semantisch zu annotieren, soll das Internet endlich richtig nützlich werden – es soll uns »verstehen«. Der Kern dieser Vision besteht darin, dass der Nutzer mit genau den Informationen versorgt wird, die *er* für nützlich erachtet. Ellenlange Suchergebnislisten, in denen die vielleicht 10 relevanten Ergebnisse wild verstreut und damit quasi unauffindbar sind, sollen der Vergangenheit angehören.

*Voraussetzung: semantisches Markup*

Dafür benötigen wir eine semantische Auszeichnung aller Dokumente des Internets – keine geringe Aufgabe! Weiterhin helfen intelligente Suchmaschinen und Agenten bei der Selektion der relevanten Dokumente. Das folgende Szenario gibt uns einen kleinen Einblick in die Möglichkeiten des Semantic Web:

*Beispiel*

Es ist Sonntag, wir stöbern im Online-Katalog einer Buchhandlung. Unsere Anfrage ergibt etwa 20 Bücher, die uns interessieren, gruppiert nach privaten und beruflichen Interessen. Wir können die gefundenen Bücher nun nach ihrer Relevanz bzgl. unserer Interessen sortieren, oder aber verschiedene Agenturen abfragen lassen, die eine Bewertung unserer Bücher vorgenommen haben. Diese Agenturen haben sich zuvor anhand einer Art Ausweis bei uns identifiziert – umgekehrt findet dieser Prozess auch statt, schließlich müssen solche Dienste ja bezahlt werden. Da wir gerade etwas knapp bei Kasse sind, möchten wir uns einige Bücher ausleihen. Wir erhalten auf Anfrage eine Liste derjenigen Bibliotheken, die bei uns zu Hause in der Nähe sind oder aber sich auf dem Weg zur Arbeit befinden. Die Öffnungszeiten der Bibliotheken lassen wir bequem mit unserem Terminkalender abgleichen. Es stellt sich heraus, dass wir am Mittwoch ohnehin ein wenig früher gehen wollten, da kann ein Sprung in die Bibliothek nicht schaden. Es könnte allerdings sein, dass wir deshalb ein wenig später zum Sport kommen – nun ja, unsere Mannschaftskameraden sind in Windeseile informiert und werden es schon nicht so genau nehmen. Schlimmer wäre es, falls wir einen Termin mit unserem Chef hätten, der wäre mit Sicherheit weniger tolerant. Das Ganze hat uns in etwa fünf Minuten gekostet, da der Vorgang komplett automatisch ablief. Jetzt fehlt nur noch unser »ok«.

Mehr zum Semantic Web und einer möglichen Realisierung mit RDF ist bei Hjelm (2001) nachzulesen.

# 7 Diskussion

*In den vorangegangenen Kapiteln haben wir die Problematik der Langzeitarchivierung digitaler Dokumente und die wichtigsten organisatorischen und technischen Lösungsansätze dazu in Grundzügen beschrieben. Hier sollen nun Konsequenzen für die praktische Umsetzung angesprochen werden, die unseres Erachtens daraus zu ziehen sind. Da die wissenschaftliche Diskussion noch im Gange ist, kann es sich nur um Überlegungen und Lösungsansätze handeln, die beim derzeitigen Wissensstand wichtig und plausibel erscheinen.*

*Abschnitt 7.1 erläutert, warum so bald wie möglich Institutionen etabliert werden müssen, welche die Aufgaben der Langzeitarchivierung systematisch und nachhaltig übernehmen können. In Abschnitt 7.2 fassen wir zusammen, was als gesichert gelten kann und wo noch Klärungsbedarf besteht. Um die Grenzen des sinnvoll Machbaren geht es in Abschnitt 7.3. Der Abschnitt 7.4 schließlich skizziert, wie die wichtigsten vorgestellten Techniken zur Langzeitarchivierung so kombiniert werden können, dass sie einander mit ihren jeweiligen Vorzügen ergänzen.*

## 7.1 Von der Notwendigkeit jetzt zu handeln

*Migrieren oder emulieren oder ...*

Leider ist die zentrale Frage nach der langfristig erfolgversprechendsten Archivierungsstrategie noch nicht entschieden: Es ist unklar, ob sich Migration oder Emulation (oder eine dritte Strategie) durchsetzen wird. Zu beiden Strategien lassen sich gewichtige Argumente für und wider nennen. Die wissenschaftliche Meinungsbildung ist im Gange und wird dem Stand der Diskussion nach wohl noch einige Zeit benötigen.

*... besser abwarten?*

Diese Situation legt es nahe zu warten, bis die Diskussion abgeschlossen ist, um unnötige Investitionen und Aufwand zu vermeiden. Genau dies kann man sich wegen der Eigentümlichkeit des Problems aber nicht mehr lange leisten. Laufend entstehen neue elektronische Formate, von denen sich die meisten zusam-

men mit ihren Abspielumgebungen (aus Hard- und Software) nur kurzzeitig halten, bis sie von neuen Formaten verdrängt oder abgelöst werden. In den alten Formaten gespeicherte Informationen sind dann nur mehr eingeschränkt zugänglich: Kurze Zeit können sie noch auf veraltetem Gerät betrachtet oder – wo das nicht zur Verfügung steht – von Spezialisten mit großem Aufwand rekonstruiert werden; mit dem Nachlassen der Magnetisierung oder beim physischen Verfall des Speichermediums sind sie endgültig verloren.

*Beispiele für Datenverluste*

Erfahrungen von Archiven, die schon länger mit digitalen Dokumenten arbeiten, belegen, dass diese Gefahr real ist. Ein bekanntes Beispiel sind die Satellitenbilder der NASA aus den 1970er Jahren, die über die Entwicklung des brasilianischen Regenwalds Auskunft geben könnten, wenn sie in ihrer elektronischen Form nicht verloren gegangen wären. Ein zweites Beispiel betrifft die Daten der amerikanischen Volkszählung von 1960. Als man sich Mitte der 1970er Jahre der Bedeutung dieser Daten bewusst wurde, waren sie nicht mehr direkt zugänglich. Es kostete viel Aufwand und dauerte drei Jahre, die Daten (mit Ausnahme von 10.000 endgültig verlorenen Datensätzen) zu rekonstruieren.

*Kulturgut* jetzt *bewahren*

Erheblich größer noch sind die Probleme digitaler Bibliotheken, deren Aufgabe es ist, digitale Dokumente aus verschiedensten Quellen und von unterschiedlichsten Formaten langfristig zu erhalten. Da in zunehmendem Maße Kulturgüter in digitaler Form produziert werden (u. a. Audio- und Videoaufnahmen, Hypertext- und Multimediadokumente), laufen wir Gefahr, schon bald Teile des zeitgenössischen Kulturguts endgültig zu verlieren, wenn nicht umgehend Vorsorge für deren dauerhafte Bewahrung getroffen wird. »Abwarten« ist hier also keine ratsame Lösung. Außerdem wird sich möglicherweise erst im praktischen Gebrauch herausstellen, welche die beste Strategie zur langfristigen Archivierung ist.

## 7.2 Was wir wissen und was zu klären bleibt

*Erfahrungen der Informatiker*

Seit Beginn des Computerzeitalters gab und gibt es viele konkurrierende und einander im stetigen Wechsel ablösende Hard- und Softwaresysteme. Informatiker leben also seit jeher mit dem Problem, ihre Daten und Programme zwischen Computern mit unterschiedlichen Formaten möglichst verlustfrei zu übertragen. Die von ihnen dafür entwickelten Konversions- und Portierungstechniken bilden die technische Basis der Langzeitarchivierungsstra-

tegien Migration und Emulation. Auch die folgenden Aussagen beruhen auf langjährigen Erfahrungen aus der Informatik. Als einigermaßen gesichert kann demnach gelten:

- Unbedingt zu meiden sind *proprietäre* Formate wie Microsoft Word, die von ihren Besitzern jederzeit geändert werden können und deren Details zum Teil sogar geheim gehalten werden. Für langfristig operierende Unternehmen wie digitale Bibliotheken stellt die einseitige Abhängigkeit von einem einzelnen Anbieter ein unakzeptables Risiko dar. Besser ist die Situation, wenn die genaue Beschreibung eines Formats offengelegt (so z. B. die Formate PostScript und PDF der Firma Adobe) und das Format der Allgemeinheit zum kostenlosen Gebrauch dauerhaft überlassen wird. Unbedenklich nutzbar ist ein solches Format zum Zweck der Langzeitarchivierung erst dann, wenn alle Rechte – insbesondere auch die Kontrolle über Änderungen am Format – in die Hände eines internationalen Normungsgremiums übergeben worden sind. Ebenfalls unproblematisch ist die Verwendung eines proprietären Formats als zusätzliche, bequeme Alternative, etwa das Anzeigen eines XML-Dokuments in einem Word-Editor.

  *proprietäre Formate meiden!*

- Es ist damit zu rechnen, dass der Bestand an digitalen Dokumenten nicht nur stetig, sondern sogar stark wachsen wird. Das schließt eine regelmäßige manuelle Bearbeitung – etwa zum Zweck der Migration – aus. Wichtig ist es also, dass Dokumentenformate nicht nur bis ins Detail bekannt, sondern auch gut für die *automatisierte Verarbeitung* geeignet sind. Das trifft z. B. für XML-Dokumente zu; hier gibt es sogar kostenfrei für jedermann verfügbare Transformationswerkzeuge.

  *ausschließlich automatisierte Verarbeitung*

- Markup beschreibt die Darstellung und/oder die Struktur der Dokumenteninhalte. Markup-Sprachen wie LaTeX und XML unterstützen die automatisierte Verarbeitung. Es gibt aber wesentliche Unterschiede: Während LaTeX auf die Erzeugung eines gefälligen Druckbilds ausgerichtet ist, beschreibt man mit XML explizit den Aufbau eines Dokuments, ohne sich um das Layout zu kümmern. Dazu als Beispiel die Formel $\frac{ab}{b}$ und deren LaTeX-Beschreibung `$\frac{a b}{b}$`: Man erkennt, dass Zähler und Nenner klar voneinander abgesetzt sind, aber der Multiplikationsoperator des Produkts im Zähler genauso wie im Schriftbild weggelassen wird. Dies macht (in Verbindung mit anderen Gründen) Probleme, wenn man ver-

  *explizit beschreibende Formate verwenden*

sucht, die LaTeX-Beschreibung in ein Format zu konvertieren, in dem Multiplikationsoperatoren explizit dargestellt werden. Für die Langzeitarchivierung ist der *beschreibende Ansatz* von XML daher besser geeignet.

*möglichst wenige Formate*

- Um den eigenen Arbeitsaufwand zu minimieren, müssen digitale Bibliotheken bestrebt sein, mit *möglichst wenig Formaten* auszukommen. Autoren, die ihre Werke für die Nachwelt erhalten sehen möchten, werden bereit sein, ihr Dokument in einem gängigen Standardformat zu produzieren (oder es nachträglich zu konvertieren). Bei dieser Arbeitsverteilung ist auch die Authentizität des eingereichten digitalen Dokuments gesichert. Voraussetzung ist allerdings, dass für die ausgewählten Formate einfach benutzbare Editoren und Werkzeuge zur Verfügung stehen.

*erweiterbare Formate*

- Ein dazu gegenläufiger Trend ergibt sich aus der großen, weiter wachsenden Zahl verschiedener Dokumentenarten. Neben Textdokumenten gibt es Bild-, Audio-, Videodokumente, die mit Hypertextdokumenten zu Multimediadokumenten verbunden werden. Diese Dokumentenarten zerfallen weiter in spezialisierte Unterarten, Textdokumente z. B. in Bücher, Berichte, Briefe und viele mehr. Um nicht für jede dieser Arten ein eigenes Dokumentenformat (mit Bearbeitungswerkzeugen) bereitstellen zu müssen, sehen einige Markup-Sprachen *Erweiterungsmechanismen* vor: In LaTeX gibt es nach dem Vorbild von Programmiersprachen die Möglichkeit, Makros zu definieren; XML-Dokumente enthalten (oder verweisen auf) Beschreibungen ihrer eigenen, dem Verwendungszweck angepassten Struktur. Auch hier ist der deskriptive Ansatz von XML für die Langzeitarchivierung geeigneter.

*textbasierte Metadaten-Formate*

- Metadaten müssen jederzeit verlässlich verfügbar sein. Zur Darstellung sollte daher ein einfaches, textbasiertes Format auf der Grundlage eines etablierten Standards wie ASCII oder Unicode verwendet werden. Wieder bieten sich Markup-Sprachen aus der XML-Familie als aussichtsreiche Kandidaten an: etwa RDF (Resource Description Framework) oder Topic Maps in Verbindung mit Dublin Core.

*Trend zu Markup-Sprachen*

In der obigen Diskussion tauchen immer wieder Markup-Sprachen auf, vor allem XML und seine Co-Standards. Tatsächlich erscheinen uns diese Formate für die Langzeitarchivierung besonders geeignet zu sein – daher die ausführliche Darstellung in Kapitel 6. Derzeit bestehen hier noch Defizite im Bereich der Präsentation.

Von der angekündigten Umstellung des Microsoft Office Pakets auf XML-Formate ist jedoch eine positive Signalwirkung zu erwarten.

*Organisation gemäß OAIS und NEDLIB*

Ähnlich klar zeichnet sich mit den Standards OAIS und NEDLIB ein einheitlicher organisatorischer Rahmen ab. Der homogene Übergang von OAIS zu NEDLIB weist OAIS als eine geeignete Basis für notwendige Verfeinerungen aus. Insbesondere die in OAIS angedachten Kooperationsmöglichkeiten werden künftig vermutlich eine große Rolle spielen.

*gesellschaftliche und juristische Aspekte*

Klärungsbedarf besteht nicht nur bezüglich der geeignetsten Archivierungsstrategie, sondern auch bei den gesellschaftlichen und juristischen Problemen, die wir schon in Abschnitt 1.5 angerissen haben. Zu den rechtlichen Aspekten des verwandten Gebiets »Langzeitarchivierung in der Verwaltung« gibt es einschlägige internationale Beiträge, deren detailliertere Betrachtung den Rahmen dieses Buches sprengen würde. Zu nennen wären hier u. a. der DoD-Standard 5015.2-STD[1] oder MoReq (*model requirements for the management of electronic records*).[2] Die anderen Fragen können nur von den Beteiligten selbst geklärt werden. Hilfreich mag dabei eine Abgrenzung des aus technischer Sicht sinnvoll Machbaren sein, wie wir sie im nächsten Abschnitt vornehmen.

*DoD 5015.2-STD und MoReq*

## 7.3 Was realistischer Weise anzustreben ist

*Zeithorizont: was ist »lang«?*

Eine wichtige Frage ist die nach dem angestrebten *Zeithorizont*: In einschlägigen Publikationen findet man auf der einen Seite astrophysikalische Zeitskalen, auf der anderen Seite auch pragmatische Beschränkungen auf kürzere Zeiträume wie die nächsten 100 Jahre. Angesichts der Tatsache, dass die gesamte Entwicklung elektronischer Rechenanlagen innerhalb der letzten 60 Jahre stattgefunden hat, erscheinen konkrete Prognosen und Pläne für einen längeren Zeitraum zumindest gewagt. Andererseits geht es um die Bewahrung unseres digitalen Kulturguts auf unbestimmte Zeit. Man sollte daher einen absehbar kurzen Planungshorizont mit einer potenziell unbegrenzten Erhaltungsdauer kombinieren und von einer Aufgabe sprechen, die »von Generation zu Generation weitergegeben« wird. Während technische Neuerungen wie in der Informationstechnik üblich zügig einfließen können, sollten grundlegende Entscheidungen wie die für die verwendete Ar-

[1] siehe http://jitc.fhu.disa.mil/recmgt/p50152s2.pdf.

[2] siehe http://www.cornwell.co.uk/moreq.pdf.

chivierungsstrategie über einen Zeitraum von Jahrzehnten konsequent befolgt und evaluiert werden, damit Planungssicherheit besteht und übereilte Entscheidungen vermieden werden. Eine verbindliche Planung von Archivierungskonzepten über 50 Jahre hinaus erscheint angesichts der rapiden Entwicklung der Informationstechnik nicht seriös. Was man aber sofort leisten kann und sollte ist, digitale Dokumente in aus heutiger Sicht Erhaltbarkeit versprechende Formate zu bringen und mit der systematischen Erprobung der vorgeschlagenen Archivierungsstrategien zu beginnen, damit die nächste Generation auf den gewonnenen Erfahrungen aufbauen kann.

*Alles aufheben oder selektieren?*

Neue Lösungen erfordert auch die Frage, was aufbewahrt werden kann und muss. Während früher Bücher teils über längere Zeiträume ohne Pflege auf verstaubten Dachböden überlebten, bis sie entdeckt und in Sammlungen aufgenommen wurden, gehen digitale Dokumente verloren, wenn ihre Erhaltung nicht aktiv betrieben wird. Damit stellt sich die Frage, ob ein digitales Dokument als aufhebenswert zu betrachten ist, praktisch schon kurz nach seiner Erstellung. Wegen der enorm angestiegenen und gleichzeitig immer billiger verfügbaren Speicherkapazitäten könnte man versucht sein, wenig Selektion zu betreiben und fast alles aufzubewahren. In der ersten Phase mag dies sinnvoll sein, um Erfahrungen zu sammeln. Es gibt aber gute Gründe, auf Sicht wesentlich selektiver vorzugehen und den Bestand nicht grenzenlos wachsen zu lassen:

*hohe Migrationskosten*

- Wenn zumindest teilweise Migration als Erhaltungsstrategie eingesetzt wird, dann steigen die regelmäßig anfallenden Aufwände für Konversion und – noch wesentlicher – für die nicht automatisierbare Qualitätssicherung proportional zum Datenumfang an. Das ist besonders kritisch, da es sich um laufende Kosten handelt.

*»Nadel im Heuhaufen« finden*

- Mit dem Umfang des Bestands wächst ebenso der für Erschließungsmaßnahmen erforderliche Aufwand. Wie man derzeit an der Entwicklung des Internet beobachten kann, wird es in einem wachsenden, nicht systematisch erschlossenen Datenbestand immer schwieriger, die sprichwörtliche »Nadel im Heuhaufen« zu finden.

*Ausfallsicherheit erfordert Redundanz*

- Eine digitale Bibliothek ist ein komplexes, seiner Natur nach auf dauerhaften, störungsfreien Betrieb angewiesenes System. In Krisenzeiten notwendige Sparmaßnahmen bedrohen die Qualität der Bestandspflege bis hin zur vollständigen Vernichtung des Bestands. Eine gewisse Ausfallsicherheit ist

vermutlich nur durch redundante Speicherung zu erreichen. Das wiederum wird die Kosten abhängig vom Umfang der Daten in die Höhe treiben.

*Abgrenzung zum Dokumentenkontext*

Eine weitere Frage betrifft die *Abgrenzung* eines digitalen Dokuments zu seiner Umgebung. »Klassische« Textdateien sind ebenso in sich abgeschlossen wie Bücher oder Tonträger herkömmlicher Art. Auch Multimedia-Dokumente sind häufig vollständig in dem Sinn, dass sie zusammen mit ihren verschiedenen Bestandteilen gespeichert werden. Hinweise auf Internetstellen, wie sie z. B. in diesem Buch gegeben werden, kann man als zeitlich befristet nutzbare Anmerkungen auffassen, deren Inhalt deshalb aber nicht zum Buch gehört. Schwieriger wird die Abgrenzung bei Hypertextdokumenten im Internet: Wo genau ist hier die Grenze? Oder wenn ein digitaler Dokumentarbericht eine E-Mail-Korrespondenz zwischen agierenden Personen der Zeitgeschichte enthält – was macht man dann mit angehängten digitalen Dokumenten oder mit eingefügten Internet-Links? Handelt es sich um einen Bericht über wirtschaftliche Zusammenhänge, dann könnte eines der angefügten digitalen Dokumente eine Spreadsheet-Tabelle sein, deren Inhalt sich womöglich aus dem Inhalt einer online verfügbaren Datenbank berechnet. In Abschnitt 5.4 haben wir Bemühungen dargestellt, Internetbezüge und Inhalte dauerhaft verfügbar zu halten. Ein konservativeres Vorgehen bestünde darin, beim Einbringen eines digitalen Dokuments in die Bibliothek gleichzeitig aktuelle Fassungen referenzierter externer Dokumentteile »einzufrieren« und zusammen mit dem Dokument zu speichern.

*erforderliches Kontextwissen*

Wenn man an Archivierung über sehr lange Zeiträume denkt, dann stellt sich auch die Frage nach dem *Kontextwissen*, das benötigt wird, um die gespeicherten Dokumente überhaupt interpretieren zu können. Das Kontextwissen betrifft die gemeinsame kulturelle Basis zwischen Autor und Leser, also Fakten der Zeitgeschichte ebenso wie die Details der Lebensumstände (Berufe, Essen, Kleidung, Wohnen etc.) und das aktuelle Sprachverständnis. Da sich dieses Kontextwissen fast unmerklich langsam verändert und eine Weile aus Zeitdokumenten gut rekonstruierbar ist, können wir die Lösung dieses Problems getrost aufschieben.

*gestaffelte Speicherungsqualität*

Eine Frage, die man bewusst angehen sollte, weil sie das Volumen der gespeicherten Daten erheblich beeinflusst, betrifft die *Qualität*, in der digitale Dokumente aufbewahrt werden. Digitale Fassungen wichtiger historischer Dokumente wie der amerikani-

schen Verfassung oder der Emser Depesche würde man sicher in der besten verfügbaren Faksimile-Qualität speichern wollen. Bei anderen Dokumenten aber wie dem Abdruck einer E-Mail-Korrespondenz wird es in der Regel nicht auf die äußere Form, sondern nur auf den übermittelten Text ankommen (die Darstellung einer E-Mail überlassen wir ja dem Mail-Programm des Empfängers). In solchen Fällen genügt es, auch nur den reinen Text zu speichern (gegebenenfalls mit einfachen Auszeichnungen wie Fettdruck). Dadurch lässt sich nicht nur ein großer Teil des benötigten Speicherplatzes sparen, sondern auch Migrationsschritte auf Dauer wesentlich leichter durchführen. Aus diesen Gründen lohnt sich der Aufwand, eingescannte Dokumente mit OCR-Software und manueller Kontrolle in ein textuelles Format umzuwandeln. Bei digital entstandenen Dokumenten entfällt dieser Zusatzaufwand; aber auch hier sollte bewusst ein passendes Format für die Langfristspeicherung gewählt werden. Die Akzeptanz für ein solches Vorgehen lässt sich bestimmt verbessern, wenn man berechtigten Anforderungen nach Authentizität an den Stellen und in dem Umfang entgegenkommt, in dem sie benötigt wird. Dazu kann man eine Palette von standardisierten Erhaltungsformaten anbieten, die bei Textdokumenten zwischen originalgetreuer Faksimiledarstellung und reiner Texterhaltung Zwischenstufen vorsieht, bei denen wesentliche Textattribute erhalten bleiben.

## 7.4 Ein kombinierter Ansatz

*Bitstrom fehlerfrei erhalten*

Angenommen, ein heute erstelltes digitales Dokument soll zu einem Zeitpunkt betrachtet werden, zu dem keines der heute dafür verwendbaren Abspielsysteme (aus Hard- und Software) mehr verfügbar ist. Dann muss der Bitstrom des Dokuments fehlerfrei erhalten sein – entweder der Bitstrom des Originals oder der Bitstrom eines migrierten Abbilds. Selbst kleine Fehler gefährden die Lesbarkeit des gesamten Dokuments. Dieser sehr strengen Anforderung wird man wahrscheinlich durch automatisierte Kopier- und Qualitätskontrollmaßnahmen auch auf Dauer entsprechen können.

*der Emulationsansatz*

Um ein im Originalzustand erhaltenes Dokument in ferner Zukunft betrachten zu können, benötigt man eine neue Abspielumgebung, die das erhaltene Dokument wie in der ursprünglichen Abspielumgebung darstellt. Der Emulationsansatz verfolgt das Ziel, solche Abspielumgebungen auf künftigen Rechnern systematisch bereitzustellen, z. B. mit Hilfe so genannter virtueller Maschi-

nen. Die *Authentizität* der so in die Zukunft transportierten Dokumente ist nicht zu übertreffen. Auch bezüglich des *Erhaltungsaufwands* hat dieses Vorgehen Vorteile: Auf künftige Rechnertypen müssen nur die Emulatoren portiert werden; die Dokumente selbst müssen dabei nicht verändert werden. Kritiker bezweifeln allerdings, dass der Emulationsansatz praktisch durchführbar ist. Aber selbst wenn das gelingt, gibt es noch Probleme: Eine per Emulation erhaltene Abspielumgebung wäre per Definition genauso zu bedienen wie heute, also z. B. per Tastatur und Maus über die heute üblichen graphischen Bedienschnittstellen. Es darf mit Fug und Recht bezweifelt werden, dass Menschen des 24. Jahrhunderts mit unseren dann vermutlich archaischen Artefakten umgehen können.

*der Migrationsansatz*

Die Vor- und Nachteile des Migrationsansatzes sind zu denen des Emulationsansatzes *komplementär*: Migrierte Dokumente befinden sich stets in einer auf aktuellen Plattformen mit aktuellen Programmen verwendbaren Form und stehen damit für alle Arten der Weiterbearbeitung zur Verfügung. Nachteilig ist hier, dass sich bei Migration unbemerkt kleine Veränderungen einschleichen können. Bei einer größeren Zahl solcher Schritte würde sich fast zwangsläufig Authentizitätsverlust ergeben. Da die Zielformate künftiger Migrationsschritte nicht erahnbar sind, ist es schwer, solche Schritte systematisch zu planen. Und bei jedem solchen Schritt fällt erheblicher Aufwand an: Geeignete Transformationswerkzeuge entwickeln, jedes einzelne Dokument dieses Formats bearbeiten und anschließend mit (manuellen!) Stichproben die Güte des Ergebnisses sicherstellen.

*Standarddatenformate*

Ideal für die Langzeitarchivierung wären (wie schon dargestellt) standardisierte Markup-Formate zu jedem Dokumententyp. Die Hoffnung, mit solchen Formaten auszukommen, ist relativ gering: Zum einen widerspräche eine derartige Normung den wirtschaftlichen Interessen der miteinander konkurrierenden Software-Anbieter. Zum anderen wird durch den andauernden Wettbewerb erst schrittweise erarbeitet, was später einmal standardisiert werden könnte. Eine vorzeitige Festlegung wäre dabei ein Innovationshindernis. In einzelnen Bereichen gibt es aber interessante Formate, die zumindest als Alternativen zu proprietären Formaten anzusehen sind, z. B. MathML für die Formeldarstellung oder XSL für Desktop-Publishing. Auf Markup-Standards wie RDF, Topic Maps und Dublin Core zur semantischen Behandlung von Metadaten haben wir bereits hingewiesen. Für das Bibliothekswesen kann sich daraus ein Standard ergeben. Die derzeit in verschiedenen Gebieten zu beobachtende Ent-

wicklung branchenspezifischer Standarddatenformate stärkt diese Hoffnung.

*ein kombinierter Ansatz*

Die Zeit drängt, keiner der beschriebenen Ansätze erfüllt alle Anforderungen, aber die Ansätze ergänzen einander sehr gut in ihren Vorzügen. Wir schlagen daher vor, das Problem der Langzeitarchivierung möglichst bald mit einem kombinierten Ansatz anzugehen, um die aktuellen digitalen Dokumente zu erhalten und um gleichzeitig die notwendigen praktischen Erfahrungen zu erwerben, auf deren Basis später eine sachlich fundierte Entscheidung möglich sein wird. Ein solcher kombinierter Ansatz umfasst folgende Bestandteile:

*Bitstrom und Metadaten sichern*

1. Bitströme werden von ihren physikalischen Trägermedien abgelöst, gleichzeitig die Emulations-Infrastruktur (virtuelle Maschine, Betriebs- und Anwendungssoftware) gespeichert und die Metadaten erfasst. Zusätzlich kann eine langfristig haltbare Sicherheitskopie von Bitstrom und Metadaten auf Mikrofilm gespeichert werden.

*Originalzustand durch Emulation erhalten*

2. Der Originalzustand wird durch Emulation langfristig unverfälscht erhalten. Dabei ist keine Verarbeitung des Dokumentenbestands nötig, wohl aber Portierung der Emulatoren auf neue Plattformen.

*Migration von Dokumenten und Metadaten*

3. Durch Migration werden grundsätzlich die Metadaten und die Beschreibung der Abspielumgebung erhalten. Da solche Daten für den laufenden Zugriff in der Datenbank des Bibliotheksverwaltungssystems gehalten werden, handelt es sich um eine *Datenbankmigration*. Durch Migration werden zusätzlich zur Erhaltung durch Emulation laufend sämtliche Dokumente in einem aktuell lesbaren Format gehalten. Außerdem werden bei Bedarf von den emulierten Dokumenten aktuelle »Gebrauchskopien« abgeleitet. Auf diese Weise ist nach einer Reihe von Migrationsschritten bzw. Portierungen der Emulationsumgebung ein aussagefähiger Vergleich beider Techniken möglich.

*standardisierte Markup-Formate*

4. Markup-Sprachen werden zur Beschreibung der Metadaten und – wo immer das geht – als Standardformate für digitale Dokumente eingesetzt, um den Aufwand in einem überschaubaren Rahmen zu halten.

# Teil II

# Einschlägige Initiativen und Projekte

# 8 Markup in der Praxis

*Im ersten Teil dieses Buches haben wir in Kapitel 6 verschiedene Dokumentenbeschreibungs-Sprachen vorgestellt. In diesem Kapitel beantworten wir schließlich die Frage: Und wer nutzt das alles nun tatsächlich und wie? Wir werden hierfür einige Projekte beschreiben, die sich mit der Annotation von Dokumenten in der Praxis beschäftigen.*

*Zunächst werden wir die wichtige Frage klären, wie Metadatensätze definiert werden können. Dazu stellen wir in Abschnitt 8.1 mit Dublin Core einen sehr weit verbreiteten Metadatenstandard vor und zeigen, wie wir ihn nutzen können, um unsere Dokumente mit standardisierten Metadaten zu versehen. Danach werden wir Projekte und Initiativen beschreiben, die sich der Archivierung digitaler Dokumente verschrieben haben. Hierbei verfolgt das VERS-Projekt (vgl. Abschnitt 8.2) einen hybriden Ansatz, bei dem die Dokumente selbst in PDF und die Metadaten in XML abgelegt werden. Wir werden hierbei nicht auf Migrationsaspekte eingehen, die wir gesondert in Abschnitt 9.1 behandeln. Die Text Encoding Initiative (vgl. Abschnitt 8.3) verfolgt einen rein XML basierten Ansatz, bei dem komplette Texte nach XML überführt werden. In Abschnitt 8.4 widmen wir uns der Arbeit der Research Libraries Group (RLG). Abschließend stellen wir mit Pandora in Abschn 8.5 ein Projekt vor, das sich ganz explizit dem Speichern und Archivieren von Hyperdokumentmengen im World Wide Web widmet.*

## 8.1 Die Dublin Core Metadata Initiative (DCMI)

*standardisierte Metadaten*

Zu einem Dokument können wir eine Reihe von Metadaten angeben, die dieses Dokument näher beschreiben, und so einen Teil seiner Semantik erfassen. Nun gibt es je nach Anwendung sehr unterschiedliche Informationen, die es wert sind, als Metadaten gespeichert zu werden. Um Metadaten austauschen zu können, ist es jedoch notwendig, ein möglichst einheitliches, aber dennoch er-

weiterbares Schema für Metadaten zu entwickeln. Dieses Ziel verfolgen eine Reihe von Initiativen, unter ihnen das Warwick Framework,[1] das Pittsburgh Project,[2] das SPIRT Metadata Project[3] oder die in diesem Abschnitt vorgestellte Dublin Core Metadata Initiative (DCMI 2002). Wir haben Dublin Core ausgewählt, da es in der Welt der digitalen Bibliotheken große Bedeutung erlangt hat.

*Dublin Core Metadata Initiative*

Die Dublin Core Metadata Initiative (DCMI) ist ein Konsortium, das sich das Ziel gesetzt hat, eine erweiterbare Basismenge von Metadaten-Elementen zusammenzustellen, die ein Dokument für die elektronische Archivierung ausreichend beschreiben können. Hierbei wurde vor allem Wert auf eine saubere Beschreibung der Basis-Elemente, Erweiterbarkeit, Plattformunabhängigkeit und Verwendung bestehender Standards gelegt. Weiterhin sind die Dublin Core-Elemente in verschiedenen Sprachen veröffentlicht, darunter Englisch, Finnisch, Deutsch, Japanisch, Französisch (DCMI 1999).

*Dublin Core-Standard (DC)*

Der Dublin Core-Standard (DC) umfasst 15 Elemente, die ein elektronisches Dokument näher beschreiben. Jedes Element ist optional und kann mehrfach vorkommen. Es handelt sich bei DC sozusagen um den kleinsten gemeinsamen Nenner, den eine Gruppe von Experten gefunden hat. Unter ihnen finden sich beispielsweise Bibliothekare, Informatiker und Historiker. Zu den DC-Elementen gehören u. a. die Elemente

*DC Elemente*

- `Title`: für den Namen eines Dokuments,
- `Creator`: für denjenigen, der das Dokument erschaffen hat,
- `Subject`: für Stichworte, die den Inhalt des Dokuments beschreiben,
- `Description`: für eine etwas ausführliche Inhaltsbeschreibung des Dokuments, z. B. ein Abstrakt,
- `Format`: für das Format, in dem das Dokument vorliegt, z. B. PDF oder XML,
- `Date`: für das Erscheinungsdatum des Dokuments oder
- `Language`: für die (natürliche) Sprache, in der das Dokument geschrieben wurde.

[1] Nähere Informationen zum Warwick Framework findet man unter http://www.dlib.org/dlib/july96/lagoze/07lagoze.html.

[2] Nähere Informationen zum Pittsburgh Project findet man unter http://www.archimuse.com/papers/nhprc/.

[3] Nähere Informationen SPIRT Metadata Project findet man unter http://rcrg.dstc.edu.au/research/spirt/.

*Beispiel*

Beispielsweise könnten wir das vorliegende Buch wie folgt mittels Dublin Core beschreiben:

- `Title`: Langzeitarchivierung
- `Creator`: Uwe Borghoff
- `Creator`: Peter Rödig
- `Creator`: Jan Scheffczyk
- `Creator`: Lothar Schmitz
- `Subject`: Langzeitarchivierung
- `Subject`: Einführung
- `Subject`: elektronische Dokumente
- `Description`: Eine etwas längere Beschreibung des Buches, die wir Ihnen hier ersparen möchten – der geneigte Leser weiß ja schon, worum es geht
- `Format`: application::PDF
- `Date`: 2003-01-01
- `Language`: ge

*kontrollierte Vokabulare*

Hier ist zu beachten, dass wir für die Beschreibung des Formats, des Datums und der Sprache nicht irgendeine Notation gewählt haben. Bei der Beschreibung des Formats haben wir uns am MIME-Standard orientiert (MIME 1996), während wir für das Datum und die Sprache jeweils einen ISO-Standard verwendet haben (ISO 1998, ISO 2000a). Das Verwenden von Standards ist besonders nützlich, da sie ein Vokabular definieren, von dem man annehmen darf, dass es weithin verstanden und akzeptiert wird. Wir werden später in diesem Abschnitt auf solche *kontrollierten Vokabulare* zurückkommen, wenn es darum geht, die Bedeutung von Dublin Core-Elementen zu verfeinern.

*Semantik von DC-Elementen*

Wir haben in den vorherigen Abschnitten erwähnt, dass Dublin Core ein Standard ist, der die Semantik von atomaren Dokumentbausteinen festlegt. Diese atomaren Bausteine sind die Dublin Core-Elemente. Die Semantik jedes einzelnen Elements ist durch zehn Attribute festgelegt. Die gebräuchlichsten davon sind:

- `Name`: der Name des Elements,
- `Registration Authority`: wer hat das Element definiert,
- `Language`: die (natürliche) Sprache, in der das Element beschrieben wird,
- `Definition`: die eigentliche Definition (Semantik) des Elements und
- `Comment`: noch ein Kommentar zur Definition und zur Anwendung des Elements.

*Beispiel* Mit den oben aufgeführten Attributen wird beispielsweise die Semantik des Elements `Title` wie folgt beschrieben, wobei wir die originale Beschreibung auf Englisch angeben.

- `Name`: Title
- `Registration Authority`: Dublin Core Metadata Initiative
- `Language`: en
- `Definition`: A name given to the resource.
- `Comment`: Typically, a Title will be a name by which the resource is formally known.

*Semantikdefinition als Fließtext?* Wir sehen also, dass die Semantik eines Elements (wieder) durch eine natürliche Sprache angegeben wird, mit den bekannten Nachteilen. Es sprechen jedoch mindestens zwei gewichtige Argumente für diese Art und Weise der Semantikbeschreibung:

1. An irgendeiner Stelle müssen wir anfangen, die Semantik von Elementen zu definieren. Ein rein formaler Ansatz auf Basis der Wahrheitswerte `Wahr` und `Falsch` wäre wohl zu kompliziert.
2. Dublin Core wurde von einer Reihe von Experten definiert, die auf die Verständlichkeit der Definitionen achten. Weiterhin wurde Dublin Core in eine Reihe weiterer (natürlicher) Sprachen übersetzt, sodass wir Dublin Core auch verstehen, wenn wir nicht Englisch sprechen können.

*Dublin Core und RDF* Die Definition der Dublin Core-Elemente lässt sich mittels eines RDF-Schema darstellen. Wir verzichten an dieser Stelle auf eine detaillierte Darstellung, da sie den Rahmen dieses Buches sprengen würde. Stattdessen wollen wir angeben, wie wir die Dublin Core-Elemente in Semantik-Beschreibungen mittels RDF nutzen können. Generell entspricht ein Dublin Core-Element einem RDF-Prädikat und der Wert des Elements einem RDF-Objekt. Das Dokument, das durch Dublin Core-Elemente beschrieben wird, entspricht einem RDF-Subjekt.

*Beispiel* Wir können nunmehr unser Buch unter Nutzung von Dublin Core gemäß des in Abb. 8.1 skizzierten RDF-Graphen beschreiben. Man beachte hier, dass wir für die Dublin Core-Elemente den Namensraum `dc` vorgesehen haben. Der Graph aus Abb. 8.1 lässt sich in XML wie folgt codieren:

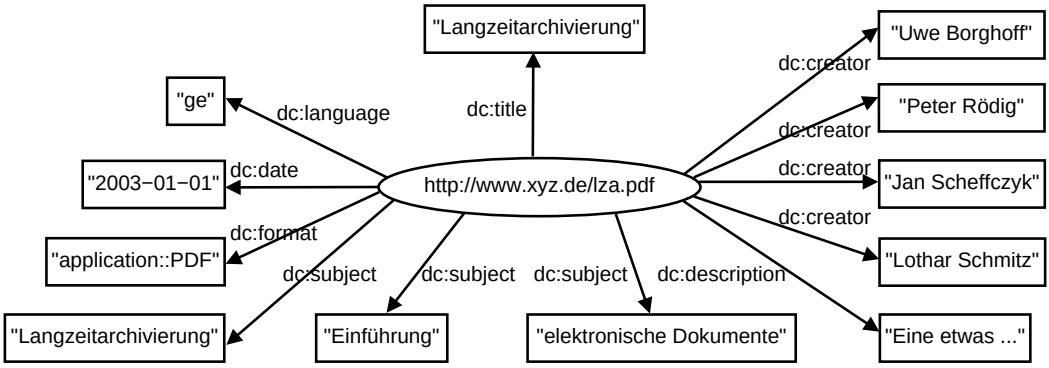



**Abbildung 8.1**
*Dublin Core als RDF-Graph*

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf=
         "http://www.w3.org/.../22-rdf-syntax-ns#"
         xmlns:dc=
         "http://purl.org/dc/elements/1.1/">
  <rdf:Description>
    <dc:title>Langzeitarchivierung</dc:title>
    <dc:creator>Uwe Borghoff</dc:creator>
    <dc:creator>Peter Rödig</dc:creator>
    <dc:creator>Jan Scheffczyk</dc:creator>
    <dc:creator>Lothar Schmitz</dc:creator>
    <dc:subject>Langzeitarchivierung</dc:subject>
    <dc:subject>Einführung</dc:subject>
    <dc:subject>elektronische Dokumente
                </dc:subject>
    <dc:description>
      Eine etwas längere Beschreibung des Buches,
      die wir Ihnen hier ersparen möchten -
      der geneigte Leser weiß ja schon,
      worum es geht.
    </dc:description>
    <dc:format>application::PDF</dc:format>
    <dc:date>2003-01-01</dc:date>
    <dc:language>ge</dc:language>
  </rdf:Description>
</rdf:RDF>
```

Für diejenigen Eigenschaften, die mehrfach verwendet wurden, lassen sich RDF-Container verwenden. In Abbildung 8.2 haben wir den Container `Bag` verwendet, der für eine Menge steht.

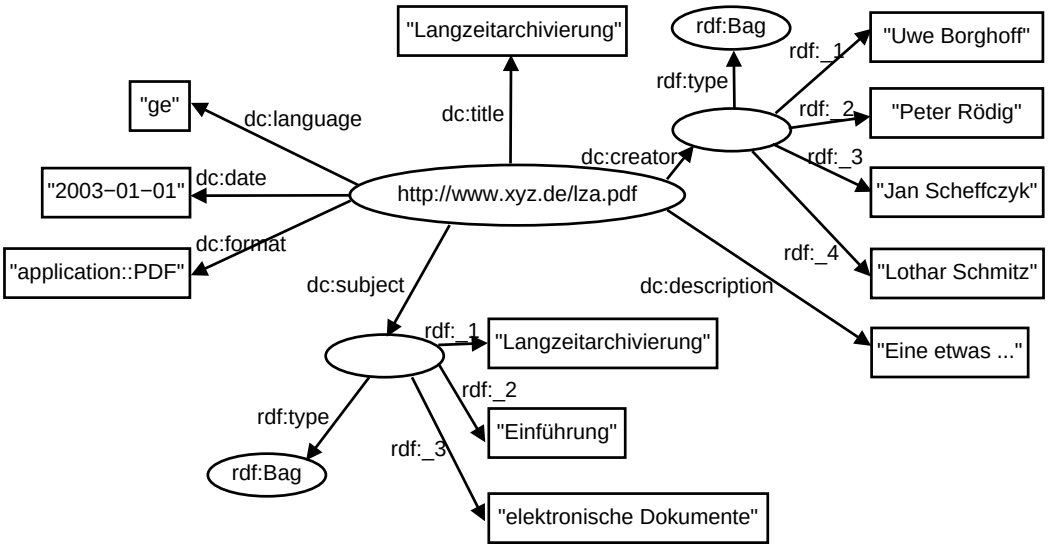



**Abbildung 8.2**
*RDF-Bags für mehrfach angegebene Dublin Core-Elemente*

*DC erweitern*

Im vorangegangenen Teil dieses Abschnitts haben wir gesehen, wie wir Dokumente mittels Dublin Core-Elementen näher beschreiben können und wie wir dies in RDF ausdrücken können. Im Folgenden wird es darum gehen, Dublin Core zu erweitern. Dies ist immer dann notwendig, wenn wir die Semantik der Dublin Core-Elemente verfeinern wollen oder ein spezielles Codierungsformat (z. B. für das Datum) vorgeben wollen.

*Qualifier*

Dublin Core sieht für die Zwecke der Elementverfeinerung und der Vorgabe eines Codierungsformats so genannte *Qualifier* vor. Zu jedem Dublin Core-Element kann es mehrere Qualifier geben. Das DCMI-Konsortium hat Vorschläge für Qualifier ausgearbeitet. Hier einige Beispiele:

*Beispiele*

- Verfeinerungen des Elements `Description`:
  `Table Of Contents`: das Inhaltsverzeichnis,
  `Abstract`: ein Abstrakt (kurze Zusammenfassung).

- Verfeinerungen des Elements `Date`:
  `Created`: wann wurde das Dokument erschaffen,
  `Modified`: wann wurde es zuletzt verändert.
- Codierungsformate des Elements `Date`:
  `W3C-DTF`: Format des W3C (entspricht dem ISO-Standard 8601),
  `DCMI-Period`: spezielles DCMI-Format zur Codierung von Zeiträumen.
- Codierungsformate des Elements `Format`:
  `IMT`: Internet Media Types – Codierung gemäß MIME-Standard.
- Codierungsformate des Elements `Language`:
  `ISO 639-2`: Codierung gemäß ISO-Standard.

*Definition von Qualifiern*

Es lassen sich natürlich auch eigene Qualifier definieren. Jeder Qualifier ist definiert durch die folgenden fünf Eigenschaften:

- `Name`: eine eindeutige Identifizierung des Qualifiers und
- `Label`: ein (allgemein verständlicherer) Name des Qualifiers,
- `Definition`: die Definition des Qualifiers als gut verständlicher Text,
- `Comment`: weitere Informationen zur Nutzung des Qualifiers,
- `See Also`: ein Link zu weiteren Informationen über den Qualifier.

*Beispiele*

Als Beispiel geben wir die Definitionen für die Qualifier `Created` und `W3C-DTF` für das Element `Date` an. Wir verwenden wieder die originalen Definitionen auf Englisch.

- Verfeinerungen des Elements `Date`:
  `Created`:
  - `Name`: created
  - `Label`: Created
  - `Definition`: Date of creation of the resource
- Codierungsformate des Elements `Date`:
  `W3C-DTF`:
  - `Name`: W3CDTF
  - `Label`: W3C-DTF
  - `Definition`: W3C Encoding rules for dates and times – a profile based on ISO 8601
  - `See Also`: http://www.w3.org/TR/NOTE-datetime

*Qualifier als RDF*

Die obigen Definitionen können wieder als RDF-Schema angegeben werden. In einem solchen RDF-Schema wird dann ein Qualifier als `rdf:type` definiert und wir können ihn in konkreten RDF-Sätzen nutzen. Auf die explizite Angabe eines RDF-Schemabeispiels verzichten wir jedoch aus Platzgründen.

*Beispiel*

Stattdessen wollen wir uns wieder auf die Nutzung der einmal definierten Dublin Core-Qualifier konzentrieren. Unser Beispiel enthält mindestens drei Ansatzpunkte für Qualifier, nämlich die Codierung des Formats, des Datums und der Sprache. Nehmen wir an, wir hätten die RDF-Typen `IMT` (für das Format), `W3CDTF` (für das Datum) und `ISO639-2` (für die natürliche Sprache) mittels RDF-Schema definiert und ihnen den Namensraum `dcterms` zugewiesen. Dann können wir die RDF-Sätze aufstellen, die in den Abb. 8.3–8.5 als Graphen dargestellt sind. Die Graphen in diesen Abbildungen können dann in den RDF-Graphen aus Abb. 8.2 eingebunden werden.

**Abbildung 8.3**
*Formatangabe gemäß MIME-Standard*

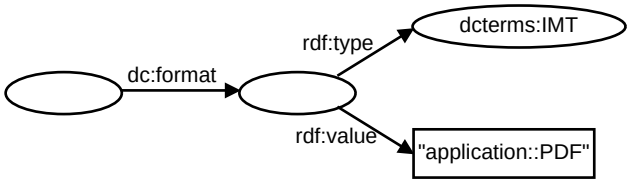



**Abbildung 8.4**
*Datumsangabe gemäß W3C-DTF-Standard*

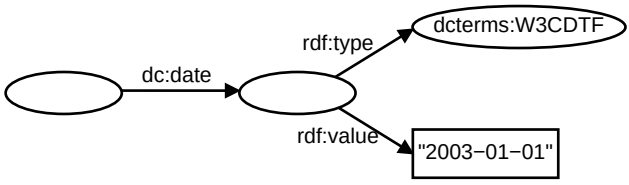
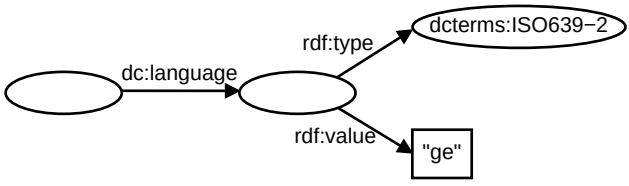



**Abbildung 8.5**
*Sprachangabe gemäß ISO 639-2-Standard*




*Wer nutzt DC?*

Wir können in diesem Buch nicht auf alle Projekte und Institutionen eingehen, die Dublin Core einsetzen. Allein die Internetseite der DCMI[4] listet mehr als 70 Projekte auf, die Du-

[4] siehe http://dublincore.org/projects/subject.shtml.

blin Core-Elemente verwenden. Darunter sind Projekte aus verschiedensten Gebieten wie Kunst, Bibliotheken, Bildung und Erziehung, Umwelt, Mathematik, Medizin und andere Wissenschaften. Unter den Bibliotheken, die Dublin Core nutzen, finden wir u. a. den Electronic Library Image Service for Europe (ELISE II), das Ukrainian Library Association Metadata Project, die National Library of Russia (St.-Petersburg) und die Russian State Library (Moscow), die Florida International University Digital Library, die National Library of Education (USA), die Everglade Information Network & Digital Library, EULER (European Libraries and Electronic Resources in Mathematical Sciences), den Digital Library Catalog, die Francois Rabelais University Libraries, die State Library of Queensland, die Monticello Electronic Library, die National Library of the Netherlands (Koninklijke Bibliotheek), das Bibliotheksservice-Zentrum (BSZ) Baden-Württemberg (Südwestdeutscher Bibliotheksverbund), die Washington Digital Library, die australische Virtual Engineering Library (AVEL) und die Electronic Visualization Library.

## 8.2 Die Victorian Electronic Records Strategy (VERS)

In Behörden fallen eine Reihe von Dokumenten an, die elektronisch archiviert werden können und sollen. Die *Public Record Office Victoria* (PROV) ist die Archivierungsbehörde des australischen Bundesstaates Victoria. Sie hat das Projekt *Victorian Electronic Records Strategy* (VERS) ins Leben gerufen (VERS 1998).

*Ziel: Langzeitformat für elektronische Dokumente*

Ziel von VERS ist es, (behördliche) Papierdokumente in elektronische Dokumente umzuwandeln und sie in einem Langzeitformat inklusive ihrer Metadaten und Kontextinformationen abzulegen. Der Fokus des Projekts lag dabei auf der Entwicklung eben dieses Langzeitformats und einer minimalen Menge von hinzuzufügenden Metadaten unter Nutzung vorhandener Standards. Im Rahmen von VERS wurde ein Prototyp zur Archivierung behördlicher Dokumente entwickelt. Auf dessen Basis wurden dann Vorschläge für ein behördliches Archivierungssystem gemacht, die wir hier kurz vorstellen.

*Archivierung*

Ein Papierdokument wird zunächst gescannt und dann inklusive der Metadaten in einem so genannten *Record* abgelegt. Die Metadaten werden notfalls von Menschenhand hinzugefügt. Eine detaillierte Darstellung des VERS-Archivierungsprozesses enthält Abschnitt 9.1.

*VERS-Record*

Ein Record wird als abgeschlossene Einheit betrachtet, die sich gewissermaßen selbst beschreibt und so alle wichtigen Informationen enthält, die man für einen späteren Zugriff benötigt. Eine schematische Darstellung eines Record sehen wir in Abb. 8.6. Wir können uns einen VERS-Record als »Zwiebel« vorstellen, bei der jede Schale beschreibt, wie die nächste Schale aufgebaut ist. Ein VERS-Record besteht aus einer Einleitung, einem Objekt-Record und mehreren digitalen Signaturen,[5] die die Echtheit des Record bescheinigen. Ein Objekt-Record enthält zum einen Dokumente und zum anderen Metadaten, soweit sie allen enthaltenen Dokumenten gemein sind. Ein Dokument besteht aus seinen eigenen Metadaten und kann in verschiedenen Formaten vorliegen. Ein Format enthält wiederum Metadaten, die den Formataufbau beschreiben, und (endlich) die Inhalte, die ursprünglich zu archivieren waren.

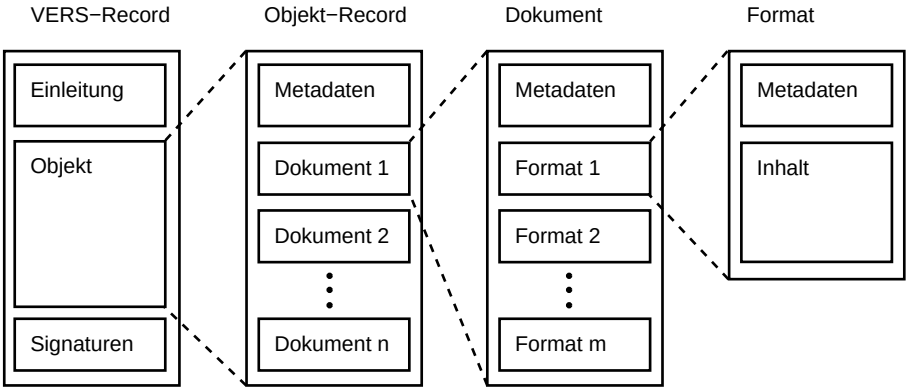



**Abbildung 8.6**
*Schematische Darstellung eines VERS-Record*

*Dokumentenformate*

Einfache Dokumente können in verschiedenen Formaten vorliegen wie Word, einfacher Text oder verschiedene Bildformate. Als Standardformat für einfache Dokumente schlägt VERS aufgrund seiner Flexibilität und weiten Verbreitung PDF vor. Für komplexere Dokumente, beispielsweise Datenbanken, wird XML vorgeschlagen. Im Rahmen von VERS wurde eine spezielle XML DTD entwickelt, mit deren Hilfe XML-Dokumente erstellt werden können, die Datenbanktabellen nachbilden.

[5] Digitale Signaturen werfen eine eigene Problematik bei der Langzeitarchivierung auf, die wir in diesem Buch nicht erschöpfend bearbeiten können.

*Codierung des VERS-Record*

Ein VERS-Dokument sowie seine übergeordneten Records werden komplett in XML dargestellt. Die eigentlichen Dokumenteninhalte werden in die XML-Datei (textuell) eingefügt. Da es sich bei PDF um ein Binärformat handelt, müssen PDF-Dateien zunächst in ein Textformat konvertiert werden. Dieses Verfahren ist jedoch nicht weiter schwierig und ähnelt der Codierung von binären E-Mail-Anhängen. Wenn wir das Zwiebelmodell abermals strapazieren wollen, wird jede »Zwiebelschale« in XML dargestellt. Für das »Zwiebelinnere« wird PDF bei einfachen Dokumenten bzw. wiederum XML bei komplexen Objekten wie Datenbanken verwendet.

*Beispiel*

Im Folgenden werden wir einen VERS-Record im XML-Format am Beispiel unseres Buches beschreiben. Der Aufbau der XML-Datei entspricht im Wesentlichen der schematischen Darstellung aus Abb. 8.6. Wir werden die vier großen Blöcke nacheinander von links nach rechts behandeln. Wir haben die Record-Darstellung an vielen Stellen durch XML-Kommentare abgekürzt und uns auf die wesentlichen Teile beschränkt. Insgesamt fällt jedoch auf, dass die Darstellung in XML für eine strukturell klare, wenn auch nicht unbedingt kompakte Darstellung sorgt.

*VERS-Record in XML*

Ein VERS-Record in XML besteht neben einer Versionsangabe aus mehreren Signaturblöcken (Element `SignatureBlock`) und einem Objekt (Element `Object`). In einem Signaturblock finden wir beispielsweise Angaben zum verwendeten Signaturformat und die eigentliche Signatur. Das Objekt seinerseits besteht aus einer Einleitung (Element `ObjectPreamble`), die etwa Informationen zum Objekttyp, seinem Sinn und Zweck, seinem Erstellungsdatum, oder seinem Format enthält. Der zweite Teil eines Objekts ist dann der Objekt-Record (Element `ObjectContent`).

```
<VERSRecord>
 <Version>2.0</Version>

 <SignatureBlock>
  <FormatDescription>
   Der Inhalt dieses Objekts wurde signiert nach
   dem Standard NIST FIPS-186 und verwendet einen
   1024-Bit-Schlüssel.
  </FormatDescription>

  <SignedBy>Max Mustermann</SignedBy>
  <SignersPublicKey>
    <!-- Der öffentliche Schlüssel von Max M. -->
```

```
  </SignersPublicKey>

  <Signature>
    <!-- Die eigentliche Signatur -->
  </Signature>
 </SignatureBlock>

 <Object>
  <ObjectPreamble>
   <!-- Typ, Zweck, Erstellungsdatum, Format -->
  </ObjectPreamble>

  <ObjectContent>
   <Record>
    <!-- Der Objekt-Record -->
   </Record>
  </ObjectContent>
 </Object>
</VERSRecord>
```

*Objekt-Record in XML*

Kommen wir nun zum zweiten großen Block, dem Objekt-Record (Element `Record`). Wie schon aus Abb. 8.6 ersichtlich, besteht ein Objekt-Record aus den Record-Metadaten (Element `RecordMetadata`) und einer Reihe von Dokumenten (Element `Document`). Die Record-Metadaten gliedern sich in eine genauere Identifizierung des Record selbst (Element `HandleLayer`) und eine Beschreibung, wie sich der Record zu anderen Records verhält – seinem Kontext (Element `ContextualLayer`). Zur Identifizierung kann der Record beispielsweise mit einem eindeutigen Namen, einer Beschreibung und Stichworten versehen werden. Die Kontextbeschreibung kann auf den eindeutigen Namen eines anderen Record verweisen, etwa auf eine alte Version. In unserem Beispiel verweisen wir auf die alte Version 0.9 unseres Buches. Wir könnten aber auch auf Records verweisen, die diejenigen Bücher repräsentieren, die in unserem Literaturverzeichnis am Buchende aufgeführt sind.

```
<Record>
 <RecordMetadata>
  <HandleLayer>
   <RecordKey>
    <RecordIdentifier>LZA V1.0</RecordIdentifier>
   </RecordKey>
```

```
   <Description>
    <Name>Record: Langzeitarchivierung</Name>
    <Keywords>
     Langzeitarchivierung, Einführung,
     elektronische Dokumente
    </Keywords>
   </Description>
  </HandleLayer>

  <ContextualLayer>
   <Relationship>
    <RelationShipType>neuer als</RelationShipType>
    <Recordkey>LZA V0.9</RecordKey>
   </Relationship>
  </ContextualLayer>
 </RecordMetadata>

 <Document>
  <!-- Gekapseltes Dokument -->
 </Document>
</Record>
```

*archiviertes Dokument*

Der dritte große Block eines VERS-Record enthält das zu archivierende Dokument, codiert in möglicherweise mehreren Formaten (Element `Encoding`). Zusätzlich werden Metadaten zum Dokument selbst abgelegt (Element `DocumentMetadata`), die eine Beschreibung des Dokuments und seiner Herkunft enthalten. Die Beschreibung eines Dokuments kann mit mehreren Metadatenmodellen erfolgen. In unserem Beispiel werden wir das Modell des *Australian Government Locator Service* (AGLS) verwenden, das Dublin Core-Metadaten nutzt, die wir schon im vorherigen Abschnitt kennen gelernt haben. Wir verwenden den Namensraum `dc` für die Dublin Core-Metadaten, um Namenskonflikten aus dem Weg zu gehen.

```
<Document>
 <DocumentMetadata>
  <Description>
   <Name>Langzeitarchivierung</Name>
   <AGLSMetadata>
    <!-- Dublin Core-Metadaten -->
    <dc:title>Langzeitarchivierung</dc:title>
    <!-- siehe Abschnitt 8.1 -->
   </AGLSMetadata>
```

```
  </Description>

  <DocumentSource>
   <Text>
    Dieses Dokument wurde mit LaTeX erstellt und
    dann nach PDF konvertiert.
   </Text>
  </DocumentSource>
 </DocumentMetadata>

 <Encoding>
  <!-- Die Dokumenteninhalte in einem
       bestimmten Format -->
 </Encoding>
</Document>
```

*Format des archivierten Dokuments*

Kommen wir zum letzten großen Block, dem Format (Element `Encoding`), das die zu archivierenden Dokumenteninhalte in codierter Form enthält. Zu jedem Format gibt es neben dem eigentlichen Code (Element `ActualDocument`) auch Metadaten, die das Format beschreiben (Element `EncodingMetadata`). Die Metadaten beschreiben sowohl das Format als auch Möglichkeiten, wie dieses Format angezeigt werden kann. Der Code ist für uns Menschen unleserlich, da es sich um PDF handelt.

```
<Encoding>
 <EncodingMetadata>
  <FileEncoding>
   <Text>
    Das PDF-Dokument (Version 1.4) wurde nach dem
    Standard Base64 als Text codiert.
    Informationen hierzu gibt das
    IETF RFC 2045, Abschnitt 6.8,
    "Base 64 Content-Transfer-Encoding".
   </Text>
  </FileEncoding>
  <FileRendering>
   <Text>
    Den Prozess der Codierung einfach umkehren
    und als PDF anzeigen, z. B. mit
    Acrobat Reader(R) ab Version 5.0.
   </Text>
  </FileRendering>
 </EncodingMetadata>
```

```
 <ActualDocument>
  MDAwMDAgbiAKMDAwMDcyOTk3NCAwMDAwMCBuIAowMDAwNzMw
  <!-- den Rest wollen wir Ihnen hier ersparen -->
 </ActualDocument>
</Encoding>
```

## 8.3 Die Text Encoding Initiative (TEI)

*Ziel: Archivierung von Texten*

Die *Text Encoding Initiative* (TEI) wurde 1987 als eine internationale Initiative von Philologen gegründet. Wie der Name schon vermuten lässt, ist es das Ziel, *Texte* zu codieren und nicht beispielsweise Bilder oder Filme. Die TEI produzierte einen internationalen interdisziplinären Standard, der es ermöglichen soll, alle Arten von literarischem und linguistischem Text elektronisch in einem einheitlichen Format zu erfassen. So soll insbesondere Bibliotheken, Museen oder Verlagen geholfen werden. Die TEI stellt also nicht nur ein Dokumentenformat zur Verfügung, sondern hat auch Richtlinien erlassen, die sich mit dem gesamten Prozess des elektronischen Publizierens befassen, also der Dokumentenanalyse, der Digitalisierung, der Textauszeichnung und der Publikation (TEI 2002). Da es uns in diesem Kapitel um die Anwendung von Markup-Sprachen geht, werden wir uns vor allem auf die Richtlinien zur Textauszeichnung beschränken.

*Ablage kompletter Dokumente in XML*

Ziel der Textauszeichnung ist es, *komplette* Dokumente im TEI-Format abzulegen und mit relevanten Informationen auszuzeichnen. Dies steht im Gegensatz zum Dublin Core-Standard, der nur die Metadaten von Dokumenten erfasst und nicht die Dokumente selbst. Das bevorzugte Dokumentenformat der TEI für die Auszeichnung ist SGML bzw. XML. Dazu hat die TEI zu ihren Richtlinien mehrere XML DTDs veröffentlicht, mit deren Elementen Dokumente ausgezeichnet werden können. Die DTDs lassen sich in drei Gruppen kategorisieren:

*TEI-DTDs*

- Das *Core Tag Set* stellt sehr allgemeine Elemente bereit, die jedes TEI-Dokument enthalten kann bzw. soll.
- Verschiedene *Base Tag Sets* stellen Basis-Elemente bereit, die sich je nach Genre eines Dokuments unterscheiden. So gibt es beispielsweise die Base Tag Sets `TEI.prose` für die Auszeichnung von Prosatexten, `TEI.verse` für die Auszeichnung von Lyrik, `TEI.dictionaries` für die Auszeichnung von Wörterbüchern oder auch `TEI.mixed` zur Auszeichnung

von Texten, die in mehrere Genres fallen. Pro Dokument kann dabei nur ein Base Tag Set gewählt werden.

- Die *Additional Tag Sets* schließlich definieren eher technische Elemente, die wir zusätzlich in einem Dokument verwenden dürfen. Pro Dokument dürfen mehrere Additional Tag Sets gewählt werden, hier einige Beispiele: Im Additional Tag Set `TEI.certainty` sind Elemente enthalten, mit denen sich ausdrücken lässt, mit welcher Sicherheit eine bestimmte Information zutrifft. Das Additional Tag Set `TEI.nets` stellt Elemente für Graphen und Netze zur Verfügung, während `TEI.names.dates` Elemente zur Auszeichnung von Namen und Daten zur Verfügung stellt.

*Konvertierung nach XML mittels Formatvorlagen*

Bevor wir jedoch zur eigentlichen Dokumentenauszeichnung kommen, sollten wir uns überlegen, wie wir bereits vorliegende elektronische Dokumente nach XML konvertieren können. Die meisten Text-Dokumente liegen nicht in XML vor, sondern in einem proprietären Format wie Microsoft Word oder anderen Textverarbeitungsformaten. Glücklicherweise bieten die meisten Textverarbeitungen *Formatvorlagen* an. So generieren Textverarbeitungen Inhaltsverzeichnisse meist auf der Basis hierarchischer Formatvorlagen für Überschriften. Wir sollten Formatvorlagen nutzen, um Dokumente in der Textverarbeitung zu formatieren. Jeder Formatvorlage kann dann ein XML-Element zugeordnet werden und so steht einer automatischen Konversion nach XML nicht mehr viel im Wege. Es sei allerdings auch darauf hingewiesen, dass nur diejenigen Informationen nach XML konvertiert werden können, die schon in der Textverarbeitung vorhanden sind. Beispielsweise könnte eine einfache Leerzeile in einem Gedicht sowohl den Übergang zwischen Gedichtüberschrift zu Gedicht als auch eine neue Strophe oder gar das Ende des Gedichts bedeuten. Zeichnen wir dagegen die einzelnen Strophen mit einer Formatvorlage *Strophe* aus, so kann zumindest diese Strukturinformation in XML erhalten werden.

*TEI Lite*

Die TEI hat inzwischen über 400 Elemente definiert, von denen jedoch die meisten für ein spezielles Dokument nicht benötigt werden. Es gibt auch vereinfachte Varianten der TEI-DTDs, die unter dem Namen TEI Lite veröffentlicht sind; vgl. etwa die Ausarbeitung von Burnard und Sperberg-McQueen (2002). Die TEI Lite DTD trägt einem in der Informatik typischen Phänomen Rechnung: der »Featuritis«. Nur ein Bruchteil der zur Verfügung gestellten Möglichkeiten einer Software erbringen für 90 Prozent der Anwender 90 Prozent des angestrebten Nutzens. Dieser Bruchteil

für sich genommen ist natürlich wesentlich einfacher benutzbar und erlernbar. Der Anspruch der TEI Lite DTD ist es, die überwiegende Mehrheit der Benutzeransprüche zu erfüllen und dabei gegenüber den eigentlichen TEI-DTDs deutlich leichter handhabbar zu sein. Den beschränkten Umfang können wir auch aus der Dokumentation der TEI-DTD erkennen. Während die Dokumentation der TEI-DTDs (in der Version P4) über 1.000 Seiten umfasst, kommt die TEI Lite DTD (momentan) mit 59 Seiten aus. Wir werden uns im Folgenden auf einige Elemente der TEI Lite DTD beschränken.

*Markup-Genauigkeit*

Zunächst lässt sich feststellen, dass die TEI (Lite) DTD uns viele Freiheiten lässt, Texte zu codieren, insbesondere bezüglich der Markup-Genauigkeit und Markup-Tiefe. So können viele Elemente einfachen Prosatext enthalten oder aber durch Sub-Elemente verfeinert werden. Je nach Anwendungsgebiet werden wir also mal eine genaue und mal eine grobe Auszeichnung vornehmen.

*Aufbau eines TEI-Dokuments*

Ein TEI-Dokument besteht im Allgemeinen aus einem Kopf und seinem Inhalt. Wir können uns die Struktur eines TEI-Dokuments also wie folgt vorstellen:

```
<TEI.4>
 <teiHeader>
  <!-- Metadaten für das TEI-Dokument -->
 </teiHeader>

 <text>
  <!-- Codierung des eigentlichen Textes
       (weiter unten) -->
 </text>
</TEI.4>
```

*Dokumentkopf* `teiHeader`

Wir wollen uns zunächst auf den Kopf konzentrieren, der die Metadaten zu einem TEI-Dokument enthält und durch das Element `teiHeader` ausgezeichnet wird. Der `teiHeader` enthält viele Informationen, die wir schon aus Dublin Core oder dem VERS-Projekt kennen. Es handelt sich im Einzelnen um folgende Elemente:

- `fileDesc`: eine bibliographische Beschreibung des Textes
- `encondingDesc`: eine Codierungsbeschreibung des Textes
- `profileDesc`: eine Beschreibung nicht-bibliographischer Aspekte, wie der verwendeten (natürlichen) Sprache

- `revisionDesc`: eine Versionsgeschichte des vorliegenden Textes

Der Kopf eines TEI-Dokuments folgt also folgender Struktur.

```
<teiHeader>
 <fileDesc>
  <!-- bibliographische Beschreibung -->
 </fileDesc>

 <encodingDesc>
  <!-- Beschreibung der Codierung -->
 </encodingDesc>

 <profileDesc>
  <!-- nicht-bibliographische Aspekte -->
 </profileDesc>

 <revisionDesc>
  <!-- Versionsgeschichte -->
 </revisionDesc>
</teiHeader>
```

*bibliographische Beschreibung*

Wir wollen uns im Folgenden auf einige Aspekte der bibliographischen Beschreibung und der Beschreibung der Codierung beschränken. Die bibliographische Beschreibung enthält die üblichen Informationen:

- `titleStmt`: Titel und Autor eines Textes
- `editionStmt`: die momentan vorliegende Edition
- `publicationStmt`: der Verlag, in dem der Text ursprünglich erschienen ist
- `extent`: die Größe der codierten Datei

*Beispiel*

Wir wollen hier nicht genauer auf die einzelnen Elemente eingehen, sondern uns lieber mit einem Beispiel befassen – dem vorliegenden Buch. F ür unser Buch könnten wir folgende bibliographische Informationen im `fileDesc` – Element festhalten:

```
<fileDesc>

 <titleStmt>
  <!-- Titel des Buches -->
  <title>Langzeitarchivierung</title>
  <title>Methoden zur Rettung ...</title>
```

```
  <!-- Autoren -->
  <author>Uwe Borghoff</author>
  <author>Peter Rödig</author>
  <author>Jan Scheffczyk</author>
  <author>Lothar Schmitz</author>

  <!-- Verantwortlichkeiten -->
  <respStmt>
   <!-- verantwortlich für die Codierung ist -->
   <resp>codiert von</resp>
   <name>Max Mustermann</name>
  </respStmt>
 </titleStmt>

 <!-- Auflage des Buches -->
 <editionStmt>
  <edition>1. Auflage</edition>
 </editionStmt>



 <!-- Umfang des Buchtextes -->
 <extent>
  13985 Sätze, 103 MegaBytes
  <!-- eine reine Phantasiegröße -->
 </extent>

 <publicationStmt>
  <!-- herausgebender Verlag -->
  <publisher>dpunkt.Verlag</publisher>

  <!-- Verfügbarkeitsinformationen -->
  <availability>Copyright 2003, dpunkt.Verlag
                </availability>

  <!-- Datum der Veröffentlichung in ISO 8601 -->
  <date>2003-01-01</date>
 </publicationStmt>

</fileDesc>
```

*Codierungsbeschreibung*

Die Beschreibung der Codierung des Textes im Element `encodingDesc` kann neben einer Projektbeschreibung (Element `projectDesc`) und einer Beschreibung der Vorgehensweise (Element `editorialDecl`) auch eine Liste derjenigen Elemente enthalten, die für die eigentliche Textauszeichnung verwendet wurden (Elemente `tagsDecl` und `refsDecl`). Angesichts der vielen Elemente der TEI-DTDs und der unterschiedlichen Nutzungsmöglichkeiten (mehr dazu weiter unten) kommt den Elementen `tagsDecl` und `refsDecl` eine besondere Bedeutung zu. Dabei enthält das Element `tagsDecl` Informationen darüber, welche TEI-Elemente überhaupt verwendet wurden und wie diese Elemente (zumindest grob) *dargestellt* werden sollen. Das Element `refsDecl` informiert, zu welchem Zweck welche TEI-Elemente verwendet wurden.

*Beispiel*

Hier eine Beschreibung der TEI-Codierung unseres Buches:

```
<encodingDesc>

 <projectDesc>
  Codierung als Beispiel für TEI
 </projectDesc>

 <editorialDecl>
  Die originalen LaTeX-Dateien wurden mittels
  eines Perl-Skripts nach XML (TEI) überführt.
 </editorialDecl>

 <tagsDecl>
  <!-- Das Element text kommt genau einmal vor -->
  <tagUsage gi="text" occurs="1"/>
  <!-- Das Element hi wird kursiv dargestellt -->
  <tagUsage gi="hi" render="italic">
   wird zur graphischen Unterscheidung genutzt
  </tagUsage>
 </tagsDecl>

 <refsDecl>
  Es gibt nicht mehr als drei Hierarchiestufen in
  der Textgliederung, die durch die Elemente div1,
  div2 und div3 ausgezeichnet sind.
 </refsDecl>

</encodingDesc>
```

Der Einfachheit halber haben wir innerhalb der beiden Elemente `editorialDecl` und `refsDecl` Prosatext verwendet. Die TEI-DTD (nicht TEI Lite) stellt hierfür auch *wesentlich* formalere Beschreibungsmethoden zur Verfügung.

*Dokumenteninhalt*
`text`

Wir wollen uns nun der Auszeichnung des Dokumenteninhalts zuwenden. Für diesen Zweck stellt TEI das Element `text` zur Verfügung, in dem wir neben dem eigentlichen Dokumenteninhalt (Element `body`) auch Informationen zu Titelseite oder Vorwort (Element `front`) sowie zu Anhängen (Element `back`) auszeichnen können.

```
<text>
 <front>
  <!-- Titelseite, Vorwort, Inhaltsverzeichnis -->
 </front>

 <body>
  <!-- der eigentliche Dokumenttext -->
 </body>

 <back>
  <!-- Anhänge, Glossar, Index, Bibliographie -->
 </back>
</text>
```

Wir wollen uns im Folgenden auf den eigentlichen Dokumenteninhalt konzentrieren und insbesondere auf die Möglichkeiten zur Auszeichnung der Textgliederung und der Textdarstellung eingehen.

*Dokumentgliederung*

TEI stellt sehr allgemeine Gliederungs-Elemente zur Verfügung, die die Form `divn` haben, wobei `n` die Gliederungsebene darstellt (eine Zahl zwischen 0 und 7). Die Gliederungsebene kann weggelassen werden, wenn wir an einer Gliederungshierarchie nicht interessiert sind. Über das Attribut `type` wird gegebenenfalls näher spezifiziert, um welche Art Gliederungsstufe es sich handelt. In einem Gedicht werden wir sicherlich Werte wie `Strophe` oder `Vers` verwenden. Bei einer Auszeichnung von Goethes »Leiden des jungen Werthers« werden wir vermutlich `Buch` und `Brief` bevorzugen. Für unser Buch dagegen bieten sich eher `Teil`, `Kapitel` und `Abschnitt` an. Jedes `divn` Element enthält üblicherweise ein Element `head` für den Titel oder die Überschrift und weitere Elemente für den Inhalt.

*Beispiel*

Für unser Buch könnten wir also folgende Gliederung aufstellen:[6]

```
<body>
 <div1 type="Teil">
  <head>Methodischer Überblick</head>
  ...
 </div1>
 <div1>
  <head>Einschlägige Initiativen und ...</head>
  ...
  <div2 type="Kapitel">
   <head>Markup in der Praxis</head>
   ...
   <div3 type="Abschnitt">
    <head>Die Text Encoding Initiative (TEI)</head>
    <!-- In diesem Abschnitt befinden wir uns
         momentan -->
    <!-- Der erste Absatz -->
    <p>Die Text Encoding Initiative (TEI) ...</p>

    <!-- Der zweite Absatz -->
    <p>Ziel der Textauszeichnung ist es, ...</p>
   </div3>
  </div2>
 </div1>
</body>
```

Wie das Beispiel zeigt, gibt es zur Feingliederung in einzelne Absätze das Element `p`.

*Spezialität: Lyrik*

Da die TEI-DTDs ursprünglich von Philologen entwickelt wurden, finden wir viele Möglichkeiten, Gedichte auszuzeichnen. Beispielsweise können einzelne Verszeilen mittels `l` ausgezeichnet werden oder direkte Rede mittels `sp`, wobei der Sprecher selbst mit dem Element `speaker` ausgezeichnet wird. Es gibt noch weitaus mehr Elemente in der TEI (Lite) DTD, um einzelne Textpassagen auszuzeichnen. Hier können wir jedoch nicht alle vorstellen.

*Darstellungsaspekte*

Stattdessen wollen wir noch kurz darauf eingehen, wie die *Darstellung* des Dokumenteninhalts ausgezeichnet werden kann. Der aufmerksame Leser wird hier aufhorchen. War es nicht eine der

[6] Es reicht aus, das `type`-Attribut lediglich einmal zu verwenden, da davon auszugehen ist, dass dieselbe Gliederungsebene innerhalb eines Dokuments immer den gleichen Typ hat.

Grundideen von XML, Inhalt und Darstellung zu trennen? Für die Langzeitarchivierung müssen wir diesen Anspruch dann hintenanstellen, wenn die Darstellung für die Archivierung wichtig ist. Eine Möglichkeit, die Darstellung eines Elements zu beeinflussen, haben wir bereits im Dokumentkopf `teiHeader` kennen gelernt. Mit dem Element `tagUsage` können wir u. a. festlegen, wie bestimmte Elemente dargestellt werden sollen. Für eine individuelle Festlegung der Darstellung können wir das Attribut `rend` verwenden, das in jedem Element erlaubt ist. Als Beispiel wollen wir den ersten Absatz dieses Abschnitts auszeichnen, um ihn selbst fett und die Abkürzung TEI kursiv (unter Verwendung des Elements `hi`) darzustellen:

```
<p rend="bold">
 Die Text Encoding Initiative (<hi>TEI</hi>) ...
</p>
```

*zusätzliche Hierarchie: physische Umbrüche*

Die Darstellung eines Textes umfasst jedoch nicht nur die Darstellung von einzelnen Wörtern oder Absätzen, sondern auch Seiten- oder Zeilenumbrüche. Seiten definieren innerhalb eines Dokuments sogar eine *zusätzliche* (physische) Hierarchie neben der logischen Gliederung. Eine hierarchische Darstellung der Seitenumbrüche verbietet sich in XML angesichts der Tatsache, dass ein XML-Dokument eine baumartige Struktur hat und so nur genau *eine* Hierarchie zulässt. Wollen wir Seitenumbrüche zusätzlich zur logischen Gliederung auszeichnen, müssen wir auf *leere* XML-Elemente zurückgreifen, die bei der Strukturprüfung eines XML-Dokuments nicht von Bedeutung sind. Genau diesen Ansatz verfolgen die TEI-DTDs und sehen die leeren Elemente `pb` für einen Seitenumbruch und `lb` für einen Zeilenumbruch vor.

*XML vs. PDF*

Insgesamt fällt auf, dass der Darstellungsaspekt eines Textes bei einer Codierung in der TEI-DTD bei weitem nicht so detailliert und genau ist, als wenn wir denselben Text in einem PDF-Dokument ablegen würden. Es können beispielsweise keine Schriftarten eingebunden werden im Unterschied zu PDF. Dafür jedoch kommt der strukturelle Aspekt bei einem XML-Format wesentlich besser zur Geltung (mit in Abschnitt 6.1.4 angesprochenen Vorteilen).

*Wer nutzt TEI-DTDs?*

Wie schon bei Dublin Core können wir nicht auf alle Projekte eingehen, die die TEI-DTDs nutzen. Die Internetseite der TEI[7] listet (momentan) über 90 Projekte auf. Darunter sind Institutionen und Projekte aus verschiedensten Gebieten, beispielsweise

[7] siehe http://www.tei-c.org/Applications/index-subj.html.

Geschichte, Wörterbucherstellung, Bildung, viele historische Einrichtungen, Sprachwissenschaften, Literaturwissenschaften und Religion. Unter den Bibliotheken, die die TEI-DTDs nutzen, finden wir u. a. die American Theological Library Association, die Library of Congress, die Bodleian Library, die digital Library of Southern Literature, die Internet Library of Early Journals, die Legacy Tobacco Documents Library, die Miguel de Cervantes Digital Library und die Indiana University Library Electronic Text Resources (LETRS).

*TEI vs. Dublin Core*

Wenn wir die TEI-DTDs mit Dublin Core vergleichen, fällt zunächst der unterschiedliche Ansatzpunkt auf. Geht es bei Dublin Core um die Auszeichnung von *beliebigen* Dokumenten mit externen Metadaten, so konzentriert sich die TEI auf die Codierung von kompletten *Text*-Dokumenten mittels eingebettetem Markup. Es gibt jedoch noch einen sehr wichtigen Unterschied bezüglich der Definition der *Semantik* der einzelnen Elemente. Dublin Core-Elemente sind nach einem festen Schema definiert, das wir auch in einem RDF-Schema darstellen können. Dublin Core-Elemente eignen sich somit sehr gut als atomare Basis-Elemente für die semantische Dokumentenbeschreibung. TEI-Elemente hingegen sind lediglich durch den Fließtext innerhalb der DTD-Dokumentation beschrieben. Diese Beschreibung ist bei weitem nicht so formal und widerspruchsfrei wie die Beschreibung der Dublin Core-Elemente. Dennoch gibt es auch innerhalb der TEI Bestrebungen, RDF oder Topic Maps in der Zukunft zu nutzen. Allerdings wird die Semantikbeschreibung der einzelnen TEI-Elemente schon aufgrund ihrer Anzahl ungleich aufwändiger als bei Dublin Core.

*TEI-Consortium*

Im Dezember 2000 wurde das so genannte TEI-Consortium gegründet. Es handelt sich hierbei um eine nichtkommerzielle Organisation, die die Arbeit der drei Jahre früher begonnenen internationalen Initiative der Philologen fortführen soll. Das Hauptziel des TEI-Consortiums ist es, möglichst viele Nutzer anzusprechen und den TEI-Standard zu verbreiten. Weiterhin wird der Standard kontinuierlich verbessert und an neue technische Gegebenheiten angepasst, z. B. an die Nutzung von RDF-Schema zur Semantikdefinition der Elemente oder von anderen XML-Dialekten (wie SVG) zur Auszeichnung von Nicht-Text-Dokumenten.

## 8.4 Die Research Libraries Group (RLG)

In diesem Abschnitt gehen wir auf die einschlägigen Programme und Projekte der Research Libraries Group (RLG) ein. Wir stellen einzelne Aktivitäten und die Zukunftsziele der RLG vor.

*Was ist die RLG?*

Die Research Libraries Group[8] ist eine nichtkommerzielle Vereinigung mit dem Ziel, die Zugänglichkeit zu Informationen zu verbessern, die wichtig für Forschung und Lehre sind. Sie wurde 1974 gegründet. Bereits 1975 – mit den Universitäten von Columbia, Harvard und Yale sowie der New York Public Library als den tragenden Mitgliedsinstitutionen – war die RLG ein Vorreiter in der Lösungsentwicklung für Probleme, die Forschungssammlungen und ihre Benutzer bezüglich Erfassung, Auslieferung und Erhaltung elektronischer Materialien haben. Heute ist die RLG eine internationale Vereinigung von Universitäten und Colleges, nationalen und öffentlichen Büchereien, Archiven, historischen Gesellschaften, Museen und unabhängigen Forschungssammlungen.

*Primary Sources Archive*

In Ergänzung zu den zwei Langzeitprogrammen *Shared Ressources (Shares)* und *Preservation (Preserv)* bearbeiten die Mitglieder eigene Projekte, um Dienstleistungen in einigen Schlüsselgebieten zu verbessern. Andere Hauptgebiete sind so genannte *Primary Sources*-Archive, spezielle Sammlungen, Manuskripte, seltene Bücher, Museumsobjekte, Kunstwerke und andere seltene oder einzigartige kulturelle Artefakte.

*Eureka, Zephyr und RLIN*

Um die Aktionen ihrer Mitglieder zu unterstützen, hat die RLG Datenbanken und Online-Dienste eingeführt, die auch von Nichtmitgliedern aus aller Welt genutzt werden können. Für diese wurden spezielle Schnittstellen entwickelt: Eureka für den campusweiten Zugang von Studenten und Fakultäten; Zephyr, ein Z39.50-Server, der Benutzern von anderen Systemen den Zugriff und die Suche ermöglicht, und das so genannte *Research Library Information Network* (kurz RLIN).

*Ziele der RLG*

Seit 1996 haben sich dabei folgende Ziele für die RLG herauskristallisiert:

1. Die RLG will in der Forschung u. a. folgende Fragen klären: Wie soll sie gelenkt bzw. geführt werden? Welche Dienste zur Unterstützung der Forschung von Institutionen werden angeboten? In welcher Umgebung wird die Forschung gelenkt bzw. geführt?
2. Die RLG will Dienste bereitstellen zum Zusammenschluss globaler Datenbestände.

[8] siehe http://www.rlg.org/rlg.html.

3. Die RLG will Bibliotheksdienste bereitstellen zur gemeinsamen Benutzung der zusammengeschlossenen globalen Datenbestände.

Darauf aufbauend sind für die nächsten Jahrzehnte drei Schlüsselinitiativen entstanden, die wir nachfolgend kurz vorstellen werden.

## Das Shares-Programm

Diese erste Initiative ermöglicht Institutionen, ihre eigenen Sammlungen mit Materialien zu erweitern, die anderen RLG-Mitgliedern gehören. Dabei gibt es zwei wichtige Aspekte: (i) Zugriff auf solche Partnersammlungen und (ii) Kosten der Zustellung von Dokumenten.

*Task Force on International Interlibrary Lending*

Die Ausweitung des Zugriffs auf internationale Ressourcen erfordert ein Umdenken bei der gemeinsamen Ressourcennutzung. Hierum kümmert sich die *Task Force on International Interlibrary Lending*. Strategien zur technischen Weiterentwicklung werden von der *Advisory Group on Resource Sharing Technical Strategy* ausgearbeitet.

*MARC-AMC*

Die RLG hat großes Interesse, mit Mitgliedern zusammenzuarbeiten, die Sondersammelgebiete besitzen und diese Sammlungen für das Online-Angebot freigeben. Hierfür wurde ein spezielles Markup-Format für die Online-Beschreibungen eingeführt, das MARC-AMC. Ein Beispiel für ein derartig mit Metadaten versehenes Sondersammelgebiet ist *Marriage, Women and the Law 1815-1914*. Es gelang damit, eine einzigartige Sammlung zu diesem Thema zu erstellen (dabei wurden auch über 200.000 Bilder erfasst) und diese für viele wissenschaftliche Arbeiten nutzbar zu machen. Es wurden wichtige Erfahrungen in der Auswahl, der Digitalisierung und der Schaffung derartiger Sammlungen gemacht. Man hat dabei u. a. gelernt, eine organisatorische und technische Umgebung zu schaffen, die die Langzeitzugänglichkeit und Erhaltung sicherstellt.

Die RLG bemüht sich auch, Communities – also Benutzergruppen mit gemeinsamen Interessengebieten – zusammenzuführen. Nachfolgend geben wir einige Beispiele für solche Communities.

*East Asian Studies Community*

- Die RLG arbeitet seit über 20 Jahren mit Bibliothekaren und Spezialisten aus Ostasien zusammen. Für die *East Asian Studies Community*[9] wurde vor allem der Online-Zugriff

[9] siehe http://www.rlg.org/eas/index.html.

auf ostasiatische Materialien verbessert, die in der Originalsprache katalogisiert wurden. Die wichtigste Errungenschaft dieser Zusammenarbeit war jedoch die Einführung eines einheitlichen Zeichensatzes, der all diejenigen Zeichen darstellen kann, die in China, Japan, Korea und Taiwan benutzt werden. 1987 wurde der ostasiatische RLIN-Zeichensatz der nationale Standard für ostasiatische Schriften in Amerika und heißt seitdem *East Asian Character Code Set for Bibliographic Use* (EACC).

- Die *Jewish And Middle East Studies Community* (JAMES)[10] schuf eine international zugängliche RLIN-Datenbank, die Tausende von Katalogaufzeichnungen enthält. Diese enthalten Beschreibungen, die in den Originalsprachen abgefasst sind: Arabisch und Hebräisch. Seit 1988 unterstützt die RLG Schriften für Hebräisch, Jüdisch, Ladino und andere jüdische Sprachen. 1992 wurde die Unterstützung um die arabischen Schriften erweitert, also Arabisch und Persisch. Außerdem hat die RLG wichtige Beiträge geleistet bei der Implementierung so genannter Rechts-nach-links-Schriften.

*Jewish And Middle East Studies Community*

- Die Teilnehmer der *Jewish Studies Community* beschäftigen sich mit jüdischen Studien, die mit Geschichte, Kultur, Text, Religion, Sprache und Literatur, Geographie, Ökonomie und Politik zusammenhängen. Momentan liegt der Schwerpunkt auf der Katalogisierung von Material in hebräischer Schrift.

*Jewish Studies Community*

- Bei den Aktivitäten der *Middle East Studies Community* liegt der Schwerpunkt auf dem Gebiet des mittleren Ostens und Nordafrikas. Ergebnisse dieser Arbeit sind internationale Standards für arabische und erweiterte arabische Zeichensätze, nationale Standards bei der Katalogisierung, Beiträge zu einer internationalen Datenbank und – in Zusammenarbeit mit Buchverlagen – die Ergänzung von RLIN um wichtige Aufzeichnungen.

*Middle East Studies Community*

## Das Preserv-Programm

Diese zweite Initiative umfasst Projekte, die Institutionen bei ihren Bemühungen unterstützen, gefährdete Forschungsmaterialien zu erhalten und den Zugriff auf sie zu verbessern. Das Programm hat mittlerweile rund hundert Mitglieder. Der Informationsaustausch basiert im Wesentlichen auf elektronischen Diskussionsforen, zu denen alle Mitglieder Zugang haben.

[10] siehe http://www.rlg.org./james/index.html.

*zunächst Mikrofilmtechnologie*

Informationserhaltung war und ist ein Hauptziel der RLG seit ihrer Gründung im Jahre 1974. Ungefähr 20 Jahre lang lag das Hauptaugenmerk auf der Mikrofilmtechnologie. In dieser beträchtlichen Zeitspanne wurden vielfältige Erfahrungen gemacht und dokumentiert. Dazu gehören die technischen Richtlinien sowie Verfahrensrichtlinien für die Produktion von stabilen, langlebigen Mikrofilmen, die Schaffung von Negativen (Mikrofilm) von über 145.000 Bänden und 1.200 Sammlungen, die Einrichtung und Verwaltung einer Lagermöglichkeit für Negative, die Schulung in der Verwaltung von Mikrofilmprojekten zur Informationserhaltung und Workshops für die notwendige Qualitätskontrolle sowie die Veröffentlichung von Richtlinien für Mikrofilmaufnahmen von Büchern, Serien, Archiven und Manuskripten.

*jetzt digitale Archivierung*

Mittlerweile hat sich die Strategie etwas gewandelt. Die Schwerpunkte liegen nun u. a. bei der digitalen Archivierung, der digitalen Bildverarbeitung und der Erhaltung mittels magnetischer Medien. In der Zukunft werden folgende Ziele verfolgt:

*Benutzung digitaler Medien*

- *Benutzung digitaler Medien als eine Erhaltungsstrategie:* Es werden Strategien für die permanente Instandhaltung und Speicherung von zu erhaltenden digitalen Dateien entwickelt; dazu gehört auch eine Zusammenarbeit mit der *Commission on Preservation and Access* (CPA) und anderen wichtigen Partnern. Die RLG analysiert Anforderungen an die Erhaltung digitaler Bilder (Einführung von Richtlinien für Bildbearbeitung, Entwicklung von Modellworkflows etc.), bildet Administratoren aus und sorgt für die Erhaltung von Sammlungen durch das Anlegen digitaler Kopien.

*Problembewusstsein*

- *Schaffung des Problembewusstseins bei der Erhaltung magnetischer Medien:* Die RLG erforscht und dokumentiert Anforderungen an die Erhaltung magnetischer Medien. Administratoren werden entsprechend geschult.

*internationale Zusammenarbeit*

- *Internationale Zusammenarbeit bei der Erhaltung:* Die eigenen Aktivitäten sollen weltweit bekannt gemacht werden, um Wiederholungsaufwand zu vermeiden, Aktivitäten voranzutreiben und die Zusammenarbeit zu fördern. Die Zusammenarbeit bei der Forschung wird durch eine gemeinsame Forschungs- und Testagenda gefördert. Eine generelle Durchsetzung von internationalen Standards und Protokollen wird angestrebt.

## Die Art and Architecture Group (AAG)

*Was ist die AAG?*

Im Jahre 1979 haben sich zehn amerikanische Universitäten und Museumsbibliotheken zur ersten RLG *Art and Architecture Group* (AAG)[11] zusammengeschlossen. Heute engagieren sich weltweit über 80 Institutionen mit ihren Erfahrungen, ihren Ideen und ihrer Mitarbeit in dieser Gruppe. Der Gesamtumfang der zusammengeschlossenen Sammlungen und das Streben aller AAG-Mitglieder nach gemeinschaftlichen Lösungen hat die RLG-Datenbanken zu einer einzigartigen Quelle für kunstbezogene, bibliographische Forschung gemacht. Innerhalb der Gruppe gibt es viele Konvertierungsprojekte, um andere an ihren Katalogen online teilhaben zu lassen. Weiterhin werden Markup-Standards für Kunstausstellungskataloge entwickelt. Auch die koordinierte Entwicklung von Sammlungen und den damit verbundenen Metadaten-Anforderungen an einen Katalog, die ins Monographic Series Project mündete, gehören zu den einschlägigen Aktivitäten. Ebenfalls wurde SCIPIO erschaffen. Hierbei handelt es sich um eine Datenbank, die bibliographische Aufzeichnungen der Verkäufe von Kunstgegenständen und seltenen Büchern nachvollziehbar macht.

*AAG Steering Committee*

Die Gruppe wird durch das *Art and Architecture Group Steering Committee* (AAGSC) geleitet, das einen Handlungsplan mit den Eingaben der Teilnehmer herausgibt. Die Mitglieder tauschen ihre Erfahrungen durch ein elektronisches Diskussionsforum aus und treffen sich einmal im Jahr. Die Gruppe arbeitet an zwei Projekten.

- Das *Monographic Series Cooperative Cataloging Project*[12] ist ein typisches Markup-Projekt. Es geht um die kooperative Katalogisierung von Publikationen.

  *Monographic Series Cooperative Cataloging Project*

- Das zweite Projekt zielt auf Museen ab.[13] Hier stützt man sich auf die Erfahrungen, die man im Projekt *Record Export for Art and Cultural Heritage* (REACH) Jahre zuvor gewonnen hat. Ferner fließen Erfahrungen aus dem Projekt *Visual Resources Sharing Information Online Network* (VISION) mit ein – eine Zusammenarbeit von RLG, *Visual Resources Association (VRA)* und dem Getty Information Institute, die zum Ziel hatte, standardisierte Metadaten und kontrollierte Vokabulare für Bilder von Kunstwerken zu schaffen.

  *VISION*

[11] siehe http://www.rlg.org/aag/index.html.
[12] siehe http://www.rlg.org/aag/aagmono.html.
[13] siehe http://www.rlg.org/strat/projmusres.html.

Zunächst hat man versucht herauszufinden, ob man Informationen über Museumsobjekte aus Verwaltungssystemen herauslösen und für die Forschung nutzbar machen kann. Heute erstellt man digitale Bilder von Kunstgegenständen. Zu diesen Rohdaten werden Metadaten hinzugefügt, die es Studenten, Wissenschaftlern und Unterrichtenden erleichtern, gezielt auf das digitale Material zuzugreifen. Ein sehr aufwändiges Unterfangen!

*AAG-Datenbanken*

Aus der Arbeit der AAG sind mehrere Datenbanken hervorgegangen. Dazu gehören u. a. die *Art Museum Image Consortium* (AMICO) Library, das schon erwähnte SCIPIO, die *Bibliography of the History of Art* sowie der *Avery Index for Architectural Periodicals*.

*Zusammenarbeit mit der Digital Library Federation*

Die Hauptarbeit der RLG bis 2003 ist und bleibt die Hilfestellung für die Partnerinstitutionen bei der Erfüllung ihrer Mission. Die Hilfestellung umfasst die Entwicklung eines Modells eines digitalen Archivs, das insbesondere auf die Bedürfnisse von RLG-Mitgliedern angepasst werden soll. Dabei werden Auswahl-, Ablage-, Speicherungs-, Erhaltungs- und Zugangskomponenten untersucht und entsprechende Richtlinien geschaffen. Weiterhin sollen bestimmte Auswahlmechanismen und -werkzeuge gefunden und internationale Verbindungen zwischen digitalen Ressourcen ausgebaut werden. Deshalb arbeitet die RLG hier mit anderen Organisationen zusammen, z. B. mit der *Digital Library Federation*.

## 8.5 Pandora

*Pandora und NLA*

Schon sehr früh rückte der Wunsch, ganze Bereiche des World Wide Web schnappschussartig zu archivieren, ins Blickfeld der Forscher. Um sich den Problemen, die eine derartige Langzeitarchivierung mit sich bringen würde, zu widmen, wurde bereits 1996 das Projekt Pandora von der *National Library of Australia* (NLA) initiiert. Das Ziel dieses Projekts war und ist die Entwicklung und Umsetzung von Verfahren zur Eingabe und Archivierung ausgewählter Online-Publikationen Australiens sowie die Sicherstellung des Langzeitzugriffs auf diese elektronischen Dokumente. Der Name Pandora steht für *Preserving and Accessing Networked Documentary Resources of Australia.*

*Ergebnisse von Pandora*

Bis heute liegen folgende Ergebnisse vor, von denen die wichtigsten anschließend ausführlicher behandelt werden.[14]

[14] siehe http://pandora.nla.gov.au/pandora/.

- *Auswahlrichtlinien für Archivdokumente:* Schon Anfang 1996 – noch bevor Pandora offiziell startete – wurden durch das Auswahlkomitee für australische Online-Publikationen (*Selection Committee on Online Australian Publications* – SCOAP) Auswahlrichtlinien für zu archivierende Dokumente festgelegt und veröffentlicht.
- *Einrichtung der so genannten Electronic Unit:* Diese Abteilung mit Spezialisten der National Library of Australia wurde eingerichtet, um sich der Entwicklung von Verfahren zur Langzeitarchivierung und deren Umsetzung zu widmen. Außerdem ist es ihre Aufgabe, die Auswahlrichtlinien bei der Archivierung der australische Online-Publikationen im Pandora-Archiv zu berücksichtigen.
  Diese Abteilung wirkte entscheidend an der Entwicklung einer Datenbank zur Metadatenspeicherung mit.
- *Spezifikation des allgemeinen Funktionsmodells:* Aus einem Dienstmodell entstand 1997 ein allgemeines Funktionsmodell. An diesem sollte sich die zukünftige Entwicklung des Pandora-Archivs orientieren.
  In diesem Modell wurden bereits zentrale Komponenten eines funktionierenden komplexen Archivsystems analysiert und weiter spezifiziert.
- *Spezifikation des logischen Datenmodells:* Auf das Funktionsmodell aufbauend entstand ein Datenmodell zur Administration des Pandora-Archivs.
- *Einbettung in das Digital Services Projekt:* Das Digital Services Projekt wurde 1998 mit der Absicht ins Leben gerufen, eine technische Infrastruktur zu schaffen, die in der Lage sein würde, traditionelle Dokumente mit Hilfe der Digitaltechnik verfügbar zu machen. Andererseits sollte sie auch weiterhin die Langzeitarchivierung und den Zugriff auf diejenigen Dokumente ermöglichen, die nur in digitaler Form vorliegen.
- *Nationale Verknüpfung und internationale Zusammenarbeit:* Aus der Erkenntnis, dass die Archivierung von Dokumentenmengen im World Wide Web die Fähigkeiten einer einzelnen Institution bei weitem übersteigt, resultierten Gespräche mit der Staatsbibliothek Victoria. International besteht ein reger Informationsaustausch mit Staatsbibliotheken in den Niederlanden, Norwegen, Schweden, Finnland, Kanada und auch Deutschland.
- *Reformierung der gesetzlichen Grundlagen:* Da die Gesetzgebung in einigen Staaten Australiens vorschrieb, Publikati-

onen in eigens für bestimmte Themenbereiche vorgesehenen Bibliotheken zu archivieren, war es nötig, die entsprechenden Gesetze zu ändern. Solange die entsprechenden Änderungen nicht vorgenommen waren, wurde zwischen der Bibliothek und dem Verfasser eines Dokuments ein individueller Vertrag geschlossen, der eine Archivierung erlaubte und gleichzeitig durch Zugriffseinschränkungen die geschäftlichen Interessen des Verfassers wahren sollte, beispielsweise durch eine Frist bis zur endgültigen Freigabe des Dokuments im gesamten Verbund.

- *Persistent URLs:* Als einen Schritt auf dem Weg zur Langzeitarchivierung mit Langzeitzugriff gelang es der National Library of Australia, die schon in Abschnitt 5.4.2 besprochenen Persistent URLs zu etablieren.

## Die Auswahlrichtlinien

*SCOAP-Richtlinien*

Wie bereits erwähnt, wurden durch SCOAP-Richtlinien festgelegt, nach denen die Auswahl der zur Archivierung bestimmten Dokumente erfolgen sollte. Mit dieser Auswahl soll vor allem die Qualität der Dokumente im Archiv sichergestellt werden. Es sollen aber auch die verfügbaren Ressourcen berücksichtigt und Überschneidungen mit anderen Archiven vermieden werden. Ein wichtiges Kriterium bei der Auswahl ist die Beachtung der technischen Schwierigkeiten bei der Langzeitarchivierung. Daher gelten folgende Richtlinien:[15]

*Medien*

Dokumente können in gedruckter Form, als Online-Version und auf Datenträgern (CD, Disketten, Magnetbändern) verfügbar sein. Da die technischen Schwierigkeiten in dieser Reihenfolge zunehmen, wird der gedruckten Form der Vorzug vor Online-Versionen gegeben und diesen der Vorzug vor Datenträgern. Nur wenn eine Version signifikante zusätzliche Informationen enthält, wird diese Reihenfolge durchbrochen.

*World Wide Web*

Da Seiten im World Wide Web häufig modifiziert und abgeändert werden, aber keine komplette Historie aller Änderungen angefertigt wird, ist stets neu zu entscheiden, mit welcher Frequenz ein archiviertes Dokument des World Wide Web aktualisiert werden muss. Um für eine Auswahl zur Archivierung in Frage zu kommen, muss ein Dokument – gemäß den aufgestellten Richtlinien von SCOAP – zu einem großen Teil

[15] siehe http://www.nla.gov.au/scoap/guidelines.html.

- inhaltlich dem Thema Australien gewidmet sein oder
- von sozialer, politischer, kultureller, religiöser, wissenschaftlicher oder ökonomischer Relevanz für Australien sein und von einem australischen Autor verfasst sein oder
- von einer anerkannten nationalen Persönlichkeit verfasst sein und dabei zum internationalen Wissensschatz beitragen.

*Kriterien für die Archivierung*

Die örtliche Herkunft der Dokumente ist für die Auswahl belanglos. Die entscheidende Rolle spielt der Inhalt des Dokuments. Sehr viel Wert wird bei der Auswahl auch auf den wissenschaftlichen Gehalt und die Reputation des Verfassers gelegt. Auf diese Weise wird mit der Langzeitarchivierung auch ein Langzeitwert für die wissenschaftliche Verwendung erreicht. Neben solchen Titeln werden auch diejenigen archiviert, die für die Zukunft insofern Bedeutung haben könnten, als sie experimentell neue Technologien und Verfahrensweisen verwenden.

*Dokumentenkontext*

Ferner wird bei der Dokumentenauswahl der breitere Kontext der gesamten Seite beachtet, in dem sämtliche (auch externe Verknüpfungen) verfolgt werden. Dabei kann es vorkommen, dass externe Verknüpfungen mitarchiviert werden müssen, um diesen Kontext vollständig zu erhalten. Wo immer möglich, wird die gesamte Dokumentenumgebung auf Komponenten heruntergebrochen, die als Einzelteile den Auswahlkriterien entsprechen und getrennt archiviert werden. Ist dies nicht möglich, da sich vielleicht nur die gesamte Dokumentenumgebung in einen entsprechenden informativen Kontext einordnen lässt, wird davon natürlich Abstand genommen.

Für bestimmte Inhalte und Dokumentenarten existieren eigene Auswahlregeln. Beispiele dafür werden im Folgenden aufgelistet:

*weitere Auswahlregeln*

- Digitalisierte Materialien wie Bilder, Tonaufnahmen sowie Manuskripte werden generell nicht archiviert, da erwartet wird, dass staatlich gestützte Organisationen ihr eigenes digitales Material selber archivieren – es sei denn, (i) es existieren Zweifel über die zukünftige Verfügbarkeit der Organisation, da ihre finanzielle Zukunft unsicher ist, oder (ii) das digitalisierte Material befindet sich auf dem Server einer anderen als von einer staatlich mitfinanzierten, archivierenden Institution.
- Andere Protokolle wie FTP oder Gopher werden zwar nicht aktiv nach potenziellen Archivdokumenten durchsucht, soll-

te aber die National Library of Australia von derartigen Dokumenten unterrichtet werden, können diese auch in Pandora archiviert werden.

- Entwürfe und Arbeiten, die noch nicht abgeschlossen sind, werden generell nicht archiviert, auch wenn die Auswahlkriterien voll zutreffen. Die einzige Ausnahme bilden literarische Manuskripte.
- Newsgroups, Informationsdienste u. ä. werden nicht in das Archiv aufgenommen.
- Zeitungen und Zeitschriften, die im Prinzip nur Material veröffentlichen, das schon im Druck veröffentlicht wurde, werden nicht archiviert.
  Dagegen werden Zeitschriften, die nur im World Wide Web existieren, eventuell in Ausschnitten in das Archiv aufgenommen, je nachdem, um welchen Inhalt es sich handelt und ob die Auswahlkriterien zutreffen.

## Das allgemeine Funktionsmodell

*allgemeines Funktionsmodell*

Obwohl sich Pandora nicht direkt am OAIS-Referenzmodell orientiert, lassen sich viele Übereinstimmungen erkennen. Das allgemeine Funktionsmodell (*business process model*) entstand aus einem Dienstmodell, das zunächst die Anforderungen an das Pandora-Archiv formulierte. Das allgemeine Funktionsmodell schließlich modellierte daraus die benötigte Funktionsweise und beschreibt dabei folgende Funktionskomponenten.

*Dokumentenauswahl*

Die Dokumentenauswahl lässt sich vor allem mit dem Bereich des Erzeugens und der Dokumentenaufnahme aus OAIS vergleichen. Der erste Schritt der Selektion besteht im Auffinden potenziell zu archivierender Dokumente. Dafür verantwortlich ist ausschließlich die Electronic Unit, deren Mitglieder zu diesem Zweck regelmäßig das World Wide Web durchsuchen, Vorschläge anderer Mitarbeiter der National Library of Australia aufgreifen und Anträgen auf Zuweisung einer ISSN (International Standard Serial Number) bzw. einer CIP (Cataloguing In Publication) nachgehen.

*Prüfphasen*

Nachdem ein Dokument gefunden wurde, wird zunächst überprüft, ob diese Publikation bereits bearbeitet wurde, ob das Dokument in gedruckter oder anderer Form bereits im Bestand der National Library of Australia ist oder ob der Autor ein äquivalentes Dokument auch in gedruckter Version vorzuweisen hat. Trifft dies alles nicht zu, wird das Dokument zur Archivierung und zum Vergleich mit den Auswahlkriterien vorgemerkt. Dabei werden sämt-

liche Details aller Prüfphasen stets festgehalten, um auch andere Abteilungen und Institutionen mit den gesammelten Informationen versorgen zu können. Nachdem das Dokument anhand der Auswahlkriterien untersucht worden ist, wird es entweder sofort abgelehnt oder zur Archivierung zugelassen. Sollte das Dokument noch nicht genug Informationen enthalten, um sofort eine Entscheidung treffen zu können, wird es weiter beobachtet bzw. der Autor für weitere Informationen kontaktiert.

*Persistent URL*

Sobald ein Dokument für das Archiv zugelassen ist, wird ihm ein Persistent URL zugeteilt. Gleichzeitig wird eine interne Identifikationsnummer zugewiesen, die es den Archivaren erlaubt, schnell ein digitales Objekt aus dem Archiv zu holen. Des Weiteren werden beschreibende Metadaten gesammelt und aufgezeichnet, welche u. a. Folgendes beinhalten: Titel, Produzent, (P)URLs, Identifikatoren (wie eine ISSN), Zugriffsrestriktionen. Weitere Metadaten können dem logischen Datenmodell entnommen werden. In jedem Fall wird Kontakt zum Autor aufgenommen. Dabei werden Fragen des Copyright geklärt und Zugriffsbeschränkungen festgelegt. Jegliche Kommunikation (prinzipiell via E-Mail) wird dabei aufgezeichnet und zusätzlich zum Dokument gespeichert. Sollte an diesem Punkt kein Kontakt zum Autor aufzubauen sein, wird der momentane Stand der Bearbeitung eingefroren und diese gegebenenfalls zu einem späteren Zeitpunkt fortgesetzt. Abschließend wird das Dokument im nationalen Bibliotheksverzeichnis mit einem Auszug der Pandora-Metadaten verzeichnet, um es allgemein zugänglich zu machen.

*Benutzerprofile*

Für verschiedene Benutzertypen mit unterschiedlichen Benutzungsprofilen wurden bestimmte Restriktionen definiert. Es sollte für alle Benutzergruppen prinzipiell nur einen Zugangspunkt geben, der bereits über eine webbasierte Schnittstelle gegeben war, nämlich dem WebPAC – *Online Public Access Catalogue´s Web-based interface*. Zugriffsebenen wurden gemäß folgenden Prämissen festgelegt: Der Zugriff sollte prinzipiell kostenlos sein, standardisierten Beschränkungen (s. Tabelle 8.1) für geschützte Dokumente unterliegen und zumindest für den internen Benutzer der Bibliothek möglichst immer frei sein. Dabei werden außer der Veröffentlichung keine weiteren Dienste (Kopien, usw.) angeboten.

Auch beim Archivmanagement gibt es Ähnlichkeiten mit OAIS. Vor allem im Bereich Datenmanagement und Speicherung kann man deutliche Überschneidungen erkennen. Es gibt fünf Kategorien, in die sich das Archivmanagement aufteilen lässt:

| **Dokument** | **öffentlich** | **teilweise kommerziell geschützt** | **kommerziell geschützt** | | **gesperrt** |
|---|---|---|---|---|---|
| **Benutzer** | | **registrierte und unregistrierte Benutzer** | **nicht registriert** | **registriert** | |
| interner Benutzer | uneingeschränkter Zugriff | uneingeschränkter Zugriff | kein Zugriff für 3, 5 oder 10 Jahre | freier Zugriff | kein Zugriff |
| externer Benutzer | uneingeschränkter Zugriff | kein Zugriff für 3, 5 oder 10 Jahre | kein Zugriff für 3, 5 oder 10 Jahre | | kein Zugriff |

**Tabelle 8.1** *Ebenen des Benutzerzugriffs für verschiedene Dokumentenklassen und Benutzer*

*Ähnlichkeiten mit OAIS*

1. *Hinzufügen neuer Publikationen zum Archiv:* Hier werden vor allem Metadaten gesammelt und gespeichert sowie das Dokument in den nationalen Bibliotheksbestand aufgenommen (Eintrag in die entsprechende Datenbank). Es werden vorangegangene Schritte überprüft (beispielsweise ob der Persistent URL korrekt erzeugt wurde) und schließlich eine Startseite für das Dokument erzeugt (diese hat einen standardisierten Aufbau und wird für jedes Dokument mit individuellem Inhalt erzeugt). Abschließend wird dem Autor der neu verfügbare Link zum Archiv mitgeteilt. Sollte es beim Herunterladen von Dokumententeilen zu Fehlern kommen, wird ein Bericht für die Electronic Unit geschrieben.
2. *Ermöglichen des Zugriffs auf gespeicherte Dokumente:* Hier werden vor allem Einträge in den internen Suchmaschinen erzeugt, aber auch wichtige Teile der Abspielumgebung archiviert, die benötigt werden, um das Dokument zu betrachten (u. a. Plug-ins, Browser, Dekomprimierer). Zu dieser Kategorie gehört auch das Testen der archivierten Softwarekomponenten, eventuelle Konvertierungen von Nicht-Standard- zu Standardformaten und schließlich das Setzen der Zugriffsbeschränkungen.
3. *Aktualisieren archivierter Dokumente:* Aufgaben, die hier auf den Archivar zukommen, sind das Hinzufügen neuer Abschnitte oder Versionen des Dokuments. Es kann aber auch vorkommen, dass erneut Konvertierungen zwischen Formaten vorgenommen, Metadaten aktualisiert oder die Einstiegsseite verändert werden müssen.

4. *Modifizieren archivierter Dokumente:* Diese Kategorie ist entscheidend für die angebotene Qualität, da hier sichergestellt wird, dass alle Informationen auf dem neuesten Stand sind. Es werden u. a. Registrierfunktionen deaktiviert, Verknüpfungen aktualisiert oder ungewollte Inhalte entfernt.
5. *Löschen archivierter Dokumente:* Das Löschen von Dokumenten kann aus drei Gründen notwendig werden: (i) wenn Fehler bei der ersten Beschaffung des Dokuments auftreten, (ii) wenn rechtliche Gründe eine weitere Archivierung verbieten oder (iii) wenn eine gedruckte Version des Dokuments aufgefunden wird. Notwendige Arbeitsschritte, um den neuen Status des Dokuments zu reflektieren, umfassen dann das Entfernen aller Inhalte aus dem Pandora-Archiv, das Deaktivieren des Persistent URL sowie das Aktualisieren der Metadaten.

*reports und report requests*

Wie im OAIS-Referenzmodell vorgeschlagen, werden in Pandora sowohl Berichte (*reports*) als auch Anfragen für solche Berichte (*report requests*) realisiert. Derartige Berichte werden in allen Phasen und Komponenten der Archivierung benötigt. Beispiele sind etwa Statusberichte, Vergleichsberichte oder technische Berichte im Allgemeinen.

*Aktenverwaltung*

Bei dem Problem der Aktenverwaltung handelt es sich hauptsächlich um die schon erwähnte Archivierung der Korrespondenz mit den Autoren. Da die Speicherung jedoch nicht zusammen mit Pandora erfolgt, müssen alle Datensätze mit dem zugehörigen Dokument verknüpft werden; eine sehr fehlerträchtige Methode, zumindest was die Konsistenz- und Vollständigkeitsanforderungen betrifft.

## Das logische Datenmodell

*logisches Datenmodell*

Da das Pandora-Archiv nicht isoliert ist, sondern mit einer Reihe anderer Bibliothekseinrichtungen in Zusammenhang steht, müssen auch die Daten sehr flexibel und untereinander verknüpft sein.

Das gesamte logische Datenmodell (*logical data model*) beschreibt die Zusammenhänge zwischen allen Archivdaten als Einheit. Das Modell lässt sich in Funktionseinheiten aufgliedern, die die Funktionsweise des Archivs widerspiegeln und in Zusammenhang mit dem allgemeinen Funktionsmodell betrachtet werden sollten. Man findet dort u. a. folgende Datenstrukturen und Prozeduren (wir bleiben hier bei den ursprünglichen englischen Bezeichnungen):

*Datenstrukturen und Prozeduren*

- ❏ Dokumentenbeschreibung (descriptive attribute, descriptive attribute type);
- ❏ Verhandlungs- und Auswahlstatus (publisher, organisation, selection status, decision rule, negotiation status, event type);
- ❏ Prozeduren für das Herunterladen des Dokuments in das Archiv (capture procedure, capture procedure type, gathering frequency, gathering regime, source URL),
- ❏ Kopien- und Zugriffsmanagement (browser type, copy, copy of issue, file, file type, format, format type, issue, process, process type, publisher preferred browser),
- ❏ Copyright- und Zugriffskontrolle (access profile, access right, NLA copyright warning, publisher copyright statement, user class).

Die Einheiten des logischen Datenmodells werden, wie an folgendem Beispiel gezeigt, dokumentiert und beschrieben: Das Zugriffsprofil speichert die Zugriffsrechte für eine Benutzergruppe. Es gilt dabei:

*Zugriffsprofile*

1. Ein Zugriffsprofil muss durch genau eine Benutzergruppe klassifiziert sein;
2. ein Zugriffsprofil muss durch genau ein Zugriffsrecht klassifiziert sein;
3. ein Zugriffsprofil existiert für genau eine Kopie einer Publikation.

*Datenstrukturelemente*

Es werden folgende Datenstrukturelemente beschrieben: die Publikations-ID, die Kopie-ID, das Zugriffsrecht, die zugehörige Benutzergruppe, der aktuelle Zeitstempel, die entsprechende Gültigkeitsdauer, der Zeitstempel der letzten Überarbeitung sowie das Zugriffspasswort des Autors. Aus dieser Art der Beschreibung lassen sich die Funktionalität des Gesamtarchivs und die Anforderungen an ein kommerzielles Programm für den Einsatz in einer der Funktionseinheiten gut ablesen.

## Das Digital Services Project

*Digital Services Project*

Das Digital Services Project [16] wurde ins Leben gerufen, um in Zukunft mit Hilfe von Digitaltechnologie den Zugriff auf Dokumente in traditioneller Form zu ermöglichen, aber auch, um die Langzeitarchivierung und den Zugriff auf Dokumente zu ermöglichen,

[16] siehe http://www.nla.gov.au/dsp/.

die nur in digitaler Form vorliegen. Dieses Projekt wurde als *buy not build* ausgelegt, d. h., sämtliche Komponenten des Systems – Software wie Hardware – werden gemäß einer vorgegebenen Spezifikation eingekauft. Zu diesem Zweck wurde im Dezember 1998 eine Informationsschrift (*request for quotation*) veröffentlicht, die die Anforderungen an dieses Projekt detailliert beschreibt. Darin wird auch davon gesprochen, dass das Pandora-Archiv, so wie es heute besteht, vollständig in ein neues System, das alternative Beschaffungsprozeduren verwendet und kompatibel zum Digital Services Project ist, migriert werden soll. Des Weiteren wird es ein zusätzliches Speichersystem für digitale Objekte geben, welches vor allem die umfangreichen Audioaufnahmen der National Library of Australia aufnehmen und verwalten soll. Kritiken und Anregungen von öffentlicher Seite wurden bis Februar 1999 gesammelt und ausgewertet. Im Anschluss wurden im Juni und August desselben Jahres zwei weitere Schriften veröffentlicht, die die Anforderungen an ein Speichersystem für Metadaten, ein effizientes Suchsystem und ein System zum Management digitaler Objekte detailliert beschreiben. Ziel ist es, sich einen Überblick über die verfügbaren kommerziellen Produkte zu verschaffen, die den gestellten Aufgaben gewachsen sind.

Zum Zeitpunkt unserer Recherchen wurde das Pandora-System durch eine eigene Entwicklung der National Library of Australia ersetzt, die neuere kommerzielle Programme einsetzt, um auf aktuellem Stand zu bleiben. Es gibt derzeit noch kein integriertes Gesamtsystem, das die Anforderungen an das Speichersystem für digitale Objekte vollständig erfüllt. Den eigentlichen Problemen der Langzeitarchivierung – durch die Weiterentwicklung von Technologie- und Softwarestandards – wird mit Überlegungen zur Konvertierung in Standardformate (Migration) und einem Ansatz zur Emulation durch eine Liste benötigter Software Rechnung getragen. Aber eine endgültige Lösung wurde auch bei Pandora noch nicht gefunden. Derzeit wird über eine Kombination aus Migration und Emulation nachgedacht.

Das Open Archival Information System (OAIS) wurde zwar im Zusammenhang mit Pandora diskutiert (Parallelen sind deutlich!), jedoch in einigen Punkten als unpassend befunden. OAIS dient jetzt als Checkliste für wichtige Funktionen.

# 9 Migration in der Praxis

*Dieses Kapitel beleuchtet die Migration anhand von vier konkreten Projekten. Zunächst diskutiert Abschnitt 9.1 die Migrationsaspekte im VERS-Projekt, das wir im vorhergehenden Kapitel schon einführend besprochen haben. Abschnitt 9.2 stellt anschließend das Projekt »Preserving the Whole« vor, wie es von der Commission on Preservation and Access initiiert wurde, um Formate zu finden, die mit der größten Wahrscheinlichkeit eine Migration sozialwissenschaftlicher, statistischer Daten und zugehöriger Dokumentation in künftige technische Umgebungen erlauben. Angeregt durch eine Anfrage des Council on Library and Information Resources (CLIR) führt schließlich die Cornell University Library das Projekt »Risk Management of Digital Information« durch und sieht als das brennendste Problem nicht die Instabilität von Medien oder überalterte Hardware an, sondern die Veraltung von Datenformaten und zugehöriger Software. Diesem spannenden Projekt widmen wir uns in Abschnitt 9.3. Schließlich beschreiben wir in Abschnitt 9.4 ein umfangreiches Projekt zur Problematik der Datenbankmigration. Erste Ergebnisse einer Umsetzbarkeitsuntersuchung werden vorgestellt.*

## 9.1 Migrationsaspekte im VERS-Projekt

*Migration bei VERS*

In diesem Abschnitt wollen wir auf die *Migrationsaspekte* im VERS-Projekt eingehen. Lassen wir uns zunächst noch einmal das Zwiebelmodell eines VERS-Record ins Gedächtnis rufen, wie wir es in Abschnitt 8.2 bereits beschrieben haben.[1] Dort sind mindestens drei Fragen offen geblieben: Wie kommen wir eigentlich zu dieser »Zwiebel«, wie erhalten wir sie über lange Zeiträume und wie können wir später auf sie zugreifen? Diese Fragen gehen offensichtlich Hand in Hand mit dem DSEP-Modell und treten dort bei folgenden Prozessen auf:

[1] vgl. auch Abb. 8.6 auf S. 156.

- *Delivery & Capture* und *Ingest*
- *Archival Storage* und *Preservation*
- *Access*

*Ziel: weitgehender Verzicht auf Migration*

Wir wollen im Folgenden anhand der obigen Punkte die Migrationsaspekte innerhalb des VERS-Projekts vorstellen. Generelles Ziel des VERS-Projekts ist es, auf fortschreitende Migration der Dokumenteninhalte möglichst zu verzichten und stattdessen gleich auf sehr wenige, möglichst langlebige *Standardformate* zu setzen. Dokumente werden so bei der Archivierung einmalig in ein Standardformat migriert. Innerhalb des Archivs werden digitale Medien natürlich immer wieder aufgefrischt. Dieser Vorgang bewirkt jedoch kaum eine Verfälschung des Dokumenteninhalts.

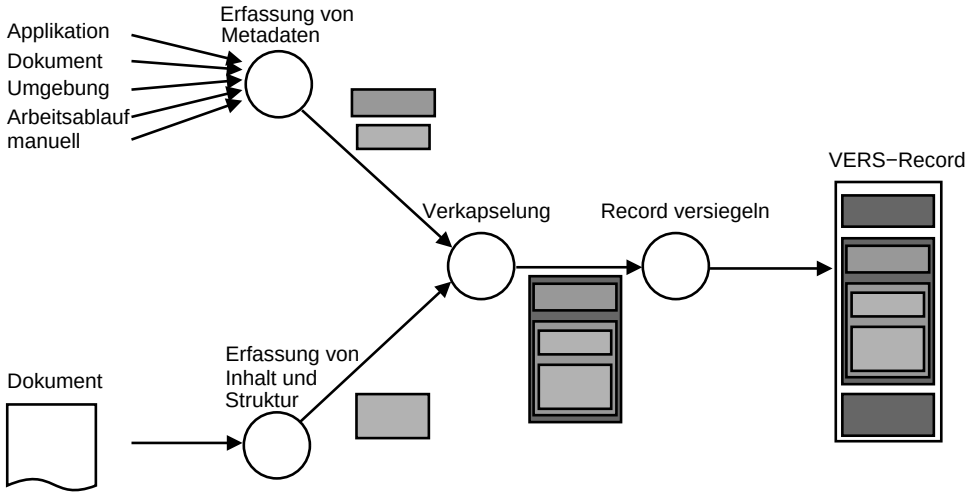



**Abbildung 9.1** *Der Weg zum VERS-Record*

*Erstellung eines VERS-Record*

Abbildung 9.1 zeigt den Prozess der Erstellung eines VERS-Record (engl. *Record Capture*). Zunächst wird das zu archivierende Dokument in ein Standardformat von VERS umgewandelt – also migriert. Je nach Art des Originaldokuments kann dieser Prozess automatisch (Archivierung von E-Mails), halbautomatisch (Archivierung von Textverarbeitungs-Dokumenten wie Microsoft Word) oder manuell geschehen (Archivierung von Papierdokumenten). Als Standardformate werden PDF für einfache Dokumente und XML für komplexere Dokumente (wie Datenbanken; vgl. auch Klettke und Meyer (2002)) verwendet. E-Mails werden beispielsweise automatisch durch den Mailserver nach PDF umgewan-

delt. Dokumente, die mit einer Textverarbeitung erstellt wurden, können über einen virtuellen PostScript-Drucker »gedruckt« werden, wobei eine PostScript-Datei erstellt wird. Diese PostScript-Datei lässt sich dann leicht nach PDF umwandeln. Allerdings geht auf diesem Wege die Struktur des Dokuments verloren, da PostScript keine Strukturinformationen unterstützt. Besser ist in diesem Fall ein direkter PDF-Export, wie er beispielsweise von Adobe für gängige Textverarbeitungen angeboten wird.

*Erfassung von Papierdokumenten*

Papierdokumente werden zunächst gescannt, um danach ihren textuellen Inhalt per OCR (engl. *Optical Character Recognition*) zu erfassen. Der erkannte Text und das zugehörige gescannte Bild können dann in einer PDF-Datei zusammen gespeichert werden. Bei der Umwandlung nach PDF bleibt so auch das Erscheinungsbild des Originals weitgehend erhalten und OCR-Fehler lassen sich eventuell später noch beheben. Datenbanktabellen werden nach XML umgewandelt. Hierbei ist darauf zu achten, dass nicht nur die eigentlichen Daten nach XML migriert werden, sondern auch zusätzliche Informationen, beispielsweise Formeln oder Schlüsseleigenschaften bestimmter Felder.

*Datenformate: PDF und XML*

Die Formate PDF und XML erfüllen die Kriterien für die Langzeitarchivierung aus Abschnitt 3.2 weitgehend und kommen so als Ziele einer Migration für die Langzeitarchivierung in Frage. PDF und XML sind vollständig spezifiziert, öffentlich zugänglich und frei von Patenten. XML wurde vom W3C standardisiert, einem der größten Standardisierungsgremien im Internet. PDF hingegen wurde von der Firma Adobe spezifiziert – somit scheint PDF das Standardisierungskriterium zunächst nicht zu erfüllen. Es lässt sich jedoch feststellen, dass sich PDF zum De-facto-Standard für druck- und anzeigbare Dokumente entwickelt hat. Weiterhin existiert momentan kein standardisiertes Dokumentenformat, das ähnliche Eigenschaften wie PDF bietet.

*Gewinnung von Metadaten*

Zu einem VERS-Record werden eine Reihe von Metadaten benötigt, die zum größten Teil *automatisch* erstellt werden sollten. Weitestgehend möglich ist das für diejenigen Metadaten, die den Aufbau des Record und seine Entstehung beschreiben. Eine Reihe von bibliographischen und kontextbezogenen Metadaten werden dem zu archivierenden Dokument entnommen oder ergeben sich aus seiner Entstehungsumgebung, z. B. dem Absender einer E-Mail, dem Autor eines Textverarbeitungs-Dokuments oder einer Liste der verwendeten Literatur. Der Autor eines Textverarbeitungs-Dokuments wird beispielsweise anhand seines System-Logins oder über eine Chipkarte identifiziert. Weitere Metadaten können aus Arbeitsabläufen generiert werden, bei denen

innerhalb verschiedener Arbeitsschritte einzelne Metadatenfragmente erzeugt werden. Letzten Endes bleibt noch die aufwändige manuelle Methode, Metadaten per Hand einzugeben. Dies ist bei eingescannten Papierdokumenten oft erforderlich.

*Verkapselung*

Sind Dokument und Metadaten erfasst, so werden sie zusammen in einem Objekt-Record gespeichert. Diese Verkapselung stellt einen weiteren Migrationsschritt dar, ist aber vollkommen verlustfrei und reversibel. Technisch gesehen werden die Metadaten und das Dokument in eine XML-Datei geschrieben, wobei PDF-Dokumente zunächst in ein Textformat umgewandelt werden müssen. Dieser Vorgang ist jedoch unkritisch, da hierfür der Standard *Base64* verwendet wird. Base64 wird beispielsweise zur Codierung von binären E-Mail-Anhängen verwendet und stellt in diesem Umfeld seine Alltagstauglichkeit täglich unter Beweis.

*Versiegelung*

Um die Authentizität des Record zu gewährleisten, wird er im letzten Schritt versiegelt. Dieser Schritt ist gerade bei behördlichen Dokumenten wichtig, bei denen es ja oftmals um die authentische Nachvollziehbarkeit von Entscheidungen geht. Im Wesentlichen wird hier die digitale Signatur derjenigen Person eingelesen, die den Record erstellt hat. Dies kann beispielsweise über eine Chipkarte geschehen, die die digitale Signatur dieser Person enthält. Die digitale Signatur wird nun zusammen mit dem schon erstellen Objekt-Record in einem VERS-Record gespeichert.

*Verändern eines VERS-Record*

Eine besondere Bedeutung kommt der nachträglichen Eingabe von Metadaten zu einem schon vorhandenen Record zu. Bei der Erstellung eines Record sind nicht immer alle Metadateninformationen vorhanden oder es ergeben sich nachträglich Änderungen wie Speicherort, zusätzliche Schlüsselwörter oder neue Kontextinformationen. Um die Authentizität des VERS-Record nicht zu beeinträchtigen, findet in einem solchen Falle das Verfahren aus Abb. 9.2 Anwendung. Die neuen Metadaten werden wie oben beschrieben eingelesen und in Form einer zusätzlichen »Zwiebelschale« einfach an den bestehenden Record angehängt.

*Ablage eines VERS-Record*

Ist ein VERS-Record erst einmal erstellt, so wird er in einem *Repository* abgelegt. Zunächst wird er auf einer Festplatte zwischengespeichert, bis genügend Records zusammengekommen sind, um sie auf eine CD zu brennen. Das Repository benötigt weiterhin Datenbanken zur Verwaltung der einzelnen CDs. Dazu gehört u. a. eine Datenbank, die Informationen darüber enthält, auf welcher CD welcher Record abgelegt ist. Weiterhin gibt es eine Datenbank, die die (physischen) Aufbewahrungsorte der einzelnen CDs beschreibt. Diese werden entweder in einem Jukeboxsystem oder im Archiv abgelegt.

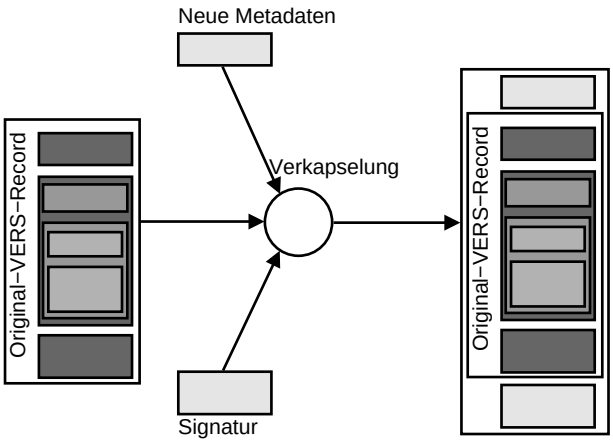



**Abbildung 9.2** *Zusätzliche Metadaten zu einem VERS-Record*

*Auffrischen der Medien*

Wie schon in Abschnitt 3.2 beschrieben, müssen digitale Medien von Zeit zu Zeit aufgefrischt werden. Hierbei werden Records, die auf einer CD vorhanden sind, einfach auf ein neues Medium kopiert. Nach einem zusätzlichen Prüflauf, der die alte CD mit der neuen CD vergleicht, muss die alte CD zerstört werden, um die Datenintegrität zu gewährleisten. Weiterhin müssen die Repository-Datenbanken aktualisiert werden. Der gesamte Prozess geschieht vollkommen automatisch. Eine zusätzliche Datenbank enthält Informationen darüber, in welchen Zeitabständen die CDs aufgefrischt werden müssen. Es gibt natürlich auch noch die üblichen Backups der im Repository vorhandenen CDs.

*Löschen*

Nicht alle Sachverhalte müssen bis in alle Ewigkeit erhalten werden. Es gibt also eine Reihe von Records, die nach einer Weile aus dem Archiv gelöscht werden. Wie aber kann ein Record gelöscht werden, der sich auf einem Medium befindet, das nicht wieder beschrieben werden kann? Um den Aufwand und die Kosten in Grenzen zu halten, kann nicht jedesmal, wenn ein Record gelöscht werden soll, die gesamte CD, auf der er enthalten war, neu gebrannt werden (ohne den zu löschenden Record). Stattdessen bietet sich folgendes Verfahren an: In der Datenbank, die die Records verwaltet, werden Informationen abgelegt, die festhalten, dass der betreffende Record gelöscht werden soll und wer dies angeordnet hat. Danach kann auf den Record nicht mehr über die normalen Zugriffsmechanismen zugegriffen werden, da diese ja die Datenbank benötigen, um den Record zu »finden«. Dennoch ist der Record noch physisch auf der CD vorhanden, sodass der Löschvorgang im Notfall auch wieder rückgängig gemacht werden könnte. Der gelöschte Record wird dann beim nächsten Auffri-

schen einfach nicht mehr berücksichtigt, was einem tatsächlichen physischen Löschen gleichkommt, da die alten CDs ja zerstört werden. Das eben beschriebene Verfahren kommt uns schon sehr bekannt vor, wenn wir zu denjenigen Macintosh- oder Windows-Nutzern gehören, die den »Papierkorb« benutzen.

*Zugriff auf einen VERS-Record*

Abschließend wollen wir die Frage klären, wie wir auf die Informationen in einem VERS-Record zugreifen können. Bildlich gesprochen geht es also darum, die Zwiebel im Repository zu finden und sie danach zu schälen. Ein VERS-Record kann leicht über die im Repository vorhandenen Datenbanken gefunden werden. Hinzu kommt natürlich noch ein Zugriffskontrollmechanismus auf Basis von Chipkarten. Schließlich soll nicht jeder (geheime) behördliche Dokumente einsehen können. Die Datenbanken sorgen auch dafür, dass auf gelöschte, aber noch physisch vorhandene Records nicht zugegriffen werden kann. Weiterhin werden Zugriffe aufgezeichnet.

*Suche*

Neben einer einfachen Suche innerhalb der Datenbanken helfen die Kontextinformationen, zu einem (schon gefundenen) Record verwandte Records zu finden. Hierbei wäre eventuell auch ein Topic-Map-basierter Ansatz sinnvoll. Eine weitere Möglichkeit stellt die Volltextsuche innerhalb der archivierten Records dar. Dieser Vorgang setzt jedoch voraus, dass entweder die VERS-Records während des Suchvorgangs »ausgepackt« werden (siehe unten) oder der komplette Text eines archivierten Dokuments extra gespeichert wird, z. B. in einer Indexdatenbank. Der zweite Ansatz erscheint wesentlich praktikabler, da hier bei der Suche nicht ständig auf die CDs zugegriffen werden muss.

*Anzeige*

Das »Schälen« einer einmal gefundenen Zwiebel gestaltet sich sehr einfach. Es müssen lediglich die (reversiblen) Schritte bei der Erstellung eines VERS-Record rückwärts vollzogen werden. Stellen wir uns einfach die Pfeile in Abb. 9.1 in umgekehrter Richtung vor. So kommen wir sehr leicht zu einem Dokument in binärem PDF oder einer in XML codierten Datenbank.

*Diskussion*

Die Verantwortlichen des VERS-Projekts gehen davon aus, dass PDF- oder XML-Dokumente auch zukünftig angezeigt werden können, beispielsweise mit dem Adobe Acrobat Reader oder mit gängigen Internet-Browsern. Was aber, wenn es in Zukunft keine Programme mehr gibt, die PDF oder XML interpretieren können? Nun, zum einen ist dieser Fall wohl recht unwahrscheinlich, da es sich bei PDF und XML um sehr weit verbreitete und gut dokumentierte Standardformate handelt, die nicht so bald aussterben werden. Wenn dieser Fall doch eintreten sollte, hilft ein emulativer Ansatz weiter: Man emuliert dann eine Umgebung, auf

der die letzten Programme laufen, die PDF und XML interpretieren können.

## 9.2 Preserving the Whole

*Hintergrund: Migration sozialwissenschaftlicher Daten*

Dieses Projekt nahm konkrete Formen an, als die *Commission on Preservation and Access* im Jahr 1996 die Social Science Library und das Social Science Statistical Laboratory an der Universität Yale beauftragte, Formate zu identifizieren und zu evaluieren, die mit der größten Wahrscheinlichkeit eine Migration sozialwissenschaftlicher, statistischer Daten und zugehöriger Dokumentation in künftige technische Umgebungen erlauben (Green *et al.* 1999).

*vom Mainframe zum PC*

Seit 1972 werden an der dortigen Bibliothek maschinenlesbare Daten gesammelt. In der Ära des Mainframe beschränkte sich die Migration auf ein Auffrischen (Refreshing), also ein Umkopieren ohne Eingriff in die logische Struktur der Formate. Mitte der neunziger Jahre tauchte mit dem Einzug PC- und Unix-basierter Systeme die Frage nach plattformunabhängigen Formaten auf, wobei der systemunabhängigen Archivierung und der Erhaltung zugehöriger Dokumentation weniger Beachtung geschenkt wurde.

*numerische Daten mit Papierdokumentation*

Zur Bearbeitung dieses vernachlässigten Aspekts wird ein zweigleisiger Ansatz gewählt, der einerseits die Konvertierung der zugehörigen Dokumentation auf Papier in eine digitalisierte Form vorsieht und andererseits eine Migration digitalisierter numerischer Daten in ein Archivformat. Die Wichtigkeit der zugehörigen Dokumentation wird besonders betont. Fragen zur Evaluierung von Speichermedien und zum geistigen Eigentum spielten bei diesem Projekt keine Rolle. In der sozialwissenschaftlichen Gemeinde besteht die Auffassung, dass die Daten frei zugänglich sein sollten.

*Gefahr auch für numerische Daten!*

Dem Projekt ging eine Literaturrecherche voraus, die sich auf die Erhaltung sozialwissenschaftlicher, numerischer Daten und zugehöriger Dokumentationen konzentrierte. Bemerkenswert ist hierbei, dass sich auf dem Gebiet der numerischen Daten keine Informationen finden ließen. Eine Nachfrage bei der *National Archives and Records Administration* (NARA) ergab, dass dort die numerischen Daten im ursprünglichen Format aufbewahrt und nur auf Anfrage konvertiert werden. Ausgehend von den Studien zu textuellen Daten wurde der Schluss gezogen, dass numerische Daten durch die Abhängigkeit von Soft- und Hardware genauso gefährdet sein werden. Zur Behandlung der *Codebücher* (*codebooks*, eine in der Sozialwissenschaft gebräuchliche Bezeichnung für Metadaten) wird auf den Ansatz des *Journal Storage Project*

(JSTOR)[2] verwiesen, bei dem die Zeitschriftenseiten als Bild angeboten werden und zusätzlich mit Hilfe von OCR einen Index erhalten haben.

*Migrationsprozess*

Zunächst soll die Behandlung der Daten beschrieben werden. Ziel hierbei war es, einen Migrationsprozess für numerische Daten von Rechnerbändern zu hard- und softwareunabhängigen Formaten zu entwickeln und zu evaluieren sowie den Nutzen der resultierenden Formate zu bewerten. Dazu wurde der Migrationsprozess in neun einzelne Schritte zerlegt:

1. Ausstattung identifizieren
2. Kopieren von Dateien auf Mainframe-basierten Medien auf lokale Festplatten
3. Prüfen der Dokumentation
4. Definition des Formats *column binary*
5. Entwicklung einer standardmäßigen Klassifikation für die Benennung von Variablen
6. Einlesen der Datendateien mit Hilfe von SAS und SPSS
7. Identifikation von Migrationsformaten
8. Umcodierung der Datendateien mit SAS
9. Erzeugen von Datendateien im Format *spread ASCII* ohne Umcodierung

*Datenumfang*

SAS (ursprünglich ein Akronym für *Statistical Analyses System*) und SPSS (*Statistical Package for the Social Sciences*) sind proprietäre Programme zur Auswertung statistischer Daten. Die anderen Begriffe werden wir in der näheren Beschreibung der einzelnen Schritte erläutern. Für die Bearbeitung des Projekts wurden zehn *Roper Reports* von damals 200 ausgewählt. Hierbei handelt es sich um Meinungsumfragen der Roper Starch Worldwide Inc., die zehnmal jährlich in den Vereinigten Staaten durchgeführt werden und 200 bis 300 Variable umfassen und von 1.500 bis 2.000 Personen beantwortet werden. Die als Beispiel dem Projektbericht beigefügte Dokumentation, die im Wesentlichen aus dem Muster eines Fragebogens besteht, umfasst 39 DIN-A4-Seiten.

*1. Schritt*

Im 1. Schritt wurden als Ausstattung für dieses Projekt IBM-Mainframes mit Bandzugriff (Quelle) und Intel-PCs mit Windows für Workgroups in einem Novell-Netzwerk (Ziel) identifiziert.

*2. Schritt*

Der 2. Schritt bestand aus mehreren Teilschritten. Zuerst wurden die Dateien im Format *column binary* (siehe 4. Schritt) von den alten Magnetbandspulen auf zuverlässigere Cartridges übertragen (Auffrischung). Danach wurden die Dateien auf die Fest-

[2] siehe http://www.jstor.org.

platte des Mainframe kopiert und anschließend mittels Dateitransfer im Binärmodus auf die Festplatte des Novell-Servers, auf der die Software SAS installiert war.

*3. Schritt*

Im 3. Schritt war als primäre Dokumentation eine Kopie des zugehörigen Fragebogens auszuwerten. Dadurch war eine Zuordnung von Lochkartennummer, Spaltennummer und Position der Lochung zu den einzelnen Befragten, zu den Fragen und zu vorgesehenen Antworten möglich. Von den Fragebögen gibt es verschiedene Varianten, die unterschiedliche Teilmengen der Fragen oder Fragen in unterschiedlicher Reihenfolge enthalten.

*4. Schritt*

Zum Verständnis des 4. Schritts ist die Erläuterung des Formats *column binary* erforderlich, das noch auf den IBM-Lochkarten basiert, die 80 Spalten und 12 Zeilen besitzen. Im Gegensatz zu *noncolumn binary*, das nur eine Variable pro Spalte zulässt, sind hier bis zu 12 Variablen pro Spalte möglich, also pro Lochkarte maximal 960 Variablen. Da die unterschiedlichen Fragen ggf. eine unterschiedliche Anzahl von Bit-Positionen auf der Lochkarte benötigen, kommt es zu einem Versatz bezüglich der Spalten, d. h., die physische Lochnummer der 80x12-Matrix stimmt i. d. R. nicht mit der *logischen* Lochnummer überein, so wie sie auf dem Fragebogen aufgeführt ist. Die logischen Lochnummern 11 bzw. X und 12 bzw. Y werden oft für Antworten vom Typ »weiß nicht«, »keine Antwort« oder »nicht anwendbar« genutzt. Diese Zuordnung, wie anhand eines Ausschnittes einer Mehrfachantwort in Tabelle 9.1 angedeutet, war Aufgabe des 4. Schritts.

**Tabelle 9.1** *Zuordnung der physischen zur logischen Lochung*

| mögliche Antworten | physische Lochung (Spalte/Zeile) | logische Lochung |
|---|---|---|
| i. A person being irresponsible | 47/12 | 9 |
| j. Influence of friends | 47/3 | 0 |
| k. Alcohol abuse | 47/2 | 11 bzw. X |
| l. What people see in movies | 47/1 | 12 bzw. Y |
| m. The advertising and marketing of toy guns, etc. | 48/4 | 1 |
| n. Guns being to easy to get | 48/5 | 2 |

*5. Schritt* Im 5. Schritt wurden die Variablen in Klassen eingeteilt, die den Typen der Fragen entsprechen. Dadurch ergibt sich ein Schema für die *templates*, die die Umcodierung der Daten im 8. Schritt erleichtern, und für die Logik, die hinter den Variablen steht. Außerdem wird eine einheitliche und konsistente Benennung der Variablen erreicht. Die vier Grundtypen der Variablen sind:

1. Reguläre numerische Variablen: Sie reichen von 0 bis 9. Bei einer Antwort könnte für »Ja« eine 1 stehen und für »Nein« eine 2.
2. Numerische Variablen mit speziellen Werten: Wie oben, nur dass ein oder zwei spezielle Werte wie »weiß nicht« existieren können, die im logischen Loch 11 bzw. X oder 12 bzw. Y aufgezeichnet werden.
3. Fragen mit Mehrfachantwort: Aus einer Liste können mehrere Punkte ausgewählt werden, die bei einer Auswahl mit einer 1 codiert werden und sonst mit 0, womit sich eine Liste binärer Variablen einschließlich möglicher spezieller Werte ergibt.
4. Fragen mit einfacher Antwort: Hier darf nur ein Punkt ausgewählt werden.

Mehrere Spalten sind dabei für die Auswahl erforderlich, die mehr als zwölf Punkte umfassen (oder mehr als zehn Punkte, wenn Loch X und Y für spezielle Worte wie »weiß nicht« reserviert sind).

*6. Schritt* Im 6. Schritt wurden die Dateien im Format column binary mit Hilfe des Programms SAS eingelesen, wobei unter mehreren möglichen Formaten zum Einlesen das Eingabeformat *PUNCH.d* gewählt worden war. Dieses Format liest genau ein Bit der ursprünglichen Daten, wobei »d« die Zeilennummer und ein zusätzliches Präfix die Spaltennummer bestimmt.

*7. Schritt* Im 7. Schritt wurden folgende Dateiformate ausgewählt, um den Migrationsprozess zu testen:

1. SAS-Systemdateien mit umcodierten column-binary-Daten mit und ohne Zwischenvariablen,
2. SAS-Systemdateien mit gekürzten Integer-Byte-Längen (drei bzw. vier Bytes),
3. SAS-Exportdateien mit umcodierten column-binary-Daten,
4. ASCII-Dateien erzeugt, aus den umcodierten column-binary-Daten,
5. ASCII-Dateien der binären Datenmuster der Originaldateien, als *spread* ASCII bezeichnet.

Gesichtspunkte bei der Auswahl waren die Fähigkeit zum Lesen des Formats column binary, die Verfügbarkeit von geeignetem Personal, die Portierbarkeit des Ausgabeformats, Speicherplatzanforderungen und Auswirkungen auf die Langzeitarchivierung. Bei einem Vergleich des Speicherbedarfs zeigt sich eine Spanne von 121 bis 817 Prozent der Originalgröße. Das ebenfalls geeignete Softwaresystem SPSS kam wegen mangelnder Erfahrung des Personals nicht zum Einsatz.

*8. Schritt*

Im 8. Schritt fand die Umcodierung der Daten statt, wobei die Daten der einzelnen Lochung – gelesen mit PUNCH.d und gespeichert in Zwischenvariablen – endgültigen Variablen, die die Logik der einzelnen Frage repräsentieren, zugeordnet wurden. Dies geschah mit Hilfe der im 5. Schritt erwähnten Templates, die vereinfacht schematisch folgendermaßen aussehen:

```
Q10_1 PUNCH.1
Q10_2 PUNCH.2
IF Q10_1=1 and Q10_2=0 THEN Q10 = 1
IF Q10_1=0 and Q10_2=1 THEN Q10 = 2
IF Q10_1=1 and Q10_2=1 THEN Q10 = 3
```

Die SAS-Makros für die konkreten Fragebögen wurden durch eine Anpassung der Templates mit einem gewöhnlichen Texteditor realisiert. Auf eine weitergehende Automatisierung des Prozesses der Umcodierung wurde verzichtet, weil die Behandlung von irregulären Dateien, z. B. mit verschiedenen Datentypen innerhalb einer Spalte, zu aufwändig gewesen wäre und weil bestimmte Fehlersituationen eine Bewertung benötigen, die nicht einem Programm überlassen werden sollte. Zur Qualitätssicherung wurden u. a. alle Variablen im endgültigen Datensatz geprüft, da die fehlerhafte Codierung einer einzigen Variablen im Konvertierungsprogramm die Integrität des ganzen Datensatzes zerstören kann. Für diese Prüfung wurde die Anzahl der Werte einzelner Variablen mit den Angaben der so genannten *xrays* verglichen. Hierbei handelt es sich um einen von Roper gelieferten Ausdruck, der für jede Lochkartenposition die Summe der gelochten Bits angibt. Abschließend wurden die umcodierten Daten in SAS-System- und SAS-Exportdateien gesichert. Für das Erzeugen von ASCII-Dateien wurde die SAS-Anweisung *PUT* verwendet, welche den Vorteil hat, platzsparende Dateien zu liefern und als *data dictionary* zu dienen. Das Übersetzungsprogramm *DBMSCOPY* hingegen konnte zwar auch ein data dictionary erzeugen, jedoch keine kompakten ASCII-Dateien.

*9. Schritt*

Der 9. Schritt kann als Alternative zum 8. Schritt betrachtet werden. Die Umcodierung beschränkt sich dabei auf eine Konvertierung des Formats column binary in ASCII-Zeichen, d. h., die binäre Struktur wird beibehalten. Jeder Lochkartenposition der column-binary-Datei wird mittels SAS eine individuelle Variable zugeordnet, die dann mit dem Befehl PUT in ein ASCII-Zeichen umgesetzt wird. Da keine Interpretation des Datenstroms wie im 8. Schritt vorgenommen wird, kann die Konvertierung als einfache Iteration über die Lochkarten und deren Spalten und Zeilen realisiert werden. Vom Nutzer werden nur wenige Parameter, z. B. die Anzahl der Lochkarten oder die Record-Länge der Originaldaten, abgefragt, um mittels eines C-Programms die entsprechenden SAS-Makros zu generieren.

*Bewertung der Migrationsergebnisse*

Nun zur Bewertung der durch die Migration erreichten Ergebnisse. Die Endnutzer empfanden es als Erleichterung, keine Eingabeanweisungen mehr für das Format column binary erstellen zu müssen. Mit der Lokalisierung der einzelnen Variablen in den Dateien im Format spread ASCII gab es auch keine Probleme. Die Übertragung von column binary in SAS- bzw. SPSS-Export-Dateien wurde als attraktive Option bewertet, weil diese Formate einfach zu nutzen und auf viele Betriebssysteme portierbar sind sowie in andere softwarespezifische Formate konvertiert werden können. Dem steht jedoch eine Reihe von Nachteilen gegenüber. Der Prozess der Umcodierung ist aufwändig und fehleranfällig und stellt hohe Anforderungen an die Bearbeiter, die auch Unregelmäßigkeiten und Fehler zu bewerten haben. Außerdem ist der Prozess sehr zeitintensiv. Darüber hinaus ist für das Lesen dieser Formate proprietäre Software erforderlich, sodass keine Garantie besteht, spezifische Versionen auch in Zukunft lesen zu können, was teure Migrationsschritte zur Konsequenz haben könnte. In diesem Zusammenhang wird das komprimierte Format *Dualabs* genannt, das für die Volkszählung in den Vereinigten Staaten 1970 verwendet wurde und auf den meisten Rechnern im Jahr 1999 nicht mehr lesbar war.

*proprietäre Formate SAS und SPSS*

*spread ASCII*

Nur das Format spread ASCII kommt in Frage, wenn ein Standard für die Archivierung als noncolumn binary, nichtproprietär und das Original verlässlich wiedergebend gefordert ist. Als Nachteil stehen eine Steigerung des Speicherplatzbedarfs um mindestens 600 Prozent sowie die Notwendigkeit der Konvertierung der binären Struktur gegenüber. Zusätzlich muss die Dokumentation ergänzt werden, um die Konvertierung zu beschreiben. Die Zuordnung zu Variablen und Werten unter Zuhilfenahme der Dokumentation (Fragebogen) bleibt dem Nutzer überlassen.

*column binary*

Das Originalformat column binary stellte sich für die Langzeitarchivierung attraktiver heraus als ursprünglich vermutet worden war, da es sparsam mit Speicherplatz umgeht, die Originalcodierung erhält, direkt mit der Dokumentation korrespondiert, im Binärformat zwischen verschiedenen Rechnern transferiert und mit Statistikpaketen auf allen für den Test eingesetzten Plattformen gelesen werden kann. Andererseits ist es schwierig, Information zu finden und zu entschlüsseln, wie mit SAS bzw. SPSS dieses Format gelesen werden kann. Die Handbücher für die PC-Version enthalten dazu nichts mehr, was das mögliche Auslaufen des Formats andeutet, entgegen der Ankündigung einer bestimmten Unterstützung durch SAS. Daher wird bei einer Aufbewahrung der Originaldateien die Mitgabe von Beispielprogrammen, die den Lesevorgang beschreiben, empfohlen. Insgesamt wird aber dem Format spread ASCII gegenüber diesem Vorgehen der Vorzug gegeben.

*Migration von Dokumentation in Papierform*

Die zweite Hauptaufgabe dieses Projekts war die Behandlung der in Papierform vorliegenden Dokumentation. Dazu wurden unterschiedliche Ansätze verfolgt. So wurde versucht, mittels der OCR-Software *TextBridge Pro* die gescannten Dokumente in ASCII-Texte umzuwandeln. Die Auswahl fiel aus praktischen Gründen auf diese Software, da das Personal bereits Erfahrung hatte und andere Produkte keine signifikanten Vorteile aufwiesen. Als problematisch stellte sich dabei die Ausgangsqualität dar, die u. a. durch mehrfaches und schlechtes Kopieren, durch handschriftliche Ergänzungen und wechselnde Fonts gekennzeichnet war. Insgesamt erwies sich der Prozess als sehr aufwändig und im Ergebnis nicht brauchbar. Neben dem Fehlen der handschriftlichen Ergänzungen lag das an der schlechten Interpretation von Tabellen und Spalten, dem mühsamen Setzen von Scanner- und OCR-Parametern, der besonders schwierigen Einteilung in Seiten und der Abhängigkeit verfügbarer Werkzeuge vom jeweils gewählten Ausgabemodus (z. B. nur WordPerfect oder Word für den Modus text proofing). Das Ablegen der gescannten Bilder im komprimierten Rastergraphik-Format *CCITT Group 3* wurde als ungeeignet betrachtet, da eine Nachbearbeitung schlechtere Ergebnisse liefert als die direkt gescannten Bilder.

*PDF und TIFF*

Als weiterer Ansatz wurde die Erzeugung von PDF-Dateien mittels *Adobe Capture* untersucht, wobei das hybride Format *image + text* als das am meisten geeignete gesehen wird, da es einerseits durchsuchbaren Text und andererseits eine genaue Wiedergabe des Dokuments einschließlich der bereits erwähnten handschriftlichen Ergänzungen liefert. Die Dokumente wurden

seitenweise eingescannt und seitenweise in Dateien mit dem Format *Tagged Image File Format* (TIFF) abgelegt, welche neben den endgültigen PDF-Dateien, für deren Erstellung die TIFF-Dateien dokumentenweise zusammengeführt worden waren, für die Archivierung gespeichert wurden. Für die endgültigen PDF-Dateien war die durch den OCR-Prozess aus TIFF erzeugte Textschicht mit dem integrierten Produkt *Capture Reviewer* in Hinblick auf eine Textsuche – z. B. nach Begriffen im Fragetext oder nach Variablennamen – zu überarbeiten. Auch bei diesem Ansatz war der Aufwand für die Nachbearbeitung als zeitraubendster Prozess stark vom Zustand des Ausgangsmaterials abhängig. Hinderlich sind dabei u. a. schattiger Hindergrund, gekrümmte Textzeilen, gestauchte Fonts, gezackte Kanten bei den Buchstaben und wiederum handschriftliche Ergänzungen. Für die Bearbeitung einer Dokumentation wird eine Zeitspanne von einer bis achteinhalb Stunden angegeben. Durchschnittswerte wurden auf Grund der kleinen Testmenge nicht gebildet. Deswegen konnte auch keine umfassende Evaluierung der Textsuche durch die Nutzer erfolgen. Neben den Vorteilen, die PDF-Dateien bei der Verteilung und Erhaltung bieten, werden auch Nachteile gesehen. So fehlen Strukturierungshilfen zur Manipulation, Ausgabe und Analyse von Metadaten, insbesondere was die Berücksichtigung numerischer Daten anbelangt. Außerdem wird die Abhängigkeit von proprietärer Software als kritisch eingeschätzt, obwohl es sich um einen veröffentlichten und populären Standard handelt. Hier wird ein Vergleich mit dem Format *dBase* gezogen, das schnell aus der Mode kam und große Probleme mit umfangreichen Sammlungen von CD-ROM-Produkten, verteilt durch die Regierung der Vereinigten Staaten, verursachte. Das TIFF-Format wird als geeignete Zwischenlösung betrachtet, bis die OCR-Technologie weiter fortgeschritten sein wird. Das *Inter-university Consortium for the Political and Social Research* (ICPSR) hat entschieden, gedruckte Dokumentation vorerst in diesem Format zu halten.

*HTML, SGML und XML*

Eine Umsetzung der Dokumentation in die Formate HTML bzw. SGML oder XML – obwohl in Ergänzung mit anwendungsbezogen standardisierten Elementen als beste Lösung erachtet – war im Rahmen dieses Projekts (noch) nicht realisierbar. Ursachen lagen in der schlechten Qualität der Ergebnisse des OCR-Prozesses und im Arbeitsaufwand für das Setzen semantischer Marken.

*Empfehlungen für Datenproduzenten*

Abschließend werden Empfehlungen für die Datenproduzenten formuliert. Sie sollen von der Wichtigkeit der Aspekte Systempflege und Langzeiterhaltung überzeugt werden. Auf Grund der gemachten Erfahrungen wird für statistische Daten das einfachste

Format, also eines ohne komplexe interne Strukturen, als das geeignetste empfohlen, nämlich *flat ASCII*, ergänzt um die maschinenlesbare Dokumentation in einem nichtproprietären und plattformunabhängigen Format. Zusätzlich dringend empfohlen werden auch Programmieranweisungen für SAS und SPSS, die zeigen, wie Rohdaten in Systemdateien umgewandelt werden. Das ganze Paket sollte mit maschinenlesbaren Metadaten versehen werden, sodass der gesamte Inhalt und die Struktur studiert werden können. Die Metadaten sollten auch eine über mehrere Studien bzw. Befragungen hinwegreichende Analyse der Daten erlauben. Ebenfalls sollte eine Überführung in Analyse- und Betrachtungswerkzeuge bzw. in statistische Software und Datenbanken unterstützt werden und zwar mittels eines Formatstandards, der konsistente Marken (tags) zur Kennzeichnung entsprechender Elemente bereitstellt. In diesem Zusammenhang wird auf die *Data Documentation Initiative* (DDI)[3] des ICPSR hingewiesen, die auf Basis von XML-Standards zur Dokumentation numerischer Datenbanken entwickelt.

*»lessons learned«*

Dieses Projekt zeigt deutlich, dass numerische Daten ohne Metadaten wertlos sind. Weiterhin wird sichtbar, wie viele unterschiedliche Formate trotz des relativ homogenen Datenbestands bei einem Migrationsansatz für eine Langzeitarchivierung zu untersuchen und nach verschiedenen Kriterien zu bewerten sind. Hervorzuheben ist bei diesem Projekt, dass auch Fehler in den Ausgangsdaten als Kriterium herangezogen wurden. Eine Einordnung der Migrationsschritte in ein konzeptionelles Modell wie OAIS wurde nicht vorgenommen. Abschließend wurde festgestellt, dass die zunehmende Integration von Daten und Dokumenten sowie der Einsatz von komplexen Datenbank-Managementsystemen neue Fragen aufwerfen, z. B. über die Nutzbarkeit von Inhalten außerhalb solcher Systeme.

## 9.3 Risk Management of Digital Information

*Datenformate und Software als Hauptrisiko*

Angeregt durch eine Anfrage des *Council on Library and Information Resources* (CLIR)[4] aus dem Jahr 1998 führte die *Cornell University Library* (CUL) dieses Projekt durch (Lawrence *et al.* 2000). Die Unterbezeichnung lautet: *A File Format Investigation*. Als brennendstes Problem werden nicht die Instabilität von Medien oder überalterte Hardware angesehen, sondern die Ver-

[3] siehe http://www.icpsr.umich.edu/DDI.

[4] siehe http://www.clir.org.

altung von Datenformaten und Software. Einerseits besteht die stillschweigende Annahme, dass digitale Bibliotheken ihnen anvertraute, elektronische Informationen erhalten. Andererseits fehlen konkrete Anleitungen, um dieses Material zu verwalten. So muss fallweise entschieden werden, was erhalten und wie priorisiert werden soll, und welche Technik hierfür einzusetzen ist. Auf Basis eigener Erfahrungen entwickelte die CUL Methoden und Werkzeuge zur Risikobewertung, die sich an den praktischen Aspekten der Migration einer im Alltagsbetrieb befindlichen Bibliothek orientieren, wobei eine tiefer gehende Diskussion des Nutzens von Emulation oder Migration für die Langlebigkeit ausgenommen war. Zur Erhaltung und Migration ergab die Literaturrecherche vier Arbeiten mit konkretem Projektbezug, darunter das im vorhergehenden Abschnitt besprochene Projekt *Preserving the Whole*. Zur Risikobewertung war unter der bibliotheksspezifischen Literatur nichts zu finden, dafür jedoch im Bereich der Informatik und speziell bei der Softwareentwicklung. Auffallend sind dort die unterschiedlichsten Definitionen von Risiko und Bewertungsmaßstäben.

*Risikoklassen*

Die Projektarbeiten waren von der Prämisse geleitet, dass die Migration in der Praxis ein fehleranfälliger Prozess ist. Um ein Bewertungsschema für das Risiko zu entwickeln, wurde der Prozess in beschreibbare und quantifizierbare Schritte zerlegt. Daraus ergeben sich drei Risikoklassen:

- *Die mit digitalen Beständen verbundenen allgemeinen Risiken:* Diese betreffen die Existenz institutioneller Unterstützung, die Finanzierung, die Systemhardware und -software, das Personal zum Verwalten der Archive sowie das politische und rechtliche Umfeld.
- *Die mit dem Datendateiformat verbundenen Risiken:* Diese umfassen die internen Strukturelemente der Dateien, die durch eine Migration geändert werden.
- *Die mit dem Prozess der Dateikonvertierung verbundenen Risiken:* Die Konvertierungssoftware könnte falsche Ergebnisse erzeugen, wobei die Fehler sowohl grob als auch subtil sein können.

*Hilfsmittel zur Risikobewertung*

Die Untersuchung dieser drei Klassen führte zur Entwicklung von drei Hilfsmitteln:

- einem Handbuch zur Risikobewertung für die Migration einer Gesamtsammlung,

- einer Lesesoftware, um spezifische Dateien oder Dateisammlungen auf Formatelemente hin zu prüfen, die mit einem Risiko verbunden sind und
- einer Testdatei im Format *wk1*, um für bekannte Format- und Datenelemente Konvertierungssoftware zu testen (bei wk1 handelt es sich um ein älteres Format des Tabellenkalkulationsprogramms *Lotus 1-2-3*[5]).

*ein Handbuch zur Risikobewertung*

Im Folgenden sollen die einzelnen Hilfsmittel näher betrachtet werden. Das Handbuch dient dazu, begrenzte personelle und finanzielle Ressourcen auszugleichen. Es soll einen systematischen Ansatz liefern, Risiken zu identifizieren und zu bewerten. Für die Bewertung wurden eine Skala für die Wahrscheinlichkeit eines Risikos und eine weitere zum Messen der Auswirkungen definiert, wobei kein Anspruch auf wissenschaftliche Präzision erhoben wird. Vielmehr geht das Entwicklerteam davon aus, dass die Entscheidung, ob eine Migration durchzuführen ist, nicht von einer einzigen Zahl abhängig gemacht wird. Das Handbuch, das ausdrücklich nochmals auf die Risiken einer Migration hinweist, besteht aus sechs Abschnitten:

1. *Migration – Fragestellungen und Optionen:* Darstellung der Migration als eine Erhaltungsstrategie und Diskussion von fünf Gründen, digitale Informationen zu migrieren. Die Definition der Migration ist an dieser Stelle etwas weiter gefasst; so zählt auch die routinemäßige Auffrischung dazu.
2. *Risikobewertung und Messung:* Neben unterschiedlichen Betrachtungsweisen des Begriffs Risiko werden konkrete Skalen für das Eintreten eines Risikos und dessen Auswirkungen vorgestellt sowie die Abbildung der beiden Skalen auf eine zweidimensionale Entscheidungsmatrix.
3. *Bewertung des Quell- und Zielformats:* Formate und Konvertierungsprogramme werden in Kategorien eingeteilt. Außerdem werden konkrete Fragen zur Eignung der Formate aus Nutzer- und Archivierungssicht gestellt.
4. *Bewertung des Systems:* Hier werden Fragen zur Hardware, zur Betriebssystemsoftware, zur Datenkompression und zur Sicherheit gestellt.
5. *Metadaten:* Hier werden Fragen zu Sachverhalten gestellt, die für die Präsentation digitaler Informationen erforderlich sind. Dazu wird u. a. auf das OAIS-Referenzmodell und auf DCMI (vgl. Abschnitt 8.1) verwiesen.

[5] jetzt IBM statt Lotus, aber trotzdem noch unter http://www.lotus.com.

6. *Bewertung der Organisation:* Wegen der unterschiedlichen Ausprägung von Organisationsstrukturen werden nur vier Kernfragen betrachtet: Planung der Erhaltung, Finanzen, Personal und Nutzer.

*drei Kategorien von Konvertierungssoftware*

Zur Bewertung des mit Dateiformaten verbundenen Risikos wurde die Konvertierungssoftware in drei Kategorien eingeteilt:

- projektspezifische Programme
- kommerzielle Programme für ausgewählte Anwendungen
- kommerzielle Allzweckprogramme

*Bewertung der drei Kategorien*

Bei der ersten Kategorie kann umfangreiches Wissen der Archivare eingebracht werden. Dieser Möglichkeit stehen lange Entwicklungszeiten und hohe Kosten gegenüber, während kommerzielle Programme kostengünstig sind und viele Funktionen bieten, aber wenig Informationen über den Übersetzungsvorgang liefern. Im Weiteren wurden bei der Bewertung von Dateiformaten zwei Risikozonen identifiziert, nämlich die eigentliche Konvertierung und die Behandlung großer, heterogener digitaler Sammlungen. Für die Vervollständigung der Risikobewertung wird aus pragmatischen Gründen der Einsatz von Testdateien vorgeschlagen. Eine direkte Analyse des Abbildungsprozesses scheitert häufig schon an der Nichtzugänglichkeit des Quellcodes kommerzieller Konvertierungssoftware und an fehlender Expertise. Zur Analyse großer Dateisammlungen wurde das Programm *Examiner* entwickelt, das ASCII- und Binärdateien nach risikobehafteten Elementen durchsucht sowie deren Häufigkeit und die vorab bestimmte Risikoklasse entsprechend einem gesetzten Schwellwert anzeigt. Der Quellcode dieses Programms ist öffentlich zugänglich.

*Risiken des Konvertierungsprozesses*

Weitere Riskobetrachtungen werden bei der Abwicklung des Konvertierungsprozesses als notwendig erachtet, z. B. bei der Auswahl kommerzieller Softwareprodukte. So könnten trotz korrekter Konvertierung fehlende Eigenschaften wie die Fähigkeit zur Stapelverarbeitung den Zeitrahmen eines Vorhabens gefährden.

*Metadaten*

Auch in diesem Projekt wurde die Bedeutung von Metadaten hervorgehoben. Anhand von Beispielen wurde die unterschiedliche Ausprägung von Metadaten dargestellt. So können Metadaten in einem anderen Format als die eigentlichen Daten oder zusätzlich in mehreren digitalen Objekten gespeichert sein, wobei z. B. eines den Inhalt beschreibt und ein anderes strukturelle Informationen wie die Zusammengehörigkeit einzelner Dateien. Gegebenenfalls ist nach der Migration eine Aktualisierung der Metadaten

erforderlich, um z. B. Verweise auf Dateinamen auf Grund einer Typkonvention anzupassen.

*Fallstudie zu TIFF und RDO*

Wir veranschaulichen im Folgenden einige der bisher aufgezählten Aspekte anhand zweier im Rahmen des Projekts durchgeführter Fallstudien. Eine Studie beschäftigte sich intensiv mit der Migration der *Ezra Cornell Papers*, einer Sammlung von etwa 30.000 Bildern, die im *Tagged Image File Format* (TIFF) auf Festplatte gespeichert waren und mit einem Xerox-System gescannt wurden, welches die strukturellen Metadaten, z. B. Seitennummern, in dem proprietären Format *Raster Document Object* (RDO) ablegte. Folgende Projektziele wurden formuliert:

- Identifikation risikobehafteter Dateiattribute
- Bewertung der Notwendigkeit, das vorhandene TIFF-Format 5.0 in die aktuelle Version 6.0 umzuwandeln
- Erkundung des Status einer sich abzeichnenden Revision zu TIFF 7.0
- Bewertung der Risiken, die mit dem Überspringen einer Version verbunden sind
- Bewertung der Risiken und des Datenverlustes durch die Konvertierung von RDO in ein offenes Format, hier *Cornell Digital Library* (CDL)

*Risikoklassen mit Beispielen*

Hierzu führte das Projektteam umfangreiche Recherchen und Analysen zu den Formaten TIFF und RDO durch, aber auch zu anderen Formaten und Projekten, die u. a. Begründungen für eine Konvertierung von TIFF 4.0 zu TIFF 5.0 lieferten. Ergebnis dieser Analyse war u. a. eine in Risikoklassen eingeteilte Beispielsammlung, die ausschnittsweise in Tabelle 9.2 dargestellt ist.

*Bewertung von TIFF*

Obwohl sich die Beschaffung von Informationen zu alten und aktuellen Versionen von TIFF als einfach erwies, war die Kontaktaufnahme zum Entwicklerteam von TIFF 7.0 bei der Firma Adobe erfolglos. Neben den Vorteilen des TIFF-Formats wie Plattformunabhängigkeit, Unterstützung durch zahlreiche Anwendungen und Speicherung von Metadatenwurden in der Flexibilität auch Nachteile gesehen, z. B. die nicht standardisierten Marken (engl. *tags*) und die Fähigkeit, Daten in der Byte-Ordnung *MSB* (*most significant byte*, d. h., das hochwertigste Byte steht an der niedrigsten Adresse) oder *LSB* (*least significant byte*, umgekehrte Reihenfolge wie bei MSB) speichern zu können. Nach einer ausführlichen Evaluierung der Spezifikationen der Versionen 5.0 und 6.0 und verschiedenen Tests bezüglich Qualität und Nutzbarkeit sah das Projektteam kein Risiko, auf diese Migration zu verzichten.

| Risikoklasse | Beispiele (Ausschnitt mit je einem Beispiel) |
|---|---|
| Stabilität des Inhalts (content fixity) | Das Dateiformat ist mit einer neuen Kompressionsmethode verbunden, die die Anordnung der Bits verändert. |
| Sicherheit | Die Formatmigration betrifft digitale Wasserzeichen, Stempel oder andere kryptologische Techniken zur Sicherstellung der Stabilität des Inhalts. |
| Kontext und Integrität | Verweise auf andere Dateien, z. B. mit Metadaten oder Varianten, werden durch die Migration geändert. |
| Referenzen | Dateierweiterungen ändern sich auf Grund einer Aktualisierung des Dateiformats – mit Auswirkungen auf den URL. |
| Kosten | Die Langzeitkosten sind nicht vorhersehbar, weil jeder Migrationszyklus unterschiedliche Schritte erfordern kann, die von der Art der Migration, z. B. Routine oder Paradigmenwechsel, abhängen. |
| Personal | Das Personal hat ungenügendes technisches Wissen. |
| Funktionalität | Die Eigenschaften, die mit einem neuen Format eingeführt werden, können die Erstellung von Versionen, z. B. Druckversionen, beeinflussen. |
| Recht | Copyright-Regelungen können die Nutzung abgeleiteter Versionen, z. B. solche mit besserer Auflösung, einschränken. |

**Tabelle 9.2** *Risikoklassen mit Beispielen*

*Bewertung von RDO*

Während die Projektgruppe das Format TIFF auf die Beobachtungsliste gesetzt hatte, wurde die Migration von RDO in ein anderes Format u. a. deshalb in Angriff genommen, weil das zum Erzeugen und Interpretieren bzw. Navigieren erforderliche Softwaresystem *Xerox Documents on Demand* (XDOD) nicht weiter betrieben werden sollte. Eine RDO-Datei, die mit XDOD beim Scan-Vorgang angelegt wird, enthält strukturelle Informationen eines Bilddokuments und Verweise auf die Dateien, die die einzelnen Bilder seitenweise repräsentieren. Da die Firma Xerox die Spezifikation von RDO nicht offengelegt hatte, war der Einsatz eines Export-Werkzeugs dieser Firma erforderlich, um ASCII-Dateien zu erzeugen, die dann mit einem selbst entwickelten Perl-Skript in das als Request for Comments (RFC Nr. 1691) veröffentlichte Format CDL transferiert werden konnten. Für die Risikobewertung wurde keine spezifische Software entwickelt, weil alle RDO-Dateien einen identischen Aufbau hatten. Die strukturellen Meta-

daten der beiden Dateiformate wurden manuell verglichen; dabei ergaben sich keine Unterschiede. Das Projektteam hätte aber auch Informationsverluste in Kauf genommen, da man das Beibehalten einer obsoleten Version eines proprietären Formats als wesentlich risikoreicher einschätzte.

*Fallstudie zu wk1*

Die zweite Fallstudie beschäftigte sich mit ökonomischen Daten des *United States Department of Agriculture* (USDA), die in 250 Datenserien mit insgesamt knapp 9.000 Dateien im wk1-Format online auf Festplatte und offline auf CDs und Disketten gespeichert waren. Ziele dieser Untersuchung waren:

- Evaluierung der Migration des Dateiformats als Strategie zur Sicherung des Zugriffs auf numerische Daten
- Identifikation der risikobehafteten Dateiformatkomponenten
- Identifikation der zugehörigen Risiken durch den Migrationsprozess
- Evaluierung der Migrationssoftware

*Lücken in den Spezifikationen*

Die Analyse des wk1-Formats zeigte die Wandlung von einem einfachen Tabellenkalkulationsformat zu einem interaktiven Datenbehälter, in den z. B. Hyperlinks und *ActiveX-Controls*, also eigenständige Programme, eingebaut werden können. Trotz dieser Erweiterungen waren die Lotus-Programme abwärtskompatibel geblieben. Lediglich bei dem ebenfalls untersuchten Format *xls* des Tabellenkalkulationsprogramms Excel wurden nicht mehr alle Elemente nach xls konvertiert. Inwieweit dies jedoch auch das alte wk1-Format betraf, war schwer zu prüfen. Ebenso musste zur Kenntnis genommen werden, dass über die Versionen hinweg offensichtlich keine vollständigen Spezifikationen mehr existierten. Hier wurde Handlungsbedarf auch für andere wichtige Tabellenkalkulationsprogramme gesehen.

*Test von Konvertierungsprogrammen*

Um den Konvertierungsvorgang zu bewerten, wurden spezifische Werkzeuge entwickelt. Hierbei handelte es sich u. a. um eine wk1-Testdatei auf Basis der Handbücher von Lotus 1-2-3, Release 2.2, und der veröffentlichten Dateiformatspezifikationen. Die Testdatei enthielt 23 Testprozeduren, die sich u. a. der maximalen Textlänge innerhalb einer Zelle, den arithmetischen Funktionen und speziellen Formaten, z. B. für Währungen, widmeten. Das Projektteam vermutete, dass dieses Release 2.2 die ältesten wk1-Dateien in der zu untersuchenden Sammlung erzeugt hatte. Nun stand eine Grundlage zur Bewertung von Konvertierungssoftware zur Verfügung. Die Vergleichsmethode mit den Ergebnisdateien war eine visuelle Inspektion, die ungefähr drei Stunden benötig-

te. Bewertet wurden die zwei kommerziellen Produkte *DataJunction 7.0* und *ConversionsPlus 4.5* (CP 4.5) mit folgenden Schwerpunktkriterien:

- Quell- und Zielformate
- Genauigkeit der Konvertierung
- Datendekomprimierung
- Stapelverarbeitung
- Erstellung von Fehlerberichten

DataJunction konnte zwar eine schnelle und genaue Konvertierung in das xls-Format durchführen, aber fehlende Fähigkeiten zur Datendekomprimierung und Stapelverarbeitung schränkten die Eignung ein. Während beide Programme einen großen Umfang an Quell- und Zielformaten bewältigten und Fehlerberichte liefern konnten, tauchten bei CP 4.5 Fehler u. a. bei der Behandlung von bestimmten Gleitkommazahlen auf. Daraufhin wurde mit dem weiteren Werkzeug Examiner die Dateisammlung auf dieses kritische Element hin untersucht. Da 96 Prozent der Dateien betroffen waren, wurde von einer Migration mittels CP 4.5 in das xls-Format abgesehen. Die Einordnung dieser Konvertierung erfolgte in die zweithöchste von fünf Risikoklassen.

*Risikobewertung*

Insgesamt wurden bei dieser Fallstudie drei Optionen für eine Risikoanalyse näher betrachtet:

*Abwärtskompatibilität*

- Die gute Unterstützung durch Lotus (zurzeit IBM) auch für ältere Formate wurde positiv bewertet, ebenso die Möglichkeit, ältere Anwendungsprogramme unter dem Betriebssystem DOS oder im DOS-Emulator unter Windows 95/98/NT laufen zu lassen. Trotz eines reduzierten Marktanteils wurde das Risiko dieser Option als niedrig eingeschätzt.

*Migration zu xls*

- Hier wurden die Marktführerschaft bei Windows- und Macintosh-Betriebssystemen sowie die breite Nutzerbasis hervorgehoben. Für die nächsten zehn Jahre wurde weiterhin eine intensive Nutzung vorausgesagt. Außerdem erhoffte man sich bei einer eventuellen Ablösung von Excel genügend Unterstützung durch Microsoft. Die bei der Konvertierung mit Excel 97 erzielten Ergebnisse anhand der wk1-Testdatei und weiteren zufällig gewählten Dateien waren positiv, sodass auch das Risiko dieser Option als niedrig betrachtet wurde. Für die Konvertierung selbst wurde Excel jedoch wegen der fehlenden Möglichkeit der Stapelverarbeitung als schlechte Lösung eingestuft.

*Migration zu ASCII*

- Das ASCII-Format wurde für große numerische Dateiarchive favorisiert, da es leicht manuell durchsucht und von vielen Programmen gelesen werden kann. Gestützt auf eine 30-jährige Erfahrung, wurde in ASCII auch für die kommenden 50 Jahre ein gebräuchliches Format gesehen. Unter entsprechenden Umständen wurde im Vergleich zur wiederholten Migration die einmalige Migration nach ASCII als kostengünstiger eingeschätzt. Bezüglich der Tabellenkalkulation Lotus war zwar eine Überführung auf primitivem Niveau möglich, indem die Werte der Tabelle auf eine Matrix mit x- und y-Koordinaten abgebildet werden. Aber Formeln, Funktionen, Zeiger und lange Zeichenketten, die je nach Konvertierungssoftware gekürzt wurden, hätten getrennt aufgezeichnet werden müssen, was bei großen Sammlungen unpraktisch wäre. ASCII wurde dennoch als eine risikoarme Alternative für wk1-Dateien eingeschätzt, falls die anderen beiden Optionen nicht realisierbar sind.

*Metadaten*

Zusätzlich wurden die Metadaten einer Risikobetrachtung unterzogen, wozu immerhin sechs Arten von Quellen zu untersuchen waren. In zwei Fällen war eine Anpassung erforderlich, die sich jedoch auf eine Aktualisierung der Erweiterung der Dateinamen beschränkt hatte und somit als risikoarm eingeschätzt wurde. Jedoch sollten die Metadaten zumindest um folgende Punkte ergänzt werden:

- Datum der Migration oder Modifikation
- betroffene Datenserien
- Beschreibung des Konvertierungsprozesses
- Identifizierung der Konvertierungssoftware
- Identifizierung der Risiken und ergriffene Maßnahmen
- Personen, die die Migration oder Modifikation beaufsichtigen

*Quantifizierbarkeit des Risikos*

In den abschließenden Betrachtungen zum Projekt wurde festgestellt, dass das Risiko einer Migration quantifizierbar ist, wobei das Risiko in Abhängigkeit vom Projektumfeld schwanken kann – manchmal sogar stark – und je nach Einflussgröße, z. B. Dateiformat, Hardware oder Personal, unterschiedlich genau bestimmt werden kann. Die größte Herausforderung lag bei der Interpretation des Risikos, um beispielsweise ein akzeptables Maß festzulegen. Obwohl Werkzeuge zur Risikobewertung Erfahrung und gutes Urteilsvermögen nicht ersetzen konnten, wurde eine Weiterentwicklung empfohlen. Was die Konvertierungssoftware anbelangt, so konnte keine Empfehlung für ein bestimmtes kostengüns-

tiges, kommerzielles Programm gegeben werden, aber das Potenzial wurde hoch eingeschätzt, auch angesichts der sonst anfallenden Kosten für Eigenentwicklungen.

*das Problem der Spezifikationen*

Der schwierigste Teil des Projekts war die Beschaffung vollständiger und verlässlicher Dateiformatspezifikationen. Beim TIFF-Format wurde beispielsweise beklagt, dass es keine klare, umfassende Beschreibung der Dateiformatspezifikationen oder Informationen über proprietäre Marken gab. Inwieweit die bei Adobe registrierten, aber nicht veröffentlichten Marken, die die zugänglichen Spezifikationen erweitern, in eine erwartete neue Formatversion eingehen, blieb unsicher. Es wurde daher das Konzept der offenen Spezifikationen unterstützt und empfohlen, dem koordinierten Zugang zu den relativ statischen, veröffentlichten Spezifikationen und den dynamischen, nicht standardisierten Elementen mehr Aufmerksamkeit zu widmen. Als sehr risikoreich wurden die öffentlichen (public domain) Archive im Internet angesehen, da sie keinen Plan vorwiesen, wie sie die Information erhalten, und schon auf Grund ihrer Natur als unzuverlässig eingeordnet wurden. Für den Aufbau verlässlicher und gepflegter Bestände mit Dateiformatspezifikationen, Dokumentationen und zugehöriger Software wurde dringender Bedarf gesehen.

*Formate bestimmen den Migrationsprozess!*

Insbesondere die beiden Fallstudien zeigen, welch detaillierte Betrachtungen in technischer und organisatorischer Hinsicht für eine Migration erforderlich sind, und wie stark die Gestaltung einzelner Migrationsschritte einschließlich Risikobewertung von einem spezifischen Format abhängen kann. Zu denken gibt auch der Hinweis, dass Spezifikationen offensichtlich sogar bei den Entwicklungsfirmen nicht vollständig verfügbar sind. Für die Erarbeitung von Migrationsstrategien ist zu berücksichtigen, dass die Rechte an einem Format den Eigentümer wechseln können, was mit einer Änderung der Lizenzpolitik verbunden sein kann. Weiterhin zeigt ein Blick in die Spezifikation von TIFF6.0,[6] wie sich auch technische Begriffe, die in Dokumentationen oder Programmen verwendet werden, mit der Zeit wandeln. So bezeichnet der Begriff Marke (engl. *tag*) nur noch die identifizierende Nummer eines Feldes, aber nicht mehr das gesamte Feld einschließlich Wert.

[6] siehe http://www.adobe.com/Support/TechNotes.html.

## 9.4 Datenbankmigration

*Loslösung digitaler Publikationen von ihren Ursprungsmedien und Überführung in ein Datenbank-Managementsystem*

Dieses durch die Deutsche Forschungsgemeinschaft (DFG) geförderte Projekt zur Langzeitarchivierung zeigt die Entwicklung eines technischen Konzepts, das durch Detailuntersuchungen kritischer Punkte und durch praktische Versuche ergänzt wird. Die vorgeschlagenen Lösungen orientieren sich an den Aufgaben einer wissenschaftlichen Universalbibliothek. Bei dem im Jahr 2002 abgeschlossenen Gemeinschaftsprojekt lieferte die Bayerische Staatsbibliothek (BSB)[7] eine fachbezogene Problemanalyse und eine Reihe von Denkanstößen, die für das Institut für Softwaretechnologie der Fakultät für Informatik an der Universität der Bundeswehr München (UniBwM)[8] als Grundlage dienten. Mit Hilfe der Bibliothek der UniBwM konnten vor Ort die praktischen Versuche durchgeführt werden. Konkret wurde untersucht, was ein kommerzielles Standarddatenbank-Managementsystem bei der Langzeitarchivierung leisten kann, insbesondere wenn digitale Objekte von ihren ursprünglichen Medien gelöst und voll in ein Datenbank-Managementsystem (DBMS) integriert werden können. Dieser Prozess wird für wichtige Medien und Formate unter den Aspekten der Verwaltung, der Nutzung und der Archivierung diskutiert. Abgesehen von der Migration auf Datenträgerebene bedeutet der Einsatz eines DBMS keine Festlegung auf eine Migrations- oder Emulationsstrategie.

### 9.4.1 Motivation

**Motivation 1: Endgültiges Lösen digitaler Objekte vom ursprünglichen Medium**

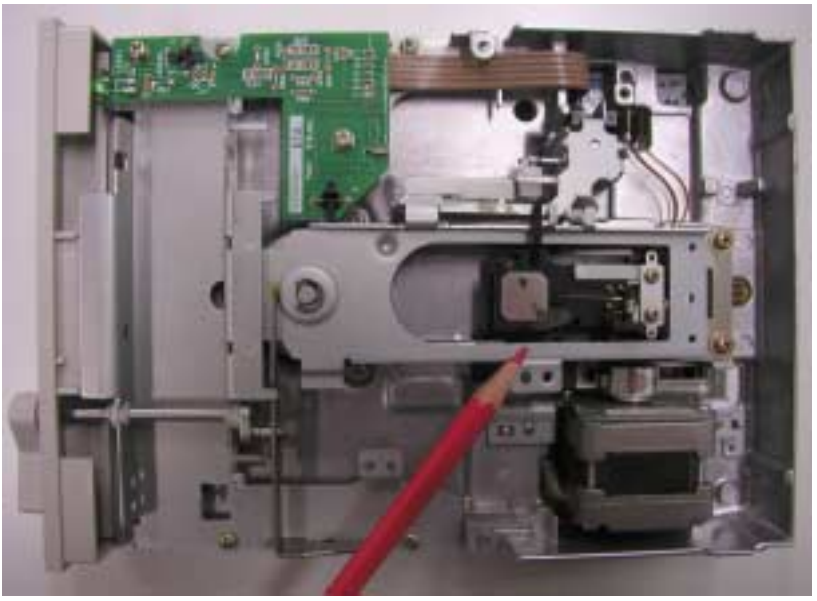
*technische Normierung durch Loslösung*

Unabhängig vom Einsatz eines DBMS bringen das endgültige Lösen digitaler Objekte von ihren ursprünglichen Medien und die Übertragung in ein aktuelles System eine Reihe von Vorteilen. So können zukünftige Migrationsschritte, die der Fortschritt in der Speichertechnologie erfordert, räumlich und technisch konzentriert vorgenommen werden. Die physischen Eigenschaften der Medien (Abnutzung, Alterung, Beschädigung durch unsachgemäßen Gebrauch – s. auch Abb. 9.3) und der technische Stand ihrer zugehörigen Hardware und hardwarenahen Software zum Lesen müssen nicht mehr berücksichtigt werden. Es findet in dieser Hinsicht eine technische und altersmäßige Normierung statt. Da die

[7] siehe http://www.bsb-muenchen.de.
[8] siehe http://ist.unibw-muenchen.de/LZA.

Medien den Nutzern entzogen werden, ist der Missbrauch, z. B. durch Austausch, oder die absichtliche oder unabsichtliche Manipulation von beschreibbaren Medien ausgeschlossen, z. B. durch Viren oder versehentliches Überschreiben. Der Nutzer hat den Vorteil, dass er nicht mehr auf ausgeliehene Medien stößt und auch keine spezielle Hardware zum Lesen benötigt. Teilweise wird sogar ein schnelleres und angenehmeres Arbeiten ermöglicht, z. B. geringere Lautstärke oder verminderter Energieverbrauch bei mobilen Geräten. Und beim Nutzer nicht mehr anzutreffende, überalterte Geräte geben keinen Anlass, Publikationen auszusondern.

**Abbildung 9.3**
*Hardware-Schaden durch Korrosion*



**Motivation 2: Einsatz eines Datenbank-Managementsystems**

*Integrität und Wirtschaftlichkeit*

Bei der Langzeitarchivierung fallen mit Sicherheit große und komplex strukturierte Datenbestände an. Ein DBMS bietet dazu eine Reihe von Diensten an, die das Management dieser Bestände unterstützen oder gar erst wirtschaftlich ermöglichen. Durch die Abbildung eines Modells der realen Welt auf ein logisches Schema mit Hilfe von Datendefinitionssprachen und durch entsprechende Datenmanipulationssprachen erlauben DBMS das gezielte, effiziente und sichere Operieren auf Datenbeständen. Dabei garantiert das Datenbanksystem entsprechend dem vorab definierten Schema die Persistenz und Integrität der Daten, auch bei Mehrbenutzerbetrieb. Ebenso ist die Integrität bei Hard- und Softwarefeh-

lern weitgehend gesichert, hauptsächlich indem nicht vollständig ausführbare Transaktionen (z. B. bei einem Absturz des Betriebssystems) zurückgesetzt oder ausgehend von einem gesicherten, konsistenten Datenbestand, als *Backup* bezeichnet, nachgeholt werden (z. B. bei einem Fehler der Festplatte). Der Prozess der Wiederherstellung wird im Allgemeinen als *Recovery* bezeichnet.

*Abstraktion von konkreten Speichermedien*

Neben diesen allgemeinen Vorteilen, die bereits sehr häufig für Katalog- und Ausleihsysteme genutzt werden, liegt der spezielle Nutzen für die Langzeitarchivierung in der Konzentration künftiger Migrationsschritte, die durch die Speichertechnologie bedingt sind, in einem einzigen System. Das DBMS stellt die logische Schnittstelle zu den Medien für die permanente Speicherung dar. Im Idealfall ist es für den Nutzer und Verwalter nicht sichtbar, wie die Persistenz eines digitalen Objekts – auch langfristig – realisiert wird. Hierfür sind der Datenbankdesigner und der Datenbankadministrator verantwortlich. Das Datenbanksystem sollte hierzu die nötigen Abstraktionshilfen bieten.

*Online-Verfügbarkeit digitaler Objekte*

Aus Sicht der Verwaltung ist es vorteilhaft, dass die digitalen Objekte übergreifend analysiert werden können. Die digitalen Objekte sind dadurch jederzeit für die Generierung von Metadaten, für die Erstellung von Varianten (z. B. komprimierte Fassung für ein Previewing), für eine Konvertierung (z. B. auf Grund technischer Obsoletheit irgendeines Formats) oder für eine direkte, inhaltsbezogene Recherche zugänglich.

*Konsistenz mit den Metadaten*

Darüber hinaus sichert das DBMS die Konsistenz zwischen den eigentlichen Daten und den Metadaten sowie ggf. den passenden Softwareumgebungen einschließlich Emulationssoftware. Die eigentlichen Daten können in abgetrennten eigenen Behältern in ihrer ursprünglichen und/oder einer konvertierten Form als Bitstrom aufbewahrt werden. Ein digitales Objekt kann mit seinen Metadaten oder einer Teilmenge davon als neue logische Einheit in einem datenbankunabhängigen Format exportiert werden. Dabei kann bei Bedarf durch Auflösung aller Referenzen eine abgeschlossene physische Kapsel generiert werden.

*Überlebensfähigkeit der Datenbanktechnologie*

Die Überlebensfähigkeit von Datenbank-Managementsystemen als Technologie kann als sehr hoch eingeschätzt werden, da für alle Informationssysteme die Forderung nach Persistenz, Integrität sowie einfachem und schnellem Zugang grundsätzlich gegeben ist. Und es kann sogar mit einer Unterstützung der Migration für bestimmte Datentypen gerechnet werden.

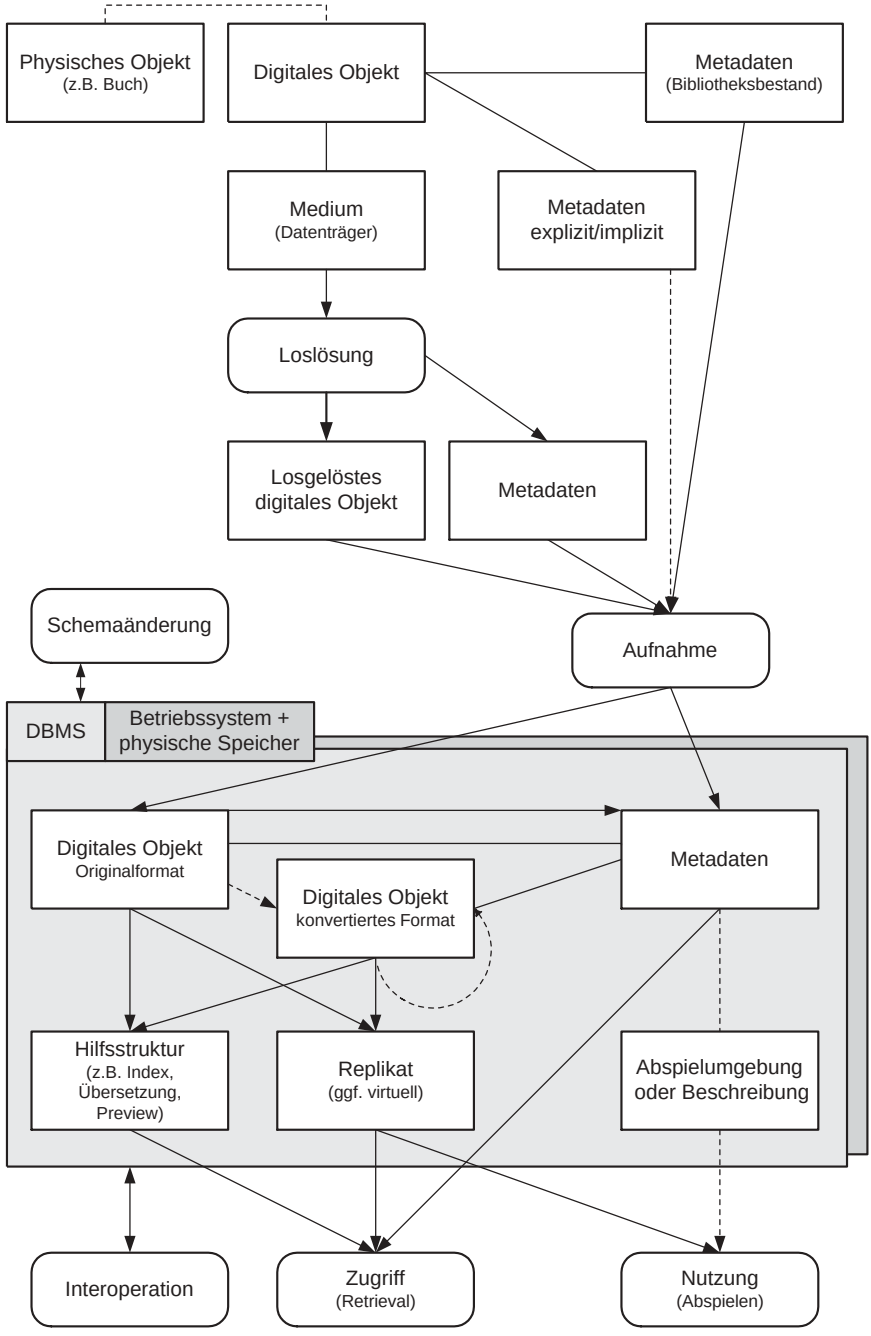



**Abbildung 9.4**
*Gesamtarchitektur: Prozesse und Objekte*

### 9.4.2 Gesamtarchitektur

*wichtige Prozesse und Objekte*

Es werden die wichtigsten Prozesse und Objekte in einem Gesamtüberblick dargestellt, wobei schwerpunktmäßig digitale Objekte betrachtet werden, die an Medien gebunden und somit nicht direkt online verfügbar sind (vgl. Abb. 9.4). Daher ist der Aufnahme in das DBMS der Loslösungsprozess vorgeschaltet, welcher nicht nur ein losgelöstes digitales Objekt erzeugt, sondern je nach Methodik und Werkzeug auch Metadaten generieren kann, die bisher nur implizit vorhanden sind und durch die Ablösung bedingt neue, meist technische Metadaten erfordert. Der Prozess der Aufnahme soll bestehende Metadaten z. B. aus Bibliothekskatalogen übernehmen und ggf. auch explizit und implizit im digitalen Objekt vorhandene Metadaten aufbereiten. Diese Schritte können aber auch nach Aufnahme in das DBMS erfolgen; der aufwändige Zugriff auf das Medium entfällt. Das DBMS selbst ist nun das Archiv und soll soweit wie möglich von Betriebssystemen und physischen Speichern abstrahieren, um das Gesamtmanagement zu erleichtern. Die Metadaten im DBMS unterstützen sowohl die Erschließung und Rekonstruktion (Abspielbarkeit) intellektueller Inhalte als auch alle Aufgaben der Verwaltung einschließlich der Langzeitaspekte. Das Objekt Abspielumgebung beschreibt, dass eine passende Abspielumgebung direkt im DBMS gespeichert sein kann oder aber nur eine Beschreibung davon, wobei dann die Zuständigkeit für eine Ablaufumgebung beim Nutzer liegt. Bei einem Konvertierungsansatz mit Berücksichtigung der jeweils aktuellen und gängigen Systeme entfällt die Abspielumgebung. Das Modell schließt nicht aus, dass Emulationen auch auf konvertierten Formaten Anwendung finden, die in mehreren Versionen vorhanden sein können. Das Objekt Hilfsstruktur soll den Zugriff auf intellektuelle Inhalte erleichtern; in diesem Sinne kann es als eine spezielle Art von Metadaten gesehen werden. Das Objekt Replikat soll zeigen, dass der Nutzerzugriff nicht direkt auf die digitalen Objekte erfolgt, z. B. aus Gründen der Zugriffsgeschwindigkeit oder der Sicherheit, wobei die Replikate nicht unbedingt physische Kopien sein müssen, sondern auch spezielle Sichten mit eingeschränkten Rechten sein können.

*Ausprägungen digitaler Objekte*

Abhängig von den ursprünglichen digitalen Objekten, auf deren Form man – wie in anderen Modellen erhofft oder angenommen – keinen Einfluss hat, können digitale Objekte im DBMS eine sehr unterschiedliche Ausprägung haben; diese kann von einem einfachen, in sich abgeschlossenen Informationsobjekt bis hin zu einer technisch heterogenen Sammlung von Objekten mit ei-

genem Ordnungssystem (z. B. Dateisystem) einschließlich eigener Abspielumgebung reichen. Der Prozess Interoperation soll die Zusammenarbeit mit anderen Systemen ggf. auch mit eigenen alten Systemen sicherstellen und kann alle Objekte im DBMS betreffen, wobei das DBMS die koordinierende und kontrollierende Instanz ist.

*OAIS*

Eine Zuordnung zu den Elementen des OAIS-Referenzmodells ist nicht ohne weiteres erkennbar. Dies liegt an der Virtualisierung durch das DBMS. So existieren das *Archival Information Package* (AIP) und das *Dissemination Information Package* (DIP) nicht mehr explizit. Es wird die Fähigkeit des DBMS ausgenutzt, Informationen in geeignete Module zerlegt speichern und nach Bedarf mit Hilfe von Verbundoperatoren zusammensetzen zu können. Der Zusammenhalt der Module untereinander wird vom DBMS gesichert, wenn die Regeln im Datenbankschema deklariert sind.

*Beschreibung der wichtigsten Prozesse*

In der vorgestellten Architektur wird sichtbar, dass dem DBMS eine tragende Rolle zugeordnet wird. Welche konkreten Dienste ein DBMS übernehmen kann und was beim Loslösungsprozess beachtet werden muss, wird im Folgenden diskutiert. Dazu werden die Prozesse Loslösung, Zugriff, Nutzung und Interoperation sowie als spezieller Teil der Aufnahme die Speicherung im DBMS näher vorgestellt. Abschließend werden die Erzeugung von Metadaten, die mehreren Prozessen zugeordnet sein kann, und die Langzeitarchivierung in Verbund mit einem DBMS in Hinblick auf Schemaänderungen und physische Speicherung untersucht.

### 9.4.3 Endgültiges Lösen digitaler Objekte vom ursprünglichen Medium

*Vielfalt der Speichermedien*

**Ausgangsmedien** In den letzten Jahrzehnten hat sich eine Vielzahl von Datenträgern und Formaten für die permanente Speicherung von Daten gebildet. Der Hersteller einer Diskettenkonvertierstation nennt allein für Disketten die Anzahl von über eintausend Formaten.[9] Für eine Bibliothek im klassischen Sinne dürfte die Problematik nicht so extrem sein, da digitale Medien lange Zeit vor allem als Beilage zu Print-Medien dienten. Die Beilagen waren aus praktischen Gründen beim Versand oder der Aufbewahrung im Buch an bestimmte physische Grenzen gebunden, und aus Sicht der Verleger bzw. der Herausgeber sollte eine breite Verwendbarkeit gegeben sein.

[9] GEVA GICO 8620 Benutzerhandbuch, Kopie ohne Datum.

**Kontext und Struktur der Medien** Für den Entwurf eines Datenbankschemas ist die Entwicklung eines konzeptionellen Modells erforderlich. Es muss u. a. den Kontext und die internen Strukturen des Mediums in Hinblick auf die Loslösung des digitalen Objekts beschreiben, sofern sie für den vom Erzeuger beabsichtigten Zugang erforderlich sind.

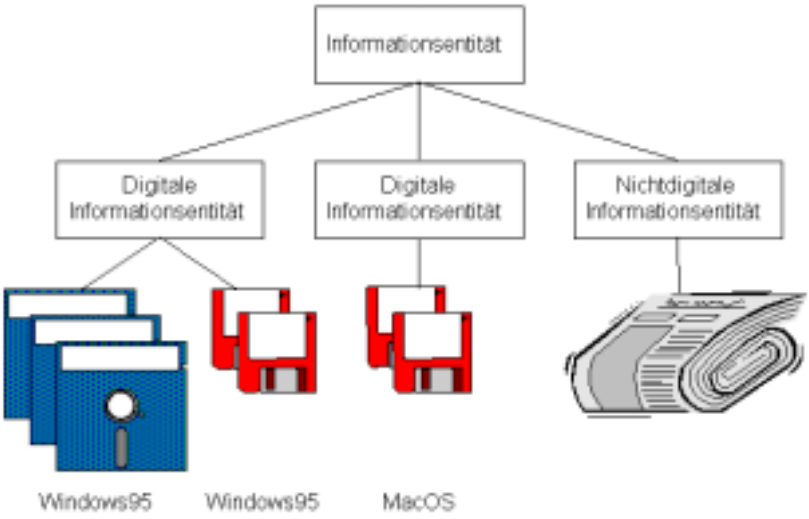



**Abbildung 9.5**
*Kontext der Medien: ein Beispiel für Redundanz und Unterstützung mehrerer Zielumgebungen*

Zunächst zum Kontext eines Mediums (vgl. auch Abb. 9.5): Der einzelne Datenträger soll hierzu genau eine digitale Informationsentität oder aber nur einen Teil davon (z. B. eine einzelne Diskette eines Satzes von zusammengehörigen Datenträgern, die auch von unterschiedlichen Typ sein können) repräsentieren.

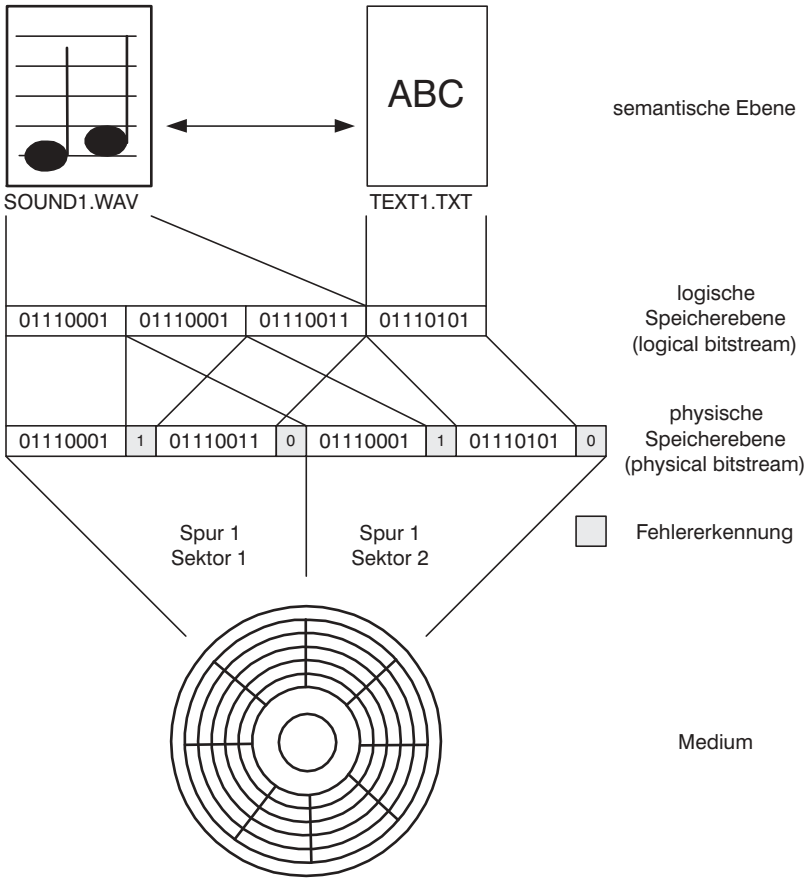
**Erklärung:** Unter *Informationsentität* verstehen wir eine vom Erzeuger oder von einer Bibliothek definierte Informationseinheit, deren Komponenten digitaler und nichtdigitaler Art sein können. Dieses Modell unterstellt insoweit Abgeschlossenheit, als die Medien nicht Teil einer weiteren Informationseinheit sein können. Insbesondere in der Übergangsphase von Speichertechnologien kommt es vor, dass Datenträger von unterschiedlichem Typ eine identische digitale Informationsentität repräsentieren. Diese Redundanz kann in der Regel beseitigt werden. Zu beachten ist jedoch, dass Datenträger von unterschiedlichem Typ eine identische Informationseinheit, aber keine identische digitale Informationseinheit repräsentieren können: Ein Datenträger vom Typ A kann für ein Zielsystem X bestimmt sein und ein Daten-

träger vom Typ B für ein Zielsystem Y. Dieser Sachverhalt ist nicht immer aus den expliziten Metadaten bzw. Beschriftungen oder Beilagen der Datenträger erkennbar. Andererseits kann ein einzelner Datenträger mehrere Zielsysteme unterstützen. Bei Disketten geschieht dies oft physisch, indem die einzelnen Seiten für unterschiedliche Zielsysteme vorgesehen sind, bei CDs und DVDs geschieht dies auf der logischen Speicherebene, die durch das Zielsystem interpretiert wird.

*nichtdigitale Metadaten*

Für die Erzeugung von Metadaten muss berücksichtigt werden, dass sich relevante Informationen auch direkt auf dem Medium, auf dem Aufbewahrungsbehälter oder in den zugehörigen Print-Medien befinden können.

**Abbildung 9.6**
*Vereinfachtes Schichtenmodell zur internen Struktur von Medien*




*Struktur der Medien*

Nun zu den internen Strukturen eines Mediums (vgl. auch Abb. 9.6): Der einzelne Datenträger besitzt in sich eine eigene spe-

zifische logische Struktur, die den vom Erzeuger beabsichtigten Zugang sicherstellt, z. B. dateiübergreifende Hyperlinks oder die Synchronisation von Bilddaten mit Audiodaten. Unter dieser *semantischen Ebene* befindet sich eine für den Anwender i. d. R. unsichtbare Struktur, die die Rekonstruierbarkeit auf der *logischen Speicherebene* (logical Bitstream) sicherstellt, z. B. die Zuordnung von Datenblöcken zu Dateien. Unter dieser Schicht liegt die *physische Speicherebene* oder *Aufzeichnungsebene* (physical bitstream), die überhaupt erst das sichere Lesen und Schreiben von Bitströmen von bzw. auf ein physisches Medium durch Transformationen und Ergänzungen der logischen Speicherebene ermöglicht, z. B. durch Fehlerkorrektur-Bits. Daneben sind ggf. medienspezifisch digitale Zusatzdaten vorhanden, die von den Schreibaktionen des Erzeugers abhängig (z. B. Volume-Deskriptoren) oder unabhängig (z. B. Hersteller eines CD-R-Rohlings) sind.

**Loslösung** Im Allgemeinen braucht die unterste Ebene nicht übertragen zu werden. Oberhalb dieser Ebene sind zwei grundlegende Fälle zu unterscheiden.

*Übertragung als Image*

Im ersten Fall werden alle anderen Ebenen in ihrer ursprünglichen Form als so genanntes *Image* (Abbild) in eine einzige Datei (in Sonderfällen auch mehrere Dateien) übernommen, im Gegensatz zum üblichen Kopieren, z. B. mit dem Microsoft-Explorer, jedoch unbeeinflusst vom Datei- und Betriebssystem des imageerzeugenden Systems. Entsprechend dem Ausgangsmedium werden solche Dateien auch als *Disk-* oder *Tapeimage* bezeichnet. Da nun das ursprüngliche Gerät und die hardwarenahe Software nicht mehr zur Verfügung stehen, müssen diese emuliert werden, z. B. durch virtuelle Laufwerke.

*Übertragung in ein neutrales Strukturformat*

Der zweite Fall besteht aus einer semantikerhaltenden Übertragung in ein neutrales Strukturformat wie XML – statt z. B. einem hierarchischen Dateisystem – oder ein passendes Datenbankschema. Dies wäre eine Migration aus einem logischen Speicherformat und ggf. einem konkreten Dateisystem oder sonstigem Ordnungssystem. Bei diesem Ansatz entfällt die Emulation der Geräte und ggf. des dazugehörigen Dateisystems bzw. Ordnungssystems. Es wird durch abstraktere Mechanismen ersetzt.

*Bewertung der beiden Methoden*

In beiden Fällen findet keine Konvertierung der eigentlichen Inhaltsdaten (content migration) statt. Die Komplexität des Zugriffs sowohl auf der Anwenderebene (z. B. Datei als semantisches Strukturierungselement) als auch auf der Speicherebene dürfte eine Überführung in abstraktere Darstellungen jedoch häufig verhindern. Andererseits gibt es bereits Emulationen, die genau ein

Image eines Datenträgers als Eingabe erwarten. Dieser Sachverhalt hat Konsequenzen für die Archivierungsstrategie. Die Konvertierung einer Datei innerhalb eines Image könnte beispielsweise die gesamte Speicherstruktur zerstören.

Bei den oben genannten digitalen Zusatzdaten ist wahrscheinlich eine Einzelfallentscheidung zweckmäßig, die u. a. vom Umfang und der Qualität abhängen kann.

*manipulierte und beschädigte Medien*

Besondere Behandlung erfordern manipulierte oder beschädigte Medien. Die Unterscheidung ist nicht immer leicht zu treffen, da durch die Manipulationen, die bis in die unterste Schicht reichen können, Fehler oft absichtlich, im Wesentlichen aus Kopierschutzgründen, erzeugt werden sollen. Im Bereich der CD-DA hat dies in letzter Zeit exzessive Maße angenommen. Dieses Problem sollte, abgesehen von rechtlichen Gesichtspunkten, wegen des hohen technischen Aufwands auf einer vertraglichen Ebene gelöst werden.

*die wichtigsten Medien*

Nun zu den für eine Universalbibliothek wichtigen Medien und ihren Eigenschaften, die für den Loslöseprozess von Bedeutung sind.

*Disketten*

**Diskette** Disketten waren das dominierende Austausch- und Verteilmedium der letzten 20 Jahre. Ursprünglich wurden Disketten ab 1971 zur Versorgung von Großrechnern mit System- und Diagnosesoftware im 8-Zoll-Format eingesetzt (Brandel 1999). Obwohl auch Heimcomputer damit ausgestattet waren, blieb der Einsatz weitgehend dem professionellen Bereich vorbehalten. Die Ablösung im Heimcomputerbereich erfolgte ab 1976 durch das 5,25-Zoll-Format, das ab 1987 mit Einführung des IBM PS/2 allmählich durch das 3,5-Zoll-Format ersetzt wurde. Obwohl an die heutigen Floppy-Controller mit dem entsprechenden Bus-Kabel noch ein 5,25-Zoll-Laufwerk angeschlossen werden kann, ist dieses Format praktisch verschwunden. Ähnliches zeichnet sich durch den Preisverfall der optischen Medien und der zugehörigen Lese- und Schreibgeräte sowie durch die zunehmende Vernetzung für das 3,5-Zoll-Format ab. Neben den dargestellten *Formfaktoren* (Zoll-Formate) sind für das Lesen das Layout der *Spuren* (tracks) und *Sektoren* (sectors), das Aufzeichnungsverfahren und das Dateisystem entscheidend. Bei den Aufzeichnungsverfahren sind die wichtigsten *MFM* (Modifizierte Frequenzmodulation) und *GCR* (Group Code Recording). Während bei den Standard-PCs MFM und beim 8-Zoll-Format der Vorgänger von MFM, nämlich *FM* (Frequenzmodulation), üblich ist, verwendeten viele 8-Bit-Heimcomputer (u. a. der Commodore C64 mit seiner großen Ver-

breitung und auch die frühen Apple-Rechner) GCR und zwar bei allen Formfaktoren. Zu beachten ist, dass damit aufgezeichnete Daten auf keinen Fall mit den Controllern der Standard-PCs gelesen werden können. Zum Lesen und Erzeugen eines Image muss entweder die Originalhardware, die hinsichtlich der Aufzeichnungsverfahren teilweise hybrid ist, komplett bereitgestellt oder das Originallaufwerk mit spezieller Software und ggf. Hardware (z. B. über die parallele Schnittstelle) an ein aktuelles System angeschlossen werden. Eine weitere Möglichkeit bei 5,25- und 3,25-Zoll-Formaten ist in vielen Fällen die Nutzung von aktuellen Standardlaufwerken mit Hilfe spezieller Laufwerkscontroller und Software. Zu beachten ist noch, dass unterschiedliche Werkzeuge unterschiedliche Images erzeugen können, was u. a. an einer unterschiedlichen Einschätzung der Relevanz von Zusatzdaten, z. B. Disk-ID, liegt.

*CD*

**Compact Disc** Compact Discs (CD) unterscheiden sich nicht nur in der Speicherkapazität und im physikalischen Aufzeichnungs- und Leseverfahren von Disketten, sondern auch konzeptionell bezüglich der ursprünglichen Zielsetzung, des Grades an Standardisierung und der Möglichkeit, datenträgerbezogene Beschreibungen aufzunehmen.

*CD-DA*

Als *CD-DA* (Digital Audio) von Sony und Philips 1980 entwickelt (Rosenbaum 2001) war sie ursprünglich nicht als Speichermedium für Rechner vorgesehen, sondern für Abspielgeräte, die das digitale Signal in ein analoges umwandeln. Wichtig aus der Sicht des Loslösungsprozesses ist, dass es keine exakte Adressierbarkeit der logischen Speichereinheiten, nur eine eingeschränkte Fehlererkennung und -korrektur und kein Dateisystem gibt, was für das Abspielen eines Audiostroms in einem CD-Player auch nicht erforderlich ist.

*CD-ROM und ISO 9660*

Dies änderte sich mit der Spezifikation der *CD-ROM* (Read Only Memory) 1983 von Sony und Philips. Neben der Standardisierung der technisch notwendigen Erweiterungen für Rechnerdaten wurde im Jahr 1986 ein plattformunabhängiges Dateisystem, bekannt als ISO 9660[10], definiert, das zusätzlich die Aufnahme datenträgerbezogener Beschreibungen vorsieht. So können im *PVD* (Primary Volume Descriptor) bibliographische Daten direkt oder in einer Datei, auf die von dort aus verwiesen wird, gespeichert werden.

[10] ECMA, Volume and File Structure of CDROM for Information Interchange, Genf, Reprinted 11.98, 12.1987.

*Erweiterung zu ISO 9660 und proprietäre Formate*

Auf Grund der Restriktionen von ISO 9660, z. B. die Beschränkung der Länge von Dateinamen, der Schachtelungstiefe von Dateiverzeichnissen oder der zulässigen Dateiattribute, fanden mehrere Erweiterungen statt. So berücksichtigt das *Rock Ridge Interchange Protocol* (RRIP)[11] die Belange des *POSIX*-Dateisystems und die *Joliet*-Spezifikation[12] die Belange von Microsoft-Windows. Um die CD-ROM auch als Boot-Medium nutzbar zu machen, wurde von IBM und Phoenix die *El Torito*-Spezifikation[13] entwickelt, die auch die Verwaltung mehrerer so genannter *Boot Disk Images* erlaubt (Diskettenabbilder deswegen, weil in der PC-Welt Disketten das ursprüngliche Boot-Medium sind). Neben den reinen *HFS*-CD-ROMs (Hierarchical File System), die ohne Zusatzsoftware nur von Apple-Rechnern gelesen werden können, gibt es auch von der Firma Apple definierte Erweiterungen[14] zu ISO 9660. Alle genannten ISO-kompatiblen Erweiterungen können, obwohl sie keinen einheitlichen Erweiterungsmechanismus verwenden, auf einer einzigen CD vorkommen, wobei sie von den Zielsystemen automatisch passend interpretiert werden.

*CD-R*

Mit der Definition des Standards für wiederbeschreibbare CDs 1990, als *CD-R* (Recordable) bezeichnet, wurde auch die Erstellung von kleinen Serien, z. B. für Dissertationen oder Tagungsberichte, wirtschaftlich. Dieser Standard führt keine neuen Formate für die eigentlichen Inhalte ein.

*weitere CD-Formate*

Neben den Hauptformaten DA und ROM gibt es noch weitere Formate. Für multimediale Anwendungen wurden Formate entwickelt, die im Wesentlichen die Synchronisation von Daten und Audioinformationen sowie die Kompression von Audio- und Videoinformationen betreffen wie MPEG-1 oder *VBR* MPEG-2 (Variable Bit Rate). Die *Tracks* (maximal 99) als logische Unterteilung einer CD können auch von unterschiedlichem Typ (mixed mode) sein, z. B. ein Track mit Daten und weitere mit Audio.

*Zusatzdaten auf CD*

Für den Prozess der Loslösung, wie in Abb. 9.4 skizziert, können auch die acht so genannten *subcode channels* oder *subchannels* P bis W wichtig sein, die je nach Format eine unterschiedliche

[11] Institute of Electrical and Electronics Engineers, Inc. (IEEE), IEEE P1282, Rockridge Interchange Protocol, Draft Standard Version 1.12, Piscataway, 08.07.1994.

[12] Microsoft Corporation, Joliet Specification, Extensions for Unicode Version 1, 22.05.1995.

[13] Phoenix Technologies, IBM, »El Torito« Bootable CD-ROM Format Specification Version 1.0, January, 1995.

[14] Apple, Technical Note FL36, Apple Extension to ISO 9660, 01.07.1989.

Rolle spielen können. Für CD-ROMs sollten sie keine direkte Bedeutung haben, ansonsten können sie Nutzdaten (auch beschreibender oder identifizierender Art) enthalten. Im Channel Q, der auch für die Zeitangaben zuständig ist, können Tracks durch *ISRC* (International Standard Recording Code mit Land, Eigentümercode, Jahr, Seriennummer) und ganze CDs durch *UPS/EAN* (Universal Product Code/European Article Number) eindeutig identifiziert werden. Es gibt jedoch Hinweise, dass dies nur spärlich genutzt wird.[15]

*UDF*

Als medienspezifisches Dateisystem ist auch *UDF* (Universal Disk Format) möglich; es wird unter dem Punkt *Digital Versatile Disc* (DVD) beschrieben, weil es dort die deutlich größere Bedeutung hat.

*Erzeugung eines Image*

Wie bei den Disketten wird i. d. R. spezielle Software für die Extraktion bzw. für die Erzeugung eines Image benötigt. Da das Brennen von CDs zum »Volkssport« geworden ist, besteht eine große Auswahl an Werkzeugen, wobei berücksichtigt werden muss, dass unterschiedliche Formate erzeugt werden. Zielsetzung der meisten Werkzeuge ist auch nicht die Erzeugung eines (normierten) Image, sondern eine neue CD, möglichst unter Umgehung aller Kopierschutzmechanismen. Für den Loslösungsprozess relativ problemlos sind CD-ROMs mit einem ISO-Dateisystem.

*DVD*

**Digital Versatile Disc** Digital Versatile Discs (DVD) sind der CD konzeptionell sehr ähnlich. Gegenüber der CD haben sie ein einheitliches physikalisches Format und mit dem *Universal Disk Format* (UDF)[16] auch ein einheitliches, übergeordnetes logisches Medienformat (Medienstruktur und Dateistruktur). Auch Video- und Audiodaten, die genau spezifiziert sind, sollen in Dateien gehalten werden (DVD-ROM mit abgemagertem UDF). Im Gegensatz zu ISO 9660 berücksichtigt UDF die Heterogenität bestehender Dateisysteme nicht durch die Definition einer gemeinsamen Teilmenge der Eigenschaften, sondern durch die Definition spezifischer Erweiterungen und deren Behandlung durch fremde Systeme. Darüber hinaus ist eine zu ISO 9660 kompatible Erweiterung, als *UDF Bridge Disk* bezeichnet, vorgesehen. Dominiert bisher die Video-Anwendung (DVD-Video), so dürfte die reine DVD-ROM zunehmend Verbreitung finden, wobei Softwaredistributio-

[15] Padus Incorporated, DiscJuggler User's Guide, Document status: 3.50.10, 15.01.2002.

[16] Optical Storage Technology Association (OSTA), Universal Disk Format Specification, Revision 2.01, 15.03.2000.

nen oder Enzyklopädien schon seit einiger Zeit angeboten werden. Die DVD-Audio ist bisher eher selten; die Abspielgeräte sind noch relativ teuer. Im Gegensatz zur CD-DA ist sie, wie viele DVD-Videos, mit Bonusmaterial (im so genannten *ROM-Bereich*) angefüllt. Dies bringt natürlich ein neues Maß an technischer und inhaltlicher Heterogenität mit sich. Der Preisverfall bei den DVD-Brennern dürfte das Medium auch für kleine Auflagen attraktiv machen.

### 9.4.4 Dienste eines Datenbank-Managementsystems

Im Folgenden wird dargestellt, welche Aufgaben ein kommerzielles Standard-Datenbanksystem im Kontext der Langzeitarchivierung übernehmen kann.

*logische Datenbankmodelle*

**Datenbanktechnologie – Überblick** Das vorherrschende logische Datenbankmodell ist zurzeit das Relationenmodell mit der standardisierten Datenbanksprache *SQL*[17]. Während zu Beginn vor ca. 25 Jahren nur einfache Datentypen und -strukturen zur Verfügung standen, finden seitdem ständig Erweiterungen statt. So werden Konzepte der Objektorientierung, angetrieben durch die Entwicklung objektorientierter Programmiersprachen und Datenbanksysteme, in so genannten *objektrelationalen* Systemen unterstützt. Semantisch reichere Modelle sind auch dadurch möglich geworden, dass selbstdefinierte Funktionen und Prozeduren, als *stored functions* bzw. *stored procedures* bezeichnet, innerhalb eines Datenbanksystems definiert, verwaltet und ausgeführt werden können. Ebenso wird die Dezentralisierung unterstützt, indem sich Datenbanken, als *verteilte Datenbanksysteme* bezeichnet, unter Beibehaltung einer logischen Sicht physisch verteilen lassen. Außerdem bieten Datenbanken Unterstützung für eine Einbindung in das Internet. Der Zugriff über die plattformunabhängige Programmiersprache Java, die wie bereits beschrieben in einer virtuellen Maschine läuft, ist client- oder serverseitig mit der Schnittstelle *JDBC* (Java Database Connectivity) möglich. Zur Verwaltung großer unstrukturierter oder semistrukturierter Datenobjekte wurden neue Datentypen u. a. zur Unterstützung von Multimediaanwendungen bereitgestellt. Für den Datenaustausch zwischen heterogenen Datenbanksystemen bekommt XML eine starke Bedeutung. Auch diese Technik wird in relationale Daten-

[17] ISO/IEC 9075-1:1999 Information technology – Database languages – SQL – Part 1: Framework (SQL/Framework), 28.01.2000.

banken integriert. Parallel dazu gibt es spezialisierte Systeme für die Objektorientierung (durch ein Firmenkonsortium standardisiert)[18] und ebenso für XML (Schöning 2001), wobei erstere eindeutig ein Nischendasein führen. Die folgenden Ausführungen beziehen sich auf relationale DBMS.

*Speicherung im DBMS als large object (LOB)*

**Speicherung im DBMS** Als Produkt des Loslösungsprozesses kann ein Image des Datenträgers in Form einer Datei entstehen, die als Zwischenlager für die exakte Bitfolge der Nutzdaten des Datenträgers dient. Genau diese Bitfolge soll im Datenbanksystem gespeichert werden. Dazu kann man sich der so genannten *LOBs* (large object) bedienen, die nochmals nach *BLOBs* (binary large object) und *CLOBs* (character large object) differenziert werden. Beides sind Standardtypen eines relationalen Datenbanksystems und können wie jeder andere Typ bei der Definition einer Relation benutzt werden. Das Image steht nach dem Laden in einem Feld vom Typ BLOB voll unter der Kontrolle des Datenbanksystems. Mit einigen Einschränkungen stehen, soweit für diesen Typ sinnvoll, alle Datenbankoperationen zur Verfügung. Insbesondere ergeben sich folgende Vorteile: Zugriffsrechte können einheitlich vergeben werden, das Image steht unter der Transaktionskontrolle des Datenbanksystems, Backup- und Recovery müssen nicht mehr getrennt auf Dateisystemebene durchgeführt werden und die Konsistenz zwischen Metadaten und Image wird entsprechend dem definierten Datenbankschema gesichert.

*Repräsentation der logischen Struktur eines digitalen Objekts*

Gelingt es, die innere Struktur des Image aufzulösen oder findet man eine brauchbare Struktur bereits auf den Datenträgern vor, so kann eine Abbildung auf semantisch reichere Schemaobjekte des Datenbanksystems stattfinden. Ein Beispiel könnte die Übertragung von Audiotracks einer CD in Datenbankfelder vom Typ BLOB sein. Die dazugehörigen Metadaten werden in geeigneten Feldern gespeichert, wozu es bereits Werkzeuge zur Unterstützung der Definition von Datenstrukturen gibt.[19] Das Ganze kann dann als eigener Typ, hier AUDIO genannt, mit spezifischen Funktionen innerhalb der Datenbank definiert werden. Die Definierbarkeit eigener Typen mit dazugehörigen Methoden ist eine Eigenschaft der Objektorientierung, die die kompakte Darstellung fachlich zusammengehöriger Sachverhalte (Daten + Methoden) er-

[18] Object Database Management Group (ODMG),
siehe http://www.odmg.org.

[19] Oracle interMedia Annotator User's Guide Release 9.0.1.
siehe http://www.oracle.com.

laubt und somit die Entwicklung und Pflege von Anwendungen erleichtert.

*Einschränkungen und Weiterentwicklungen*

Eine Einschränkung besteht zurzeit darin, dass die Größe der LOBs in den aktuell verfügbaren Systemen begrenzt ist (Größenordnung zwei bis vier Gigabyte), was auch für die Dateigröße etlicher Dateisysteme gilt. Sofern keine Auflösung der inneren Struktur möglich ist, wie eben anhand der Audiotracks einer CD erläutert, kann als Abhilfe ein Binärstrom gesplittet werden, was eine Erweiterung des Datenbankschemas und einen Mechanismus zum Zusammenführen erfordert. Alternativ dazu kann innerhalb einer Relation ein Verweis auf eine externe Datei definiert werden. Das ist vor allem unter dem Aspekt der Konsistenzsicherung als negativ zu beurteilen. Um diesen Nachteil zu beseitigen, versucht man, externe Datenbestände unter die Kontrolle eines DBMS zu bringen.[20] Unter dem Gesichtspunkt der Langzeitarchivierung sind bei dieser Lösung die einheitliche Zugriffskontrolle, die Einbindung in die Backup- und Recovery-Mechanismen des DBMS und die Erhaltung der referenziellen Integrität besonders wichtig. So ist u. a. zu verhindern, dass Dateien auf Betriebssystemebene unkontrolliert umbenannt oder gelöscht werden.

*Erschließung von Inhalten, Erzeugung von Metadaten*

**Zugriff** Ein einfacher Zugriff auf Daten und Metadaten ist nicht nur für den Nutzer wünschenswert, sondern er ist Grundvoraussetzung für eine Beherrschung der Archivierungsaufgaben. Immerhin sind tausende von Dateiformaten – zumindest vor einer möglichen Reduzierung oder Normierung – zu verwalten, so wie es beispielsweise eine *Monitor-Technologie* erfordern würde. Vor allem mit Hilfe der mächtigen Sprachkonstrukte eines DBMS können digitale Objekte übergreifend und unter Ausnutzung bereits vorhandener Metadaten zur Bildung geeigneter Teilmengen identifiziert, ausgewertet und bearbeitet werden. Dies ist insbesondere dann von Vorteil, wenn Metadaten auf Grund neuer Anforderungen oder zum Zeitpunkt des Ladens fehlender Werkzeuge oder sonstiger Ressourcen nachträglich erstellt werden müssen. Eine weitere Perspektive ist die querschnittliche Ad-hoc-Erschließung von intellektuellen Inhalten, z. B. mit Werkzeugen der Mustererkennung oder Spracherkennung, wie sie jetzt schon mit einfacher Funktionalität angeboten werden. Voraussetzung ist allerdings, dass die Daten für das Datenbanksystem verfügbar sind. Dieser

[20] ISO/IEC 9075-9:2001 Information technology – Database languages – SQL – Part 9: Management of External Data (SQL/MED), 07.06.2001.

Aspekt wird unter dem Stichwort Langzeitarchivierung beleuchtet.

*Abspielumgebungen und Zusatzinformationen im DBMS*

**Nutzung** Die zeitlich unbeschränkte Nutzbarkeit (Abspielbarkeit) ist das eigentliche Ziel der Archivierung. Ist also der Zugriff auf das passende digitale Objekt mit Hilfe der Datenbank erfolgreich gewesen, so muss eine passende Abspielumgebung zur Verfügung stehen. Im Idealfall wird eine geeignete Abspielumgebung gleich mitgeliefert. Aus heutiger Sicht mag dies bei einem *PDF*-Betrachter rechtlich und technisch sehr einfach sein. Komplizierter wird es bereits, wenn ein Image mit einem nicht mehr gebräuchlichen Dateisystem aufzulösen ist und die darin verwendeten Dateien aber noch verwendet werden sollen. Was eine passende Abspielumgebung ist, sollte zumindest aus den Metadaten direkt oder indirekt erkennbar sein. Gegebenenfalls ist die Navigation durch eine komplexe Informationsstruktur notwendig. Vorteil eines Datenbanksystems ist es ferner, dass zugehörige technische Informationen einfach redundanzfrei abgelegt werden können und nicht jedem digitalen Objekt separat mitgegeben werden müssen, was auch unter dem Gesichtspunkt der langfristigen Wartbarkeit sehr nützlich ist. Prinzipiell könnten vollständige Speicherkapseln effizient verwaltet und dem Nutzer in einer geeigneten Form zur Verfügung gestellt werden. Hierbei ist zu beachten, dass das Datenbankschema selbst Teil der Metadaten ist und entsprechend erklärt sein muss.

*Nutzung von Image*

Zum Komplex Image kann aus heutiger Sicht festgestellt werden, dass für viele obsolete Hard- und Softwaresysteme auf aktuellen Plattformen brauchbare Emulationen existieren, die Images als Eingabe akzeptieren und die Inhalte auch weiterverarbeiten können. »Brauchbar« ist dabei ein sehr dehnbarer Begriff, der stark von der Anwendung abhängt und sicherlich zusätzlich subjektiver Natur ist. Für neue Technologien wie CD oder DVD stehen mit virtuellen Laufwerken und Dateisystemtreibern Lösungen zur Verfügung, die Images so behandeln, als wären ihre Datenträger und Abspielgeräte physisch vorhanden. Unter Linux funktioniert dies für eine Reihe von Image-Formaten standardmäßig mit Hilfe so genannter *loop devices*. Da die Bindung von LOBs an gängige Programmiersprachen standardisiert ist und von konkreten Produkten unterstützt wird, können Programme für die Bearbeitung von Images (z. B. zur Installation) entwickelt werden.

*Interoperabilität durch Flexibilität*

**Interoperation** Um Interoperabilität herzustellen, bieten DBMS eine Reihe von Möglichkeiten an. Relationen und dazugehörige Operationen sind auch hier die Grundlage für ein großes Maß an Flexibilität bei der Strukturierung bzw. Restrukturierung logischer Schemata. Damit verbunden sind auf Basis bestehender Relationen die einfache Definierbarkeit und Verwaltung *logischer Sichten* (views) nach Innen und Außen, die sowohl struktureller (z. B. Weglassen eines Attributs einer Relation) als auch wertbezogener Art sein können (d. h., der Wert eines Attributs bestimmt nach einer definierten Regel die ausgewählte Teilmenge). Gleichzeitig kann dabei eine Umbenennung der Schemaobjekte vorgenommen werden. So kann beispielsweise aus dem Attribut (Spaltenname) einer Relation mit dem Namen Titel ein neuer virtueller Name wie DC.Title (vgl. Dublin Core) generiert werden, wenn das eine externe Anwendung erfordert. Ebenso ist es möglich, die Werte von Attributen mittels normaler Relationen und Operationen bestimmten Anforderungen anzupassen. Ein Beispiel könnte eine Generalisierung des etwas unbekannteren *MIME*-Inhaltstyps *L24* zu dem bekannteren Typ *audio* sein (MIME = Multipurpose Internet Mail Extensions). Hiermit soll angedeutet werden, dass sich einfach strukturiertes »Wissen«, in diesem Fall ontologischer Art, direkt auswertbar in einer Standard-Datenbank halten lässt. Weitergehende Konzepte (z. B. Deduktion) sind aber in kommerziellen Systemen nicht zu finden. In Hinblick auf eine Internationalisierung ist die integrierte Unterstützung unterschiedlicher Zeichensätze und Datumsformate nicht zu unterschätzen. Logische Sichten erzeugen keine Redundanzen. Sie sind eigenständige Objekte innerhalb des Datenbanksystems. Dies hat zur Folge, dass auf ihnen auch individuelle Zugriffsrechte vergeben werden können, was die Generierung und Verwaltung von Elementen für die Kooperation erheblich erleichtert. Durch die Unterstützung von XML wird nicht nur eine internetgerechte Darstellung der Datenbankabfragen, zumindest bezüglich der Metadaten, wesentlich erleichtert, sondern auch der explizite Austausch von Daten. Vor allem steht ein datenbank-, hersteller- und betriebssystemunabhängiges, mit Semantik anreicherbares Format zur Verfügung.

*Erzeugung von Metadaten*

**Erzeugung vom Metadaten** Die Qualität der Metadaten ist, unabhängig von der konkreten Strategie für die Langzeitarchivierung, nach heutigen Rahmenbedingungen von entscheidender Bedeutung. Dies spiegelt sich auch in den vielfältigen Bemühungen wider, geeignete Metadaten zu finden und zu standardisieren (OCLC 2001, DDB 2001). Auch wenn sich der Umfang der als not-

wendig erachteten Metadaten in weiten Grenzen bewegt, ist schon der Aufwand bei der unteren Grenze nicht zu unterschätzen, insbesondere unter Berücksichtigung der im letzten Vierteljahrhundert aufgelaufenen Masse an Dokumenten, der unbefriedigenden Aussonderungskriterien und der bisher nur spärlich vorhandenen technischen Metadaten.

*Quellen für Metadaten*

Ein Ansatz ist die Erweiterung der bei der Datenträgermigration notwendigen Schritte. So können bei der Erzeugung der Images Protokolle erzeugt werden. Da diese nicht unmittelbar in einer geeigneten (hinreichend formalisierten) oder einer einheitlichen Form (weil häufig verschiedene Werkzeuge erforderlich sind) vorliegen werden, ist eine Nachbearbeitung erforderlich. Das Image kann aber auch nach dem Loslösungsprozess analysiert werden: entweder mit Werkzeugen, die für das ursprüngliche Medium konzipiert sind (z. B. CD-Katalogisierer), durch Fähigkeiten des Zielsystems (z. B. Anzeige in einem Dateisystembetrachter), die auch emuliert sein können, oder durch spezielle Software, die Images direkt auswerten kann, z. B. durch Analyse des PVD einer CD-ROM. Im Allgemeinen werden auch hier die Analyseergebnisse in einer nacharbeitungsbedürftigen Darstellung vorliegen. Die entsprechenden Routinen könnten vom Datenbanksystem verwaltet werden. Ist die Aufbereitung der Protokolle und Analyseergebnisse nicht möglich, können sie in ihrer ursprünglichen Form im Datenbanksystem i. d. R. als CLOB gespeichert werden; es gibt aber bereits Werkzeuge, die für bestimmte neuere Formate XML-Dokumente generieren.

*nichtdigitale Metadaten*

Oft liegen Metadaten nicht in digitaler Form vor. Hier ist zu überlegen, ob eine Überführung in die Datenbank automatisierbar ist, was die Digitalisierung und Interpretation der Inhalte voraussetzt.

*technisches Hintergrundwissen für die Erzeugung von Metadaten*

Insgesamt ist sehr viel technisches Hintergrundwissen erforderlich, um Metadaten explizit zu machen. Bei historischem Material ist es aufwändig, eine korrekte und vollständige Dokumentation zu beschaffen[21] und bei neuerem Material ist eine steigende Komplexität festzustellen, u. a. durch eine zunehmende Multimedialität mit immer anspruchsvolleren Abspielumgebungen.

[21] Die Beschaffung verlässlicher Information zu Dateiformaten wurde im Projekt Risk Management of Digital Information (s. Abschnitt 9.3) als schwierigster Teil gesehen. Dort wurden aber im Wesentlichen nur zwei Formate untersucht.

*Sicherung der Persistenz*

**Langzeitarchivierung** Die Sicherung der Persistenz der digitalen Objekte ist nun Aufgabe des DBMS. Dabei ist das DBMS selbst auch nur ein digitales Objekt und unterliegt sogar noch strengeren Gesetzen als die Archivierungskandidaten. Es muss mit all seinen anvertrauten Daten auf alle Fälle am Leben erhalten werden, d. h., es muss regelmäßig auf neue Systemumgebungen portiert und an neue Standards und Anforderungen angepasst werden. Viele Aufgaben werden durch den Datenbankhersteller erledigt, doch anwendungsbezogene Aspekte oder die Bewältigung konzeptioneller Mängel der Datenbanktechnologie und Defizite einzelner Produkte bleiben in der Verantwortung des Nutzers. Unter dem Gesichtspunkt der Langzeitarchivierung sind Schemaänderungen und die physische Speicherung zu diskutieren.

*logisches Datenbankschema*

Das logische Datenbankschema sollte ein genaues und aktuelles Abbild eines konzeptionellen Modells sein, das notwendige Daten, Funktionen und Regeln für eine bestimmte Aufgabe genau beschreibt. Konzeptionelle Modelle unterliegen Änderungen, die durch die reale Welt bestimmt werden und technischer, organisatorischer oder kultureller Art sein können. So entstehen z. B. nicht nur ständig neue Begriffe, sondern es findet darüber hinaus ein mehr oder weniger schneller Wandel der Inhalte statt, z. B. bei Erweiterungen von Dateinamen. Außerdem werden die Beschreibungsmöglichkeiten für konzeptionelle Modelle und logische Datenbankschemata weiterentwickelt. Dies erfordert eine laufende Anpassung der Datenbankschemata, um eine schleichende Degeneration des DBMS, wie sie in der Praxis häufig zu beobachten ist, zu verhindern. Daher ist bereits beim Entwurf auf ein hohes Maß an Flexibilität zu achten, wofür es Regeln und Heuristiken gibt. Flexibilität kann u. a. dadurch erreicht werden, dass Redundanzen vermieden oder explizit kontrolliert werden, was jedoch das Erkennen funktionaler Abhängigkeiten voraussetzt. Die Forschungsarbeiten zu dem unter Langzeitgesichtspunkten wichtigen Thema der Schemaänderungen sind noch nicht abgeschlossen.

*physische Speicherung*

Die physische Speicherung der digitalen Objekte stellt unter Langzeitgesichtspunkten auch für die Datenbanktechnologie eine große Herausforderung dar. Dies liegt hauptsächlich an der Begrenztheit der Speicherkapazität online-fähiger Speichermedien oder der Speicher- und Verarbeitungsfähigkeit des DBMS bezüglich sehr großer Datenbestände, was zur Folge hat, dass bestimmte Teile der Datenbank, i. d. R. auf *Tertiärspeicher*, z. B. Magnetbänder, ausgelagert werden müssen. Nach dem heutigen Stand der Technik geht durch mangelnde Abstraktion von physischen Speicherungsbelangen ein großer Vorteil eines DBMS, nämlich

die Transparenz, verloren: Man hat auf der logischen Ebene keinen einheitlichen Zugriff mehr, da die ausgelagerten Teile in einer deutlich gröberen Struktur vorliegen. Dies hat insbesondere Konsequenzen für die Sicherung der globalen Konsistenz. Zwei wichtige Fälle hierbei sind die Änderung der internen Speicherstrukturen durch den Datenbankhersteller und die Änderung des logischen Datenbankschemas, die auch eine direkte Änderung der Daten erforderlich machen kann. Aber auch die Wartung und Nutzung der Daten werden durch den erschwerten Zugriff erheblich behindert, besonders wenn digitale Objekte querschnittlich analysiert werden müssen und vorab keine geeigneten Teilmengen gebildet werden können.

*Datenbankarchivierung in anderen Disziplinen*

Diese Problematik ist auch für andere Anwendungsgebiete relevant. Viele Konstruktions- und Produktdaten sind durch *CAD*-Systeme (Computer Aided Design) erzeugt worden und liegen somit in digitaler Form vor. Der Wunsch nach Wiederverwendung von Komponenten oder langfristige Wartungsverpflichtungen, z. B. bei atomtechnischen Anlagen, machen auch hier Lösungen erforderlich. Herbst (1997) beschreibt, welche Dienste ein DBMS für ein so genanntes *anwendungsorientiertes DB-Archivieren* bieten müsste. Unter anderem wird ein *Archivzugriff* gefordert, d. h., archivierte Daten müssen langfristig über ein DBMS mit dessen Operationen zugänglich sein. Schaarschmidt (2001) diskutiert die Archivierung als Datenbankdienst, wobei das Archiv vom DBMS kontrolliert wird und die Archivierungsfunktionalität im DBMS integriert ist. Solange kommerzielle Standard-Datenbanksysteme jedoch keine vollständigen Lösungen für dieses Problem anbieten, muss eine Übergangsstrategie entwickelt werden. Hierzu wurden im Rahmen dieses Projekts einige Lösungen skizziert.

*Änderung interner Speicherstrukturen*

Um der Problematik der Änderung der internen Speicherstrukturen durch den Datenbankhersteller zu entgehen, könnte eine Speicherung der LOBs in externen, aber durch die Datenbank kontrollierten Dateien eine Alternative sein. Dabei ist u. a. zu bedenken, dass auch die konkreten Dateisysteme einer Weiterentwicklung, z. B. zur Steigerung der Zugriffsgeschwindigkeit oder Erhöhung des Zugriffsschutzes, unterliegen und dass Änderungen der internen DBMS-Speicherstrukturen nicht so häufig sind wie die üblichen Versionswechsel.

*Multimediadaten im DBMS*

Da in DBMS immer höherwertigere Datentypen z. B. für Multimedia angeboten werden, kann bei einer Migration mit Unterstützung durch den DBMS-Hersteller gerechnet werden, sofern

ein kommerzielles Interesse auf Grund genügender Nachfrage besteht.

### 9.4.5 Detailuntersuchungen und praktische Versuche

*Voruntersuchungen der Bayerischen Staatsbibliothek*

Die in den Arbeitsberichten der Bayerischen Staatsbibliothek (Dörr 1999a, 1999b) angestellten Analysen geben konkrete Hinweise auf Probleme, die im Umgang und in der Bereitstellung elektronischer Medien in einer Universalbibliothek auftreten. Hervorzuheben sind:

- die beschränkte Einflussnahme auf Erzeuger elektronischer Publikationen
- ein hohes Maß an hard- und softwaretechnischer Heterogenität bei den Formaten für Dateien und Dateisystemen
- die zunehmende Komplexität der Systemvoraussetzungen durch Multimedialität
- die mangelhafte Dokumentation des Bestands und der Nutzbarkeit digitaler Medien

*Schwerpunkte der Untersuchungen*

Um die Umsetzbarkeit und den Nutzen des Konzepts der Datenbankmigration zu untersuchen, wurden am Institut für Softwaretechnologie der Universität der Bundeswehr München folgenden zwei Komponenten näher erforscht:

- die Loslösung der digitalen Objekte von ihren ursprünglichen Medien (Datenträgern)
- die Überführung der losgelösten digitalen Objekte in ein kommerzielles DBMS zur integrierten Bereitstellung, Verwaltung und Archivierung

Neben der Auswertung von Studien und technischen Unterlagen fanden auch praktische Versuche statt. Diese sollten nicht nur die technische Machbarkeit belegen, sondern auch Hinweise zum erforderlichen Aufwand und weiteren Untersuchungsbedarf geben (Rödig *et al.* 2002). Schwerpunkte waren:

- der Prozess der Loslösung digitaler Objekte
- die Gewinnung technischer Metadaten
- der Entwurf eines formalisierten konzeptionellen Modells als Grundlage für einen Datenbankentwurf
- Teile eines Datenbankentwurfs einschließlich Implementierung

*Werkzeuge für den Versuch*

Dazu wurde untersucht, welche verfügbaren kommerziellen und nichtkommerziellen Software- und Hardwareprodukte die Prozesse der Loslösung und der Erzeugung technischer Metadaten unterstützen können.

*»Der Teufel steckt im Detail«*

Die folgende Schilderung der Details belegt, mit welch vielfältigen Problemen man bei der Archivierung digitaler Medien rechnen muss. Ob diese Probleme nur für den betrachteten Entwicklungszeitraum charakteristisch sind, ist derzeit noch nicht abzusehen.

### Loslösung

*Wissen und Geräte für Disketten schwinden*

Durch die Loslösung wird eine technische und zeitliche Normierung der Medien erreicht. Am problematischsten ist die Situation bei Disketten, weshalb dieser Komplex auch etwas tiefer analysiert wurde. Es schwindet nicht nur das Wissen über diese Technologie, sondern allmählich auch die nötige Ausstattung. An der Bayerischen Staatsbibliothek war kein System mit 5,25-Zoll-Laufwerk mehr verfügbar. Die Hersteller von 8- und 5,25-Zoll-Laufwerken nennen zudem eine Lebensdauer der Komponenten (component life) von nur fünf Jahren.[22] Insbesondere in der Anfangsphase der Diskettennutzung waren auf der physischen und logischen Ebene viele unterschiedliche Formate in Gebrauch. Außerdem sind 8-Zoll- und 5,25-Zoll-Disketten wegen der nicht verschließbaren Leseöffnung und der flexiblen Hülle leicht Beschädigungen ausgesetzt. Bei einer Sichtung der Datenträger an der Bibliothek der UniBwM wurde festgestellt, dass ein großer Teil der 5,25-Zoll-Disketten nicht mit einer Schreibschutzmarke zum Verschließen der Schreibschutzkerbe versehen war. Es gibt zwar Disketten mit Hüllen, die von Haus aus keine Schreibschutzkerbe besitzen, was aber von den Publizierern nur selten genutzt wurde. Eine 8-Zoll-Diskette hingegen ist schreibgeschützt, wenn die Schreibschutzkerbe, die nicht immer an der gleichen Stelle sitzt, nicht geschlossen ist; aber nicht alle Laufwerke haben einen entsprechenden Sensor. Da die Datenträger anfangs relativ teuer waren, wurden teilweise sehr unkonventionelle Methoden verwendet, um den Speicherplatz voll auszunutzen. So wurden beispielsweise einseitige 5,25-Zoll-Disketten in Laufwerken mit einem Le-

[22] NEC Nippon Electric Co. Ltd, FD1165-F Floppy Disk Drive, Performance Specifications, Kopie ohne Datum und Teac, TEAC FD-55GFR, 5,25-high/normal densities, double sided mini flexible disk drive, specification, Kopie ohne Datum.

sekopf gedreht, um auch die aus produktionstechnischen Gründen beschichtete Rückseite zu beschreiben.

*Klassifizierung von Disketten – Formfaktoren*

Im Folgenden werden die Disketten nach Eigenschaften klassifiziert, die für die Loslösung wichtig sind. Das augenscheinlichste Merkmal bei Disketten ist deren Formfaktor mit Maßen, die gewöhnlich in Zoll angegeben sind. In den ECMA-Standards (European Computer Manufacturers Association)[23] hingegen und auch in Bibliothekskatalogen werden Millimeter bzw. Zentimeter verwendet. Die wichtigsten Formfaktoren sind 8, 5,25 und 3,5 Zoll bzw. 200, 120 und 90 mm. Nach den Zoll-Maßen sind auch die Laufwerke eingeteilt. Neben dem Begriff Formfaktor ist ebenso Format üblich.

*Aufzeichnungsverfahren*

Zur weiteren Klassifizierung der Disketten bietet sich die physische Speicherebene (Aufzeichnungsebene) an. Bei Disketten dominieren drei Aufzeichnungsverfahren:

- Frequenzmodulation (FM)
- Modifizierte Frequenzmodulation (MFM)
- Group Code Recording (GCR).

Diese Verfahren bestimmen, wie der sequenzielle Datenfluss synchronisiert wird. Dies ist u. a. notwendig, weil auf Grund von Schwankungen der Umdrehungsgeschwindigkeit des Laufwerks dessen einzelne Signale nicht einem beliebig festen Zeitfenster zugeordnet werden können. FM wird auch als *IBM 3740 single density format* und MFM als *IBM 34 double density format* bezeichnet. Alle Aufzeichnungsformate sind bei allen Formfaktoren zu finden, wobei FM praktisch aber nur beim 8-Zoll-Format eine Rolle spielt. Durch die Dominanz der IBM-PCs und deren Clones wurde das GCR-Verfahren verdrängt, das bei den frühen Apple- und Commodore-Rechnern zu finden war; diese wurden in der Folge mit hybriden *FDCs* (Floppy Disk Controller) ausgestattet.

*single sided und double sided*

Eine weitere Einteilung für 8-Zoll und 5,25-Zoll-Formate ist die einseitige (SD single sided) oder die beidseitige (DS double sided) Beschreibbarkeit. Sehr selten sind einseitige 3,25-Zoll-Disketten. Bei einigen 8-Zoll-Formaten kann aus der Lage des Indexloches auf ein- oder beidseitige Beschreibbarkeit geschlossen werden. Teilweise sind bei den größeren Formfaktoren auch Aufkleber (Labels) vorhanden.

[23] seit 1994: ECMA International - European association for standardizing information and communication systems.

Eine weitere Unterteilung ist die Schreibdichte der Medien:

*Schreibdichte*

- single density (SD) bei: 3,5, 5,25, 8 Zoll
- double density (DD) bei: 3,5, 5,25, 8 Zoll
- high density (HD) bei 3,5, 5,25 Zoll
- extra high density (ED) bei 3,5 Zoll.

Sehr selten sind 3,5-Zoll-Disketten mit SD. Disketten mit ED (4 MByte unformatiert) haben sich nicht durchgesetzt. Die 3,5-Zoll-HD[24] hat eine Identifikationsöffnung, ebenso die 3,5-Zoll-ED,[25] jedoch an anderer Stelle. Die 3,5-Zoll-DD[26] ist hingegen geschlossen. Für eine Kapazitätsplanung kann davon ausgegangen werden, dass 1,44 MByte (HD 3,5 Zoll IBM-formatiert) i. d. R. das Maximum darstellen.

*Spuren, Sektoren und Spiralen – Low-Level-Formatierung*

Strukturell unabhängig von den bisher genannten Eigenschaften ist die Einteilung der Disketten in Spuren (tracks) und Sektoren (sectors), wobei die Sektoren die kleinste adressierbare Einheit darstellen. Nimmt man die Position des Lese- und Schreibkopfes (engl. *head*) bei zweiseitig genutzten Medien noch hinzu, hat man eine eindeutige Adressierungsmöglichkeit. Die Anzahl der Spuren und die Speicherkapazität einer Spur bzw. der Sektoren hängen vom Aufzeichnungsverfahren (z. B. Anzahl der notwendigen Synchronisationsbits oder vergrößernde Umcodierungen der Nutzerdaten), vom Medium (z. B. zulässige Anzahl von Spuren pro Zoll bzw. *tpi* – tracks per inch) und von der Hardware (z. B. Fähigkeit des Laufwerks) ab. Für die Verwaltung und Synchronisation der Sektoren sind zusätzlich Informationen notwendig. Das mögliche Layout wird eigentlich nur durch die physischen Parameter begrenzt. So kann die Anzahl der Sektoren oder deren Größe – auch innerhalb einer Spur – unterschiedlich sein. Beim Commodore Standardformat (GCR 170 KByte formatiert) befinden sich z. B. in den äußersten Spuren 21 Sektoren und in den innersten Spuren 17 Sektoren mit konstant 256 Bytes pro Sektor.[27]

[24] ECMA, Data Interchange on 90 mm Flexible Disk Cartridges Using MFM Recording at 7 958 ftprad on 80 Tracks on Each Side - ISO Type 301, 2nd edition, Genf, 12.1988.

[25] ECMA, Data Interchange on 90 mm Flexible Disk Cartridges Using MFM Recording at 15 916 ftprad on 80 Tracks on Each Side - ISO Type 302, Genf, 12.1987.

[26] ECMA, Data Interchange on 90 mm Flexible Disk Cartridges using MFM Recording at 31 831 ftprad on 80 Tracks on Each Side - ISO Type 303, Genf, 12.1990.

[27] Commodore Büromaschinen GmbH, DISK DRIVE 1570/71, Bedienungshandbuch, Frankfurt, 1985; siehe auch Abb. 9.7.

Der Prozess des Anlegens dieser Struktur wird als Formatieren bezeichnet (z. B. mit dem Befehl *fdformat* bei Linux). Da hiermit in der Praxis oft auch das (gleichzeitige) Anlegen eines Dateisystems verstanden wird, kann der Begriff *Low-Level-Formatierung* zur Verdeutlichung verwendet werden. *Hardsektorierte* Disketten (für jeden Sektor eine feste Markierung) oder Disketten mit spiralförmiger Aufzeichnung werden in einer Universalbibliothek als eher unwahrscheinlich eingeschätzt.



**Abbildung 9.7**
*Commodore 128 mit 1570 Disk Drive*

*Dateisysteme – High-Level-Formatierung*

Mit dieser blockorientierten Struktur kann nun ein Dateisystem, das auch schreibend wahlfreien Zugriff erlaubt, angelegt werden. Dieser Prozess wäre eigentlich als *High-Level-Formatierung* zu bezeichnen (z. B. mit dem Befehl *mkfs* – make file system – bei Linux). Es können auch ohne Dateisystem Daten auf die Diskette geschrieben werden; Adressierungseinheit sind dabei die Sektoren. Bootdisketten sind in diesem Kontext die wichtigste Ausprägung.

### Praktische Durchführung

*Sichtung der Medien im Bibliotheksbestand*

Insgesamt gibt es also eine beträchtliche Vielfalt, insbesondere auf der Ebene der Formatierung. Wie geht man nun praktisch vor? Ein erstes Kriterium für die Beschaffung von notwendiger Apparatur ist die Verbreitung von Hardware beim Publikum. Es kann zwar heute von einer Dominanz der IBM-kompatiblen PCs ausgegangen werden, aber auch Heimcomputer (die dann ebenfalls als PCs bezeichnet wurden) wie die Commodore-Rechner hatten eine sehr

große Verbreitung. Und Apple ist heute noch am Markt präsent. Wenig hilfreich sind die Kataloge der beiden Bibliotheken BSB und UniBwM, wenn es um eine technische Klassifizierung geht. Aber indirekt, z. B. durch Zeitraum und Thematik, kann auf das Vorhandensein bestimmter Formate geschlossen werden. Als besonders nützlich erwiesen sich Gespräche mit Mitarbeitern, die seit langem in der Katalogisierung tätig sind. Einen großen Vorteil bietet die separate Aufbewahrung der digitalen Datenträger; ein Augenschein liefert einen ersten guten Überblick. Dieser Vorteil wurde an der Bibliothek der UniBwM genutzt, um gezielt eine Hard- und Softwareumgebung aufzubauen. Zu erkennen war in Hinblick auf Disketten:

- hoher Anteil an 5,25-Zoll-Disketten, trotz vorhergehender Aussonderungen
- sehr kleiner Anteil an 8-Zoll-Disketten
- häufig fehlende Hinweise auf der Hülle zu High- oder Low-Level-Formaten
- Abwesenheit obsoleter Formfaktoren wie 3 Zoll aus dem Bereich Hobby-Computer
- Vorhandensein älterer bzw. nicht PC-spezifischer Formate – u. a. einige exotische Disketten (s. auch Abb. 9.8).



**Abbildung 9.8**
*Exotische Disketten*

*Hard- und Softwareumgebung*

Auf dieser Basis wurde versucht, im Rahmen der vorhandenen Ressourcen eine praktikable Umgebung zur Loslösung zu installieren. Grundsätzliche Alternativen waren die Verwendung der originalen Hard- und Software mit einer Verbindung zu einem

aktuellen System oder eine Erweiterung aktueller Systeme, die möglichst viele Fälle abdeckt. Es wurde der zweite Weg realisiert, wobei die originale Umgebung – am besten mit Dokumentation – eine nützliche Ergänzung ist. Somit kann die Zahl der notwendigen Hard- und Softwarekomponenten reduziert, eine aufwändige Vernetzung vermieden und der Prozess insgesamt beschleunigt werden. Vorhandene Originalausstattung wird geschont; sie dient der Überprüfung oder der Behandlung von Spezialfällen.

**Abbildung 9.9**
*Catweasel-Controller*



*Hard- und Software zum Lesen von Disketten*

Für das Lesen von Disketten wurde der *Catweasel-Controller* gewählt (s. auch Abb. 9.9), der von der Firma Individual Computers vertrieben wird. Er kann mit Hilfe der mitgelieferten Software aus einer Reihe von Formaten mit den Aufzeichnungsverfahren GCR und MFM Images erzeugen (z. B. CBM 1541 5,25 Zoll single-sided 170 KB, Amiga 3,5 Zoll double-sided 880 KB, Apple Macintosh 3,5 Zoll single-sided 400 KB) und für eine Teilmenge Dateien auflisten (z. B. CBM 1571 DOS 2A SS+DS, Amiga OFS Old File System). Er ist in einem Standard-PC mit *ISA-Bus* (Industry Standard Architecture) installierbar (maximal vier Stück) und kann zwei Laufwerke bedienen. Angeschlossen wurden ein übliches 3,5-Zoll- und ein 5,25-Zoll-PC-Laufwerk (s. Abb. 9.10). Auf die Installation eines 8-Zoll-Laufwerks wurde vorerst verzichtet. Es sollte, sofern keine GCR-Aufzeichnung zu interpretieren ist, prinzipiell an einen FDC eines Standard-PCs angeschlossen

werden können, der das FM-Aufzeichnungsverfahren (noch) unterstützt. Dies gilt für MicroPD765-kompatible Controller und auf jeden Fall für den 82072[28] und 82077AA[29] von Intel. Dabei muss der 50-polige Bus der 8-Zoll-Laufwerke auf den 32-poligen Bus der kleineren Laufwerke angepasst werden (Wagemuth 1987). Die aktuelle Bedeutung des 8-Zoll-Formats ist schwer abzuschätzen. Einerseits waren die Laufwerke für den Massengebrauch zu teuer, andererseits spielte es in bestimmten Anwendungen wie der medizinischen Bildverarbeitung eine wichtige Rolle.



**Abbildung 9.10**
*Standard-PC mit unterschiedlichen Floppy-Disk-Controllern*

[28]Sridhar Begur, Intel Corporation, Designing with the 82072 CHMOS Single-Chip Floppy Disk Controller, Santa Clara, 09.1987.

[29]Intel Corporation, CHMOS Single-Chip Floppy Disk Controller 82077A, Santa Clara, 10.1989.

*fehlerhafte Medien*

Weitere Fragen werfen fehlerhafte Medien auf: Wieviel Aufwand soll überhaupt investiert werden und wie behandelt man partiell lesbare Medien? Für das Lesen größerer Mengen wurde versucht, ein akzeptables Maß zu finden, jedoch ohne Bewertung des intellektuellen Inhalts. So wurde beispielsweise der Parameter, der die Anzahl der Leseversuche bestimmt und standardmäßig auf 10 gesetzt ist, auf maximal 100 bei 5,25-Zoll-Disketten und auf 50 bei 3,5-Zoll-Disketten begrenzt aus Gründen der Zeitersparnis (nach 8 Versuchen wird eine zeitaufwändige Rekalibrierung des Lesekopfes durchgeführt) und der Schonung der in begrenzter Stückzahl vorhandenen Laufwerke. Außerdem ließen die Vorversuche bei einem höheren Wert keine wesentlichen Verbesserungen mehr erwarten. Die Images partiell lesbarer Medien wurden grundsätzlich übernommen. Einzelne Fehler bedeuten nicht gleich Unbrauchbarkeit, da Informationen teilweise technisch oder semantisch redundant sind bzw. die Fehler mehr oder weniger einfach geheilt werden können oder nur abgegrenzte Bereiche betreffen. Den Grad der jeweiligen Einschränkung für eine künftige Nutzung im Einzelfall zu beurteilen, ist sehr aufwändig. Die Software von Catweasel protokolliert Fehler auf Ebene der Sektoren. Dieses Protokoll soll als Teil der Historie mit dem digitalen Objekt gespeichert werden und steht somit einer späteren, möglicherweise teilweise automatisierbaren Auswertung zur Verfügung.

*weitere Werkzeuge*

Ergänzend wurde ein weiterer Standard-PC, der sowohl mit Linux als auch mit Windows NT 4.0 gebootet werden kann, mit einem 5,25-Zoll-Laufwerk ausgestattet. Unter Linux steht eine Reihe von Emulationen zur Verfügung. Damit ist u. a. eine Prüfung der von Catweasel erzeugten Images möglich. Unter Windows NT wurde *WinImage* als weiteres Programm zur Erzeugung von Images, das u. a. gezielte Lesewiederholungen zulässt, installiert. Es unterstützt zwar nicht die gleichen Formate wie Catweasel (prinzipiell keine GCR-Formate), hilft aber bei den gemeinsamen Formaten, Lesefehler besser zu beurteilen. Dies ist zweckmäßig, da die Lesbarkeit von Disketten, wie aus der früheren Praxis bekannt, sehr stark vom konkreten Laufwerk abhängen kann und die Lesemechanismen, z. B. die Ansteuerung des Laufwerkscontrollers, nicht offen dokumentiert sind.

*Protokollierung des Loslösungsprozesses*

Für den Erfassungsvorgang wurde ein Protokoll entworfen, das in Teilen Grundlage für das Laden der Datenbank ist. Neben der unabdingbaren Identifikation und der strukturellen Zusammengehörigkeit der digitalen Objekte, sollten auch die auf den Medien vorhandenen Hinweise erfasst werden und zwar in drei Ebenen: die Low-Level-Formatierung, die High-Level-Formatierung

und die Systemvoraussetzungen. Die getrennte Aufbewahrung der digitalen und der zugehörigen nichtdigitalen Medien ist hier ein Nachteil. Auf eine Auswertung der nichtdigitalen Medien wurde aus Kapazitätsgründen verzichtet. Angestrebt wird, die ersten zwei Ebenen automatisch während des Loslösungsprozesses oder nachträglich z. B. über Datenbankprozeduren zu gewinnen. Zusätzlich sollten Fehler und Besonderheiten aufgezeichnet werden können. Die Erfassung deskriptiver Metadaten, also der Beschreibung des intellektuellen Inhalts und der üblichen Metadaten, war nicht Teil des Versuchs.

*die Praxis*

Das ursprüngliche Laufwerk wurde nach einer Häufung von Lesefehlern, auch bei neueren 5,25-Zoll-Disketten, ausgetauscht. Dieses Laufwerk war nach Vorversuchen ausgewählt worden, weil es auch zu groß geratene Disketten – so etwas gibt es – aufnehmen konnte. Ein Knick zwischen Entlastungskerbe und Leseöffnung kann ein Hinweis darauf sein.

*Überprüfung der Ergebnisse*

Zur Überprüfung der mit Catweasel erzeugten Images wurde u. a. das Werkzeug *disk image maintenance utility c1541* des *Versatile Commodore Emulator* (VICE) unter Linux genutzt. Das rechtlich geschützte *Kernel-ROM* ist hierzu nicht nötig (vgl. Diskussion zu den rechtlichen Aspekten des Emulationsansatzes auf S. 82).

### Bewertung

*Bewertung der Versuchsausstattung*

Insgesamt kann man mit dieser Versuchsausstattung einen großen Teil der Formate der Testreihe abdecken. Es hat sich wieder gezeigt, wie wichtig bei 5,25-Zoll-Disketten ein passendes, gut justiertes Laufwerk ist. Bei einem schlechten Laufwerk sind Lesezeiten von über einer Stunde (sic!) keine Seltenheit, wobei ein Erfolg nicht sichergestellt ist. In Hinblick auf eine automatische Gewinnung von Metadaten wäre eine homogenere Darstellung der Dateisysteme und der Dateien, am besten mit einem einzigen Werkzeug, wünschenswert. Hier besteht mit Sicherheit noch Entwicklungsbedarf, insbesondere auch, was eine zuverlässige technische Klassifizierung anbelangt. Der benötigte Zeitaufwand hängt stark von der Qualität der Disketten und der Laufwerke ab. Hier einige Anhaltswerte:

- 5,25 Zoll 1,2 MByte FAT12 ca. 15 Minuten
- 3,25 Zoll 1,44 MByte FAT12 ca. 4 Minuten

Deutlich schneller geht es mit WinImage (sofern die Formate überhaupt unterstützt werden), wobei die Vergleichbarkeit durch die

Hardware der Versuchsumgebung eingeschränkt ist. Durch zusätzliche Hardware kann der Ablösevorgang parallelisiert und somit beschleunigt werden. Der Aufwand für die manuelle Erfassung technischer Angaben, die sich auf den Medien oder Verpackungen befinden, fällt bei Disketten kaum ins Gewicht.

*CDs im Versuch*

Nun zum nächsten wichtigen Medium, der Compact Disc (CD). Auch hier wurde wieder eine Sichtung eines großen Teils der separat aufbewahrten Medien vorgenommen und u. a. festgestellt, dass der Umfang der Hinweise zur Formatierung und zu technischen Voraussetzungen sehr unterschiedlich ist. Analog zu den Disketten wurde eine Reihe von Detailuntersuchen durchgeführt, was hier nicht weiter beschrieben werden soll. Nur soviel: Die Fülle der Materialien auf einem Medium und deren Heterogenität ist beeindruckend. An eine manuelle Erstellung von Metadaten ist hier nicht mehr zu denken, weswegen dem automatisierten Aufschluss von Inhalten besondere Bedeutung zukommt. Diese Aussage kann wohl auch auf DVDs übertragen werden, obwohl nur eine kleine Testserie zur Verfügung stand.

### Formales konzeptionelles Modell und Datenbankschema

*Anforderungen an ein formales konzeptionelles Modell*

Das konzeptionelle Modell muss einerseits die gegebenen physischen und logischen Belange existierender Publikationsformen berücksichtigen (Ausgangssituation) und andererseits nach der Loslösung der digitalen Objekte eine Sicht definieren, die einen semantikerhaltenden Zugriff erlaubt und genügend Möglichkeiten für eine Verwaltung auch technischer Art bietet.

*Bestimmung der Granularität digitaler Objekte*

Eine Schwierigkeit ist hierbei, dass vorab aus wirtschaftlichen und technischen Gründen nicht allgemein bestimmt werden kann, wie fein gegebene, digitale Informationen aufgelöst und zu einer digitalen *Entität* erhoben bzw. dieser zugeordnet werden können. Unter Entität wird ein Objekt verstanden, das eigenständig genutzt und mit intellektuellen Attributen versehen werden kann, z. B. ein Audiotrack einer CD-ROM mit Interpret und Codierungsverfahren oder der Bootsektor einer Betriebssystem-Diskette mit Betriebssystem und Ersteller. Diese Informationen werden in vielen Fällen nicht vorab explizit gemacht werden können. Sie bleiben somit Teil einer allgemeineren Entität. Beispiele wären etwa ein Sprachkurs mit Audio-Kommentaren und Bildern oder einführende Beispielprogramme zur Numerik verschiedener Betriebssysteme. Für diesen Fall müssen die richtigen Attribute vorgesehen werden. Beim Image einer multilingualen DVD sollten alle Sprachen explizit notiert werden können, beim Image einer

CD alle unterstützten Betriebssysteme. Für eine Konvertierung oder Emulation müssen die darunter liegenden Dateien individuell angesprochen werden können. Die allgemeineren Entitäten bilden hierbei den Einstiegspunkt. In welcher Form Datenträger Informationsentitäten repräsentieren können, wurde eingangs beschrieben.

*Übernahme bestehender bibliothekarischer Ordnungsstrukturen*

Für die Entwicklung eines konzeptionellen Modells wird man wahrscheinlich die grundlegenden Ordnungsstrukturen bestehender Bibliothekssysteme nur bedingt übernehmen können. So werden z. B. an der Bibliothek der UniBwM technisch unterschiedliche Datenträger mit identischem Inhalt einfach durchnummeriert.

*konzeptionelles Modell*

Nun soll ein konzeptionelles Modell mit dem Schwerpunkt Loslösung zur Diskussion gestellt werden. Dazu werden die wichtigsten Elemente informell beschrieben. Eine Darstellung als detailliertes Klassendiagramm in *UML*-Notation (Unified Modeling Language) findet sich in Abb. 9.11.

*Beschreibung der Klassen*

Ausgangselement ist die Informationsentität (Information Entity). Diese repräsentiert die logische Sicht des Publizierers oder einer Bibliothek. Auf dieser Ebene werden auch die üblichen bibliographischen Angaben (Explicit Meta) angesiedelt sein. Hier sind ggf. weitere Ordnungsstrukturen einzufügen. Das Element Informationsentität[30] kann aus einer oder mehreren digitalen (Digital Entity) bzw. nichtdigitalen Entitäten (Nondigital Entity), hauptsächlich Bücher, bestehen. Als nächste Stufe folgt das Element Volume (Volume), das einen konkreten Datenträger repräsentiert und nochmals in beliebig viele Dimensionen (Dimension), z. B. Seiten einer Diskette, unterteilt wird. Damit wird u. a. berücksichtigt, dass ein Volume mehrere unabhängige Ordnungssysteme oder aber nur ein in sich abgeschlossenes enthalten kann. Diese hierarchische Grundstruktur erlaubt die aus pragmatischen Gründen erforderliche Wahl einer geeigneten Aggregationsstufe insbesondere zur Beschreibung technischer Aspekte. Das Element Dimension korrespondiert einerseits mit einem oder mehreren losgelösten digitalen Objekten (Detached Digital Object) und andererseits mit einem abstrakten Ordnungssystem (Ordersystem). Dies ist im konkreten Fall ein Dateisystem (File-System) oder ein Inhaltsverzeichnis (TOC); aber auch eine absolute Adressierung ist denkbar. Das Ordnungssystem, das letztlich alle intellektuellen Inhalte umfasst, wird auf einen logischen Bitstrom (Logical Bit-

[30] In der Begrifflichkeit der OO-Modellierung präziser als eine Instanz der Klasse Informationsentität zu bezeichnen.

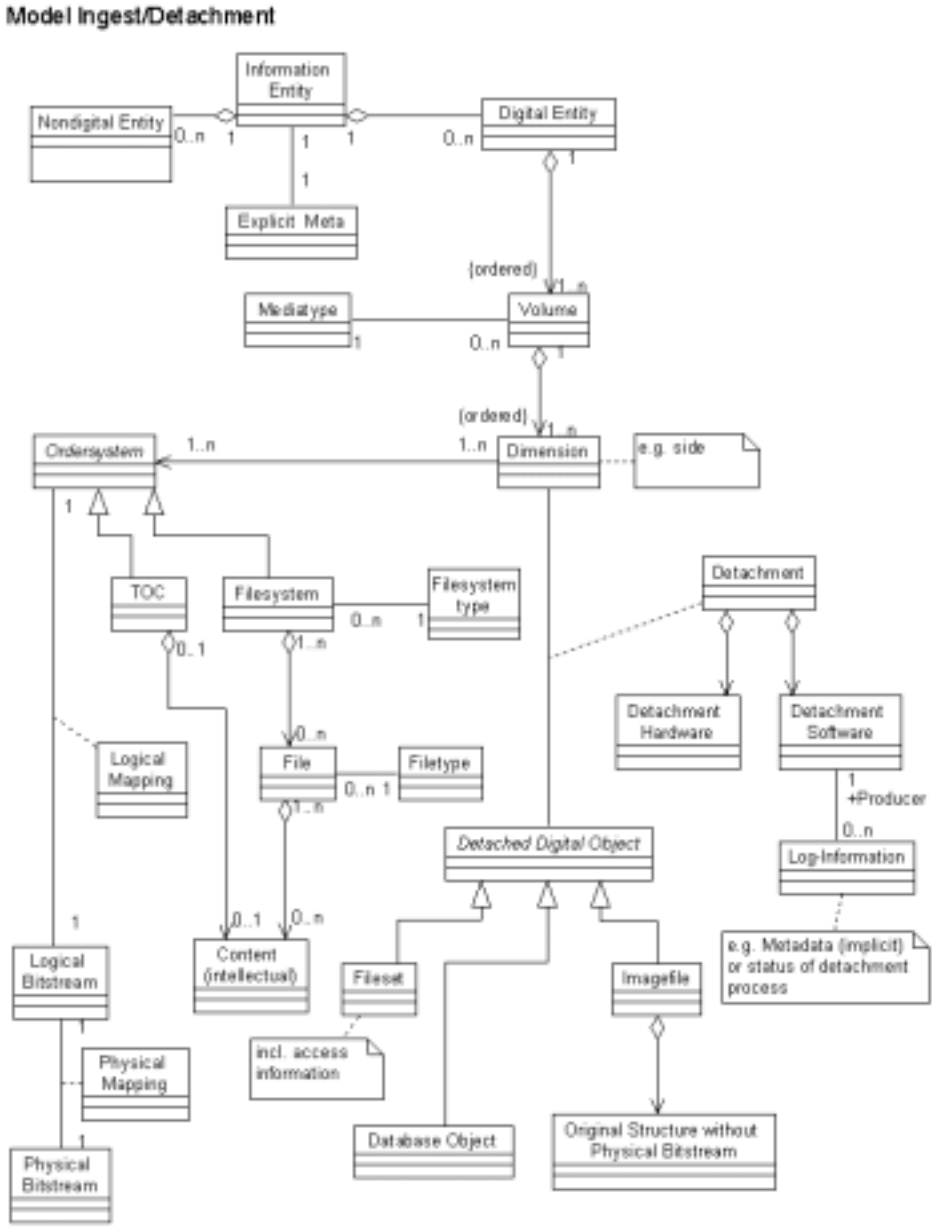



**Abbildung 9.11**
*UML-Klassendiagramm des konzeptionellen Modells für den Loslösungsprozess*

stream) und dieser dann auf einen physischen Bitstrom (Physical Bitstream) entsprechend dem vorgestellten Schichtenmodell abgebildet. Berücksichtigt wird der Sonderfall, dass das Ordnungssystem mehrere, nicht notwendigerweise zu einem Volume gehörige Dimensionen umfassen kann. Für so einen Sachverhalt wird gelegentlich der Begriff *logical volume* verwendet. Zu sehen ist weiterhin, dass eine Datei gleichzeitig mehreren Dateisystemen angehören kann.

*logische Strukturen vs. physische Strukturen*

Wichtig ist der Prozess der Restrukturierung nach der Ablösung, wie die neue Zuordnung abgelöster Objekte zu digitalen Entitäten. Die Elemente Dimension und Volume wären eigentlich nach der Ablösung nicht mehr notwendig, da sie die physische Welt der Datenträger repräsentieren. Im Einzelfall ist aber zu prüfen, wie weit die alte Struktur geändert werden darf. Eine Referenzierung beispielsweise, ggf. in einer zugehörigen nichtdigitalen Entität, könnte zu Unklarheiten führen. Eine Analogie ist vielleicht bekannt: In einem Handbuch oder Installationsprogramm zu einer Software steht, man solle die Diskette Nr. 5 einlegen und auf der CD-ROM mit den vollständigen Disketteninhalten wird nichts mehr gefunden. Die logische Rolle der Elemente Volume und Dimension ist teilweise sehr schwer zu erkennen. Die ursprüngliche Struktur sollte im Zweifelsfall standardmäßig erhalten bleiben.

*die abgelösten Objekte*

Die abgelösten digitalen Objekte (Detached Digital Object) stellen eine abstrakte Klasse dar, die konkret eine Image-Datei (Imagefile), ein Datenbankobjekt (Database Object) oder auch eine Dateimenge (Fileset) sein kann. Bei der Image-Datei ist in der Regel der physische Bitstrom die unterste Schicht,[31] bei den anderen beiden auch der logische Bitstrom (vgl. Abb. 9.6). Eine Dateimenge kann ein herausgelöstes Verzeichnis sein wie ein *VIDEO_TS* (Title Set) einer DVD-Video.

*Dokumentation des Loslösungsprozesses*

Der Loslösungsprozess (Detachment) soll dokumentiert werden können. Dafür sind die Elemente Ablösehardware (Detachment Hardware) und Ablösesoftware (Detachment Software) vorgesehen, die auch die Protokollinformationen (Log-Information) liefern. Diese können, je nach verwendeter Software, Angaben zu Dateisystemen, zu Dateiverzeichnissen und zu Fehlern in unterschiedlicher Darstellungstiefe und Form sein. Diese Angaben stehen für eine weitere Analyse und Überführung in eine einheitliche Form zur Verfügung. Vorgesehen ist zusätzlich der Fall, dass ein

[31] bis auf die Nutzdaten für die darüber liegende Schicht entfernt.

digitales Objekt mehreren Loslösungsprozessen unterzogen werden muss.

*Implementierung*

Das entwickelte und auf Basis des Produkts Oracle 9.0.1. implementierte Datenbankschema soll hier wegen des Umfangs nicht näher vorgestellt werden. Einige Beispiele zeigen die praktische Anwendung.

**Abbildung 9.12**
*Direkte Auswertung eines BLOB-Typs: Test der Konformität von CD-Images zu ISO 9660 und Inhalte des PVD*

```
Signatur
Byte Position 2-6 Sektor 16 (falls ISO, dann CD001)
Publisher Identifier (nur bei ISO 9660)
-------------------------------------------------
L-INF560/X11491/NON-1
CD001
ADDISON-WESLEY PUBLISHING COMPANY,
ADVANCED BOOK PROGRAM, 2725 SAND HILL ROAD,
MENLO PARK, CALIFORNIA 94025, (415) 854-0300
-------------------------------------------------
I-INF190/YB5322/NON-1
kein IS0 9660 Dateisystem
-------------------------------------------------
ELT400/YA3264NON-1
CD001
P/S SOFTWARE UND COMPUTERTECHNIK
```

*Gewinnung von Metadaten aus einem CD-Image*

Ein wichtiger Aspekt ist die Gewinnung und Generierung von Metadaten. Dies kann direkt anhand der Images geschehen oder indirekt anhand der Protokollinformationen. Ein Beispiel für eine direkte Auswertung ist in Abb. 9.12 zu sehen. Mit Hilfe einer in der Datenbank gespeicherten Prozedur (stored procedure) werden CD-Images auf Konformität mit dem ISO-Standard getestet und ggf. der *Publisher Identifier* angezeigt. Ein sequenzielles Auslesen des BLOB-Typs ist nicht erforderlich. Die Analyseergebnisse können direkt in einen definierten Datenbankbereich überführt werden.

*Gewinnung von Metadaten aus einem Ablöseprotokoll*

Nun folgt eine indirekte Auswertung anhand der Protokolle des Loslösungsprozesses, die als CLOB-Typ gespeichert sind. Mit einer primitiven, aber entsprechendes terminologisches Wissen voraussetzenden Textmustersuche werden digitale Objekte mit dem Commodore-Dateisystem gesucht. Ein Ausschnitt der Ergebnisliste ist in Abb. 9.13 zu sehen. Für ein ausgewähltes digitales Objekt wird daraufhin das gesamte Verzeichnis ausgelesen, wie in Abb. 9.14 gezeigt.

```
L-INF350/S10585NON-2
CBM 1541, 170 KB

L-INF350/S10585NON-1
CBM 1541, 170 KB

L-INF200/N3621NON-1
CBM 1541, 170 KB
```

**Abbildung 9.13**
*Auswertung als CLOB gespeicherter Protokolle: Suche nach Commodore-Images*

```
CatDir, V 1.1. Copyright (C) 1996, 1997
by Tobias Haustein, Aachen.

CBM 1541,
170 KB, DOS 2A SS
Inhalt des Verzeichnisses 0:
  01.01.80  00:00     5842  ANOVA.ONEWAY.PRG
  01.01.80  00:00    10160  ANOVA.TWOWAY.PRG
  01.01.80  00:00     3810  BASIC STAT.FREQ.PRG
  01.01.80  00:00     3048  BASIC STAT.PRG
  01.01.80  00:00     4318  CHISQ GOODNESS.PRG
  01.01.80  00:00     5080  CHISQ INDEP.PRG
  01.01.80  00:00     3810  ONE GROUP PROP.PRG
  01.01.80  00:00     4826  R.ONEGROUP.PRG
  01.01.80  00:00     6350  R.TWOGROUP.INDEP.PRG
  01.01.80  00:00    10922  STAT.MENU.PRG
  01.01.80  00:00     4826  T-TEST.DEP.PRG
  01.01.80  00:00     3810  T-TEST.DEP.SUM.PRG
  01.01.80  00:00     4064  T-TEST.INDEP.PRG
  01.01.80  00:00     3810  T-TEST.INDEP.SUM.PRG
  01.01.80  00:00     4318  T-TEST.ONE GROUP.PRG
  01.01.80  00:00     3048  TWO INDEP PROP.PRG
82042 Bytes in 16 Dateien.
```

**Abbildung 9.14**
*Darstellung eines als CLOB gespeicherten Protokolls: Inhaltsverzeichnis einer Commodore-Diskette*

*Darstellung der Datenbankinhalte im XML-Format*

Zum Abschluss der Auswertungen zeigt ein Auszug der Datenbank im XML-Format, wie ein neutrales Format die Interoperabilität unterstützen kann. Die Auswertung erfolgt mit Hilfe der Formulierung einer normalen SQL-Abfrage, die an einen im Datenbanksystem integrierten Generator weitergeleitet wird, in diesem Fall *XSU* (XML SQL Utility). Im Beispiel werden u. a. die auf den Medien befindlichen Hinweise zur Low- und High-Level-Formatierung aufgelistet, die in einigen Fällen (insbesondere bei alten

Medien) sehr hilfreich sind. Standardmäßig erhalten die Marken die Namen der Attribute, wie sie in den Relationen definiert sind (vgl. Abb. 9.15). Um die Bedeutung der Marke ID_LIB und des darin eingeschlossenen Wertes zu erläutern, wird eine zweite Abfrage, die wieder eine Ausgabe im XML-Format erzeugt, nachgeschoben (vgl. Abb. 9.16). Die Verwendung verständlicher Marken und Werte kann Missverständnisse bei der Interpretation und somit schwer wiegende Probleme bei der langfristigen Verwaltung vermeiden. Eine dynamische Anpassung an gebräuchliche Begriffe oder Standards kann durch das Datenbanksystem verwaltet werden. Hierzu stehen verschiedene Techniken zur Auswahl, deren Flexibilität noch zu untersuchen wäre. Dass Flexibilität nötig ist, zeigen schon die Interpretationsschwierigkeiten beim Referenzmodell OAIS, wie die Ausführungen von Lupovici und Masanes (2000) zur Abgrenzung von *Context Information* und *Provenance Information* zeigen. Vor diesem Hintergrund ist der Nutzen der hier verwendeten Namen von Relationen und ihren Attributen nicht überzubewerten.

**Abbildung 9.15**
*XML-Ausgabe mit Hinweisen zur Low- und High-Level-Formatierung auf den Medien*

```
<?xml version="1.0"?>
<DIMENSIONSET>
 <DIMENSION>
  <ID_LIB>1</ID_LIB>
  <MEDIA_FORMFACTOR>FDC 5,25</MEDIA_FORMFACTOR>
  <FORMAT_MEDIA_TEXT>Apple II+,IIe,IIc
  </FORMAT_MEDIA_TEXT>
 </DIMENSION>
 <DIMENSION>
  <ID_LIB>1</ID_LIB>
  <SHELFMARK>L-INF510/P11234NON-1</SHELFMARK>
  <MEDIA_FORMFACTOR>FDC 5,25</MEDIA_FORMFACTOR>
  <FORMAT_MEDIA_TEXT>IBM PC</FORMAT_MEDIA_TEXT>
  <OS_FS_MEDIA_TEXT>PC-DOS (Microsoft-DOS)
  </OS_FS_MEDIA_TEXT>
 </DIMENSION>
</DIMENSIONSET>
```

*Perspektiven einer Online-Verfügbarkeit digitaler Objekte*

An diesen Beispielen kann man erkennen, welche Perspektiven sich bieten, wenn die digitalen Objekte bzw. deren ausführliche Beschreibungen, beispielsweise Inhaltsverzeichnisse, online in einem Datenbank-Managementsystem verfügbar sind. Es ist auch sichtbar geworden, dass sich der Prozess der Gewinnung von Me-

```
<?xml version="1.0"?>
<LIBRARYSET>
 <LIBRARY>
  <ID_LIB>1</ID_LIB>
  <CODE>706</CODE>
  <CODEDESCR>Bibliothekssigel</CODEDESCR>
  <NAME>Universitätsbibliothek der Universität der
        Bundeswehr München</NAME>
 </LIBRARY>
</LIBRARYSET>
```

**Abbildung 9.16** *XML-Ausgabe zur Erläuterung der Marke ID_LIB*

tadaten vom Loslösungsprozess größtenteils abkoppeln lässt und dadurch die Archivierung flexibler gestaltet werden kann.

*Testumfang*

In der durchgeführten Validierung wurden alle Disketten (ca. 300 Stück) der Teilbibliotheken Elektrotechnik und Luft- und Raumfahrttechnik der UniBwM bearbeitet. Bis auf wenige harte Fälle, z. B. ein fehlendes zweites Indexloch, konnten alle gelesen werden. Durch den Einsatz zusätzlicher Hard- und Software konnte auch die Fehlerhaftigkeit von Medien mit großer Wahrscheinlichkeit bestimmt werden. Alle erstellten Images und Protokolle wurden in das Datenbanksystem eingelesen (ca. 900 Dateien). Zusätzlich wurde zu Testzwecken, z. B. für die in Abb. 9.12 gezeigte Analyse, eine Reihe von CD-Images in das Datenbanksystem überführt.

### 9.4.6 Diskussion

*DBMS als unterstützendes Werkzeug*

DBMS stellen zwar keine konkrete Strategie zur Lösung der eigentlichen Sicherung der Abspielbarkeit zur Verfügung, sie sind aber eine integrierende Basistechnologie. Für die Leistungsfähigkeit eines DBMS sind die Qualität des konzeptionellen Modells und die korrekte, vollständige und aktuelle Umsetzung in ein logisches Datenbankschema von grundlegender Bedeutung. Ein wichtiges Qualitätskriterium ist die Flexibilität, da bisher keine konsolidierten bzw. standardisierten technischen und organisatorischen Modelle für die Langzeitarchivierung zur Verfügung stehen und weil technologische Entwicklungen bzw. Anforderungen an ein System zur Langzeitarchivierung schwer abzusehen sind.

*Speicherung von Hintergrundwissen im DBMS*

Insgesamt erfordert die Langzeitarchivierung viel technisches Hintergrundwissen. Auch wenn aktuelle Datenbanksysteme nicht für alle Arten der Informationsverarbeitung geeignet sind, so können doch einerseits sehr stark formalisierte Regeln, z. B. wie zur

Sicherung der Konsistenz erforderlich, und andererseits schwach strukturierte Dokumente mit CLOB- bzw. XML-Typen, z. B. zur Unterstützung der Recherche oder zur Beschreibung einer Abspielumgebung (technische Dokumentation), gespeichert werden.

*Festlegung auf DBMS-Produkte*

Die Festlegung auf ein bestimmtes Datenbankprodukt ist eine strategische Entscheidung. Trotz aller Standardisierungsbemühungen unterscheiden sich die Produkte auch bei einem gleichen logischen Modell doch deutlich. Ein Produktwechsel ist daher i. d. R. mit erheblichem Aufwand verbunden, auch wenn von den einzelnen Herstellern Migrationshilfen angeboten werden. Es besteht das Risiko, dass ein Produkt vom Markt verschwindet. Doch bei hoher Marktdurchdringung und Qualität kann mit einer langen Übergangsfrist und entsprechender Unterstützung gerechnet werden. Von spezialisierten Systemen wird in diesem Zusammenhang abgeraten. (Objekt-)relationale Datenbanken haben sich bisher als sehr flexibel erwiesen. So wird mit der bereits angekündigten vollständigen Unterstützung von XML (Fan 2002) auf neue Formen des Publizierens und des Datenaustausches eingegangen.

*weiterer Handlungsbedarf*

Bei der Konvertierung digitaler Objekte bzw. bei der Abspielbarkeit mit Hilfe von Emulationen sind, von Ausnahmen abgesehen, sicher noch tiefer gehende technische Überlegungen und praktische Versuche erforderlich. Auf alle Fälle steht man vor einer komplexen verwaltungstechnischen Aufgabe. Hierzu kann ein Datenbanksystem, sofern ein gutes Schema existiert, wertvolle Dienste leisten. Zu wünschen übrig lassen noch die Abstraktion von den Eigenschaften physischer Medien, das Management von Schemaänderungen und die Interoperabilität. Das Management großer Objekte (LOBs) erfordert einen zusätzlichen Aufwand für Tuning-Maßnahmen. Der entfernte Zugriff auf Datenbanken stellt kein Problem mehr dar, höchstens noch die begrenzten Bandbreiten der Kommunikationswege. Auch das Loslösen der digitalen Objekte von ihren ursprünglichen Medien kann man für die meisten Fälle als gelöst betrachten, obwohl hier noch Optimierungspotenzial vorhanden ist. Problematisch stellt sich eine geeignete technische Klassifizierung der digitalen Objekte als Schlüssel für die Langzeitarchivierung dar. Das liegt an der großen Menge, der Uneinheitlichkeit und den vielfältigen Abhängigkeiten. Hier bestehen aber gute Chancen, mit einem formalisierten Begriffssystem mehr Präzision bei weniger Handarbeit zu erreichen. Somit kann auch das Wissen, das für künftige Lösungsansätze notwendig ist, besser bestimmt und gesichert werden.

# 10 Emulation in der Praxis

*Dieses Kapitel diskutiert aktuelle Studien und Experimente bei der Emulationsstrategie. Diese Strategie zielt ja darauf ab, die Originalabspielumgebungen digitaler Dokumente durch Emulation auf künftigen Computern dauerhaft am Leben zu erhalten. Abschnitt 10.1 beschreibt zunächst ein Emulationsexperiment, wie es im Rahmen des europäischen Verbundprojekts NEDLIB an der Nationalbibliothek der Niederlande (Koninklijke Bibliotheek) durchgeführt wurde. Abschnitt 10.2 stellt anschließend einen Ansatz auf Basis des sogenannten Universal Virtual Computer (UVC) vor und gibt damit eine erste Antwort auf die Risiken des Migrationsansatzes und auf die Bedenken, die gegen den ursprünglichen Emulationsansatz vorgebracht werden.*

## 10.1 Emulationsexperiment von Rothenberg

*Emulation in einer Archivbibliothek*

Dieses Experiment sollte die Brauchbarkeit des Emulationsansatzes zur Erhaltung digitaler Publikationen in einer Archivbibliothek nachweisen. Die technischen Hintergründe und die Einbindung in das DSEP-Prozessmodell sind in diesem Buch bereits beschrieben. Im Zusammenhang mit dem europäischen Verbundprojekt NEDLIB führte RAND-Europe an der Nationalbibliothek der Niederlande (Koninklijke Bibliotheek – KB)[1] im Jahr 1999 eine Kurzstudie durch, die ein iteratives Vorgehen entsprechend einem Spiralmodell vorsah (Rothenberg 2000a). Der Test folgender Hypothese war das Ziel: Die originale Hardwareumgebung, die erforderlich ist, um die Originalsoftware zur Darstellung digitaler Publikationen ablaufen zu lassen, kann kosteneffektiv mit ausreichender Genauigkeit so beschrieben werden, dass Softwareemulatoren der Originalhardware für zukünftige Plattformen – als *host* bezeichnet – entwickelt werden können. Und außerdem stellen solche Emulatoren durch das Ausführen der Originalsoftware die di-

[1] siehe http://www.kb.nl.

gitalen Publikationen dem Original hinreichend ähnlich dar, sodass sie den Ansprüchen einer Archivbibliothek genügen.

*ein iteratives Experiment in drei Stufen*

Das Experiment sollte in drei Stufen ablaufen. Dazu wurde ein Ausgangsexperiment – 1999-Iteration genannt – und zwei Folgeexperimente – die Post-1999-Iteration und die Post-2000-Iteration – mit den folgenden Variablen definiert, deren Werte je nach Projektfortschritt und vorhandenen Ressourcen anzupassen waren:

1. Typen der zu erhaltenden Publikationen bzw. Dokumente
2. Kriterien für eine erfolgreiche Erhaltung für jeden Publikationstyp im Rahmen einer Archivbibliothek
3. Tests zur Überprüfung dieser Kriterien
4. Typen von Software, um jeden Publikationstyp darzustellen
5. originale Systemanforderungen und -konfigurationen für die benötigte Software
6. Lebenszyklen der gewählten Emulationsansätze
7. Werkzeuge für Emulatoren bzw. deren Erstellung einschließlich erforderlicher Ressourcen
8. verfügbare Hardware für die Emulation
9. Einfluss nichttechnischer Fragestellungen wie geistiges Eigentum bei Software

Für die 1999-Iteration wurden hierfür folgende Werte festgelegt:

1. *Publikationstypen:* kleine, durch die KB ausgewählte Beispielserie
2. *Erhaltungskriterien:* mit der KB zu entwickeln
3. *Tests der Erhaltungskriterien:* Identifizierung und Analyse von Alternativen
4. *Darstellungssoftware:* abhängig von der Beispielserie
5. *Originalsystem:* wie an der KB und bei RAND verfügbar
6. *Lebenszyklus:* kurz, wie bei aktuellen Systemen
7. *Werkzeuge:* Freeware oder kommerzielle Standardprodukte (commercial of the shelf – COTS)
8. *Hardware:* wie an der KB und bei RAND verfügbar
9. *Nichttechnischer Einfluss:* minimal

*Bestimmung der Testmenge*

Für die Auswahl einer Testmenge für diese erste Iteration wurden einige generelle Kriterien entwickelt:

- ❑ repräsentativ bezüglich logischer Formate, Interaktivität und Arten der Standardisierung, die für eine Archivbibliothek von Interesse sind

- ❏ repräsentativ bezüglich des Bestands einer Archivbibliothek idealerweise auf Basis einer funktionalen Kategorie
- ❏ Betonung der herausfordernsten Fälle, aber Einschluss einiger Beispiele für weitere wichtige Fälle
- ❏ Auswahl einer möglichst kleinen Menge, die genügend Kategorien abdeckt und für jede Kategorie genügend Beispiele liefert

*Herausfinden der kritischen Fälle*

Die so getroffene Auswahl sollte weiter analysiert werden, um eine minimale Menge für jede Kategorie zu bestimmen und um letztendlich herauszufinden, was die wirklich anspruchsvollsten Fälle sind. Auf dieser Basis wählte die KB eine Reihe von sechs CDs und fünf Online-Zeitschriftenpublikationen im PDF-Format aus. Da die Zeitschriften aus Zeitmangel nicht weiter analysiert wurden, zeigt Tabelle 10.1 nur die CD-Publikationen. Dabei werden für das Experiment irrelevante Attribute (z. B. Autor, Herausgeber und ISDN) nicht aufgeführt.

*Eigenarten digitaler Publikationen*

Aus Zeitmangel entfiel auch die formale Definition typischer Nutzungsszenarien als Grundlage für eine Überprüfung, ob die Erhaltungsmaßnahmen einer beabsichtigten bzw. erwarteten Nutzung gerecht werden. Während sich die KB unabhängig davon einer weiteren Charakterisierung der Publikationen bezüglich Funktionalität und Verhalten widmen sollte, kümmerte sich RAND auch um unauffällige Eigenschaften, die für eine Beschreibung allgemein, über die ausgewählte Testmenge hinaus, nützlich sein könnten:

- ❏ Eigenarten oder Schwierigkeiten, die bei der Installation vermutet werden
- ❏ Verweise auf die Hardwareumgebung, z. B. auf Tastennamen
- ❏ Navigationsmöglichkeiten, z. B. Blättern (paging) oder Rollen (scrolling), auch wenn sie in der ursprünglichen Publikation nicht vorhanden waren
- ❏ Funktionalitäten, die durch eine aktuelle Darstellungssoftware geboten werden, aber in der ursprünglichen Publikation nicht vorhanden waren
- ❏ ungewöhnliche Interaktionen, z. B. Setzen von Lesezeichen oder Präferenzen, insbesondere bei einer Sicherung über Sitzungen hinweg
- ❏ Abhängigkeiten vom Originalmedium, z. B. die Abfrage einer laufenden Nummer einer CD

| Attribute | Werte |
|---|---|
| Titel/Jahr | Marine Planarians of the World, 1996 |
| Medium | 1 CD-ROM (Windows) |
| Anforderungen | IBM compatible PC, 80486 processor, 8 MB RAM, Windows 3.1 or Windows 95, VGA color monitor, CD-ROM player |
| Titel/Jahr | Cultivation and Farming of Marine Plants, 1997 |
| Medium | 1 CD-ROM (Macintosh and Windows) |
| Anforderungen | Macintosh computer with 68040 processor or PowerPC, MacOS version 7.0, 8 MB RAM, 14-inch 16-bit color monitor, quad-speed CD-ROM player / IBM compatible PC, 80486 processor, 8 MB RAM, Windows 3.1 or Windows 95, VGA 16 bit color monitor, quad-speed CD-ROM player, sound card compatible with Windows |
| Titel/Jahr | Paranasal Sinuses and Anterior Skull Base, 1993 |
| Medium | 1 CD-ROM, 1 CD-I |
| Anforderungen | IBM compatible PC, 33 MHz 486 CPU or better, 4 MB internal memory, Microsoft Windows 3.1, 8 MB available on hard disk for installation, CD-ROM player with adequate access time (less than 400 milliseconds), accelerated 256 color VGA adapter compatible with Microsoft Windows (resolution 640x480), mouse or other pointing device compatible with Microsoft Windows / CD-I: Philips CDI 220 or the portable CDI 360 player with the Digital Video (DV) cartridge, TV with scart connector, 6 inch color LCD-screen |
| Titel/Jahr | Encyclopedia of Neuroscience, 1999 |
| Medium | 1 CD-ROM (Macintosh and Windows) |
| Anforderungen | Macintosh, 68030 or better processor (Power Macintosh recommended), MacOS supported printer, 8 MB RAM (16 MB recommended), system 7 or better (7.5 recommended), double-speed CD-ROM drive (quad-speed recommended) / IBM compatible PC, 486 DX66 or better processor, 8 MB RAM (16 MB recommended), Windows 95, color monitor recommended, double-speed CD-ROM drive (quad-speed recommended), VGA video card (SVGA recommended), 8 bit sound card, Microsoft mouse or compatible, Microsoft supported printer |
| Titel/Jahr | Vraag en Aanbod Handelswijzer, 1998 |
| Medium | 1 CD-ROM |
| Anforderungen | ohne Angaben |
| Titel/Jahr | Auto Technisch Handboek, 1999 |
| Medium | 2 CD-ROM |
| Anforderungen | Windows 3.1 or Windows 95 |

**Tabelle 10.1** *CD-Publikationen für das Emulations-experiment*

- Interaktionen zwischen dem Originalmedium und der Anwendung, z. B. der erforderliche Wechsel des Mediums, um eine gesamte Datenbank abzusuchen
- Abhängigkeiten von der Originalverpackung oder beigefügter Information, z. B. der Seriennummer
- Verweise auf externe Informationsquellen wie URL insbesondere wenn sie zur Installation notwendig sind
- Fähigkeiten, in eine Datei zu drucken
- aktive Texte
- animierte Graphiken, Klang bzw. Ton (sound), Videoclips, Geisterstimmen (voiceover) oder andere visuelle Bildschirmanzeigen, bei denen die Synchronisation zwischen Klang bzw. Ton und visueller Darstellung für die Bedeutung kritisch sind
- Semantischer und funktionaler Einsatz von Farbe, z. B. für bereits besuchte Links
- Verhalten von Rahmen, Texten und Bildern, wenn die Größe des umschließenden Fensters geändert wird, z. B. der Umbruch oder das Verdecken von Texten
- ungewöhnliche Darstellungskontrollen, z. B. *font zoom*

*Besonderheiten konkreter CD-Publikationen*

Eine vollständige Beschreibung der Testmenge anhand dieser Gliederung konnte auf Grund des engen Projektrahmens nicht durchgeführt werden, aber eine Reihe von Eigenschaften wurde informell dokumentiert, was Tabelle 10.2 ausschnittsweise wiedergibt. Allen CDs ist gemeinsam, dass sie keine *Autorun*-Funktionalität, die nach dem Einlegen einen Programmstart veranlasst, besitzen. Der Nutzer muss selbst ein Initiierungsprogramm (setup) starten, das sich in unterschiedlichen Verzeichnissen befinden kann.

*Kriterien für Authentizität*

Eine weitere Aufgabe war die Entwicklung von Kriterien für die Authentizität, da auf keine akzeptierte Definition der *digitalen Erhaltung*, die die Sicherung aller Aspekte eines digitalen Dokuments berücksichtigt, zurückgegriffen werden konnte. Hierfür wurde versucht, Prinzipien zur Authentizität aufzustellen, die je nach Publikationstyp etwas variieren können. Daraus sollte dann die Ableitung spezifischer Kriterien möglich sein. Folgende Prinzipien, als Vergleich zwischen Original und zu erhaltender digitaler Publikation formuliert, wurden angegeben:

| Titel | Besonderheiten |
|---|---|
| Marine Planarians of the World | Demo kann nur durch Drücken der »Fluchttaste« (escape) gestoppt werden und besitzt besondere Navigationsknöpfe. CD ist für das Abspielen erforderlich. QuickTime for Windows muss zum Ansehen der Videos installiert sein und zum Ausdrucken muss ein Standarddrucker (default) unter Windows installiert sein. Anwendungsfenster kann nicht den ganzen Bildschirm ausfüllen. |
| Cultivation and Farming of Marine Plants | Demo kann nur durch Drücken der Fluchttaste (escape) gestoppt werden und besitzt besondere Navigationsknöpfe. CD ist für das Abspielen erforderlich. QuickTime for Windows muss zum Ansehen der Videos installiert sein und zum Ausdrucken muss ein Standarddrucker (default) unter Windows installiert sein. Anwendungsfenster kann nicht den ganzen Bildschirm ausfüllen. |
| Paranasal Sinuses and Anterior Skull Base | Erfordert unter Windows 95 das Programm update.exe auf der 3,5-Zoll-Diskette. Das Startfenster kann nicht in voller Größe gezeigt werden. Die Taste F1 für Hilfe erzeugt Fehler auf dem Bildschirm. CD ist für das Abspielen erforderlich. Video for Windows muss zum Ansehen der Videos installiert sein, und es besteht keine Druckmöglichkeit innerhalb der Anwendung. Anwendungsfenster kann nicht den ganzen Bildschirm ausfüllen, sondern nur Bilder nach Aufruf der Zoom-Funktion. |
| Encyclopedia of Neuroscience | Installationsprogramm installiert erforderliche Programme Netscape Navigator, Shockwave Plug-in und Apple QuickTime for Windows, falls noch nicht installiert. CoolTalk for Navigator kann optional installiert werden. HTML-Seite ist der Start der eigentlichen Anwendung; Netscape Navigator ist der Rahmen für die gesamte Anwendung. CD ist für das Abspielen nicht erforderlich. Aber zum Ausdrucken muss ein Standarddrucker (default) unter Windows installiert sein. Anwendungsfenster kann den ganzen Bildschirm ausfüllen. |
| Vraag en Aanbod Handelswijzer | Zur Beschleunigung optionale Installation von weiteren Daten der Anwendung; bei den Untermenüs kann nach Programmstart die Auswahl durch Anklicken einer Auswahl oder Eingabe einer Zeichenkette getroffen werden. Die Textfarben können in der Anwendung geändert werden, wobei die letzte Einstellung gesichert wird. CD ist für das Abspielen erforderlich. Video for Windows muss zum Ansehen der Videos installiert sein und zum Ausdrucken muss ein Standarddrucker (default) unter Windows installiert sein. Anwendungsfenster kann den ganzen Bildschirm ausfüllen, nicht jedoch das graphische Hauptfenster und das Texteingabefenster. |
| Auto Technisch Handboek | Hauptanwendung ist der Multmedia Viewer 2.00, der durch den bei der Installation gesetzten shortcut aufgerufen wird. CD ist für das Abspielen erforderlich, aber die Anwendungsdaten sind auf zwei CDs verteilt. Zum Ausdrucken muss ein Standarddrucker (default) unter Windows installiert sein. Anwendungsfenster kann den ganzen Bildschirm ausfüllen. |

**Tabelle 10.2**
*Besonderheiten der CD-Publikationen*

- gleich für jeden Sinn und Zweck (*intents and purposes*)
- gleiche Funktionalität und Beziehungen zu anderen Publikationen
- gleiches *Look-and-Feel*
- gleicher Inhalt (*content*)
- gleiche Darstellung (*description*)

*Rolle des Herausgebers*

Die Diskussionen mit der KB legten nahe, dass die Prinzipien die Vereinbarungen mit den Herausgebern widerspiegeln sollten. Obwohl diese Übereinkünfte bei den am NEDLIB-Projekt beteiligten Bibliotheken abwichen, erwarteten doch die meisten Verleger, dass die Publikationen in der Form wie eingereicht auch erhalten werden. Diese Aussage ist bei digitalen Publikationen jedoch diskussionsbedürftig. Für die folgenden Tests dieser Iteration kam das erste und anspruchsvollste Prinzip zur Anwendung. Dieses umfasst nicht nur die Sicht des Nutzers, sondern auch die des Erstellers, weshalb die Werkzeuge zum Modifizieren und Neuformatieren mit einzuschließen sind.

*Emulation und Hardware*

Weiterhin war herauszufinden, welche Aspekte der Hardware zu emulieren sind, um die Darstellung und Nutzung aller Publikationstypen sicherzustellen. Die Möglichkeit, digitale Publikationen auf einer Reihe von Plattformen und Konfigurationen und oft mit unterschiedlichen Peripheriegeräten und Programmen wie Browsern abspielen zu lassen, führte zur Definition eines ursprünglichen Streubereichs (engl. *native range of variability*). Eine Emulation ist dann als gültig anzusehen, wenn sie innerhalb dieses ursprünglich akzeptierten Bereichs liegt. Auf Basis dieses Begriffs, der noch nicht formalisiert ist, wurden Schritte festgelegt, um die relevanten Aspekte der Hardware zu identifizieren. Wesentlich dabei ist, dass für jeden Publikationstyp auf Grundlage der Authentizitätskriterien die so genannten erforderlichen Publikationsattribute bestimmt werden. Dazu werden die Originalpublikationen mit der originalen Hard- und Software abgespielt. Danach werden weitere verschiedene Originalprogramme einschließlich Versionen und Varianten bestimmt, die diese Attribute innerhalb eines akzeptablen Rahmens darstellen. Daraus ergibt sich der ursprüngliche Streubereich für die Attribute. Ähnlich wird die in Frage kommende Originalhardware, z. B. Plattformen und Konfigurationen, bestimmt, auf denen die vorher identifizierte Software läuft. Gegebenenfalls führt dies zu einer Erweiterung des akzeptablen Rahmens für die Darstellung der Attribute. Abschließend ist zu analysieren, wie die Attribute von den Eigenschaften

der Hardware abhängen, um letztlich die so genannten erforderlichen Hardwareattribute zu bestimmen.

*kommerzielle Emulationssoftware*

Da die Entwicklung jeglicher spezifischer Software für diese erste Phase ausgeschlossen war, kam nur (kommerziell) verfügbare Emulationssoftware in Betracht. Der Auswahlprozess soll hier nicht näher beschrieben werden, da er stark von technischen Details und spezifischen organisatorischen Problemen geprägt war. Von Bedeutung ist jedoch der Hinweis, dass wegen grundsätzlicher Vorteile eine Hardwareemulation gesucht wurde. Softwareemulationen wie *WINE* waren daher ausgeschlossen. Die Wahl fiel eigentlich auf *SoftWindows* der Firma Insignia, tatsächlich genutzt wurde jedoch *Virtual PC for Mac* der Firma Connectix.

**Erläuterung:** WINE realisiert die Programmierschnittstellen (*Application Programming Interface* – kurz API) Win32 und Win16 auf Basis von Unix. Virtual PC liefert ohne zusätzliche Hardware die Fähigkeiten eines Intel-Pentium-PCs auf einem Macintosh und kann somit Gastgeber, also host, für PC-Betriebssysteme sein, z. B. Windows 95, DOS oder Linux. Etwas verwirren könnte der Umstand, dass Virtual PC auch mit vorinstallierten Betriebssystemen geliefert wird.

*Durchführung des Experiments*

Damit waren die Vorbereitungen für das eigentliche Experiment getroffen, jedoch mit einigen Einschränkungen, wie das Fehlen formal beschriebener, typischer Nutzungsszenarien und einer vollständigen Charakterisierung der Publikationen. Daher und auf Grund vermuteter Geschwindigkeitsprobleme durch den Emulations-Overhead wurde auf einen vorgesehenen Doppelblindversuch, bei dem Nutzer und Beobachter die Testumgebung nicht kennen, verzichtet, was für eine erste Iteration allerdings als nicht gravierend bewertet wurde. Zwei Rechner dienten für den informellen Test als Umgebung: ein Intel-Pentium-PC mit Windows 95 als Originalplattform und ein Macintosh als host für eine Emulation von Windows 95. Tatsächlich konnten bei allen ausgewählten Publikationen keine signifikanten Unterschiede festgestellt werden. Sogar alle Abstürze und Hänger der Windows-Emulation traten auch in der Originalumgebung auf! Teilweise stellte sich die Emulation als robuster heraus, da sie leichter normiert werden kann, z. B. bei jedem Start einer Anwendung. Insgesamt wurden die Ergebnisse als überraschend ermutigend eingeschätzt.

*Erkenntnisse*

Die Ergebnisse dieser 1999-Iteration führten zu einer Reihe von Erkenntnissen und Empfehlungen. So konnte gezeigt werden,

dass der Emulationsansatz prinzipiell geeignet ist, um Inhalte, Verhalten und Look-and-Feel digitaler Publikationen auch in einer künftigen Umgebung zu rekonstruieren. Für den praktischen Nachweis wurden weitere Entwicklungen als notwendig erachtet:

- Spezifikationstechniken, die die Beschreibung aller relevanten Hardwareeigenschaften erlauben
- Techniken, um Emulationen ohne großen Aufwand auf beliebigen, künftigen Plattformen laufen zu lassen
- Verfeinerung der Metadaten, um beispielsweise eine Verknüpfung der Publikationen zu Software und emulierter Hardware herzustellen
- unbegrenzte Erhaltung der Metadaten in einer für den Menschen verständlichen Form
- Verfeinerung der Authentizitätskriterien und der zugehörigen Tests (zur Erinnerung: Bei der Formulierung der Authentizitätsprinzipien wird beispielsweise der unscharfe Begriff Look-and-Feel verwendet)

Um das Potenzial der Emulation gründlicher zu erforschen, wurden weitere Iterationen vorgeschlagen, unseres Wissens aber nicht mehr durchgeführt. Die Post-1999-Iteration sah neben einer Revision der oben genannten Variablen des Experiments im Wesentlichen die Entwicklung einer portablen Emulationsumgebung vor und die Post-2000-Iteration Vertiefungen und Erweiterungen besonders in Hinblick auf Langfristigkeit.

*Heterogenität und Komplexität der Publikationen als Argument für den Emulationsansatz*

Die Beschreibung der ausgewählten Publikationen zeigt anschaulich, mit welcher Heterogenität und Komplexität eine Archivbibliothek in der Realität konfrontiert wird. Genau diese Eigenschaften der Publikationen sind ein starkes Argument für den Emulationsansatz. Kritisch ist jedoch bei diesem Vorgehen, dass auch Publikationen neben der üblichen Mensch-Maschine-Schnittstelle immer mehr spezialisierte Geräte, z. B. Kommunikationseinrichtungen, ansprechen, die in dieser Art möglicherweise künftig nicht mehr zur Verfügung stehen. Die Logik dieser Hardware wird beim Ansatz von Rothenberg nicht explizit. Interessant wären die Ergebnisse der weiteren Iterationsschritte. Somit könnte u. a. der Aufwand für die Erstellung einer Emulatorspezifikation und einer logischen Sicht, wie sie für einen *Universal Virtual Computer* (UVC) erforderlich ist, verglichen werden. Der UVC-Ansatz sieht auch die Definition abstrakter Geräte vor. Dieser Ansatz ist Gegenstand der Diskussion des nächsten Abschnitts.

## 10.2 Universal Virtual Computer

*eine Antwort auf die Risiken und Mängel der Migration und Emulation*

Der Ansatz auf Basis eines *Universal Virtual Computer* (UVC) von Lorie (2000) ist eine Antwort auf die Risiken des Migrationsansatzes[2] und auf die Bedenken, die gegen den im vorhergehenden Abschnitt beschriebenen Emulationsansatz vorgebracht werden. Dort war ja die gemeinsame, gekapselte Speicherung folgender Objekte vorgesehen:

1. eine Beschreibung des für Texte genutzten Alphabets
2. eine weitgehend textuelle Beschreibung der Metadaten
3. die Daten als Bitstrom
4. als Bitstrom das Programm, das verwendet wurde, um die Daten zu speichern und zu manipulieren, einschließlich des Betriebssystems und weiterer notwendiger Komponenten
5. eine detaillierte Beschreibung der Rechnerarchitektur

Als Alternative zum 5. Objekt sieht Rothenberg eine Spezifikation für Emulatoren vor, um eine Entwicklung in der Zukunft zu ermöglichen. Lorie kritisiert dabei die Abhängigkeit vom 4. Objekt. Sie ist zwar bei einer Archivierung eines Programmverhaltens zu rechtfertigen, aber bei der Archivierung von Daten führt dies zum Overkill. Zwei Beispiele sollen dies veranschaulichen: Wenn bei der Speicherung von Bildern nur ein Interesse der Nachwelt am Endergebnis besteht, müsste trotzdem das ganze Softwaresystem zum Erstellen, Modifizieren und Retuschieren gespeichert werden. Oder wenn mit *Lotus Notes* eine E-Mail verschickt wurde, müsste für eine künftige Nutzung dieses ganze Softwaresystem, das noch etliche weitere Dienste für eine Gruppenarbeit anbietet, aktiviert werden, nur um den Inhalt wieder herzustellen. Als noch schwerwiegender wird bewertet, dass die Anwendungsprogramme nur eine bestimmte Sicht der Daten, z. B. in graphischer Form, zeigen, ohne einen direkten Zugriff auf die Daten zu geben. Somit ist es unmöglich, die Grunddaten aus dem alten System in ein neues zu überführen. Darüber hinaus wird die künftige Erstellung von Emulationssoftware auf Basis des 5. Objekts bzw. einer Spezifikation als bedenklich eingestuft, zumal die Korrektheit auf Grund der fehlenden Hardware nicht mehr festgestellt werden kann. Ohne die Wichtigkeit organisatorischer Aspekte zu verleugnen, konzentriert sich dieses Projekt auf technische Fragestellungen.

[2] Auf die Migrationsrisiken sind wir schon in Abschnitt 3.4, S. 56ff., ausführlich eingegangen.

*Prozesse der Archivierung und Wiederherstellung*

Um die geschilderten Bedenken zu umgehen, hat Lorie eine andere Methodik vorgeschlagen, wobei zwischen einer Archivierung des Programmverhaltens und einer Archivierung von datenorientierten Dokumenten unterschieden wird. Die Phasen für den zweiten Fall, der bereits genauer untersucht ist, sehen folgendermaßen aus (vgl. Abb. 10.1):

1. *Zum Archivierungszeitpunkt:* Erstellung eines Programms P, das die logische Sicht des zu archivierenden Bitstroms herstellen kann und auf dem UVC läuft, sowie zusätzlich Erstellung von Metadaten, die durch ein UVC-Programm decodiert werden können.
2. *Zum Wiederherstellungszeitpunkt:* Ausführung des erstellten Programms P mittels einer Emulation des UVC in der neuen Umgebung und Interpretation der dadurch erzeugten logischen Sicht zur Darstellung durch entsprechende Wiederherstellungsprogramme wie viewer.

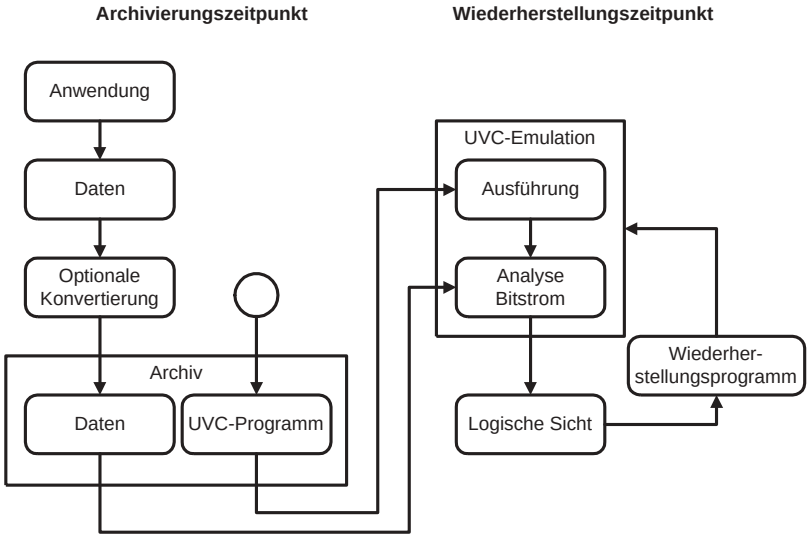



**Abbildung 10.1** *Datenorientierte Langzeitarchivierung mit UVC*

*Universal Virtual Computer (UVC) als Kernkomponente*

Die Kernkomponente ist dabei der UVC, also ein Computer, der nicht physisch realisiert werden muss. Es handelt sich dabei um einen Allzweckcomputer, der sich an der *Von-Neumann*-Grundarchitektur orientiert, wie sie von Beginn an bei Rechnern existiert. Sekundäre Eigenschaften, die beispielsweise die Ausführungsgeschwindigkeit oder die Speichernutzung optimieren, wer-

den weggelassen. Außerdem führt der UVC keine Ein- und Ausgabeoperationen aus. Diese sind durch den UVC-Emulator und durch die Wiederherstellungsprogramme zu erledigen, indem sie einen virtuellen Speicher nutzen, auf den ein UVC-Programm zugreifen kann. Der Speicher wird bitweise adressiert, das Konzept Byte und Wort gibt es nicht. Darüber hinaus ist der Speicher segmentiert, um die Speicherverwaltung und den Datenaustausch zwischen unabhängigen Programmeinheiten zu erleichtern. Über die Speichergröße und die Anzahl der Register, deren Länge variabel sein kann, werden keine Annahmen getroffen. Die Befehlslänge ist ebenfalls variabel, aber der Befehlssatz soll minimal gehalten werden. Es wird angenommen, dass die UVC-Programme sowieso in einer höheren Programmiersprache entwickelt werden.

*Emulation des UVC*

Da die Emulation des UVC in der Zukunft erstellt werden muss, ist eine präzise und verständlich bleibende Beschreibung im Rahmen einer Konvention unabdingbar. Die Erstellung selbst wird als einfach erachtet und es wird sogar angenommen, dass die Hersteller ihre Computer damit ausstatten. Prototypisch wurde bereits ein UVC-Emulator für Windows/NT auf Basis eines Intel-Prozessors realisiert (Lorie 2002).

*Logisches Modell für ein Dokument*

Für die künftige Interpretation eines Dokuments muss ein logisches Modell formuliert werden. Prämissen für den Entwurf sind die Einfachheit, sodass wenige zusätzliche Beschreibungen notwendig sind, und die Wiederherstellbarkeit als einziger Anwendungszweck. Das logische Modell sollte leicht auf die lineare Struktur eines Bitstroms abgebildet werden können, wofür auch hierarchische Strukturen, wie u. a. mit XML darstellbar, als geeignet betrachtet werden. Ein Beispiel, das der Dokumentation einer Dateivalidierung im Audioformat *WAV* entnommen ist (Lorie *et al.* 2001), hier verkürzt das Vorgehen in der Notation einer DTD[3] illustriert:

```
<!ELEMENT SONG (AUTHOR*,TRACK*,ALBUM*,DURATION,
                SAMPLINGFREQUENCY,DATA,END+)>
<!ELEMENT AUTHOR (#PCDATA)>
<!ELEMENT TRACK (#PCDATA)>
<!ELEMENT ALBUM (#PCDATA)>
<!ELEMENT DURATION (#PCDATA)>
<!ELEMENT SAMPLINGFREQUENCY (#PCDATA)>
<!ELEMENT DATA (#PCDATA)>
<!ELEMENT END (#PCDATA)>
```

[3] Die Autoren verwenden eine Notation *ähnlich* XML.

*Schemadefinition für das logische Modell*

Um dieses Schema interpretieren zu können, muss es auch beschrieben werden, wozu aber vorher das Schema für die Metadaten dokumentiert sein muss, was wiederum mit Hilfe einer DTD realisiert werden kann. Die *Level* modellieren hier die hierarchische Struktur der Record-Felder.

```
<!ELEMENT FIELDS
(LEVEL*,NAME*,DESCRIPTION*,TYPE*,VALUE*)>
<!ELEMENT LEVEL (#PCDATA)>
<!ELEMENT NAME (#PCDATA)>
<!ELEMENT DESCRIPTION (#PCDATA)>
<!ELEMENT TYPE (#PCDATA)>
<!ELEMENT VALUE (#PCDATA)>
```

*Beispiel für die Schemadefinition*

Dieses Schema ist per Konvention ein für alle mal festzulegen. Eine konkrete Ausprägung der Metadaten in Form einer XML-Datei entsprechend der DTD könnte ausschnittsweise für die zwei Felder SAMPLIGFREQUENCY und DATA wie folgt aussehen:

```
<FIELDS>
  <LEVEL>1</LEVEL>
  <NAME>SAMPLINGFREQUENCY</NAME>
  <DESCRIPTION>
    Frequency at which the song is sampled
  </DESCRIPTION>
  <TYPE>#PCDATA</TYPE>
  <LEVEL>0</LEVEL>
  <NAME>DATA</NAME>
  <DESCRIPTION>
    The actual data of the song stored in
    byte format
  </DESCRIPTION>
  <TYPE>#PCDATA</TYPE>
</FIELDS>
```

Und die Daten selbst könnten konkret für die zwei exemplarischen Felder folgendermaßen aussehen, wobei das sehr lange Datenfeld abgeschnitten wurde:

```
<SONG>
  <SAMPLINGFREQUENCY>11025</SAMPLINGFREQUENCY>
  <DATA>-6-1190-10-1-4-1-20-160-4-1-135050150-6
```

*Wiederherstellung der logischen Dokumentenstruktur*

Die eigentlichen Daten können vorab einer Formatkonvertierung, bei graphischen Daten z. B. JPEG nach TIFF, unterzogen werden, um eine künftige Decodierung zu erleichtern. In Abb. 10.1 ist dies als optionale Konvertierung bezeichnet. Um nun die als Bitstrom linearisierte Beschreibung auf einer künftigen Plattform zu interpretieren, muss zur Archivierungszeit ein UVC-Programm – kurz UVC-P – geschrieben werden. So ein Programm soll die logische Struktur und die zugeordneten Daten in einer elementaren Weise wieder erkennbar machen. Als geeignete, auch für den Menschen verständliche Darstellung werden hierarchisch gestufte Paare mit einer Marke und einem Wert erachtet. UVC-P selbst wird als einfache Zeichenfolge mit Ziffern 0 bis 9, Pluszeichen, Minuszeichen, einfachen Anführungszeichen und Leerzeichen archiviert. Der erstellte Prototyp verwendet eine ASCII-Codierung, aber eine effizientere 4-Bit-Codierung würde keine Einbuße an Einfachheit bedeuten. Die verwendete Codierung muss verbindlich in einer Konvention festgelegt werden. Für die Programmerstellung wurde ein UVC-Assembler realisiert, der alle Befehle symbolisch anspricht und einige Konstrukte höherer Programmiersprachen enthält.

*Kommunikation mit einer künftigen Umgebung*

Natürlich müssen das UVC-Programm und die UVC-Emulation mit einer neuen Umgebung zum Zeitpunkt der Wiederherstellung kommunizieren können, wozu ein eigenes einfaches Programm in der aktuellen Umgebung erforderlich ist. Dies übergibt an das Emulationsprogramm einige Parameter wie die Adresse von UVC-P und des archivierten Biststroms sowie die Speicheradressen für die Rückgabe der Marken und Werte. Dieser Aufrufmechanismus muss ein weiterer Bestandteil einer verbindlichen Konvention sein, die nun insgesamt aus vier Teilen besteht und für die Zukunft erhalten werden muss, aber nicht jedem zu archivierenden Dokument mitgegeben zu werden braucht. Mit Hilfe des UVC-Programms könnte die logische Sicht ausschnittsweise so aussehen:

```
MARKE: SAMPLINGFREQUENCY WERT: 11025
MARKE: DATA              WERT: 6-1190-10-1-4-1-20-160
```

*Interpretation und Darstellung der logischen Dokumentenstruktur*

Zur Wiederherstellung muss letztendlich die durch Marken und Werte beschriebene logische Struktur interpretiert und auf einer aktuellen Plattform dargestellt werden. Hierbei gehen die Protagonisten dieser Erhaltungsmethode davon aus, dass im allgemeinen Fall die entsprechenden Programme verfügbar sein werden und dass Ad-hoc-Implementierungen der logischen Sicht künftig auch Ad-hoc-Programmentwicklungen erfordern werden. Eine Vi-

sion ist auch die Erstellung und Verteilung von Werkzeugen. Bereits getestet wurden u. a. Formate wie JPEG, GIF und in Teilaspekten PDF (Lorie 2002).

*datenorientierte vs. verhaltensorientierte Archivierung*

Die bisherigen Ausführungen beziehen sich auf eine *datenorientierte* Archivierung. Die *verhaltensorientierte* Archivierung erfordert weitergehende Überlegungen. Weniger Probleme werden gesehen, wenn die Programme keine Interaktion mit der externen Welt besitzen wie Ein- und Ausgabe-Operationen oder Unterbrechungen (engl. *interrupts*) im Allgemeinen. Zusätzlich zu den Daten wird das Programm gespeichert, dessen Befehle ein UVC-Programm emuliert. Komplizierter wird es, wenn Ein- und Ausgabe-Operationen, z. B. Drucken, vorgesehen sind. Hierzu müsste ein weiteres UVC-Programm geschrieben werden, das diese Operationen nachahmt und die Ergebnisse im Fall des Druckens in eine Zwischenstruktur schreibt. Ist diese Struktur einfach und gut dokumentiert, sollte es möglich sein, eine Abbildung auf ein aktuelles Gerät zu erstellen, dessen konkrete Eigenschaften jetzt noch nicht bekannt sein können. Die durch das UVC-Programm erzeugte Zwischenstruktur könnte bereits an den heutigen Geräten getestet werden. Ähnlich müssten weitere abstrakte Geräte definiert werden, z. B. für Bänder, für Speicher mit wahlfreiem Zugriff, für die sequenzielle Ein- und Ausgabe von Zeichen (Bildschirme, Tastaturen) und für die Positionierung in einem Koordinatensystem (Mäuse, Cursor).

*Vorteile*

Als große Vorteile ihres Ansatzes heben die Entwickler folgende Punkte hervor:

- Durch den UVC entfällt die Emulation einer realen Maschine.
- Die Notwendigkeit, sich auf Standards für Formate zu einigen, fällt weg. Nur die vier oben genannten Konventionen müssen als Standard eingehalten werden.
- Nur die UVC-Emulation muss bei einer neuen Rechnerarchitektur geschrieben oder neu übersetzt werden. Die archivierte Information bleibt unbeeinflusst.
- Der UVC kann so allgemein und einfach gestaltet werden, dass eine Relevanz in der Zukunft gegeben ist.

*weiterer Entwicklungsbedarf*

Andererseits sehen die Entwickler noch weiteren Handlungsbedarf. So ist die Frage eines geeigneten Alphabets nicht abschließend geklärt. Die Annahme der künftigen Allgemeinverständlichkeit von ASCII wird als Restriktion gesehen. Außerdem ist das Schema zur Interpretation von Schemata noch zu rudimentär. Es

müsste um die Unterstützung von Varianten erweitert werden. Dazu wird das gegenwärtige Interesse an XML als hilfreich erachtet.

*Ansätze zur verhaltensorientierten Archivierung sind noch rudimentär!*

Auch diese Architektur kommt nicht ohne gewisse Annahmen über zukünftiges Geschehen aus. Hilfreich könnte die Unterscheidung zwischen datenorientierter und verhaltensorientierter Archivierung sein, da sie bezüglich der Komplexität ein abgestuftes Vorgehen nahelegt. Dies strukturiert vielleicht auch die Diskussion, wieviel Authentizität eigentlich erforderlich ist und welche Rolle dieser Begriff bei einer Vielzahl von heute möglichen interaktiven Ablaufumgebungen wie Browsern mit ihren eigenen Navigationsmöglichkeiten spielt. Zur obigen Kategorisierung der Archivierung ist auch anzumerken, dass immer mehr Publikationen mit verhaltensorientierten Komponenten ausgestattet werden. Hierzu zählen nicht nur Lexika und interaktives Lehrmaterial, sondern auch Dokumente, die mit umfangreichem Datenmaterial angereichert sind, das nach einer Vielzahl von Kriterien, teilweise unter Zuhilfenahme externer Programme, ausgewertet werden kann. Das UVC-Projekt, das zu den wenigen zählt, das sich intensiv mit den technischen Kernproblemen der Langzeitarchivierung auseinandersetzt, befindet sich noch in der Experimentierphase, sodass Aussagen über Kosten- und Zeitaufwände im Praxisbetrieb noch nicht vorliegen können.

# Abbildungsverzeichnis


1.1 Austauschbarkeit über Standardformate . . . . . 7
1.2 Probleme bei der Konversion von Texten . . . . . 8
1.3 Hardware-Museum an der Universität der Bundeswehr München . . . . . 17

2.1 Das Informationsmodell des OAIS-Referenzmodells . . . . . 27
2.2 Bestandteile eines Information Package . . . . . 28
2.3 Umgebung eines OAIS-Archivs . . . . . 29
2.4 Das Prozessmodell des OAIS-Referenzmodells . . . . . 30
2.5 Geschäftsgang einer digitalen Bibliothek mit einem DSEP . . 33
2.6 Das Prozessmodell eines DSEP . . . . . 33

3.1 Migration auf nichtdigitale Medien . . . . . 47

4.1 T-Diagramm eines Übersetzers . . . . . 62
4.2 Die Portierungsaufgabe . . . . . 62
4.3 Übersetzer mit neuem Syntheseteil . . . . . 63
4.4 Erzeugung des Cross-Compilers . . . . . 63
4.5 Erzeugung des Ziel-Compilers . . . . . 64
4.6 Emulator von H auf M . . . . . 70
4.7 Verwendung des Emulators . . . . . 70
4.8 Zweistufige Emulation . . . . . 71
4.9 Erzeugung eines Emulators . . . . . 71
4.10 Erzeugung auf einer anderen Plattform . . . . . 72
4.11 Kombinatorische Explosion der Emulatoren . . . . . 73
4.12 Verwendung einer virtuellen Maschine . . . . . 74
4.13 Beispielhafte Ausführungs- und Emulationskette . . . . . 74
4.14 Ausführungs- und Emulationskette mit virtueller Maschine . . 75
4.15 Ausführungs- und Emulationskette ohne virtuelle Maschine . 75
4.16 Emulation mit virtueller Maschine . . . . . 77
4.17 Aufbau der vollständigen Speicherkapsel . . . . . 78

5.1 Korrekturvorschriften . . . . . 86
5.2 Eingebettetes und externes Markup . . . . . 91

5.3 Markup-Ebenen . . . . . 93
5.4 Verweis auf ein Dokument . . . . . 103
5.5 Broken-Links-Problem . . . . . 103
5.6 Zweiteilung von Verweisen . . . . . 104

6.1 RDF-Modell . . . . . 126
6.2 RDF: Direkte Aussagen . . . . . 127
6.3 Indirekte Aussagen in RDF . . . . . 127
6.4 RDF-Schema für Abb. 6.2 . . . . . 128
6.5 Topic Map-Modell . . . . . 130
6.6 Topic Map-Beispiel . . . . . 131

8.1 Dublin Core als RDF-Graph . . . . . 151
8.2 RDF-Bags für mehrfach angegebene Dublin Core-Elemente . 152
8.3 Formatangabe gemäß MIME-Standard . . . . . 154
8.4 Datumsangabe gemäß W3C-DTF-Standard . . . . . 154
8.5 Sprachangabe gemäß ISO 639-2-Standard . . . . . 154
8.6 Schematische Darstellung eines VERS-Record . . . . . 156

9.1 Der Weg zum VERS-Record . . . . . 188
9.2 Zusätzliche Metadaten zu einem VERS-Record . . . . . 191
9.3 Hardware-Schaden durch Korrosion . . . . . 212
9.4 Gesamtarchitektur: Prozesse und Objekte . . . . . 214
9.5 Kontext der Medien: ein Beispiel für Redundanz und Unterstützung mehrerer Zielumgebungen . . . . . 217
9.6 Vereinfachtes Schichtenmodell zur internen Struktur von Medien . . . . . 218
9.7 Commodore 128 mit 1570 Disk Drive . . . . . 236
9.8 Exotische Disketten . . . . . 237
9.9 Catweasel-Controller . . . . . 238
9.10 Standard-PC mit unterschiedlichen Floppy-Disk-Controllern . 239
9.11 UML-Klassendiagramm des konzeptionellen Modells für den Loslösungsprozess . . . . . 244
9.12 Direkte Auswertung eines BLOB-Typs: Test der Konformität von CD-Images zu ISO 9660 und Inhalte des PVD . . . . . 246
9.13 Auswertung als CLOB gespeicherter Protokolle: Suche nach Commodore-Images . . . . . 247
9.14 Darstellung eines als CLOB gespeicherten Protokolls: Inhaltsverzeichnis einer Commodore-Diskette . . . . . 247
9.15 XML-Ausgabe mit Hinweisen zur Low- und High-Level-Formatierung auf den Medien . . . . . 248
9.16 XML-Ausgabe zur Erläuterung der Marke ID_LIB . . . . . 249

10.1 Datenorientierte Langzeitarchivierung mit UVC . . . . . 261

# Tabellenverzeichnis

3.1 Dokumentenkategorien und Standardformate nach Hendley (1998) . . . . . 43

5.1 Beispiele prominenter Varianten zur dauerhaften Identifizierbarkeit und Wiederauffindbarkeit von Dokumenten . . . . . 108

8.1 Ebenen des Benutzerzugriffs für verschiedene Dokumentenklassen und Benutzer . . . . . 182

9.1 Zuordnung der physischen zur logischen Lochung . . . . . 195
9.2 Risikoklassen mit Beispielen . . . . . 206

10.1 CD-Publikationen für das Emulationsexperiment . . . . . 254
10.2 Besonderheiten der CD-Publikationen . . . . . 256

# Literaturverzeichnis


BERGERON, B. P. (2002): Dark Ages II: When the Digital Data Die. New Jersey: Prentice Hall PTR

BRANDEL, M. (1999): 1971: IBM fashions the floppy. cnn. http://www.cnn.com/TECH/computing/9907/08/1971.idg/

BURNARD, C. LOU, SPERBERG-MCQUEEN, M. (2002): TEI Lite: An Introduction to Text Encoding for Interchange. http://www.tei-c.org/Lite/

CCSDS (1999): Reference Model for an Open Archival Information System (OAIS). Draft Recommendation for Space Data System Standards CCSDS 650.0-R-1, Red Book. http://www.ccsds.org/RP9905/RP9905.html

DACK, D. (2001): Persistence is a Virtue. Proc. 10th Information Online Conference, Sydney, Australia. http://www.nla.gov.au/nla/staffpaper/2001/dack.html

DCMI, DUBLIN CORE METADATA INITIATIVE (1999): Dublin Core Metadata Element Set, Version 1.1: Reference Description. DCMI Recommendation

DCMI, DUBLIN CORE METADATA INITIATIVE (2002): DCMI Metadata Registry. http://dublincore.org/dcregistry/index.html

DDB, DIE DEUTSCHE BIBLIOTHEK (2001): METADISS – Format des Metadatensatzes für Online-Hochschulschriften Der Deutschen Bibliothek und des Projekts Dissertation Online. Version 1.3

DÖRR, M. (1999): Langzeitarchivierung digitaler Medien, Arbeitsbericht für den Zeitraum 1.1.1999 – 30.6.1999. Techn. Bericht 554 922(3)/98 BIB 44 Musb 01-03. Bayerische Staatsbibliothek

DÖRR, M. (1999): Langzeitarchivierung digitaler Medien, Arbeitsbericht für den Zeitraum 1.7.1999 – 31.12.1999. Techn. Bericht 554 922(3)/98 BIB 44 Musb 01-03. Bayerische Staatsbibliothek

DUDEN (2000): Die deutsche Rechtschreibung Bd. 1, 22. Auflage. Mannheim Leipzig Wien Zürich: Dudenverlag

ENDRES, A., FELLNER, D. W. (2000): Digitale Bibliotheken. Heidelberg: dpunkt.verlag

FAN, C. (2002): XPERANTO: Bridging Relational Technology and XML. Techn. Bericht. IBM

GOLDFARB, C. F. (2002): XML Handbook. 4. Auflage, Prentice Hall

GREEN, A., DIONNE, J., DENNIS, M. (1999): Preserving the Whole: A Two-Track Approach to Rescuing Social Science Data and Metadata. Washington, DC: The Digital Library Federation (DLF)

HACKER, R. (2000): Bibliothekarisches Grundwissen. München: Saur

HAYNES, D., STRATFIELD, D., JOWETT, T., BLAKE, M. (1997): Responsibility for Digital Archiving and Long Term Access to Digital Data. British Library Research and Innovation Centre

HEDSTROM, M. (2000): Digital Preservation: A Time Bomb for Digital Libraries. Presentation at the Summer 2000 Conf., Stanford-California State Library Institute on 21st-Century Librarianship. http://www.uky.edu/~kiernan/DL/hedstrom.html

HENDLEY, T. (1998): Comparison of Methods and Costs of Digital Preservation. British Library Research and Innovation Report 106, CimTech Ltd., Univ. of Hertfordshire

HERBST, A. (1997): Anwendungsorientiertes DB-Archivieren: Neue Konzepte zur Archivierung in Datenbanksystemen. Berlin, Heidelberg, New York: Springer-Verlag

HJELM, J. (2001): Creating the Semantic Web with RDF. John Wiley, Chichester

ISO (1986): Standard Generalized Markup Language (SGML). ISO 8879:1986

ISO (1991): American Standard Code for Information Interchange (ASCII). ISO/IEC 646:1991

ISO (1998): Codes for the representation of names of languages – Part 2: Alpha-3 code. ISO 639-2:1998

ISO (2000): Representation of dates and times. ISO 8601:2000

ISO (2000): Topic Maps. ISO/IEC 13250:2000

ISO (2003): Open archival information system – Reference model. ISO/IEC 14721:2003

JONES, M., BEAGRIE, N. (2001): Preservation Management of Digital Materials: A Handbook. The British Library. http://www.jisc.ac.uk/dner/preservation/workbook/

KLETTKE, M., MEYER, H. (2002): XML & Datenbanken. Heidelberg: dpunkt.verlag

KOPKA, H. (2002): LaTeX I Einführung. Pearson Studium. s. auch die Deutschsprachige Anwendervereinigung TeX e.V. unter http://www.dante.de/

LAWRENCE, G. W., KEHOE, W. R., RIEGER, O. Y., WALTERS, W. H., KENNEY, A. (2000): Risk Management of Digital Information: A File Format Investigation. Washington, DC: The Digital Library Federation (DLF)

LIEVESLEY, D. (1995): Strategies for Managing Electonic Resources. JISC/British Library Workshop: Long Term Preservation of Electronic Materials. http://www.ukoln.ac.uk/

LORIE, R. A., CHHATPAR, C., GUPTA, A., HANUMOLU, V., PATEL, T. (2001): Long-Term Preservation of Audio Files. Techn. Bericht CS-5604. Virginia Polytechnic Institute and State University

LORIE, R. A. (2000): Long-Term Archiving of Digital Information. Techn. Bericht RJ-10185 (95059). San Jose, CA: IBM Almaden Research Center.
http://www.kb.nl/coop/nedlib/results/emulationpreservationreport.pdf

LORIE, R. A. (2002): A Methodology and System for Preserving Digital Data. Proc. 2nd ACM/IEEE-CS Joint Conf. on Digital Libraries, Portland, OR. New York: ACM Press, S. 312–319

LUPOVICI, C., MASANES, J. (2000): Metadata for Long-term Preservation. Techn. Bericht NEDLIB LB 5648 Issue 1.0. Bibliotheque nationale de France

MICHELSON, A., ROTHENBERG, J. (1992): Scholarly Communication and Information Technology: Exploring the Impact of Changes in the Research Process on Archives. American Archivist 55, 235–315

MIME (1996): Multipurpose Internet Mail Extensions (MIME) Part 1 (Format of Internet Message Bodies) & Part 2 (Media Types). RFC2045, RFC2046. http://www.isi.edu/in-notes/iana/assignments/media-types/media-types

NAGARAJAN, V., MCGOVERN, J., BOTHNER, P. (2003): XQuery. Sams

NELSON, T. H. (1997): Embedded Markup Considered Harmful. World Wide Web Journal 2:4, 129–134

OCLC, RLG (2001): Preservation Metadata for Digital Objects: A Review of the State of the Art. Techn. Bericht Issue 1.0. OCLC/RLG Working Group

OMG, OBJECT MANAGEMENT GROUP (2001): Unified Modeling Language (UML) Version 1.4. OMG Specification. http://www.omg.org/technology/documents/formal/uml.htm

OMG, OBJECT MANAGEMENT GROUP (2002): XML Metadata Interchange (XMI) Version 1.2. OMG Specification. http://www.omg.org/technology/documents/formal/xmi.htm

PDF, ADOBE SYSTEMS INCORPORATED (2001): PDF Reference Version 1.4. 3. Auflage, Addison-Wesley

POSTSCRIPT, ADOBE SYSTEMS INCORPORATED (1999): PostScript Language Reference. 3. Auflage, Addison-Wesley

POWERS, S. (2003): Practical RDF. O'Reilly & Associates

RECHENBERG, P., POMBERGER, G., Hrsg. (1997): Informatik-Handbuch. München Wien: Carl Hanser Verlag

RÖDIG, P., PFEIFFER, E., BORGHOFF, U. M. (2002): Langzeitarchivierung digitaler Publikationen. Techn. Bericht 2002-02. Fakultät für Informatik, Univ. der Bundeswehr München. http://www2.informatik.unibw-muenchen.de/LZA/lza_techrep.pdf

ROSENBAUM, O. (2001): Lexikon Optische Speichermedien: Namen, Normen und Begriffe. Berlin: Verlag Technik

ROTHENBERG, J., BIKSON, T. K. (1999): Carrying Authentic, Understandable and Usable Digital Records through Time. RAND-Europe. vgl. auch http://www.si.umich.edu/CAMILEON/index.htm

ROTHENBERG, J. (1995): Ensuring the Longevity of Digital Documents. Scientific American 272:1, 24–29

ROTHENBERG, J. (1999): Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation. Washington, D.C.: Council on Library and Information Resources (CLIR). http://www.clir.org/pubs/reports/rothenberg/contents.html

ROTHENBERG, J. (2000): An Experiment in Using Emulation to Preserve Digital Publications. RAND-Europe and Koninklijke Bibliotheek, The Hague. http://www.kb.nl/coop/nedlib/results/emulationpreservationreport.pdf

ROTHENBERG, J., Hrsg. (2000): Using Emulation to Preserve Digital Documents. The Hague: RAND-Europe and Koninklijke Bibliotheek. ISBN 906259145-0

SCHAARSCHMIDT, R. (2001): Archivierung in Datenbanksystemen: Konzept und Sprache. Stuttgart, Leipzig, Wiesbaden: Teubner

SCHÖNING, H. (2001): XML-Datenbanken. Datenbank-Spektrum 2001:1, 33–34

TASI (2000): Digital Preservation. ILRT. vgl. auch http://www.tasi.ac.uk/

TEI, TEXT ENCODING INITIATIVE CONSORTIUM (2002): Guidelines for Electronic Text Encoding and Interchange. University of Virginia Press, Edition P4. http://www.upress.virginia.edu/books/tei.html

TFADI (1996): Preserving Digital Information – Final Report. Task Force on Archiving of Digital Information. http://www.rlg.org/ArchTF

TOLKSDORF, R. (2002): XHTML und HTML – die Sprachen des Web. Heidelberg: dpunkt.verlag

TUROWSKI, K., FELLNER, K. J., Hrsg. (2001): XML in der betrieblichen Praxis. Heidelberg: dpunkt.verlag

UNICODE (2002): Members of the Editorial Committee, Unicode.org. Unicode Standard Annex 28 – Unicode 3.2. http://www.unicode.org/unicode/reports/tr28/

VAN DER WERF, T. (2000): The Deposit System for Electronic Publications: A Process Model. Techn. Bericht. Koninklijke Bibliotheek

VERS (1998): Victorian Electronic Records Strategy – Final Report. Public Record Office Victoria. http://www.prov.vic.gov.au/vers/published/final.htm

W3C, WORLD WIDE WEB CONSORTIUM (1998): Cascading Style Sheets, level 2. W3C Recommendation. http://www.w3.org/TR/REC-CSS2/

W3C, WORLD WIDE WEB CONSORTIUM (1999): HTML 4.01 Specification. W3C Recommendation. http://www.w3.org/TR/html401/

W3C, WORLD WIDE WEB CONSORTIUM (1999): Namespaces in XML. W3C Recommendation. http://www.w3.org/TR/REC-xml-names/

W3C, WORLD WIDE WEB CONSORTIUM (1999): Resource Description Framework (RDF) Modell and Syntax Specification. W3C Recommendation. http://www.w3.org/TR/REC-rdf-syntax/

W3C, WORLD WIDE WEB CONSORTIUM (1999): XML Path Language (XPath) Version 1.0. W3C Recommendation. http://www.w3.org/TR/xpath/

W3C, WORLD WIDE WEB CONSORTIUM (2000): Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation. http://www.w3.org/TR/REC-xml/

W3C, WORLD WIDE WEB CONSORTIUM (2000): Resource Description Framework (RDF) Schema Specification 1.0. W3C Recommendation. http://www.w3.org/TR/rdf-schema/

W3C, WORLD WIDE WEB CONSORTIUM (2001): Mathematical Markup Language (MathML) Version 2.0. W3C Recommendation. http://www.w3.org/TR/MathML2/

W3C, WORLD WIDE WEB CONSORTIUM (2001): Scalable Vector Graphics (SVG) 1.0 Specification. W3C Recommendation. http://www.w3.org/TR/SVG/

W3C, WORLD WIDE WEB CONSORTIUM (2001): XML Schema Part 0: Primer. W3C Recommendation. http://www.w3.org/TR/xmlschema-0/

W3C, WORLD WIDE WEB CONSORTIUM (2002): XHTML 1.0 – The Extensible HyperText Markup Language (Second Edition). W3C Recommendation. http://www.w3.org/TR/xhtml1/

WAGEMUTH, W. (1987): Anschluß gesucht, Floppy-Laufwerke zum Laufen gebracht – vom sogenannten Shugart-Bus und ähnlichen Standards. c't 1987:7, 169–176

WIDHALM, R., MÜCK, T. (2002): Topic Maps: Semantische Suche im Internet. Berlin Heidelberg New York: Springer-Verlag

WILDE, E., LOWE, D. (2002): XPath, XLink, XPointer, and XML. Addison Wesley Longman

XTM (2001): XML Topic Maps (XTM) Version 1.0. TopicMaps.Org Authoring Group XTM. TopicMaps.Org Specification

# Index


**A**

Advisory Group on Resource Sharing Technical Strategy, 172
American Association of Publishers (AAP), 107
American National Standard for Information Sciences (ANSI), 42, 107
American Standard for Coding and Information Interchange (ASCII), 12, 13, 16, 20, 38, 43, 48, 52, 78, 80, 95, 138, 194, 196–199, 201, 204, 206, 209, 264, 265
ANSI, *siehe* American National Standard for Information Sciences
Art and Architecture Group (AAG), 175
Art and Architecture Group Steering Committee (AAGSC), 175
Art Museum Image Consortium AMICO Library, 176
ASCII, *siehe* American Standard for Coding and Information Interchange
ASCII für die kommenden 50 Jahre, 209
Auffrischung digitaler Medien, 45
Australian Government Locator Service (AGLS), 159
Avery Index for Architectural Periodicals, 176

**B**

Base64, 161, 190
Bayerische Staatsbibliothek (BSB), 211, 237
Bibliography of the History of Art, 176
binary large object (BLOB), 225, 246
Bootstrapping, 64
Broken-Links-Problem, 103

**C**

CCITT Group 3, 199
CD-DA (Digital Audio), 221
CD-R (Recordable), 222
CD-ROM (Read Only Memory), 221
character large object (CLOB), 225, 229, 246, 250
codebooks, 193
column binary, 194–199
Commission on Preservation and Access (CPA), 174, 193
compact disc, 221
Consultative Committee for Space Data Systems (CCSDS), 11, 26
content migration, 47
ConversionsPlus 4.5, 208
Cornell Digital Library (CDL), 205, 206
Cornell University Library (CUL), 201
Corporation for National Research Initiatives (CNRI), 106
Council on Library and Information Resources (CLIR), 201
Cross-Compiler, 63

**D**

Darpa Agent Markup Language (DAML), 132
Data Documentation Initiative (DDI), 201
DataJunction 7.0, 208
Datenbank, 211
Datenbank der Metadaten, 92, 96
Datenbank-Managementsystem, *siehe* DBMS
Datenbankmigration, 96, 211

Datenbankmigrationsprojekt an der UniBwM, 232
DBMS, 211–213, 215, 216, 225–228, 230–232, 249, 250
als unterstützendes Werkzeug, 249
Ausprägungen digitaler Objekte, 215
BLOB, 225
CLOB, 225
Konsistenz der Metadaten, 213
logisches Schema, 230
Multimediadaten, 231
Persistenz, 230
physische Speicherung, 230
Relationales-, 225
Speicherstrukturen, 231
Transparenz, 231
und OAIS, 216
Zugriff auf Metadaten, 226
DCMI, *siehe* Dublin Core Metadata Initiative
Deposit System for Electronic Publications (DSEP), 32, 50, 52, 53, 76, 78, 80, 187, 251
Digital Library Federation, 176
Digital Object Identifier (DOI), 44, 107, 108
Digital Rights Management, 21
Digital Services Project, 177, 184
Digital Versatile Disc, 223
digitale Signatur, 156, 190
Disketten, 220
DoD 5015.2-STD, 139
DOI, *siehe* Digital Object Identifier
Dokumentenkontext, 75, 141
bei der Emulation, 75
Kontextwissen, 141
Domain Name System (DNS), 102
DRM, *siehe* Digital Rights Management
DSEP, *siehe auch* Deposit System for Electronic Publications
Archival Information Update, 36
Create New Reference Platform, 36
Define Reference Platforms, 36
Delivery & Capture, 34
Develop Packaging Designs, 35
Develop Preservation Standards & Strategies, 35
Monitor Technology, 36
Packaging & Delivery, 34
Preservation, 35
Preservation Activities, 35
Preservation Planning, 35
Dublin Core, 21, 143
Dublin Core Metadata Initiative (DCMI), 147, 148
Darstellung der Elemente mittels RDF und RDF-Schema, 150
kontrollierte Vokabulare, 149
Semantik, 149
Verfeinerung der Elementsemantik, 152
Verwendung beim Australian Government Locator Service (AGLS), 159
vs. TEI, 170
DVI, 21, 66

**E**

East Asian Character Code Set for Bibliographic Use (EACC), 173
East Asian Studies Community, 172
EBCDIC, 12
El Torito, 222
Emulation, 18, 59, 66
Archivierungsprozesse, 76
Begriffsfeld und Anwendungsbeispiele, 59
Chancen und Risiken, 81
Dokumentenkontext, 75
EDSAC-Emulator, 65
Emulationsalternativen, 67
Emulator als I-Diagramm, 69
Experiment an der Koninklijke Bibliotheek, 251
Experiment von Rothenberg, 251
in der Langzeitarchivierung, 66
JVM-Emulator, 65
Kriterien für die Authentizität, 255
mit Universal Virtual Computer (UVC), 260
Softwareschichten, 68
ursprünglicher Streubereich, 257

Varianten, 69
Verwendung virtueller Maschinen, 73
zweistufige, 70
Erhaltungsstrategie
Emulation, 18, 59
Hardware-Museen, 16
Migration, 15
Error Correcting Codes, 80
Eureka, 171
Examiner, 208
Exportieren in Fremdformate, 49
eXtensible Markup Language (XML), *siehe* XML
Ezra Cornell Papers, 205

**F**
flat ASCII, 201
Formatvorlagen für Konversion nach XML, 162

**G**
Geographic Information System (GIS), 41, 43, 44
Getty Information Institute, 175

**H**
Handle-System, 106
Handlungsbedarf bei Tabellenkalkulationsprogrammen, 207
Hardware-Museen, 16
Hierarchical File System, 222
Hilfsmittel zur Risikobewertung bei der Migration, 202
HTML, *siehe* HyperText Markup Language
human readability, 96
HyperText Markup Language (HTML), 9, 43, 87, 88, 97, 109, 110, 113–116, 118, 122–124, 200, 256

**I**
image file directory, 111
Importieren von Fremdformaten, 50
Inter-university Consortium for the Political and Social Research (ICPSR), 200
International Digital Object Identifier Foundation (IDF), 107
International Organization for Standardization (ISO), 26, 42, 107, 129, 149, 153, 154, 165, 221–224, 226, 235, 246
Interoperability of Data in E-Commerce Systems (INDECS), 107
ISO, *siehe* International Organization for Standardization
ISO 9660, 221–223, 246

**J**
Java Database Connectivity (JDBC), 224
Java Virtual Machine (JVM), 64
Jewish And Middle East Studies Community (JAMES), 173
Jewish Studies Community, 173
Joliet, 222
Journal Storage Project (JSTOR), 194

**K**
Kompatibilität, 54, 68, 80, 208
abwärtskompatible Produktfamilien, 49, 51
Kompressionsverfahren, 111
kontrollierte Vokabulare, 149
Konvertierungssoftware, 50
Kopierschutz, 21
bei der Migration, 45

**L**
Langzeitarchivierung
durch Emulation, 66
durch Migration, 37
kombinierter Ansatz, 142
Langzeitprogramme der RLG
Preservation (Preserv), 171
Shared Ressources (Shares), 171
LaTeX, 8, 21, 87, 88, 137, 138, 159
loop devices, 227

**M**
MARC-AMC, 172
Markup
Anforderungen an die Langzeitarchivierung, 95
Anwendungsmöglichkeiten für die Archivierung, 95

Bibliographische Anforderungen, 98
Ebenen, 93
eingebettetes, 91
externes, 91
Klassifikation, 87
prozedurales, *siehe auch* PDF; PostScript
semantisches, *siehe auch* RDF; Topic Maps
strukturelles, *siehe auch* XML; HTML; SGML
MathML, 21, 122, 143
media migration, 44
Metadaten, 3, 10, 11, 13, 18, 20, 21, 23, 25, 28, 34, 36, 40, 41, 51–55, 77–81, 85, 86, 92, 93, 95, 96, 98–100, 109, 112, 113, 124, 125, 132, 138, 143, 144, 147, 148, 155, 156, 158–161, 163, 170, 172, 175, 176, 181–183, 185, 189–191, 193, 200, 201, 203–207, 209, 213, 215, 216, 218, 225–229, 232, 233, 241, 242, 246, 249, 259–261, 263
bei Pandora, 177, 181
codebooks, 193
Datenbank, 92, 96
Doppelrolle bei der Migration, 51
INDECS, 107
Markup-Ebenen, 93
Migration, 15, 37–40, 48–50
Archivierungsprozess, 50
auf andere digitale Medien, 45
auf nichtdigitale Medien, 46
Begriffsfeld und Zielsetzung, 37
Chancen und Risiken, 55
Datenbank-, 211
Datenformate, 41
Doppelrolle von Metadaten, 51
durch Ändern der logischen Struktur, 47
durch Wechseln des Datenträgers, 44
Hilfsmittel zur Risikobewertung, 202
im engeren Sinn, 39
im weiteren Sinn, 39
in der Langzeitarchivierung, 40
Kopierschutz, 45
Microsoft-Office, 41
Migrationshistorie, 54
Patente, Lizenzen und Gebühren, 42
Portierung von Software, 60
Quantifizierbarkeit des Risikos, 209
Risiken des Konvertierungsprozesses, 204
Risikoklassen, 202
Schritte bei Preserving the Whole, 194
Verfälschungen bei fortlaufender Migration, 16, 56, 96
WAV, 41
zu ASCII, 38, 209
zu xls, 208
MIME, 149, 153, 154, 228
model requirements for the management of electronic records (MoReq), 139
Modellvorstellung eines Computers, 60
Monitor-Technologie, 226
Monographic Series Cooperative Cataloging Project, 175

**N**

NASA, 26, 136
National Archives and Records Administration (NARA), 193
National Library of Australia (NLA), *siehe* Pandora
native range of variability, *siehe* ursprünglicher Streubereich bei der Emulation
NEDLIB, *siehe* Networked European Deposit Library
Networked Computer Science Technical Reports Library, 106
Networked European Deposit Library (NEDLIB), 31, 32, 35, 36, 251, 257
noncolumn binary, 195, 198

**O**
OAIS, 11, 25–36, 50, 53, 76, 180, 181, 183, 185, 201, 216, 248
  Access, 31
  Administration, 30
  Archival Storage, 30
  Data Management, 30
  Information Packages, 27
  Ingest, 29
  Preservation Planning, 31
  Referenzmodell, 25
Object Management Group (OMG), 124
OCR, 142, 199
Ontologien: DAML+OIL, 132
Ontology Inference Layer (OIL), 132
Open Archival Information System, *siehe* OAIS
Open Archival Information System Reference Model, *siehe* OAIS
Open Archives Initiative (OAI), 25

**P**
PADI, *siehe* Preserving Access to Digital Information
Pandora, 176, 177, 180, 181, 183, 185
  allgemeines Funktionsmodell, 180
  Archivmanagement, 181
  Metadaten, 177, 181
  SCOAP-Richtlinien zur Langzeitarchivierung, 178
PDF, *siehe* Portable Document Format
Persistent Uniform Resource Locator (PURL), 44, 104, 108, 178, 181–183
Persistent URL, *siehe* Persistent Uniform Resource Locator
Persistenz, 100, 212, 213, 230
Pittsburgh Project, 148
Plattformunabhängigkeit, 95
Portable Document Format (PDF), 21, 37, 38, 41–43, 49, 88, 109, 110, 112–114, 122, 137, 147–149, 156, 157, 160, 169, 188–190, 192, 193, 199, 200, 227, 253, 265
  hybride Dokumente, 113
Portierung, 60
Posix, 222
PostScript, 11, 12, 21, 37, 43, 87, 88, 112, 113, 118, 137, 189
Preserving Access to Digital Information, x
Preserving and Accessing Networked Documentary Resources of Australia, *siehe* Pandora
Preserving the Whole, 193
Primary Volume Descriptor, 221
Protocol for Metadata Harvesting (OAI-PMH), 25
public domain Archive im Internet, 210
PURL, *siehe* Persistent Uniform Resource Locator

**R**
Raster Document Object (RDO), 205, 206
Rastergraphik, 110, 199
RDF, *siehe* Resource Description Framework
Record Export for Art and Cultural Heritage (REACH), 175
referenzielle Integrität, 226
Refreshment, 45, 53
Repackaging, 53
Replication, 45, 53
Research Libraries Group (RLG), 171–176
Research Library Information Network (RLIN), 171, 173
Resolution, 101
Resource Description Framework (RDF), 21, 90, 109, 126–129, 131, 132, 134, 138, 143, 150–152, 154, 170, 280
  RDF-Schema, 127
  Reifikation, 127
  semantische Suche, 132
  vs. Topic Maps, 131
  XML-Syntax, 128
RFC Nr. 1691, *siehe* Cornell Digital Library (CDL)
Rich Text Format (RTF), 41, 43, 49
Risiken des Konvertierungsprozesses bei der Migration, 204

Risikoklassen bei der Migration, 202
RLIN, *siehe* Research Library Information Network
Rock Ridge Interchange Protocol, 222
Roper Report, 197
RTF, *siehe* Rich Text Format

**S**
Scalable Vector Graphics (SVG), 21, 122
SCIPIO, 175, 176
SCOAP, 177, 178
  Richtlinien zur Langzeitarchivierung bei Pandora, 178
Selection Committee on Online Australian Publications, *siehe* SCOAP
Semantic Web, 134, *siehe auch* RDF; Topic Maps
semantisches Markup, 89
SGML, *siehe* Standard Generalized Markup Language
SMIL, 21
Speicherkapsel, 77
SPIRT Metadata Project, 148
spread ASCII, 194, 199
Standard Generalized Markup Language (SGML), 7, 21, 43, 49, 52, 88, 115, 116, 161, 200

**T**
T-Diagramm eines Übersetzers, 61
Tagged Image File Format (TIFF), 9, 43, 109–111, 199, 200, 205, 206, 210, 264
Task Force on Archiving of Digital Information (TFADI), 38, 39
Task Force on International Interlibrary Lending, 172
Text Encoding Initiative (TEI), 161, 163, 166, 167
  Codierungsbeschreibung, 166
  Dokumenteninhalt, 167
  Dokumentgliederung, 167
  DTDs, 161, 167
  Lite DTD, 162
  Markup-Genauigkeit, 163
  physische Umbrüche, 169
  TEI-Consortium, 170
  TEI-Dokument, 163
  vs. DCMI, 170
TFADI, *siehe* Task Force on Archiving of Digital Information
TIFF, *siehe* Tagged Image File Format
Topic Maps, 90, 109, 126, 129, 131, 143
  Assoziationen, 130
  Auftreten (occurence), 130
  semantische Suche, 132
  Thema (topic), 129
  und Ontologien, 131
  vs. RDF, 131
  XML-Syntax, 131
Transparenz bei DBMS, 231

**U**
UCS, 12
Unicode, 12, 13, 16, 20, 52, 78, 80, 95, 138, 222
Uniform Resource Identifier (URI), 101, 104, 108, 126, 127
Uniform Resource Locator (URL), 44, 101, 102, 104, 108, 181, 184, 206
Uniform Resource Name (URN), 44, 101, 102, 108
United States Department of Agriculture (USDA), 207
Universal Disk Format (UDF), 223
Universal Virtual Computer (UVC), 259–262, 264, 265, 284
Universal Virtual Computer (UVC), 261
Universität der Bundeswehr
  Datenbankmigration, 211
Universität der Bundeswehr, München (UniBwM), 211, 232, 233, 237, 249
  Datenbankmigrationsprojekt, 232
  Hardware-Museum, 16
URI, *siehe* Uniform Resource Identifier
URL, *siehe* Uniform Resource Locator
URN, *siehe* Universal Resource Name
ursprünglicher Streubereich bei der Emulation, 257

**V**
Victorian Electronic Records Strategy (VERS), 155
Migrationsaspekte, 187
virtuelle Maschinen, 73
Universal Virtual Computer (UVC), 260
Visual Resources Association (VRA), 175
VoiceXML, 122

**W**
W3C, *siehe* World Wide Web Consortium
Warwick Framework, 148
Web Service Description Language (WSDL), 124
World Wide Web Consortium (W3C), 42, 115, 122, 125, 126, 134, 153, 154, 189

**X**
Xerox Documents on Demand (XDOD), 206
XHTML, 122, 123
XML, 7, 21, 43, 49, 52, 87, 88, 90, 97, 98, 102, 109, 110, 113–125, 128, 131, 137–139, 147, 148, 150, 156–158, 161, 162, 169, 170, 188–190, 192, 193, 200, 201, 219, 224, 225, 228, 229, 247–250, 262, 263, 266
als Datenaustauschformat, 123
Darstellung, 122
Darstellung von Datenbankinhalten, 247
Document Type Definition (DTD), 116
Formatvorlagen für Konversion, 162
gültige Dokumente, 117
Namensräume, 120
Navigation mit XPath, 121
Suchen mit XQuery, 121
Transformation, 122
wohlgeformte Dokumente, 116
XML-Dialekte, 122
XML-Schema, 98, 118
zur Beschreibung von Nicht-Dokument-Daten, 123
XML Metadata Interchange (XMI), 124
XPath, 121
XQuery, 121
XSL, 21, 122, 143
XSL-FO, 122
XSLT, 122

**Z**
Zephyr, 171