Institut für Informatik

Diplomarbeit

Konzeption von Dokumentenservern für Digitale Bibliotheken im Hinblick auf Langzeitarchivierung und Retrieval

vorgelegt von

Sebastian Ohme

Erstgutachter: Prof. Dr. Andreas Schwill

Zweitgutachter: Dr. Andreas Degkwitz

Potsdam, den 01.10.2003

Inhaltsverzeichnis

1 Einleitung

„Elektronische Medien sind nicht archivierbar“ resümiert der amerikanische Astronom und Datenschutz-Spezialist Clifford Stoll in seinem Buch „Die Wüste Internet. Geisterfahrten auf der Autobahn“ [Stoll96] und verweist dabei auf Daten, die 1979 von der Raumsonde Pioneer vom Saturn übertragen und bei der NASA archiviert wurden. Obwohl die Daten auf vier verschiedenen Datenträgern gespeichert waren (7-Spur-Magnetband, 9–Spur-Magnetband, Lochstreifen und Lochkarte), sollen sie 1994 nicht mehr nutzbar gewesen sein, da zum einen keine entsprechenden Lesegeräte mehr verfügbar waren und die Daten auf 1,2 Millionen Magnetbändern außerdem nur noch schwer den vielen weiteren Weltraummissionen und Projekten zugeordnet werden konnten. Vom so genannten „NASA-Effekt“ wird gesprochen – die Bänder waren nicht oder nur notdürftig beschriftet ([SB96], siehe auch [Arch99]). Nur wenig besser erging es den Daten der US-Volkszählung von 1960, die beim US Bureau of the Census ebenfalls auf Magnetband abgelegt waren. Bei der Umstellung auf ein neues Speicherformat konnten sie zwar teilweise gerettet werden, Datenverlust war aber auch hier zu verzeichnen [Roth95A]. Diese Schwierigkeiten, um nur zwei Beispiele zu nennen, waren nicht nur auf technische, sondern auch auf organisatorische Defizite und eine Nichteinhaltung einfacher Archivierungsgrundsätze zurückzuführen.

Auch heute ist die Problematik akuter denn je: gerade im Zeitalter des für die Informationsbeschaffung und -verbreitung inzwischen unentbehrlichen Internets laufen Daten Gefahr, verloren zu gehen oder wegen des rasanten Fortschreitens der technologischen Entwicklungen schon bald nicht mehr verwendbar zu sein. Die *Halbwertzeit* von Informationen wird immer kürzer und das Verschwinden von Daten im WWW ist alarmierend [Germ00]. Die frühen Tage des Internets sind, genau wie viele alte Schriften, Filme und Photographien, bereits nicht mehr rekonstruierbar und für zukünftige Generationen somit verloren (siehe z.B. [LB98]). Ziel muß es daher sein, den Befürchtungen von Stoll entgegenzuwirken und digitale Materialien von Anfang an in einer Form zu konservieren, die ein Auffinden und Weiterverwerten auch in mehreren Jahrzehnten möglich macht. Entsprechende Bemühungen, wie sie u.a. in [Asch01] zusammengefaßt sind, zeigen, daß die Gesellschaft aus der Vergangenheit gelernt hat und sich der Wichtigkeit der Wahrung des kulturellen Erbes durchaus bewußt ist, daß es wegen der gigantischen Datenmengen aber auch viele finanzielle, organisatorische und vor allem technische Probleme gibt.

Probleme, die auch – und gerade – vor Bibliotheken nicht halt machen: die Anzahl elektronischer Publikationen ist in den letzten Jahren stark gestiegen – und ebenso stark mußten und müssen sich viele Arbeitsabläufe im bibliothekarischen Alltag verändern. Noch orientiert sich ein Großteil der wissenschaftlichen Bibliotheken in Deutschland an ihrer klassischen Aufgabe der Erwerbung, Bestandsvermehrung und -wahrung analoger Medien. Mit einem Fortschreiten der Digitalisierung und Vernetzung erscheint diese eher konservative Haltung aber nicht mehr tragbar: insbesondere Universitätsbibliotheken sehen sich in zunehmendem Maße mit der Aufgabe konfrontiert, elektronisch vorliegende Publikationen entgegenzunehmen, zu erschließen, zu archivieren und verfügbar zu machen. Mehr und mehr werden lokale Dokumentenserver in Form von „Institutional Repositories“ eingesetzt, die verschiedenartigste Materialien, wie Bilder, Vorlesungsskripte, Zeitschriften, Konferenzbände, Preprints u.ä. aufnehmen können.

Thomas Hilberers Aussage von 2001 ist zwar ernüchternd:

> Die „Publikationsserver deutscher Hochschulen oder deren Bibliotheken [...] stellen sich als Gemischtwarenläden ohne Profil, ohne Programm und ohne inhaltliche Schwerpunkte dar [...]. Zusammengefasst: der Mangel an Profil und das niedrige durchschnittliche Qualitätsniveau machen die bestehenden Veröffentlichungsunternehmen deutscher Hochschulbibliotheken für Wissenschaftler unattraktiv."
> [Hilb01]

... ob dem aber noch immer so ist, gilt es herauszufinden – und gegebenenfalls, soweit in diesem Rahmen möglich, für die UB Potsdam zu verändern.

1.1 Zielstellung

Die vorliegende Diplomarbeit soll die bibliothekarischen Bedürfnisse aufzeigen und darlegen, inwieweit Informatik-Kompetenz für den Aufbau und die technische Administration der angesprochenen Institutional Repositories notwendig ist. Es werden Projekte und Lösungsansätze beleuchtet, die es ermöglichen, dem Wunsch der dauerhaften Archivierung näher zu kommen. Ziel ist ein Dokumentenserver, der durch Verwendung freier, möglichst herstellerunabhängiger Software und durch den Einsatz standardisierter Protokolle und Formate eine hoffentlich langlebige Plattform für elektronisches Material jeglicher Art bietet. Mit der Schaffung eines (prototypischen) Archivs zur Sammlung wissenschaftlicher Dokumente soll selbst ein kleiner Beitrag zum Fortbestand wichtiger Erkenntnisse geleistet werden. Da eine allumfassende Lösung in diesem Rahmen natürlich unmöglich ist, werden die Anforderungen zunächst ganz allgemein für Textdokumente ausgelotet und die gewünschten Funktionalitäten ab Kapitel 5 dann konkret für elektronische Dissertationen zusammengetragen. Sich auf diesen, im universitären Umfeld besonders häufig anzutreffenden Dokumenttyp zu konzentrieren, erlaubt dabei nicht nur, den Umfang der Ausarbeitung angemessen zu beschränken, sondern ermöglicht es den Bibliotheken auch, in einem vorgegebenen Rahmen noch mehr Erfahrungen mit dem digitalen Medium zu sammeln. Mit Dissertationen ist man einfach bestens vertraut – sie stehen „[...] überall in ausreichendem Umfang zur Verfügung und ‚wachsen', je nach Größe der Universität auch in großen Mengen, automatisch nach."[1] Außerdem liegen sie bereits in digitalisierter Form vor, bevor sie gedruckt werden.

> „Es liegt [daher] nahe, sie über das Internet der Wissenschaft zugänglich zu machen. Da es sich bei solchen Arbeiten nicht selten um Forschungen handelt, die den *state of the art* zu einem bestimmten Thema beschreiben [...], neue methodische Wege aufzeigen, aktuelle Forschungsergebnisse dokumentieren [...], würde ein erheblicher wissenschaftlicher Mehrwert erzeugt, wenn sie leicht auffindbar und sofort zugänglich wären"

… so [Diep01] zum Hauptvorteil elektronischer Dissertationen, der gleichzeitig auch Hauptgrund für die Spezialisierung des Dokumentenservers auf diese Textart ist.
Langfristig gesehen gilt es allerdings, auch Zeitschriftenartikel, Preprints, Lehrmaterialien, Studien u.ä. in das geplante Repository aufzunehmen und so ist bei der Konzeption des Systems in besonderem Maße darauf zu achten, die bei der Bereitstellung der Doktorarbeiten gewonnenen Erkenntnisse möglichst problemlos auch auf andere wichtige Dokumente übertragen zu können. Die archivierten Publikationen müssen außerdem in geeigneter Weise zugriffsfähig und recherchierbar gemacht werden. Eine einfache Auflistung ist zwar wünschenswert, reicht gerade im Kontext einer Digitalen Bibliothek aber bei weitem nicht aus und so muß eine Suchmaschine ein Retrieval im Volltext und in strukturellen Daten, wie Titel oder Verfasser, ermöglichen.

[1] siehe http://www.dissonline.de/texte_html/aktuellesitua.html

2 Begriffsbestimmung

Die Möglichkeiten des Publizierens im World Wide Web bergen vielfältige Veränderungen für den wissenschaftlichen Kommunikationsprozeß in sich und stellen Bibliotheken vor völlig neue technische und organisatorische Herausforderungen. Geprägt von Digitalisierung und Vernetzung finden sich Schlagwörter, wie Multimedia, Internet oder E-Publishing, im bibliothekarischen Alltag wieder – und auch von sogenannten Digital Libraries ist die Rede. Aus analogen Medien werden *Elektronische Publikationen*, aus klassischen Archiven werden *Digitale Bibliotheken*. Doch was bedeuten diese Begriffe eigentlich und welche Vorteile (und neuartigen Probleme) bringen sie mit sich?
Die nachfolgenden Abschnitte versuchen, diesbezüglich einige Definitionen und Hintergrundinformationen zu liefern, können aber leider nur einen kleinen Einblick in die Thematik geben, da diese überaus komplex und umfangreich ist.

2.1 Elektronische Publikationen

Der Begriff „Elektronisches Publizieren“, laut [Kist88] erstmalig bereits im Jahre 1977 verwendet, läßt sich trotz seines langjährigen Bestehens noch immer nicht eindeutig und in jedem Umfeld gleichermaßen definieren – zu unterschiedlich ist die Auffassung über die Intensität des Einsatzes von Computern und entsprechend viele Definitionsversuche, Begriffsvarianten und vermeintliche Synonyme gibt es. In [Riehm92] wird das „Ideal“ des Elektronischen Publizierens als eine elektronisch integrierte Publikationskette verstanden, in der alle arbeitsteilig vollzogenen Stadien des Publizierens – vom Autor bis zum Nutzer – mit Unterstützung von Informations- und Kommunikationstechniken durchlaufen werden (siehe Abbildung 1).

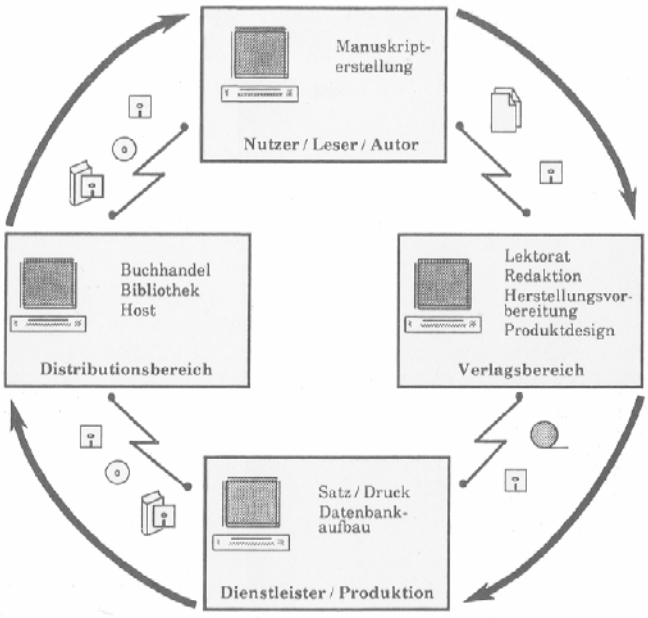


Abbildung 1: Publikationskette

Es wird allerdings auch angemerkt, daß dieses Ideal der Realität wenig nahe kommt; daß man auf eine gedruckte Publikation bisher meist nicht verzichtet und eher auf Parallelpublikationen und die Doppelexistenz von Informationen setzt. Laut [RBW88] lassen sich drei hauptsächliche Varianten des Begriffs unterscheiden:

1. Elektronisches Publizieren wird produktionsorientiert verstanden, d.h. in erster Linie als Unterstützung bei der computergestützten Herstellung von (herkömmlichen) Publikationen.
2. Elektronisches Publizieren wird distributionsorientiert verstanden, d.h. es wird vorrangig auf die über elektronische Medien vermittelte Verteilung (elektronischer) Publikationen bezogen.
3. Elektronisches Publizieren bezeichnet neuartige, innovative, elektronische Präsentations- und Publikationsformen, die mit herkömmlichen Mitteln nicht erreichbar sind (z.B. Integration von Text, Grafik, Animation und Ton in einem Dokument).

Eine ausführliche Auseinandersetzung mit verschiedenen Electronic Publishing-Begriffen und anderen, in ähnlichen Kontexten verwendeten *buzz-words*, wie **C**omputer **A**ided **P**ublishing, **D**esk**T**op **P**ublishing oder **E**lectronical **T**echnical **P**ublishing, findet sich in [RBW86].

Im Folgenden soll elektronisches Publizieren allgemein als „die Herstellung, Vervielfältigung und Verbreitung von geistigen Erzeugnissen (Mehrwertdiensten) mit Hilfe elektronischer Technologien bzw. Medien“ verstanden werden, wie es auch in [Ball00] definiert ist.
Diese vorrangig computergestützte Schaffung und vor allem auch Bereitstellung von Dokumenten gehört inzwischen zum Alltag. Die einstmals diskutierte Frage, ob ein Wechsel von gedruckten zu digitalen Publikationen überhaupt notwendig und sinnvoll sei, ist kaum noch relevant.[2] Wissenschaft muß publizieren und entscheidend für den Erfolg ist die möglichst umgehende (und daher immer häufiger elektronische) Veröffentlichung der Ergebnisse. Gerade der langwierige, oft monatelange Begutachtungsprozeß („peer-review“), den z.B. Zeitschriftenartikel vor Aufnahme in ein Fachblatt zumeist über sich ergehen lassen müssen, ist diesbezüglich eher hinderlich – und entsprechend dankbar nehmen die Autoren die Preprint-Server auf, die Anfang der neunziger Jahre aufkamen und durch die Ablage von „Vorab-Versionen“ der Aufsätze eine hohe Aktualität und sofortige Verfügbarkeit neuer Forschungsergebnisse gewährleisten. Auch Diskussionsforen, Mailinglisten und weitere Dienste des Internets, wie z.B. FTP und Gopher, ermöglichen den schnellen Informationsaustausch und werden daher rege genutzt.
Die Produzenten und Konsumenten wissenschaftlicher Literatur stehen dem digitalen Medium also durchaus wohlwollend gegenüber, zumal computergestützt erzeugte Dokumente eine Recherche im Volltext erlauben und zudem durch eine Integration z.B. von Formeln, 3D-Modellen oder Videosequenzen völlig neue inhaltliche und gestalterische Möglichkeiten bieten. Nur die Verlage, die unter anderem ihre Einnahmenquellen gefährdet sehen (Stichwort *Zeitschriftenkrise*, siehe z.B. [Kell01]) und die traditionellen Bibliotheken, die sich plötzlich mit Lizensierungsfragen und neuartigen Problemen auseinandersetzen müssen, können sich für Online-Publikationen noch nicht so recht erwärmen. Die Entwicklung wird sich aber unausweichlich auf ein Ablösen des Print-Mediums hin bewegen, wie auch von [Kell02] vermutet wird.
Arnoud DeKemp vom wissenschaftlichen Springer Verlag meinte vor einigen Jahren:

> „[...] gerade dort [im Wissenschaftsbereich] entsteht im Moment eine viel größere Bedrohung für Bibliotheken und Verlage, nämlich die Abschaffung gedruckter Publikationen durch die Nutzung digitaler Bibliotheken“ (zit. n. [Rein99]).

[2] Die Vor- u. Nachteile von Print und Online werden bereits seit geraumer Zeit ausführlich diskutiert; als Einstieg siehe z.B. [Odly95].

Aber warum diese ‚Gefahr' nicht eher als Chance sehen?

> → „Es ist unwahrscheinlich, daß die Wissenschaft mit dem gegenwärtigen System weiterarbeiten will, einem System, in dem Bibliothekare Bücher und Zeitschriften kaufen und ablegen, die vielleicht nie mehr gelesen werden. Zusätzlich werden die Zeitschriften dann gebunden und/oder mikroverfilmt, was weiteres Geld verschlingt. Dann wird zusätzlich Geld für die Lagerung der Bücher und Zeitschriften ausgegeben, bis in gar nicht so ferner Zukunft, das Papier von Säure zerfressen wird, und die Bücher und Zeitschriften dann für teures Geld konserviert werden müssen. In einer Zeit, in der die Budgets von Hochschulen zusammengestrichen werden, scheint es nur logisch zu sein, sich nach einem kostensparenderen System umzusehen",

so das durchaus einleuchtende Fazit von Robin P. Peek (zit. n. [Keitz97]).

2.2 Digitale Bibliotheken

Sammlung und Verfügbarmachung von Publikationen ist seit jeher die Aufgabe von Bibliotheken. Die weitreichenden Veränderungen der letzten Jahre durch die Kommunikationstechnologien haben deren Aufgabenspektrum allerdings verändert, welches nun über den klassischen Erwerb und Erhalt von Dokumenten hinausgeht. Sie müssen sich den neuen Bedingungen – und zunehmend auch unangenehmen Fragen ihre Zukunft betreffend stellen. Vor welchen Strukturveränderungen steht die wissenschaftliche Bibliothek heute; wie sehr muß sie ihre Dienstleistungen an die neuen Bedürfnisse ihrer Kunden anpassen? Und wie kann sie ihrer angestammten Rolle überhaupt noch gerecht werden? Sogar von einem Paradigmenwechsel ist die Rede (Einzelheiten siehe z.B. [Ball00]). Antworten finden sich, sofern dies in umfassender Weise überhaupt möglich ist, in [Zimm02] – hier sollen hingegen technische Aspekte elektronischer Archive im Vordergrund stehen und nicht die Probleme, sondern eher die Möglichkeiten aufgezeigt werden, die Digitalisierung und Vernetzung von Informationen bieten.

Bereits sehr früh wurde von [Kuhl95] auf die Bedeutung elektronischer Medien im Umfeld von Bibliotheken hingewiesen:

> „Aufgabe der Bibliothekare und Informationsspezialisten wird es zunehmend sein, das lokale Angebot in entsprechenden Servern auf informationsmethodisch kontrollierte und komfortable Weise darzustellen und es auf ebenfalls kontrollierte Weise mit der Außenwelt zu verbinden. Der Informationsprofession kommt die immer wichtigere Aufgabe zu, den Zugriff zu der unüberschaubaren Fülle an Informationen dadurch offenzuhalten, daß sie strukturiert und suchbar/navigierbar wird."

An dieser Erkenntnis hat sich bis heute nichts geändert, im Gegenteil: die Bedeutung solcher *Digitaler Bibliotheken* hat mit der breiten Verfügbarkeit des Internets weiter zugenommen und obwohl das Geld knapp und die zugrundeliegende Technik noch recht teuer ist, werden sie, wie auch von [Reil02] vermutet, schon bald einen noch größeren Stellenwert einnehmen. Zum einen ist bereits ein Preisverfall zu verzeichnen (vergl. [Arm00]), zum anderen bieten elektronische Archive im Gegensatz zu klassischen Bibliotheken den Vorteil, ein Dokument jederzeit und quasi in unbegrenzter Anzahl bereitzustellen.

Der Übergang zu einer papierlosen Informationsgesellschaft wird sich dabei sukzessive, aber wahrscheinlich nur teilweise vollziehen. Zwar steigt die Anzahl elektronisch verfügbarer

Dokumente fast exponentiell, dennoch existieren bisher keine befriedigenden, technisch ausgereiften Lösungen, die die Vorteile gedruckter Publikationen – Augenfreundlichkeit, Lesbarkeit, Benutzbarkeit, Portabilität – aufwiegen können und so ist eine radikale Abkehr von den Druckmedien derzeit noch undenkbar (vergl. auch [Zimm02]). Es gilt vielmehr, die neuen multimedialen Möglichkeiten digitaler Daten in geeigneter Weise mit den klassischen Publikationsformen zu verbinden und so eine gemeinsame, sich gegenseitig ergänzende Existenz zu ermöglichen. Nicht „Digital Library", sondern besser „Hybrid Library" sollte der hierbei an die Bibliothek anzulegende Begriff sein – neben den traditionellen Zuständigkeiten müssen diese nun auch die Verantwortung für die neuen Medien übernehmen. Doch was bedeutet Digitale Bibliothek nun eigentlich und wie ist sie insbesondere im Kontext des geplanten Dokumentenservers zu definieren?

Von der Antike bis zum 20. Jahrhundert war die Schriftlichkeit, Beständigkeit und vor allem die räumlich gegebene Ordnung bei der Aufbewahrung von Wissen charakteristisch für Bibliotheken, unabhängig davon, ob die Trägermaterialien Tontafeln, Papyrus oder Papier waren. Diese Bindung an das Material und der räumliche Bezug befinden sich mehr und mehr in der Auflösung [Bilo00] – die klassische, lokationsorientierte Definition der „Bibliothek" als Ort zur Sammlung von Büchern reicht heute nicht mehr aus, will sie mit den technischen Neuerungen und den sich damit verändernden Bedürfnissen ihrer Nutzer schritthalten. Bereits seit den 80er Jahren befinden sich neben den Printmedien auch digitale Materialen, wie Disketten, Magnetbänder oder CD-ROMs, im Bibliotheksbestand und werden durch elektronische Kataloge erschlossen. Ergänzend wurden aber auch externe bibliographische Datenbanken in das Online-Angebot aufgenommen und so hat sich der Begriff der „virtuellen Bibliothek" entwickelt, welcher genau diese Spaltung zwischen den Angeboten, die im Hause vorhanden sind und den zusätzlich eingeworbenen Zugriffsmöglichkeiten auf entfernte Angebote ausdrückt (vergl. [Rusch99]). In der Literatur existieren weitere Bezeichnungen und, ähnlich wie beim *Elektronischen Publizieren*, verschiedenartige Definitionen für eine „Digitale Bibliothek".
Während der Koordinator der „Düsseldorfer virtuellen Bibliothek", Thomas Hilberer noch unterscheidet:

> „*Digitale Bibliotheken* sind Sammlungen elektronischer (=digitaler) Informationen, die sich im Besitz und damit unter Kontrolle der betreffenden ‚realen Bibliothek' befinden;
> *Virtuelle Bibliotheken* sind Sammlungen von Verweisungen (Link-Sammlungen) auf Informationen, die sich aber als solche nicht im Besitz der betreffenden ‚realen Bibliotheken' befinden bzw. befinden müssen." [Hilb95]

und Claudia Lux eine vierstufige Entwicklung „von der traditionellen über die automatisierte und die elektronische Bibliothek hin zur virtuellen Bibliothek" beschreibt [Lux95], werden einige dieser Begriffe oft auch völlig synonym verwendet. Immer aber wird hervorgehoben, daß die physische Präsenz des digitalen Materials für den Benutzer keine Rolle mehr spielt – eine *Virtual Library* beispielsweise ist (ebenfalls nach [Lux95]) „an jedem Ort zu jeder Zeit für jedermann erreichbar. [...] Sie ist örtlich ungebunden, sie wird als Bibliothek ohne Mauern, ohne Wände bezeichnet".

Im Rahmen dieser Diplomarbeit und in Anlehnung an [EU99] soll eine *Digitale Bibliothek* als Oberbegriff für Reale, Virtuelle und Elektronische Bibliothek verstanden werden: sie ist durch die wesentliche Erweiterung um binäre Informationen gekennzeichnet und verfügt als viergegliederte Bibliothek neben den traditionellen Teilen *Verwaltung*, *Benutzung* und *Magazin* zusätzlich über eine *virtuelle Komponente*. Die *Digital Library* entwickelt sich damit

immer mehr in Richtung einer „verteilten Bibliothek“, beinhaltet wie eingangs angedeutet aber auch weiterhin gedruckte Bücher bzw. andere analog gespeicherte und publizierte Dokumente. Sie grenzt sich somit von einer reinen *Virtual Library* ab, welche sich von der klassischen Bibliothek bereits vollständig getrennt hat und sozusagen nur noch auf der elektronischen Ebene existiert.

Dieses Verständnis von einer Digitalen Bibliothek als notwendige Erweiterung der traditionellen Institution um die Erschließung und Verfügbarmachung neuer Medien, wie Bilder, Videos oder eben elektronischer Volltexte, hat also weitreichende Konsequenzen für die Bibliothekare – und das technische Personal, welches durch Bereitstellung entsprechender Dokumentenserver und Zugriffsfunktionalitäten die Grundlagen für die Wissensarchivierung schafft. [Bilo00] betont in seinen Ausführungen, daß der permanente Veränderungsprozeß durch die Verschiebungen von Arbeitsschwerpunkten aller Mitarbeiter zugunsten der elektronischen Publikationen mehr und mehr akzeptiert werden muß; trotz der damit verbundenen Verunsicherungen und der Notwendigkeit einer Prioritätensetzung zu Lasten der (gleichwohl weiterlaufenden) Bearbeitung konventioneller Literatur.
Die wissenschaftlichen Bibliotheken befinden sich in einer Phase des Umbruchs und es gehört mehr denn je zu ihrem Aufgabenspektrum, eine angemessene Informationsversorgung zu gewährleisten. In einem Papier des Wissenschaftsrates wird jedoch festgestellt:

> „Die Bibliotheken sind gegenwärtig aus vielen Gründen nur eingeschränkt in der Lage, sich diesen Umstrukturierungen zu stellen.“ [WR01]

Alleine können sie, meist mangels entsprechend geschulten, informatik-versierten Personals, den neuen technischen Anforderungen nicht oder nur unzureichend gerecht werden und so wird inzwischen auch von der DFG[3] ein „integriertes Informationsmanagement an Hochschulen durch neuartige Organisationsmodelle im Verbund von Rechenzentrum, Bibliothek, Medienzentrum sowie den Informationseinrichtungen der Fachbereiche bzw. Institute“ gefordert. [DFG02]

[3] Deutsche Forschungsgemeinschaft

3 Dokumente und Formate

Während auf den letzten Seiten noch recht allgemein auf Digitale Bibliotheken und die von ihnen bereitgestellten elektronischen Publikationen eingegangen wurde, sollen im folgenden konkrete Dokumentformate und deren spezifische Eigenschaften im Vordergrund stehen. Auch hierbei ist es unumgänglich, zunächst grundsätzliche Definitionen anzugeben und die für den späteren Einsatz relevanten Anforderungen zusammenzutragen.
Der Begriff „Dokument“ ist so alt wie die Entstehung der Schriftzeichen selbst. Das Erscheinungsbild der Dokumente hat sich im Laufe der Zeit allerdings entscheidend verändert. Bestanden die ersten Schriftstücke lediglich aus Text, eventuell durchsetzt mit Bildern, sind heutzutage „Multimediale Dokumente“ möglich – bestehend aus den verschiedenartigsten Objekten und Darstellungsformen. [Stoja00] beschreibt ein Dokument daher als einen

> „[...] physisch existenten Informationscontainer (z.B. als Papier oder Datei in einem Computer). Ein Dokument muß als Einheit speicherbar und versendbar sein und als solche auch aufgefunden, wahrgenommen (gesehen, gelesen, gehört) und verwendet werden können. Ein Dokument kann Informationen beliebiger Darstellungsform enthalten; bei Kombination von Texten, Daten, Grafik, Bild und Ton spricht man von ‚Multimedialen Dokumenten‘.“

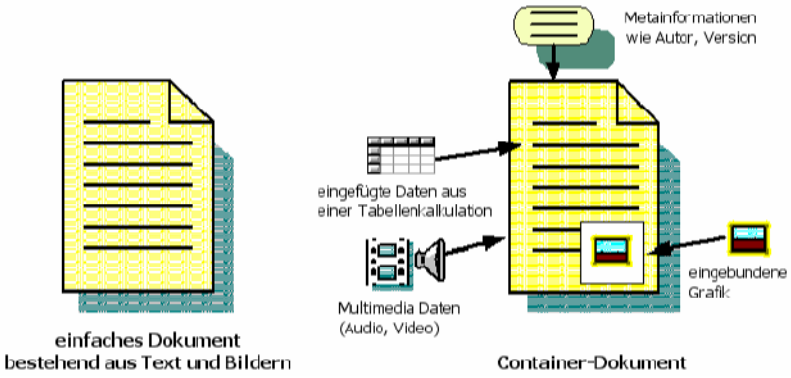


Abbildung 2: einfache und multimediale Dokumente

Die obige Definition beschränkt sich nur auf den Inhalt – jedes Dokument besitzt im allgemeinen aber auch eine spezifische hierarchische Struktur: einzelne Informationseinheiten (z.B. Passagen) werden zu einer höheren Instanz (Abschnitt, Kapitel) zusammengefaßt und meist mit einer Überschrift versehen, um sie zu klassifizieren und später besser wiederfinden zu können. Das Inhaltsverzeichnis beschreibt dabei die wichtigsten Teilstücke und ist somit eine einfache Möglichkeit zur Darstellung der logischen Struktur eines Dokuments. Weitere Architektureigenschaften von Dokumenten sind in [Blank98] zusammengefaßt. Dort heißt es u.a.:

> „Die bekannteste Norm für (Büro-)Dokumente ist die ODA-Norm[4]. Sie ‘... gliedert die Architektur auszutauschender Dokumente nach:
> - Logischer Struktur (hierarchische Einteilung des Textes, z.B. bei Briefen in Kopffeld, Textfeld und Schlußfeld) und
> - Layoutstruktur (Anordnung von Texten bzw. Bildern auf Papier oder am Bildschirm).‘ “

[4] Open Document Architecture

Bei der Auseinandersetzung mit Standards für das elektronische Publizieren nimmt die Diskussion zu Dokumentformaten eine zentrale Rolle ein. Das oben beschriebene Datenmodell, welches von einer Einheit der drei Bestandteile „Inhalt“, „Struktur“ und „Layout“ ausgeht, hat sich dabei für die formale Beschreibung als praktikabel erwiesen und so wird in [DINI01] noch einmal deutlich herausgestellt:

> „Der Begriff des Dokumentformats bezieht sich [...] nicht allein auf die Festlegung von Speicherformaten für Dateien, insbesondere für Texte, sondern auf ein Modell, das es erlaubt, verschiedene Medien zu bedienen und zu integrieren. Das Dokumentformat bildet den Rahmen, um die Zusammengehörigkeit und das Zusammenspiel sowie die Abfolge von Informationen für die Elemente eines Dokumentes wie Text, Bild, Ton, Video, Animation, Datentabellen zu erfassen.“

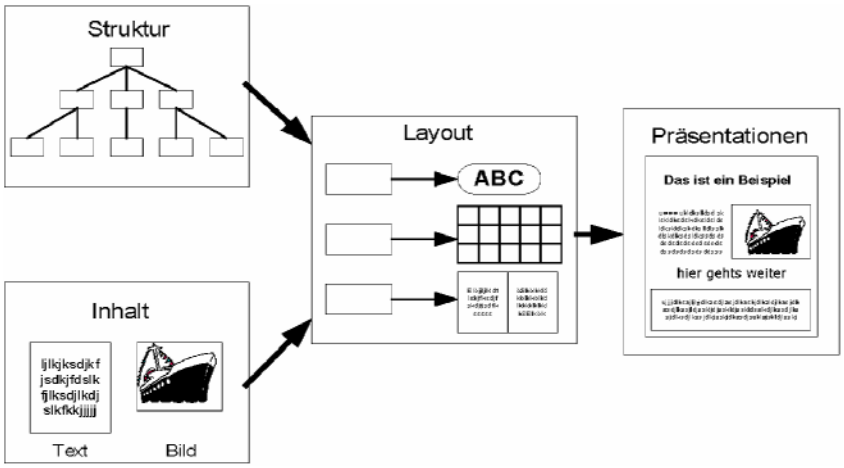


Abbildung 3: Trennung von Struktur und Inhalt

Wird der Dokumentinhalt von der Struktur getrennt aufbewahrt (z.B. durch Speicherung der strukturellen Daten in einer zusätzlichen Datei), kann bei Festlegung des verwendeten Strukturschemas und des benutzten Vokabulars ein einfacher, flexibler Austausch auch komplexer Dokumente zwischen heterogenen Systemen gewährleistet werden. Leider wurde in der Vergangenheit – ganz im Gegenteil zur grafischen Aufbereitung – kaum Wert auf die strukturelle Beschreibung von Dokumenten gelegt, was die automatische Weiterverarbeitung unnötig erschwert. Dazu aber später mehr.

Zunächst gilt es, die unterschiedlichen Anforderungen, die Produzenten und Konsumenten wissenschaftlicher Literatur an ein Dokument stellen, näher zu beleuchten. Das Augenmerk eines Autors richtet sich beispielsweise besonders auf die einfache und effiziente Erstellung von Dokumenten. Er möchte dabei nach Möglichkeit sein präferiertes Textverarbeitungssystem benutzen und auf keine multimedialen oder fachspezifischen Bestandteile verzichten. Die Bibliothek wiederum dürfte sich schwer damit tun, eine Publikation in einem veralteten Format, wie MS Works 1.0, oder bestehend aus aufwendigen multimedialen Erweiterungen, wie z.B. Videos, zu erschließen, zu archivieren und ihren Nutzern langfristig zugänglich zu machen. Rechentechnische Anforderungen sind somit ebenfalls zu berücksichtigen: die Dokumente sollten konvertierbar und möglichst einfach über das Internet austauschbar sein. Außerdem muß für deren Bearbeitung und Darstellung entsprechende Software auf möglichst vielen Rechnerplattformen verfügbar sein, um eine problemlose Nutzung zu gewährleisten. Die Konsumenten letztlich erhoffen sich umfassende Recherche- und Navigationsmöglichkeiten und vertrauen auf Integrität und Aktualität der Dokumente.

All diese Wünsche durch Wahl *eines* „Standardformats“ gleichermaßen zu befriedigen, ist derzeit allerdings unmöglich – zu unterschiedlich sind die Forderungen aller Beteiligten an die von den Speicherformaten gebotenen Funktionalitäten, wie u.a. auch von [Degen97] festgestellt wurde. Für die Erzeugung, Präsentation und Archivierung von elektronischen Publikationen steht aufgrund einer heterogenen Systemwelt mit vielen Anwendungsprogrammen und Versionen zwar eine Vielzahl verschiedener Formate zur Verfügung. Die wenigsten dieser Dateiformate sind aber für eine rechnerunabhängige Präsentation, für umfangreiche Recherchen oder für eine langfristige Aufbewahrung geeignet.

3.1 Anforderungen an Dateiformate

Die Frage nach dem idealen Format für digitale Dokumente wurde bereits vor Jahren ausgiebig diskutiert (vergl. z.B. [Schmitt96] und [Schirm98]), ist aber noch immer nicht endgültig geklärt und so ist es wichtig, die einzelnen Erkenntnisse nochmals zu bündeln, um Empfehlungen für am ehesten geeignete Dateiformate geben zu können. Ergebnis soll eine Präferenzliste sein, die Autor, Nutzer und Bibliothek gleichermaßen beim Auswahlprozeß unterstützt. Da Speicherformate im Kontext Digitaler Bibliotheken und bereitstellender Dokumentenserver eine zentrale Rolle einnehmen, müssen entsprechende Überlegungen über eine einfache Auflistung existierender Standards hinausgehen: es gilt möglichst umfassend zu klären, *warum* einige Formate für einen bestimmten Zweck besser geeignet sind, als andere.
Die im Rahmen des Projekts „Elektronisches Publizieren von Dissertationen“ der HU Berlin entstandene und oft zitierte Studienarbeit von Daniel Ohst [Ohst98] gibt dabei einen guten Überblick über die Anforderungen an die Dateiformate aus bibliothekarischer und rechentechnischer Sicht und soll aus oben genanntem Grund nachfolgend kurz zusammengefaßt werden.[5]

3.1.1 Verfügbarkeit

Unter Verfügbarkeit soll die für die Erstellung, Veränderung und Präsentation von Dokumenten notwendige Software verstanden werden. Funktionsumfang und ‚Usebility‘ können dabei, ebenso wie die jeweiligen Ansprüche der verschiedenen Autoren, erheblich variieren und so ist es schwierig, die Anzahl der benutzten Text- und Satzsysteme zu beschränken, zumal diese meist auch nicht unter allen Betriebssystemen gleichermaßen lauffähig sind. Aus rechentechnischer Sicht wäre aber gerade diese Limitierung hilfreich, da die Anwendungen gewartet werden müssen und eine kompetente Betreuung für eine Vielzahl unterschiedlicher Programme personell nur schwer zu realisieren ist. Die Kosten sind ein weiteres Kriterium für die Auswahl geeigneter Software: während die Tools und Plugins zur Betrachtung der Dokumente meist frei und oft auch für alle gängigen Plattformen, wie Windows-PCs, Apple-Rechner und Unix/Linux-Derivate, verfügbar sind, müssen Autoren für professionelle, aber überwiegend plattformabhängige Office-Pakete und Editoren nicht selten drei bis vierstellige Eurobeträge zahlen – in Bibliotheken und Rechenzentren können die Lizenzkosten so schnell ungeahnte Größenordnungen erreichen.

[5] Obwohl hier des öfteren speziell von Dissertationen die Rede ist, lassen sich die Anforderungen natürlich auch auf andere textuelle Dokumenttypen übertragen – im Kapitel 5 wird dann nochmals genauer auf die Vorteile elektronischer Hochschulschriften eingegangen.

3.1.2 Strukturierbarkeit

Von einem strukturierten Dokument spricht man, wenn die einzelnen logischen Bestandteile als solche gekennzeichnet sind und für eine spätere Auswertung, z.B. in Form einer Recherche oder für notwendige Konvertierungen, zur Verfügung stehen. Oft ist es sinnvoll, Texte nicht nur in ihrer Gesamtheit nach bestimmten Stichwörtern zu durchmustern, sondern direkt z.B. in den Überschriften oder Zitaten zu suchen. Einfache Layoutinformationen, wie „Überschriften sind fett“ oder „Zitate sind kursiv und eingerückt“, reichen dafür nicht aus – die Strukturelemente müssen explizit ausgezeichnet werden, um die Präzision von Suchanfragen zu erhöhen und die Treffermenge überschaubar zu halten. Formatvorlagen verbinden die Dokumentbestandteile und weisen einem strukturellen Element ein spezifisches Layout zu, was eine Aufbereitung der Publikation für verschiedene Ausgabemedien ermöglicht. Für den Autor kann eine detaillierte Strukturierung allerdings einen erheblichen Mehraufwand bedeuten, da er sich über den genauen Aufbau seiner Arbeit Gedanken machen und die einzelnen Bestandteile entsprechend kennzeichnen muß. Hier ist leistungsfähige Software gefragt, die diesen Prozeß umfassend unterstützt.

3.1.3 Konvertierbarkeit und Austauschbarkeit

Wie bereits erwähnt ist es schwierig und sicherlich auch nicht immer praktikabel, sich auf ein oder zwei Textverarbeitungssysteme mit einem bestimmten Dateiformat zu beschränken – zu unterschiedlich sind die Vorlieben der einzelnen Autoren – und zu ähnlich die Interessen der Anbieter kommerzieller Software. Entsprechend viele verschiedene Formate gibt es; bedingt durch die interne Struktur sind diese aber für die Archivierung oder z.B. für Recherchezwecke nicht gleichermaßen gut geeignet und so müssen Konvertierungsmöglichkeiten zur Verfügung gestellt werden, um ein vom Autor in dessen präferierten System erzeugtes Dokument möglichst verlustfrei in ein anderes Dateiformat zu überführen. Ist das Ausgangsdokument dabei gut strukturiert, ergeben sich bei einer eventuell notwendigen Konvertierung weniger Probleme, da diese dann weitestgehend automatisch und ohne vorherige oder nachträgliche Bearbeitung erfolgen kann. Der Autor muß also bereits bei der Nutzung seines Text- oder Satzsystems einige rechentechnische Vorgaben beachten, will man das erstellte Dokument im nachhinein einfach und mit geringem personellen Aufwand in andere Formate konvertieren.
Ebenso einfach sollte auch der Austausch von Dokumenten zwischen Kommunikationspartnern möglich sein, sei es nun zwischen den Komponenten eines Systems oder zwischen dem System und dem Nutzer. Ein universales Austauschformat, welches jeder Partner ‚versteht‘ und auswerten kann, ist hierbei überaus hilfreich. Beim Verschicken von Mail-Anhängen über das Internet ist es wegen fehlender 8Bit-Unterstützung beispielsweise oft noch erforderlich, die Dokumente zunächst mit Verfahren wie *uuencode*, *binhex* oder *base64* in 7Bit-ASCII-Code zu konvertieren und beim Empfänger wieder in das Ursprungsformat zu überführen. Dank moderner Mailclients geschieht diese Konvertierung automatisch und vom Anwender unbemerkt – ein Dateiformat im 7Bit-Code (z.B. RTF oder HTML) zu präferieren ist daher heutzutage nicht mehr unbedingt erforderlich. Aber natürlich spielt auch hierbei der konkrete Anwendungsfall eine entscheidende Rolle.

3.1.4 Recherchierbarkeit

Eine der bedeutendsten neuen Möglichkeiten, die digitale Dokumente bieten, ist die Recherche nicht nur in den Metadaten und einem kurzen Abstract, sondern im Volltext der gesamten Publikation. Die Vergangenheit hat allerdings gezeigt, daß eine Suche in

unstrukturierten Texten viel zu viele Dokumente mit oftmals geringer Relevanz liefert. Wichtigste Aufgabe muß es deshalb sein, Voraussetzungen für eine verbesserte *‚precicion'* bei der Recherche zu schaffen und die Antwortzeiten trotz einer Suche in großen Datenmengen gering zu halten. Nachfolgend stichpunktartig einige Forderungen an eine Recherche mit hoher Qualität der Treffermenge:

- strukturelle Suche in speziell ausgezeichneten Bestandteilen eines Dokuments (z.B. Überschriften, Autoren, Zitate, Orte, Tabellen usw.)
- Suche im gesamten Text (Volltextrecherche)
- Suche in Metadaten (Titel, Schlagwörter usw.)
- Nutzung boolescher Ausdrücke (UND, ODER, NICHT) zur Verknüpfung mehrerer Suchbegriffe
- Trunkierung von Begriffen
- Tolerierung von Schreibfehlern
- Nachbarschaftssuche
- Möglichkeit der Nutzung regulärer Ausdrücke
- Suche in mathematischen oder chemischen Formeln oder Noten
- Suche nach Bildteilen in Grafiken und Videos bzw. nach Audiosequenzen in Musikstücken (z.B. „Suche alle Bilder, die links unten einen roten Kreis enthalten")

Nicht alle diese Forderungen sind einfach umzusetzen. Insbesondere die Suche nach nichttextuellen Bestandteilen ist kompliziert; einmal implementiert bietet sie aber ungeahnte Möglichkeiten der Recherche in aufwendig gestalteten Dokumenten. In jedem Fall gilt: für das Erreichen der oben genannten Ziele ist ein Dateiformat mit starken Strukturierungsmöglichkeiten unerläßlich.
Zusätzlich dazu sollte es das zugrundeliegende System auch gestatten, durch thematisch sortierte Mengen von Dokumenten zu navigieren (Browsing). Dem Nutzer ist es dadurch möglich, die Menge der zu durchsuchenden Dokumente zu beschränken bzw. einen Überblick über das vorhandene Material zu einem Thema oder aus einem Bereich (z.B. alle Dissertationen des Instituts für Informatik) zu erhalten.

3.1.5 Darstellbarkeit und Zitierbarkeit

Abhängig von der Bestimmung eines Dokuments ist eine adäquate Präsentation in unterschiedlichen Formen und Ausprägungen notwendig. Soll die elektronische Publikation lediglich gedruckt werden, spielen eventuell eingebettete multimediale Objekte (wie z.B. Animationen) eine untergeordnete Rolle – bei der Anzeige auf dem Monitor sollten diese aber sehr wohl wiedergegeben werden. Grundsätzlich kann man folgende Forderungen an ein Präsentationsformat stellen:

- Möglichkeit der Bildschirmdarstellung sowohl des gesamten Dokuments als auch von Teilen (z.B. einzelne Seiten, Kapitel, ...)
- Ausdruckbarkeit des gesamten als auch von Teilen des Dokuments
- Identität zwischen Bildschirmdarstellung und Ausdruck
  → Zitierbarkeit (Seitenidentität zwischen Papier- und verschiedenen digitalen Versionen)
- Integrierte und standardisierte Darstellung von Sonderzeichen, Strukturen, Multimediaelementen
- Nutzbarkeit von Hyperlinks

Im wissenschaftlichen Umfeld spielen die erwähnten landes- oder fachspezifischen Sonderzeichen eine besondere Rolle: Dissertationen der Theologie können beispielsweise auch hebräische Zeichen enthalten und so sollte ein Dateiformat die Speicherung und korrekte Darstellung derartiger Informationen z.B. durch Nutzung von Standards wie Unicode ermöglichen. Multimediaelemente, wie Video- oder Audiosequenzen, werden meist gesondert gespeichert, so daß die hier betrachteten Formate in der Lage sein sollten, diese Objekte zu verlinken und zu integrieren.
Oft existieren neben einem eventuellen Papierexemplar einer elektronischen Publikation für unterschiedliche Anwendungszwecke auch verschiedene digitale Versionen des Dokuments. Um die für Autoren und Leser wichtige Zitierbarkeit sicherzustellen, müssen die Textseiten in den einzelnen Publikationsformen übereinstimmen, um Passagen eindeutig referenzieren zu können. Bei Ablage des elektronischen Dokuments in einem entsprechenden Archiv muß außerdem gewährleistet sein, daß Verlinkungen via URL[6] oder z.B. URN[7] über lange Zeit erhalten bleiben. Wildwuchs

3.1.6 Standardisierung

Grundsätzlich lassen sich die von Firmen für eigene Produkte entwickelten „Industriestandards“ von offenen Standards unterscheiden, die meist von internationalen Gremien in einem längeren Diskussionsprozeß zwischen Wissenschaftlern und Firmenvertretern definiert werden. Auch bei den Industriestandards sind die Spezifikationen häufig offengelegt, so daß man nicht unbedingt auf die vom Hersteller angebotene Software angewiesen ist, allerdings hat man so gut wie keinen Einfluß auf die zukünftige Entwicklung.
Weiterhin existiert noch eine große Anzahl proprietärer Dateiformate, die jeweils für ein spezielles Produkt (z.B. ein Textverarbeitungssystem) entworfen wurden und deren Spezifikation nur selten frei verfügbar ist. Durch ständige Änderungen und Anpassungen der Formate an neue Bedürfnisse kann ein Wildwuchs entstehen, der unkontrollierbar und schnell unüberschaubar wird. Offene oder zumindest anerkannte Industriestandards sind proprietären Dateiformaten daher vorzuziehen, um auf Veränderungen zeitnah reagieren und z.B. Konvertierungen einfacher vornehmen zu können. Außerdem lassen sich standardisierte Formate oftmals auch mit verschiedenen Programmen gleichermaßen gut erzeugen und weiterverarbeiten, was gerade im Hinblick auf eine herstellerunabhängige Langzeitarchivierung von Vorteil ist. Die Erstellung von Dokumenten mit einem proprietären Text- oder Satzsystem sollte nur dann zugelassen werden, wenn dieses auch eine Speicherung in einem besser geeigneten Dateiformat erlaubt.

3.1.7 Archivierbarkeit

Anders als bei Papierexemplaren, die entsprechend präpariert und unter geeigneten klimatischen Bedingungen durchaus 500 Jahre und mehr überdauern können, ist eine Aufbewahrung und Verfügbarmachung elektronischer Publikationen über einen solchen Zeitraum hinweg nur schwer sicherzustellen. Die Verwendung offener Standards sowie gut strukturier- und konvertierbarer Formate sind allerdings günstige Voraussetzungen für eine längerfristige Archivierbarkeit digitaler Dokumente. Ist abzusehen, daß sich die aktuell verwendete Hard- und Software (Speichermedien, Betriebssysteme, Anwendungsprogramme, usw.) wesentlich verändern wird, müssen die Daten rechtzeitig und möglichst verlustfrei in

[6] Uniform Resource Locator
[7] Uniform Resource Name (siehe Kapitel 4.3)

andere Formate überführt werden – oder es ist sicherzustellen, daß die Arbeitsumgebung selbst für die Nachwelt archiviert wird (siehe Kapitel 4.1).
Bei einem geschätzten Aufkommen von ca. 20.000 deutschsprachigen Dissertationen pro Jahr mit einem Umfang von durchschnittlich 200 Seiten ist selbst bei reinen und vielleicht sogar komprimierten Textdokumenten außerdem der Speicherplatz ein limitierender Faktor. Auch wenn Datenträger wie Festplatten oder Bänder sicherlich keine kostenkritischen Ressourcen mehr sind, ist es dennoch sinnvoll, ein angemessenes Verhältnis zwischen zu speichernder Information und Dokumentgröße zu fordern.
Aufgrund mangelnder Erfahrung in dem ‚relativ' jungen Gebiet des elektronischen Publizierens läßt sich zum gegenwärtigen Zeitpunkt leider kein „Patentrezept" für eine sichere Langzeitarchivierung angeben. Es wird daher empfohlen, nicht ausschließlich auf eine digitale Speicherung zu vertrauen, sondern immer auch ein korrespondierendes Papierexemplar zu erhalten – auch wenn dadurch wichtige Strukturinformationen, wie Videos und Hyperlinks, verloren gehen.

3.2 Bewertung einzelner Dateiformate

Nachdem auf den letzten Seiten mit Verfügbarkeit, Strukturierbarkeit, Konvertierbarkeit, Austauschbarkeit, Recherchierbarkeit, Darstellbarkeit, Zitierbarkeit, Standardisierung und Archivierbarkeit die allgemeinen, von [Ohst98] zusammengetragenen Anforderungen an Dateiformate nochmals in komprimierter Form dargelegt wurden, sollen nun konkrete, in der Praxis relevante Formate näher untersucht werden. Wegen der besonderen Ausrichtung dieser Arbeit auf Online-Hochschulschriften sollen hier vorrangig textuelle Informationen im Vordergrund stehen und für deren Lieferung, Präsentation und Speicherung[8] entsprechend geeignete Dateiformate vorgeschlagen werden. Eine Verlinkung oder gar Einbettung multimedialer Elemente, wie Grafiken, Videos oder Musik, sollte grundsätzlich möglich sein – für detaillierte Beschreibungen sei allerdings auf z.B. [Born01] verwiesen. Die nachfolgenden Ausführungen orientieren sich u.a. an den Arbeiten von [Ohst98] und [Stoja00], basieren aber auch auf eigenen Erfahrungen mit den Vor- und Nachteilen gängiger Dateiformate.

3.2.1 Microsoft Word (.doc)

Das Speicherformat des Textverarbeitungssystems Word der Firma Microsoft wurde aufgrund seiner großen Verbreitung exemplarisch für die Vielzahl der existierenden proprietären und nicht-standardisierten Dateiformate gewählt.[9] Die interne Struktur hat sich seit der Entstehung mehrfach geändert – die einzelnen Versionen sind durch Zugabe oder Wegfall bestimmter Funktionalitäten nur bedingt kompatibel zueinander. Microsoft Word als Teil einer Office-Suite ist nur für Windows und Mac-OS verfügbar und insbesondere wegen günstiger, an ein Neugerät gebundener OEM-Versionen[10] weit verbreitet. Ansonsten kann die Software als teuer bezeichnet werden, bietet aber auch einen enormen Funktionsumfang und wie jedes moderne Textverarbeitungsprogramm die Möglichkeit einer Strukturierung. Es können eigene Formatvorlagen erstellt und mitgelieferte genutzt oder modifiziert werden – allerdings mit dem Nachteil, daß Strukturierungen wie „Überschrift 3" immer auch ein bestimmtes Layout implizieren (also z.B. „Schriftart: Arial; Größe: 14; Stil: fett und unterstrichen"). Im Hinblick

[8] Diese Unterscheidung ist rein funktional zu verstehen: die Formate können teilweise oder vollständig identisch sein.
[9] weitere Vertreter dieser ‚Gattung': WordPerfect, Works, alte StarOffice-Dateien, ...
[10] Original Equipment Manufacturer

auf die geforderte Konvertierbarkeit ist diese Kopplung von Struktur und Layout problematisch, da sie wegen unterschiedlicher Formatvorlagen meist mit Qualitätsverlust verbunden und ein Austausch von DOC-Dateien im allgemeinen nur zwischen gleichen Soft- und Hardwareplattformen möglich ist. Für eine Recherche und eine Archivierung ist das Word-Format ebenfalls nur schlecht geeignet, zu sehr ist es Versionsschwankungen ausgesetzt und zu wenig ist über den internen Aufbau bekannt. Zwar existieren aktuelle Konverter und Indexierungstools, mit dem nächsten Update sind diese aber vermutlich nicht mehr verwendbar. Ein eigenständiges und sogar kostenloses Präsentationsprogramm[11] ist nur für Windows-Systeme verfügbar. Vorteile des Formats hingegen sind eine identische Bildschirm- und Druckdarstellung und die vielfältigen Integrationsmöglichkeiten von Grafiken, Tabellen, Sonderzeichen (entsprechende Fonts vorausgesetzt), Formeln und z.B. Hyperlinks.
Wegen seiner einfachen Bedienung und den ausgereiften Bearbeitungsfunktionen ist es insbesondere für die Produzenten wissenschaftlicher Literatur interessant – für plattform-unabhängige Darstellung, Retrieval und Archivierung sind andere Dateiformate besser geeignet. Den Grund dafür faßt [Mönn00] nochmals wie folgt zusammen:

> „[...] Die Herstellerfirmen legen die Formatspezifikationen selbst fest und geben diese in der Regel nicht preis. Die Umwandlung dieser proprietären Formate in andere kann daher nur in Kooperation mit den Firmen gelingen. Es besteht deshalb die Gefahr, dass es in der Zukunft nicht mehr möglich sein wird, vorhandene Texte in andere Formate zu wandeln, wenn zum Beispiel eine Firma vom Markt verschwindet oder ihr Textverarbeitungsprogramm einstellt. [...]
> Proprietäre Textverarbeitungsprogramme sind demzufolge ungeeignet für das langfristig angelegte elektronische Publizieren.“

3.2.2 ASCII-Text (.txt)

Ein Dokument, welches nur aus Zeichen des ASCII-Codes[12] besteht und außer Leerzeichen, Tabulatoren und Zeilenumbrüchen keinerlei Strukturierungsmöglichkeiten bietet, wird im allgemeinen als ASCII-Text bezeichnet. Aufgrund der geringen Komplexität ist für Erstellung, Anzeige und (nicht unbedingt identischen) Ausdruck lediglich ein einfacher Editor oder Viewer notwendig – Konvertierungen in andere Formate sind problemlos möglich. Auch umfangreiche Rechercheoptionen (boolesche Verknüpfungen, Trunkierung, reguläre Ausdrücke, ...) lassen sich relativ einfach realisieren; strukturelle Suchen sind aus oben genanntem Grund hingegen nicht oder nur über Umwege machbar. Aufgrund ihrer Standardisierung und den einfachen Konvertierungsmöglichkeiten wären ASCII-Texte für die Langzeitarchivierung gut geeignet, wegen fehlender Integrationsmöglichkeiten von Sonder-zeichen, Formeln, Grafiken, multimedialen Elementen, Hyperlinks u.ä. sind sie für die adäquate Erstellung und Speicherung wissenschaftlicher Arbeiten in der heutigen Zeit aber nicht mehr verwendbar.

3.2.3 TeX, LaTeX

TeX ist ein recht kompliziert zu bedienendes, fast nur im naturwissenschaftlichen Umfeld verbreitetes Text- und Satzsystem, welches auf 7Bit-ASCII-Code aufbaut und sich

[11] Microsoft Word-Viewer
[12] American Standard Code for Information Interchange – ursprünglich 127 Zeichen, dann erweitert auf 255. Unicode: 255*255

insbesondere durch seine einzigartige Integration von Formeln und Sonderzeichen auszeichnet. Über Befehlssequenzen, Makroerweiterungen und Stylepakete[13] werden Inhalt und Layout des Dokuments definiert. Für die Darstellung müssen TeX-Source-Files zunächst in das geräteunabhängige DVI-Format übersetzt und dann z.B. nach PostScript (siehe unten) konvertiert werden. Die benötigten Werkzeuge sind für alle gängigen Plattformen frei verfügbar; ein Austausch über das Internet samt Compilierung und Präsentation auf Empfängerseite ist problemlos möglich, sofern die gleichen Makro- und Stylepakete installiert sind. Strukturierungen lassen sich ähnlich Word-Formatvorlagen an einen Text anbringen, allerdings erfolgt hier nur eine teilweise Bindung an das Layout, da zusätzlich eine Vielzahl von reinen Layoutanweisungen zur Verfügung stehen. Die guten Strukturierungsmöglichkeiten begünstigen zwar eventuelle Konvertierungsmaßnahmen, für ausgefeilte Recherchen werden sie allerdings kaum benutzt – meist werden dafür TeX-Ausgabeformate, wie PS, herangezogen und diese indexiert. Für eine langfristige Speicherung ist TeX ebenfalls nur bedingt geeignet: zwar bietet es eine hohe Layoutqualität und wird herstellerunabhängig von vielen Personen gepflegt und weiterentwickelt; die Mächtigkeit beruht jedoch im wesentlichen auf der Verfügbarkeit von Zusatzpaketen, die somit ebenfalls in der jeweils passenden Version archiviert werden müßten. Außerdem ist TeX speziell für die Verwendung in den Naturwissenschaften ausgelegt – eine Konvertierung anderer Formate nach TeX wäre ein unvertretbarer Aufwand.

3.2.4 PostScript (.ps)

Das im Jahre 1985 von der Firma Adobe Systems Inc. vorgestellte, im allgemeinen ebenfalls auf 7Bit-ASCII-Code basierende und im Zusammenhang mit TeX bereits erwähnte PS-Format hat sich innerhalb kürzester Zeit als Industriestandard etabliert. Grund ist die Möglichkeit, textuelle und graphische Elemente geräte- und auflösungsunabhängig definieren zu können – die Ausgabe auf Drucker und Bildschirm ist damit identisch. PostScript-Dateien werden in der Regel nicht direkt erstellt und editiert, sondern entstehen durch Generierung aus anderen Formaten. Meist werden dafür Datei- oder Druckerfilter benutzt; eventuell vorhandene Strukturinformationen gehen bei der Konvertierung verloren. Wegen der starken Layoutfixierung der Seitenbeschreibungssprache ist eine Umwandlung nur in weniger komplexes ASCII oder in andere layoutorientierte Dateiformate, wie PDF (siehe unten), möglich und sinnvoll. Sonderzeichen, Formeln und Grafiken lassen sich je nach Verfügbarkeit im Ausgangsformat problemlos in PostScript integrieren. Weitere Vorteile sind die Plattformunabhängigkeit vorrangig kostenloser Treiber und Viewer und die qualitätserhaltende Verkleiner- und Vergrößerbarkeit der seitenweise ausdruckbaren Dokumente. Durch die vom Ausgabegerät unabhängige Layouttreue ist so auch eine optimale Zitierbarkeit gewährleistet; Indexierungen sind wegen fehlender Strukturinformationen hingegen nur mit größerem Aufwand möglich. Als alleiniges Archivierungsformat ist es nur unter Vorbehalt zu empfehlen: mangels Komprimierung sind PostScript-Dateien oft sehr groß und aufgrund nicht-vorhandener struktureller Auszeichnungen auch nur schwer in andere Formate zu konvertieren. Für die Speicherung des Layouts und für die notwendige Zitierbarkeit ist es als Sekundärformat aber durchaus geeignet.

[13] LaTeX als Erweiterung der ca. 300 ursprünglichen TeX-„control sequences“ um ein umfangreiches Makropaket vereinfacht die Arbeit und ermöglicht die logische Auszeichnung von Textbestandteilen.

3.2.5 Portable Document Format (PDF)

PDF als plattformunabhängiger Industriestandard[14] und Weiterentwicklung des PS-Formats erfreut sich gerade im Bereich des elektronischen Publizierens großer Beliebtheit. Anders als PostScript ermöglicht das universelle, 1993 ebenfalls von Adobe Systems konzipierte Dateiformat neben der Speicherung des Seitenlayouts auch eine Beschreibung von Inhalts-, Formular- und Hypertextstrukturen, wodurch ein Dokument wenigstens in Grundzügen gegliedert werden kann. Für die Erstellung kommen meist kommerzielle Produkte in Form spezieller ‚Druckertreiber' zum Einsatz, die den als Zwischenschritt erzeugten PS-Code ins PDF-Format umwandeln und damit eine Generierung direkt aus anderen Anwendungen, wie z.B. Microsoft Word, heraus ermöglichen. Alle Schriften, Formatierungen, Farben und Grafiken des Quelldokuments bleiben dabei erhalten, unabhängig von Textverarbeitungsprogramm und Betriebssystem. Wie bei PostScript sind Bildschirmdarstellung und Ausdruck immer identisch und wegen der Layoutorientierung von hoher Qualität. Das PDF-Format bietet aber auch zusätzliche Features: zum einen sind derart gespeicherte Dokumente stark komprimiert: zehn MB PostScript entsprechen ca. einem MB PDF. Zudem kann in PDF-Dokumenten gesucht und navigiert – und der Zugriff abgestuft beschränkt werden, z.B. durch Unterbinden der Ausdruckmöglichkeit oder des Markierens und Kopierens von Text (weitere Infos siehe [Mönn00]). Die mit Abstand am weitesten verbreitete Software zur Anzeige von PDF-Dateien ist der Acrobat Reader[15], welcher von Adobe kostenlos verteilt wird und auch als Plugin für verschiedene WWW-Browser zur Verfügung steht. Grundsätzlich gelten für das Portable Document Format ähnliche Einschränkungen wie für seinen Vorgänger: außer der Einbringung von Inhaltsverzeichnis, Kommentaren, Bookmarks und Hyperlinks bietet es keine Möglichkeiten der Speicherung struktureller Zusatzinformationen und ist daher nur für Volltextrecherchen und Umwandlungen in andere layoutorientierte Formate geeignet. Die alleinige Archivierung von PDF-Dokumenten ist aufgrund der unzureichenden Strukturierung und den sich daraus ergebenden Konvertierungsproblemen nicht empfehlenswert. Auch die Festlegung auf das Format eines Herstellers ist kritisch, da keine konkreten Aussagen über dessen Zukunft getroffen werden können. Wegen der seitenidentischen Darstellung von elektronischer und gedruckter Fassung, der damit verbundenen guten Zitierbarkeit und vor allem wegen der großen Verbreitung ist PDF für Präsentationszwecke hingegen die aktuell wohl beste Wahl.

3.2.6 Standard Generalized Markup Language (SGML)

SGML ist eine formal definierte Metasprache, die Dokumentformate beschreibt. Anders als bei den bisher besprochenen Dateiformaten steht hier nicht das Layout im Vordergrund, sondern die Definition der inhaltlichen Struktur. SGML ist aus der bereits 1969 entwickelten Generalized Markup Language hervorgegangen und wurde 1986 als ISO Standard 8879 verabschiedet. Wie auch bei GML werden Struktur und eigentlicher Text einer Publikation strikt getrennt von Angaben zum konkreten Erscheinungsbild aufbewahrt. Letzteres wird in einer separaten Stylesheet-Datei festgelegt, welche z.B. dem Element „Überschrift" das Layoutattribut „fett, 14pt" zuordnet. Der Vorteil liegt auf der Hand: ein und dasselbe SGML–Dokument läßt sich durch Angabe verschiedener Styles auf unterschiedlichen Medien ausgeben. Die Metasprache beschreibt in Form von „Document Type Definitions" außerdem

[14] Trotz ständiger Weiterentwicklung und Bindung an einen Hersteller hat sich das PDF-Format schnell verbreitet und kann als Quasi-Standard bezeichnet werden. Eigenentwicklungen sind durch offengelegte Spezifikationen möglich.

[15] inzwischen umbenannt in Adobe Reader

Klassen gleichartiger Dokumente, wie z.B. „Geschäftsbrief“ oder „Dissertation“. Eine DTD besteht dabei aus der hierarchischen Verkettung logischer Bestandteile (Absatz, Fußnote, Bildunterschrift, ...) und legt präzise fest, wie oft, in welchem Kontext und in welcher Reihenfolge diese Elemente im Dokument vorkommen können und müssen (siehe auch Abschitt XML). SGML-Datei, Styledefinition und die Tags der DTD liegen normalerweise im 7Bit-ASCII-Format vor, wodurch die Erstellung auch mit einfachsten Text-Editoren möglich ist. Komfortabler gestaltet sich die Arbeit allerdings mit kommerziellen Systemen, die zwar meist sehr teuer sind, aber dafür die zum Teil komplizierten Regeln und Abhängigkeiten zwischen den Files prüfen.[16] Konzeptbedingt ist SGML ein Beispiel für Strukturierbarkeit schlechthin. Die DTD legt genau fest, welche Elemente wo und wie im Dokument auftreten dürfen – eine Verknüpfung mit Layoutinformationen findet an dieser Stelle nicht statt. Entsprechend gut läßt sich SGML in andere Dateiformate konvertieren; die Erzeugung aus anderen Formaten ist hingegen schwieriger und nur selten automatisch zu bewerkstelligen, da Zusatzinformationen über den internen Aufbau vorliegen müssen. Gute Voraussetzungen bieten mit Formatvorlagen erstellte Dokumente – hier müssen Struktur und Layout ‚nur‘ getrennt und in die DTD- und Stylesheet-Hierarchie eingebracht werden. Obwohl sich SGML-Dateien wegen der konsequenten Auszeichnung ihrer Bestandteile äußerst gut für strukturelle Suchen eignen, wird dieser Vorteil – sicherlich nicht zuletzt wegen der geringen Verbreitung der Metasprache im Web – nur mangelhaft von existierenden Index- und Suchmaschinen ausgenutzt. Als Präsentationsformat hat sich SGML bisher ebenfalls nicht durchsetzen können, da es nicht direkt, sondern nur in Kombination mit einer passenden DTD und Styledefinition angezeigt werden kann. Spezielle Viewer sind hierfür nötig, die Sonderzeichen, Hyperlinks, multimediale Bestandteile und vor allem Monitor- und Druckbild allerdings nur selten in gleicher Weise darstellen – Zitierbarkeit ist somit nicht gegeben. Für eine langfristige Archivierung ist SGML aber dennoch hervorragend geeignet, da es ein Maximum an Zusatzinformationen zu einem Text speichert.

3.2.7 Hypertext Markup Language (HTML)

HTML ist eine auf SGML basierende Auszeichnungssprache zur Darstellung von Informationen aller Art, unabhängig vom verwendeten Rechner- und Betriebssystem. Das Format wurde im Kontext des WWW-Projekts[17] spezifiziert und 1995 in der Version 2.0 standardisiert. Die aktuelle und vermutlich letzte Version 4.0 wurde 1997 festgelegt, der Standard von den einzelnen Browserherstellern durch Einführung eigener Tags aber immer wieder aufgeweicht. Trotz der damit verbundenen Inkompatibilitäten hat sich HTML, nicht zuletzt wegen seiner einfachen Erlernbarkeit und den vielen verfügbaren WYSIWYG-Editoren[18], als Präsentationsformat im Internet durchgesetzt. Für die Darstellung ist im Gegensatz zu SGML keine separate Styledefinition notwendig, da die Webbrowser eine Standardformatierung vornehmen[19] und viele Tags alleine schon Layoutbedeutung haben. Eine strikte Trennung von Inhalt, Struktur und Layout ist hier also nicht gegeben, dennoch weisen HTML-Dokumente durch Nutzung einer spezifischen, in SGML formulierten DTD eine gute Strukturierbarkeit auf, die eine einfache Konvertierung von und in andere Formate ermöglicht. Insbesondere mit Standardformatvorlagen erstellte Publikationen lassen sich so

[16] Unter dem Stichwort „SGML aus Autorensicht: Schreiben für den Austausch oder neues Schreibparadigma“ wird in [Riehm92] das Für- und Wider dieses Formats sehr ausführlich beleuchtet.

[17] World Wide Web-Projekt am Europäischen Institut für Teilchenphysik bei Genf (CERN); begonnen 1989, Anfang 1993 gingen die ersten WWW-Server ans Netz.

[18] What You See Is What You Get

[19] die allerdings via Style-Standards, wie z.B. Cascading Style Sheets, überschrieben werden kann

weitgehend automatisch auf die Tag-Struktur von HTML abbilden. Erweiterungen dieses Tag-Sets sind wegen der festen DTD und den vordefinierten Styles allerdings nicht möglich – vom Autor angegebene zusätzliche Informationen können somit auch nicht gespeichert werden. Für Recherchen in Metadaten und dem Volltext ist HTML gut geeignet, auf Strukturinformationen basierende Index- und Suchmaschinen gibt es hingegen kaum. Wie bereits erwähnt, ist das Hauptanwendungsgebiet der Sprache daher auch eher die grafische Gestaltung von Webpräsentationen, für die HTML durch Einbindung von Grafiken, Tönen, Sonderzeichen und fast beliebigen weiteren Dokumenttypen gute Möglichkeiten bietet. Nachteilig ist, daß die Darstellung z.B. von den installierten Fonts und den Styledefinitionen des Browsers (bzw. den gesondert zu speichernden Stylesheets) abhängt und daß wegen des oft stark variierenden Druckbilds leider auch keine Zitierbarkeit gegeben ist.

3.2.8 Extensible Markup Language (XML)

Faßt man die beiden letzten Abschnitte zusammen, wird schnell klar, daß sowohl SGML, als auch HTML nur bedingt als zukunftsweisende Dokumentformate bezeichnet werden können: wegen seiner umfangreichen Spezifikation ist Ersteres für den ‚Normal-Anwender' einfach zu komplex und für den direkten Publikationsprozeß ungeeignet. Die Möglichkeiten von HTML wiederum sind aufgrund der geringen Anzahl von Tags derart begrenzt, daß es wohl „niemals in der Lage sein wird, alle Dokumente darzustellen, die jemals ins Internet gestellt oder ganz allgemein in elektronische Form umgewandelt werden sollen." [GR00] Selbst die kräftig erweiterte Version 4.0 ist noch viel zu beschränkt und statisch, um alle Bereiche der zahllosen Web-Anwendungen abzudecken (Datenbanken, Suchmaschinen, optimale Präsentation, professionelles Drucken, Datenverifizierung, usw.). Vom Web-Konsortium wurde deshalb eine Arbeitsgruppe ins Leben gerufen, welche eine „Light-Version" von SGML entwerfen sollte, die gleichzeitig eine Weiterentwicklung von HTML darstellt.[20] Ergebnis ist die „eXtensible Markup Language", welche 1998 in der Version 1.0 vom W3C verabschiedet wurde und als echte Untermenge von SGML ebenfalls als text-basierte Meta-Auszeichnungssprache zu bezeichnen ist. Bei der Entwicklung galt es, insbesondere die von der SGML Special Interest Group festgelegten und nachfolgend kurz zusammengefaßten 10 Planungsziele umzusetzen, die in [GR00] sogar als die „Zehn Gebote von XML" bezeichnet werden:

1. XML soll über das Internet einfach zu nutzen sein.
2. XML soll eine große Bandbreite an Anwendungen unterstützen.
3. XML soll mit SGML kompatibel sein.
4. Es soll einfach sein, Programme zu schreiben, die XML-Dokumente verarbeiten.
5. Die Zahl der optionalen XML-Funktionen sollte so klein wie möglich sein, bestenfalls sogar bei Null liegen.[21]
6. XML-Dokumente sollen lesbar und möglichst verständlich sein.
7. Das Design vom XML solle schnell vorbereitet sein.[22]
8. Das Design von XML soll formal und prägnant sein.
9. XML-Dokumente sollen leicht zu erstellen sein.
10. Kürze sollte bei der Auszeichnung von XML von minimaler Bedeutung sein.

[20] Die ebenfalls spezifizierte Extended Hypertext Markup Language XHTML wird nicht mehr weiterentwickelt, da sie HTML lediglich via XML definiert und keine neuen Funktionalitäten bietet.
[21] Um Inkompatibilitäten und sonstige Schwierigkeiten zu vermeiden, sind keine zusätzlichen Funktionalitäten erlaubt. Zudem sollten weltweit alle XML-Parser in der Lage sein, sämtliche XML-Dokumente zu interpretieren.
[22] Dem W3-Konsortium war es wichtig, die Spezifikation zügig abzufassen, um auf die neuen Bedürfnisse zeitnah reagieren zu können.

Viele Beschränkungen, die in HTML noch bestehen, werden in XML aufgehoben. Die Struktur wird so z.B. sehr viel konsequenter vom Layout getrennt. Wichtigster Unterschied ist wohl aber die Möglichkeit, eigene Tags zu definieren. Jede Person, jedes Unternehmen, jede Organisation usw. kann mittels einer DTD eine eigene *Sprache* für das gesamte Spektrum der anfallenden Daten definieren. XML basiert auf der Grundidee, daß Dokumente aus einer Reihe von *Entitäten* zusammengesetzt sind. Jedes dieser „Objekte" enthält mindestens ein *Element* und jedes Element kann wiederum durch null oder mehr *Attribute* (Eigenschaften) gekennzeichnet sein. Diese beschreiben die Art und Weise, wie jedes Element verarbeitet werden muß. XML-Dokumente sollten *gültig*, müssen aber *wohlgeformt* und syntaktisch korrekt sein. Was das heißt, soll an einem kurzen Beispiel verdeutlicht werden und ist insbesondere deshalb von Bedeutung, weil XML-Strukturen im Rahmen des zu konzipierenden Dokumentenservers immer wieder auftauchen werden, sei es nun in Form eines eventuellen Recherche- und Archivierungsformats (siehe Kapitel 5.1) oder z.B. als Austauschformat für Metadaten (siehe Kapitel 3.4.2).

Hier ein einfaches XML-Dokument:

```
<beispiel>
     <element1 key="value" attr="wert">
           Hello Word &ausruf;
           <el2>Inhalt</el2>
           <nix/>
     </element1>
</beispiel>
```

Um *wohlgeformt* zu sein, muß ein Dokument einigen einfachen Regeln entsprechen, die es einem XML-Prozessor ermöglichen, die Datei richtig zu analysieren. Zunächst einmal muß es ein *Stammelement* (hier „beispiel") geben, welches die gesamte Dokumenteninstanz umfaßt und in keinem anderen Element als Inhalt auftaucht. Weiterhin müssen die Tags ausgeglichen und korrekt geschachtelt sein: zu jedem Starttag (z.B. „<element1>") muß es an passender Stelle ein schließendes Endtag („</element1>") geben – es sei denn, das Element ist leer („<nix/>"). Attribute gehören in Anführungszeichen, können wie erwähnt aber auch vollständig entfallen (wie in „el2").
Gültige XML-Dokumente sind wohlgeformt *und* entsprechen einer DTD, die entweder als externe Datei referenziert oder direkt in das Dokument eingebettet werden kann. Letzteres könnte für das obige Beispiel wie folgt aussehen:

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE beispiel [
<!ELEMENT beispiel (element1)>
<!ELEMENT element1 (#PCDATA,el2,nix)>
<!ATTLIST element1
          key CDATA #REQUIRED
          attr CDATA #REQUIRED >
<!ELEMENT el2 (#PCDATA)>
<!ELEMENT nix EMPTY>
<!ENTITY ausruf "!">
]>
<beispiel>
      [...]
</beispiel>
```

Dieses Beispiel soll, ohne explizit auf die einzelnen Zeilen der DTD eingehen zu wollen, zumindest ansatzweise die umfangreichen Möglichkeiten der Auszeichnung von XML-Dokumenten verdeutlichen. Unabhängig von Layoutinformationen wird exakt spezifiziert, welche Elemente mit welchem Inhalt in welcher Reihenfolge auftreten dürfen. Die Metasprache definiert dabei nur die Struktur einer allgemeinen Dokumentenklasse und noch nicht das Dokument selbst. Dieses wird erst durch Anwendung der Regeln erzeugt und mit den konkreten Textinformationen der XML-Datei instanziiert. Die Anzeige wiederum übernehmen speziell an die jeweiligen Bedürfnisse anpaßbare ‚Formatvorlagen', die angeben, wie die einzelnen Bestandteile letztlich auf den Bildschirm oder zu Papier gebracht werden sollen. Die existierende Layoutsprache DSSSL[23] hat sich für diesen Zweck als ungeeignet erwiesen und so wird neben den im HTML-Umfeld verbreiteten und leider recht unflexiblen Cascading Stylesheets (CSS) eine relativ neue Entwicklung präferiert: die Extensible Stylesheet Language (XSL).
Durch die beschriebene Zerlegung der Dokumente in einzelne Teile wird eine Trennung von Inhalt, Struktur und Layout möglich[24] – eine Forderung, die auch SGML erfüllt, aber eben sehr viel komplizierter und bei weitem nicht so elegant (vergl. auch [Mönn00], Stichwort XML statt SGML). XML wurde entworfen, um ein einheitliches und universelles Dateiformat für ein breites Spektrum von Anwendungen zu schaffen – und erfreulich umfangreich ist das Angebot an entsprechender Literatur. Weiterführende Informationen zum Thema XML und den vielen damit verbundenen Standards, wie z.B. XPath, XLink oder XSLT finden sich daher z.B. in [WL02]; an dieser Stelle sollen abschließend nochmals die wichtigsten Eigenschaften (in Anlehnung an [Lind00]) zusammengefaßt werden:

- XML ist ein offener Standard des World Wide Web Consortium (W3C) und somit herstellerunabhängig.
- XML-Dokumente bestehen im Gegensatz zu proprietären Binärformaten aus ASCII-Text und somit dem ‚kleinsten gemeinsamen Nenner' sämtlicher Computersysteme. Durch die damit verbundene Plattform-, Programmiersprachen- und Protokollunabhängigkeit wird Portabilität zwischen heterogenen Systemen gewährleistet. Außerdem sind XML-Dokumente menschenlesbar und selbstbeschreibend.
- XML adaptiert etablierte Standards des Internets und kann so dessen Infrastruktur nutzen[25]
- XML ist problemlos erweiterbar. Es gibt keine vordefinierten Tags und so kann die Semantik einer XML-Sprache individuell für eine Anwendung entworfen werden.
- XML-Dokumente enthalten keinerlei Informationen über die Darstellung des Inhalts. Formatierungsanweisungen sind getrennt vom Inhalt in einem Stylesheet gekapselt, was Medienunabhängigkeit ermöglicht.
- XML-Dateien können, eine strukturbeschreibende DTD vorausgesetzt, mit universellen Parsern eingelesen und auf Integrität geprüft werden. Die Parser müssen nicht selbst programmiert werden, was eine Zeit- und Kostenersparnis mit sich bringt.
- XML ist in der Lage, Sonderzeichen auf Basis verschiedenster Standards zu integrieren. Dabei können z.B. direkt die Zifferncodes von ISO-8859 oder Unicode, aber auch deren verbale Umschreibung in Form von Entities genutzt werden. Spezielle Dialekte, wie CML oder MathML ermöglichen die Darstellung komplizierter chemischer und mathematischer Formeln.

---

[23] Document Style Semantic and Specification Language

[24] Ob dabei alle drei Bestandteile im XML-Dokument selbst oder in externen Dateien abgelegt werden, spielt für das Ergebnis keine Rolle.

[25] Da XML ein text-basiertes Format ist, können die Inhalte über HTTP transportiert werden, ohne daß irgendwelche Veränderungen an der Netzstruktur erforderlich sind.

- XML läßt aufgrund der guten Strukturierbarkeit präzise Suchanfragen zu und erlaubt verlustfreie Konvertierungen in andere Formate. Es eignet sich damit sowohl als Rechercheformat, als auch für Archivierungszwecke.
  → Für die Erstellung und Präsentation sind andere Formate vorzuziehen, da geeignete Editoren selten, teuer und zumeist wenig intuitiv sind, und weil Aufgrund variierender Styledefinitionen eine Zitierbarkeit nur schwer zu realisieren ist.

3.3 Auswahl geeigneter Dateiformate

Wie in Abschnitt 3.1 bereits einleitend erwähnt, gilt es nun, Vorschläge für konkret zu nutzende und die einzelnen Publikationsschritte am besten unterstützende Dateiformate zu unterbreiten. Zuvor sollen die bisher gewonnenen Erkenntnisse aber ähnlich wie in [Ohst98] tabellarisch zusammengefaßt werden, was eine Gegenüberstellung und einen besseren Vergleich ermöglicht:[26]

konkretes Dateiformat / allgemeine Anforderung	MS Word	ASCII-Text	PostScript	PDF	(La)TeX	SGML	HTML	XML
Verfügbarkeit	⊕	√	√[1]	⊕[1]	√	⊕	√	⊕
Strukturierbarkeit	⊕	–	–	–	⊕	√	⊕	√
Konvertierbarkeit	⊕	√	–	–	√	√	√	√
Recherchierbarkeit	⊕	⊕	⊕	⊕	⊕	√	⊕	√
Darstellbarkeit	⊕	√[2]	√[2]	√	√	⊕	√	⊕
Zitierbarkeit	√	–	√	√	⊕	⊕	⊕	⊕
Standardisierung	–	√	⊕	⊕	⊕	√	√	√
Archivierbarkeit	–	√	⊕	⊕	⊕	√	⊕	√

Legende:

√ mit wenigen oder keinen Einschränkungen erfüllt

⊕ mit Einschränkungen erfüllt

– nicht oder mit starken Einschränkungen erfüllt

[1]: Bearbeitung nicht möglich

[2]: nicht hypertextfähig

Diese Tabelle und die recht umfangreichen Ausführungen der letzten Seiten sollen einen Überblick über die gängigsten Textformate für elektronische Publikationen geben und deren Eigenschaften im Hinblick auf die zuvor herausgearbeiteten allgemeinen Anforderungen herausstellen. Die Liste erhebt dabei natürlich keinerlei Anspruch auf Vollständigkeit: Word, ASCII, PS, PDF, (La)TeX und die erwähnten Metasprachen stellen nur eine kleine Auswahl an verfügbaren Dateiformaten dar, lassen sich aufgrund der gebotenen Funktionalitäten aber

[26] Die von [Ohst98] übernommene dreistufige Klassifikation ist leider nicht in der Lage, konkrete Eigenschaften eines Formats genau zu erfassen. Deshalb sollte die Tabelle nicht ohne die vorhergehenden verbalen Erläuterungen gelesen werden.

sehr gut bestimmten Dokumentenklassen zuordnen, was letztlich eine Beschränkung des Dokumentenservers auf wenige zu unterstützende Formate erlaubt. Letzteres ist von entscheidender Bedeutung, will man langfristige Verfügbarkeit mit vertretbarem technischen und finanziellen Aufwand sicherstellen. Ziel muß es sein, Empfehlungen in Form einer Präferenzliste zu formulieren – oder gar verbindlich festzulegen, welche Programme und Dateiformate zu verwenden sind.

> „Es geht nicht darum ein einziges Format und Medium zu definieren, wohl aber geht es darum, Wildwuchs zu vermeiden und die Vielfalt auf ein vernünftiges Maß zu bringen, damit die Erwartungen der Autoren und der Leser auf eine nutzungsorientierte Dienstleistung erfüllt werden.“ [Leh97A]

Faßt man die Arbeit von [Ohst98], den Bericht zum Teilprojekt „Formate“ des DFG-Projekts „Dissertationen Online“ [Schmitt96] und die eigenen Analysen zusammen, kann man für die Formate der hier im Vordergrund stehenden elektronischen Hochschulschriften folgende vereinfachte Klassifikation angeben:

- **Anlieferungsformat:**
  - es soll ein weit verbreitetes, für alle zugängliches und erschwingliches Standardformat sein
  - es sollte möglichst kein proprietäres Format sein
  - Textverarbeitungsprogramme sollen das Format erzeugen können
  - es muß dem Printformat des begutachteten Prüfungsexemplars entsprechen
  - es muß dem Printformat des gedruckten Archivierungsexemplars entsprechen
  - es muß die maschinelle Weiterverarbeitung und Konvertierung in das Archivierungs- und Präsentationsformat ermöglichen

- **Archivierungsformat:**
  - es sollte ein weltweit verbreitetes Standardformat sein
  - es müssen Softwaretools zum automatischen Konvertieren in andere Formate existieren
  - aus dem Archivierungsformat muß die Originalversion (also das Anlieferungsformat) rekonstruierbar sein

- **Präsentationsformat:**
  - Betrachten am Bildschirm muß auch mit WWW-Browsern möglich sein
  - es muß eine Volltextrecherche unterstützen
  - ein gezieltes Navigieren im Dokument muß möglich sein
  - es soll dem Printformat soweit wie möglich entsprechen (insbesondere, um die Zitierfähigkeit der elektronischen Version zu erhalten)
  - das Ausdrucken des gesamten Dokuments soll möglich sein
  - das Ausdrucken von ausgewählten Teilen des Dokuments soll möglich sein

Weiterhin sollte das Präsentations-, das Archivierungs- oder ein zusätzlich zu definierendes **Rechercheformat** möglichst nicht nur eine Volltextrecherche, sondern auch eine *strukturierte* Suche zulassen (vergl. auch [Dob99]).
Die Bewertungsmatrix (siehe letzte Seite) läßt sehr gut erkennen, welches der vorgestellten Dateiformate die jeweiligen Anforderungen am ehesten erfüllt.

Für die plattformunabhängige Darstellung einer Publikation sind zum gegenwärtigen Zeitpunkt wohl insbesondere zwei Formate geeignet: PDF und HTML. Beide sind hypertextfähig, relativ einfach erstellbar und somit sehr weit verbreitet. PDF bietet eine hohe Layoutqualität und wegen identischer Papier- und Druckversion die Zitierbarkeit eines Dokuments, während HTML eine Einbettung verschiedenster multimedialer Elemente und die Generierung dynamischer Inhalte ermöglicht.

Auf das Anlieferungsformat hat man hingegen meist nur wenig Einfluß, zu unterschiedlich sind die rechen- und programmtechnischen Voraussetzungen und die Vorlieben der Autoren. Im Hinblick auf Konvertierung und langfristige Speicherung sollten hier natürlich Text- und Satzsysteme mit Strukturierungsmöglichkeiten präferiert werden, allerdings ist eine konsequente Auszeichnung der einzelnen Bestandteile mit viel Arbeit und Know How verbunden. Soll sich der Aufwand für nachträgliche Bearbeitungen in Grenzen halten, müssen die Autoren schon im Vorfeld entsprechend betreut und z.B. im Umgang mit Formatvorlagen (Word, StarOffice, ...) bzw. Makros (LaTeX) geschult werden – für die betroffene Institution eine enorme Mehrbelastung.

Gleiches gilt auch für das Archivierungsformat. Der Vergleich zeigt ganz klar, daß hierfür eigentlich nur SGML bzw. der Nachfolger XML in Frage kommt. Leider setzt deren Erstellung aber hochgradig strukturierte Dokumente voraus und so müssen die Publikationen entweder direkt mit entsprechenden, meist kompliziert zu bedienenden Editoren oder als Folge umfangreicher Konvertierungen erzeugt werden. Der Aufwand scheint derart groß zu sein, daß bisher nur wenige deutsche Universitäten SGML oder XML für die Speicherung elektronischer Hochschulschriften nutzen.[27]
Nichtsdestotrotz sind beide Formate für Archivierungs-, aber auch für Recherchezwecke langfristig am besten geeignet. Als offene Standards sind sie nicht vom Aufstieg und eventuellen Fall einer einzelnen Firma abhängig – und gerade SGML wird im professionellen Bereich des Dokumentenmanagements bereits seit vielen Jahren erfolgreich eingesetzt. Die Auszeichnungssprachen bieten ein größtmögliches Maß an Flexibilität und Strukturierbarkeit und lassen sich entsprechend gut durchsuchen und in andere Formate konvertieren. Gerade die Möglichkeit des strukturellen Retrievals in beliebigen Textbestandteilen und nicht nur in einer kleinen Menge von Metainformationen (siehe nächster Abschnitt) stellt eine völlig neue Qualität dar: es lassen sich wesentlich gezieltere Suchanfragen formulieren, die die Treffermenge minimieren und die *Precision* erhöhen.

Zusammenfassend läßt sich insgesamt also folgender Publikationsprozeß andenken: mittels Formatvorlagen oder Makros erzeugte Dokumente werden vorzugsweise nach XML in eine DTD samt zugehörigem Stylefile konvertiert. Aus der Originaldatei wird zusätzlich ein PDF-Dokument, und aus der XML-Datei – orientierend am Inhaltsverzeichnis der Arbeit – eine Menge von HTML-Files generiert. Letztere ermöglichen später ein kapitelweises Browsing und die einfache Anzeige von Rechercheergebnissen. Das PDF-Dokument wiederum kann zum Lesen oder Drucken des kompletten Volltextes dienen; archiviert und indexiert wird die XML-Datei.

[27] Hauptvertreter ist sicherlich die HU Berlin, die grundsätzlich nur speziell strukturierte Dissertationen annimmt (siehe auch Kapitel 5.1).

Diese Lösung, die von der Humboldt-Universität Berlin (unter Nutzung von SGML) auch tatsächlich angewandt wird, erscheint zukunftsweisend, ist aber leider auch recht kompliziert und aus bereits genannten Gründen für alle Beteiligten trotz möglicher Automatisierungen mit viel Aufwand verbunden.

Einen solchen Workflow ohne umfangreiche Planungen und vorherige Umstrukturierungsmaßnahmen an einer diesbezüglich eher unerfahrenen Institution wie der Universitätsbibliothek Potsdam durchsetzen zu wollen, ist daher unmöglich. Im Rahmen der vorliegenden Diplomarbeit sollen deshalb eher Vorschläge unterbreitet werden, wie eine geeignete Aufbewahrung und Verfügbarmachung wissenschaftlicher Literatur aussehen könnte. Wegen personeller Engpässe und nur wenig vorhandener Schulungsangebote kann XML als alleiniges Recherche- und Archivierungsformat zum gegenwärtigen Zeitpunkt noch nicht genutzt werden, muß für die UB Potsdam auf lange Sicht aber das zu favorisierende Dateiformat bleiben. Aus diesem Grund wird der hier implementierte Dokumentenserver auch gute Erweiterungsmöglichkeiten bieten, die es der Bibliothek erlauben, auf zukünftige Veränderungen angemessen reagieren zu können (siehe auch Kapitel 6).

→ Grundsätzlich lohnt es sich, die Entwicklungen im Bereich einheitlicher, offener Dokumentformate weiter zu verfolgen. Die Gefahr, daß binäre PDF- oder DOC-Dateien in 20 Jahren vielleicht nicht mehr lesbar sein könnten, wurde allgemein erkannt und so machen sich inzwischen auch Hersteller wie Microsoft Gedanken über die Zukunft ihrer proprietären Formate. Das auf der diesjährigen CeBIT vorgestellte Office2003 beispielsweise soll, ähnlich wie Konkurrent Star-/OpenOffice, endlich auch ein Speichern in XML erlauben. Schade nur, daß Microsoft das offene Format erneut um eigene Funktionen erweitert, die z.B. eine grafisch exakte Darstellung ermöglichen – wozu man wieder ein entsprechendes Programm benötigt. Projekte wie 1dok.org[28], die offene Standards für den Austausch elektronischer Dokumente initiieren wollen, sind schön und gut,

> „[...] ob sich Branchenriesen wie Microsoft oder Adobe [aber] darauf einlassen, ist fraglich, denn die wollen ihre hauseigenen Formate durchsetzen – als Absatzgrundlage für ihre Programme. Trotz Ankündigungen der Hersteller auf der CeBIT ist ein Esperanto für digitale Dokumente also nicht in Sicht. Dabei ist es höchste Zeit." [CHIP03]

3.4 Dokumentbeschreibung durch Metadaten

Die Evaluation gängiger Dateiformate hat ein Problem besonders deutlich gemacht: existiert keine strukturelle Auszeichnung der einzelnen Bestandteile, sind effektive Recherchen mit hoher Trefferqualität unmöglich. Da die zu archivierenden Dokumente aber oftmals nur wenig oder gar unstrukturiert vorliegen, müssen zusätzliche Informationen gespeichert werden, um den Inhalt dennoch erfassen und dieses Manko zumindest ansatzweise beheben zu können.

> „Speziell für den Bereich digitaler Publikationen im Netz (World Wide Web) haben sich Bibliothekare und Informatiker zu Arbeitsgruppen zusammengefunden, die mit sogenannten Metadaten effektive netzorientierte Erschließungsverfahren entwickeln." [Leh97B]

[28] http://www.1dok.org/de - „DIN A4" fürs Internet - Der Weg zu einem offenen Dokumentenformat.
Das Projekt wird unterstützt von der EU und dem Wirtschaftsministerium Schleswig-Holstein.

Tim Berners-Lee, der Erfinder des World Wide Web und Direktor des W3C, definiert diese „Metadaten“ dabei als „[...] maschinenlesbare Informationen über elektronische Ressourcen oder andere Dinge“ [Bern97]. Man versteht unter ihnen ganz allgemein also „Daten über Daten“, mit deren Hilfe Informationsressourcen beschrieben und dadurch besser auffindbar gemacht werden. Sie liefern grundsätzliche Angaben zu einem Dokument, wie z.B. Titel, Autor und Zeitpunkt der Veröffentlichung und reproduzieren damit – vergleichbar mit altertümlichen Zettelkatalogen – im Prinzip genau das, was an Erschließungsarbeit in den Bibliotheken seit jeher geleistet wird. Im einfachsten Fall stellen Metadaten „Attribut–Wert“-Paare dar, wie z.B. „Autor – Sebastian Ohme“ oder „Abgabedatum – 01.10.2003“. Ein effektiver Einsatz setzt allerdings, wie auch schon bei den Dateiformaten, einen gewissen Standardisierungsgrad voraus. Herkömmliche bibliothekarische Regelwerke haben inzwischen ein so hohes Komplexitätsniveau erreicht, daß sie sich auf die Fülle der Dokumente in elektronischen Netzen nur noch schwer übertragen lassen und so sind sie für die Definition standardisierter Metadatenformate ungeeignet.[29] Neue Ansätze mußten her und entsprechend viele Initiativen gab es, die Metadaten abhängig von Speicherort, Anwendungsbereich, Verwendungsmöglichkeit, Erzeugungsmethode und ähnlichen Kriterien klassifiziert und geeignete Formate vorgeschlagen haben. Beispielhaft seien hier

- Encoding Archival Description (EAD)
- Maschinelles Austauschformat für Bibliotheken (MAB)
- Computer Interchange of Museum Information (CIMI)
- Machine-Readable Cataloging (MARC)

für hochstrukturierte, differenzierte Formate und

- Summary Object Interchange Format (SOIF)
- LDAP Data Interchange Format (LDIF)
- Dublin Core (DC)

für strukturierte, weniger stark differenzierte Formate aufgeführt (siehe [AGVT98] und für Informationen bezüglich der vielen Klassifikationsmöglichkeiten z.B. [Aud02]).
Das letztgenannte „Dublin Core“-Metadatenmodell ist der dabei wohl bekannteste und am meisten diskutierte Ansatz und soll wegen seiner einfachen, kostengünstigen, aber dennoch wirkungsvollen Handhabung daher nachfolgend nochmals genauer vorgestellt werden.

3.4.1 Dublin Core Metadata Element Set

Dieses Metadatenformat ist seit 1995 in interdisziplinärem Konsens zwischen Informatikern, Wissenschaftlern und Bibliothekaren gewachsen und wurde in Anlehnung an den Ort des ersten Treffens der Entwicklergruppe, Dublin (Ohio), benannt.

> „Die Absicht war [...], Kernelemente für die Beschreibung von dokumentartigen Objekten im Fernzugriff festzulegen, um auf diese Weise ihre Identifizierung in einer Netzumgebung (Internet) und den Zugriff darauf zu erleichtern.“ [HS97]

Vereinfacht gesagt sollte also ein Minimalsatz von Erschließungselementen definiert werden, die eine verbesserte Präzision und Retrievalfähigkeit von digitalen Dokumenten ermöglichen. Die Dublin Core Metadata Initiative selbst beschreibt die Zielsetzung folgendermaßen:

> „The Dublin Core is a […] metadata element set intended to facilitate discovery of electronic resources. Originally conceived for author-generated description of Web resources, it has also attracted the attention of formal resource description communities such as museums and libraries.“ [DC97]

[29] vergl. auch http://www2.sub.uni-goettingen.de/intrometa.html

Das Format besteht in der aktuellen Version 1.1 aus 15 ressourcenbeschreibenden Hauptelementen und ist bewußt einfach gehalten, damit die Autoren unabhängig von ihrer fachlichen Herkunft die entsprechenden Metadaten gegebenenfalls selber generieren können, ohne dabei auf teure Verfahren oder die Hilfe entsprechend geschulten Personals angewiesen zu sein.

Nr.	Bezeichnung	Inhalt
1	Title	Der vom Verfasser, Urheber oder Verleger vorgegebene Name der Ressource.
2	Creator	Die Person(en) oder Organisation(en), die den intellektuellen Inhalt verantworten. Im Falle mehrerer Autoren ist jeder in einem eigenen Meta-Element zu erfassen.
3	Subject	Schlagwörter, Stichwörter oder Phrasen, die das Thema oder den Inhalt beschreiben. Das Element kann sowohl systematische Daten nach einer Klassifikation oder Begriffe aus anerkannten Thesauri enthalten.
4	Description	Eine textuelle Beschreibung des Ressourceninhalts inklusive eines Abstracts (bei dokumentähnlichen Ressourcen) oder einer Inhaltsbeschreibung (bei grafischen Ressourcen).
5	Publisher	Die Einrichtung, die verantwortet, daß diese Ressource in vorliegender Form zur Verfügung steht, wie beispielsweise ein Verleger oder eine Universität.
6	Contributors	Zusätzliche Person(en) und Organisation(en) zu jenen, die im Element *Creator* genannt wurden, die einen bedeutsamen, aber sekundären intellektuellen Beitrag zur Ressource geleistet haben (z.B. Herausgeber, Übersetzer, Illustratoren, …)
7	Date	Datumsangaben der Ressource, wie beispielsweise das Datum der letzten Änderung oder das Veröffentlichungsdatum.
8	Type	Die Art der Ressource (z.B. Dissertation, Diplomarbeit).
9	Format	Das datentechnische Format der Ressource. Dieses Feld enthält die erforderlichen Informationen für eine Verarbeitung der codierten Daten.
10	Identifier	Zeichenkette oder Zahl, die eine eindeutige Identifikation des Dokumentes ermöglicht. Bei vernetzten Ressourcen sind URLs vorgesehen.
11	Source	In dieses Element wird das gedruckte oder elektronische Werk, aus dem die Ressource stammt, eingetragen.
12	Language	Sprachcode der Sprache(n) des intellektuellen Inhalts der Ressource.
13	Relation	Verhältnis zu anderen Ressourcen, die einen formalen Bezug zu dieser Ressource haben, aber als eigenständige Resourcen existieren (z.B. Kapitel eines Buches oder Bilder eines Dokuments).
14	Coverage	Zeitliche, örtliche, flächenhafte oder ähnliche Aspekte, die zur Charakterisierung des Objektes sinnvolle Ergänzungen geben.
15	Rights	Urhebervermerk und rechtliche Bedingungen zur Nutzung dieser Ressource.

Die in der Tabelle aufgeführten Kernelemente geben nähere Informationen über den Inhalt (z.B. Titel, Beschreibung, Sprache), die Urheberschaft (z.B. Autor, Verleger, rechtliche Bestimmungen) und formale Kriterien (z.B. Datum, Format, Identifikationsnummer), sind für eine präzise Beschreibung elektronischer Dokumente und „dokumentähnlicher Objekte“ allerdings nicht ausreichend und so entschloß man sich, weitere *Subelemente* einzuführen, die, in Kombination mit sogenannten *Qualifiern*, eine Spezifizierung der Basiselemente zulassen. [Rusch97] schreib dazu:

> „Ein Qualifier – falls vorhanden – verfeinert den Wert und die Bedeutung des Elements ggf. durch den inhaltlichen Bezug zur Benutzung in einer Fachcommunity. Qualifier entstammen bekannten bibliothekarischen Standards bzw. Standardbegriffen eines Fachgebiets, zu dem die Ressource gehört. Qualifier sind wichtig, weil sie eine selbst zu definierende Spannweite zwischen allgemeinem Gebrauch und wissenschaftlichen Anspruch zulassen.“

Wie genau diese semantische Erweiterung des Dublin Core Standards syntaktisch auszusehen hat, ist nicht festgelegt. Innerhalb des Math-Net-Projekts[30] erfolgt die Klassifikation eines Dokuments beispielsweise mit Hilfe des Subelements MSC[31] und des Qualifiers SCHEME, die für den Bereich „Mathematical problems of computer architecture“ zusammen z.B. den String

```
NAME="DC.Subject.MSC" SCHEME="msc91" CONTENT="68M07"
```

ergeben.[32] Will man hingegen die Sprache einer Publikation und zusätzlich den Untertitel in einer davon abweichenden Sprache spezifizieren, kann dies beispielsweise wie folgt aussehen:

```
NAME="DC.Language" SCHEME="ISO639-2" CONTENT="ger"
NAME="DC.Title.Alternative" LANG="eng" CONTENT="Untertitel"[33]
```

Allgemein läßt sich also folgender Aufbau eines qualifizierten Dublin Core Sets angeben:

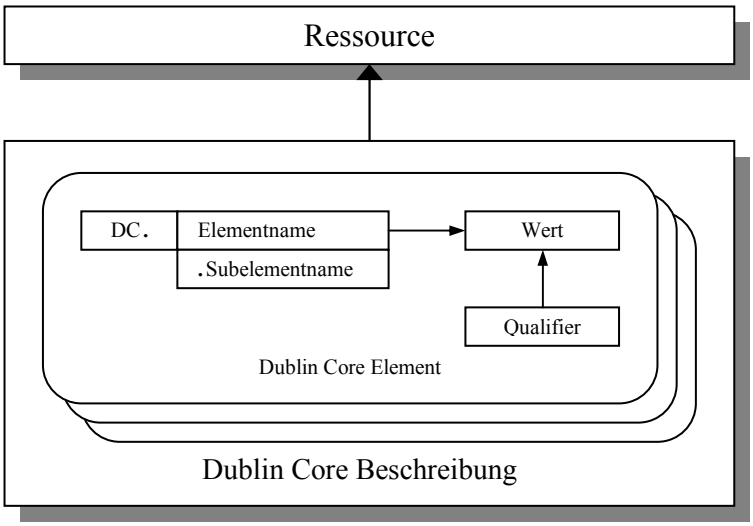


Abbildung 4: Qualified Dublin Core Set

[30] Informationsdienste für die Mathematik im Internet, http://www.mathnet.de
[31] Mathematics Subject Classification, siehe auch http://www.ams.org/msc
[32] Dublin Core kann in den verschiedensten Syntax-Formaten dargestellt werden - dies ist ein Beispiel dafür.
[33] ISO639-2 steht dabei für einen 3-Buchstaben-Code und der Qualifier LANG gibt für das Subelement Title.Alternative die Sprache an.

Abhängig von Anwendungsgebiet und Art der zu beschreibenden Dokumente existieren verschiedene Schemata, die für die Spezifizierung der Subelemente und Qualifier genutzt werden können. Für mathematische Inhalte können so z.B. die Vorgaben des bereits erwähnten Math-Net-Projekts genutzt werden – für Dissertationen wurde von Der Deutschen Bibliothek und dem Projekt „Dissertationen Online“ ein weiterer Metadatensatz entwickelt, der speziell an diesen Dokumenttyp angepaßt ist und im Kapitel 5.2 näher untersucht wird.

Bei der Speicherung und Verfügbarmachung von Metadaten lassen sich grundsätzlich drei Varianten unterscheiden (zitiert nach [Aud02]):

(1) Metadaten werden getrennt vom Dokument gespeichert und können unabhängig von diesem abgerufen werden.
(2) In einem Container werden die beschriebene Ressource und deren Metadaten verpackt. Der Container stellt die Informationen bereit, wenn sie benötigt werden.
(3) Die Metadaten sind im Dokument selbst enthalten.

Ein typisches Beispiel für Metadaten innerhalb eines Dokuments sind die META-Tags im Kopf einer HTML-Datei, die im einfachsten Fall allerdings unqualifiziert und nicht DC-konform sind. Die Beispiele der letzten Seite haben sich bereits an der zu verwendenden Syntax orientiert:[34]

```
<HEAD>
<META NAME="Author" CONTENT="Sebastian Ohme">
<META NAME="Keywords" CONTENT="Diplomarbeit, Digitale Bibliothek">
</HEAD>
```

Größter Vorteil dieser Einbettung von Metaangaben in HTML-Dokumente:
Internet-Suchmaschinen, wie Google, Altavista und Co., können den Inhalt einer Webseite viel besser erfassen und über Ranking-Mechanismen eine Art strukturierte Recherche mit höherer *Precision* anbieten. Um die Indexierung des eigenen Angebots durch Webcrawler, Robots, Harvester, aber auch lokale Searchengines bestmöglich zu unterstützen, wird das zu konzipierende System daher ebenfalls Metatags in die Frontdoor-Seiten[35] integrieren. Zusätzlich dazu werden die Dublin Core-Elemente aber auch in einer Datenbank vorgehalten, um in ihnen suchen zu können und um sie unter anderem über eine OAI-Schnittstelle der Öffentlichkeit zugänglich zu machen (siehe nächster Abschnitt und natürlich Kapitel 6).

Trotz der abschließend zusammengefaßten Vorteile von Dublin Core, wie z.B.

- Internationaler Ansatz und Verbreitung
- Interoperabilität
- Einfache Erstellung
- Erweiterbarkeit
- Keine Pflichtelemente
- Wiederholbarkeit der Elemente
- Elemente können näher beschrieben / eingeschränkt werden

stößt dieser Metadaten-Standard – wie leider auch viele andere – schnell an seine Grenzen: komplexere Zusammenhänge lassen sich nur schwer abbilden, da Relationen zwischen den Elementen nicht auf vernünftige Art und Weise ausgedrückt werden können. Als Beispiel sei

[34] Groß-/Kleinschreibung spielt normalerweise nur innerhalb des „Contents“ eine Rolle.
[35] Hiermit sind die HTML-Seiten gemeint, die den eigentlichen Publikationen vorgeschaltet sind und neben den versteckten Metadaten auch eine kurze Inhaltsangabe im Klartext enthalten.

hier die Elementgruppe „Autor, Geburtsdatum, Geburtsort und Adresse“ genannt: wurde eine Dissertation von mehreren Autoren geschrieben, läßt sich die Zugehörigkeit der einzelnen Elemente zur jeweiligen Gruppe nicht mehr eindeutig feststellen, es sei denn, daß die Elemente in der richtigen Reihenfolge direkt hintereinander geschrieben werden und daß so eine Zuordnung möglich wird.

Dies ist ein Problem, auf das [Weiß00] aufmerksam macht und welches z.B. mit Hilfe des sogenannten Resource Description Framework (RDF[36]) gelöst werden kann. Es handelt sich hierbei um eine universelle, auf XML basierende Metasprache, die „[...] eine Infrastruktur zur Codierung, zum Austausch und zur Wiederverwendung von Metadaten zur Verfügung [stellt].“ [Aud02] Da RDF im Zusammenhang mit Publikationsservern aber bei weitem (noch) nicht so verbreitet ist, wie das Dublin Core Metadata Element Set, und gerade an deutschen Einrichtungen der letztgenannte Standard dominiert, soll an dieser Stelle auf eine weitere Untersuchung verzichtet werden.

3.4.2 Open Archives Initiative

Eine moderne wissenschaftliche Informationsversorgung basiert auf einem freien Zugang zu verteilten wissenschaftlichen Informationen und setzt Standards für den Aufbau und die Interoperabilität lokaler Dokumentenserver voraus. Metadaten sollten, wie im letzten Abschnitt kurz angedeutet, daher nicht ausschließlich strukturelle Recherchen in nur einem digitalen Archiv erlauben, sondern insbesondere auch der breiten Öffentlichkeit zugänglich gemacht werden. Und genau dies ermöglicht das *Metadata Harvesting Protocol* der Open Archives Initiative. Dieses mit OAI-PMH oder gar OAI abgekürzte Protokoll stellt eine Schnittstelle zum Austausch von dokumentspezifischen Metadaten bereit und sollte ursprünglich dazu dienen, selbst-archivierende Systeme wie Publikations- und Preprint-Server untereinander stärker zu verbinden.[37] Es galt, die neuesten Forschungsergebnisse der wissenschaftlichen Community auf einfachstem Wege, ohne Zugangsbeschränkung und vor allem ohne zeitliche Verzögerung zur Verfügung zu stellen. Ein in der Zeitschrift c't erschienener Artikel vergleicht die Idee hinter OAI mit einem „Napster für die Wissenschaft“:

> „Ähnlich wie bei der Internet-Tauschbörse Napster geht es darum, das Auffinden sowie den Zugriff auf einzelne Objekte in einem System der weltweit verteilten Datenhaltung zu organisieren; im Unterschied zu Napster gibt es hier jedoch keine Probleme der Fairnis gegenüber den Urhebern, da die Autoren ihre Arbeiten selbst ins Netz stellen und nicht wie Musiker von Tantiemen oder Honoraren leben.“[38]

Das erste Treffen von Wissenschaftlern und Informatikern zum Thema Open Archive fand im Oktober 1999 in Santa Fe statt und hatte sich zum Ziel gesetzt

> „[...] to create a forum to discuss and solve matters of interoperability between author self-archiving solutions (also commonly referred to as e-print systems), as a way to promote their global acceptance.“ [Somp00]

Anfang 2001 wurde dann die Version 1.0 des Harvesting Protokolls freigegeben und nach einer ausgiebigen Testphase ist seit Juni 2002 die stabile Version 2.0 verfügbar, welche sich inzwischen für jegliche Art elektronisch publizierter Literatur eignet. Die Offenheit des Standards spielt dabei eine entscheidende Rolle, wobei *open* nicht wie bei dem Begriff der

[36] http://www.w3.org/RDF
[37] Der anfängliche Name lautete daher auch Universal Preprint Service.
[38] http://heise.de/ct/01/06/078/default.shtml

Freien Software für *kostenlos* steht (obwohl dies der Fall ist), sondern für die technische Transparenz. Anders als das komplexe Z39.50-Format, das in Bibliotheken weit verbreitet ist, setzt das OAI-Protokoll auf eine einfache Struktur in Form von XML-Metadatensätzen, die über das allseits bekannte Hypertext Transfer Protocol (HTTP) übertragen werden. Um eine minimale Interoperabilität zu gewährleisten, genügen unqualifizierte Dublin Core Metadaten (siehe letzter Abschnitt) – das Protokoll unterstützt jedoch auch andere Metadatenformate. Grundsätzlich werden über OAI zwei verschiedene Teilnehmer miteinander verbunden: die *Datenprovider* sorgen für den Aufbau von Archiven, entwickeln Mechanismen und Richtlinien zur Aufnahme der Dokumente sowie zu ihrer sicheren Aufbewahrung und machen diese über Metadaten recherchierbar. *Serviceprovider* wiederum implementieren Endnutzerdienste, indem sie diese Daten verfügbar machen – beispielsweise durch den Aufbau von thematischen Suchmaschinen (vergl. auch [Zimm02]).[39] Die nachfolgenden, aus [MK03] entnommenen Abbildungen veranschaulichen das Zusammenspiel:

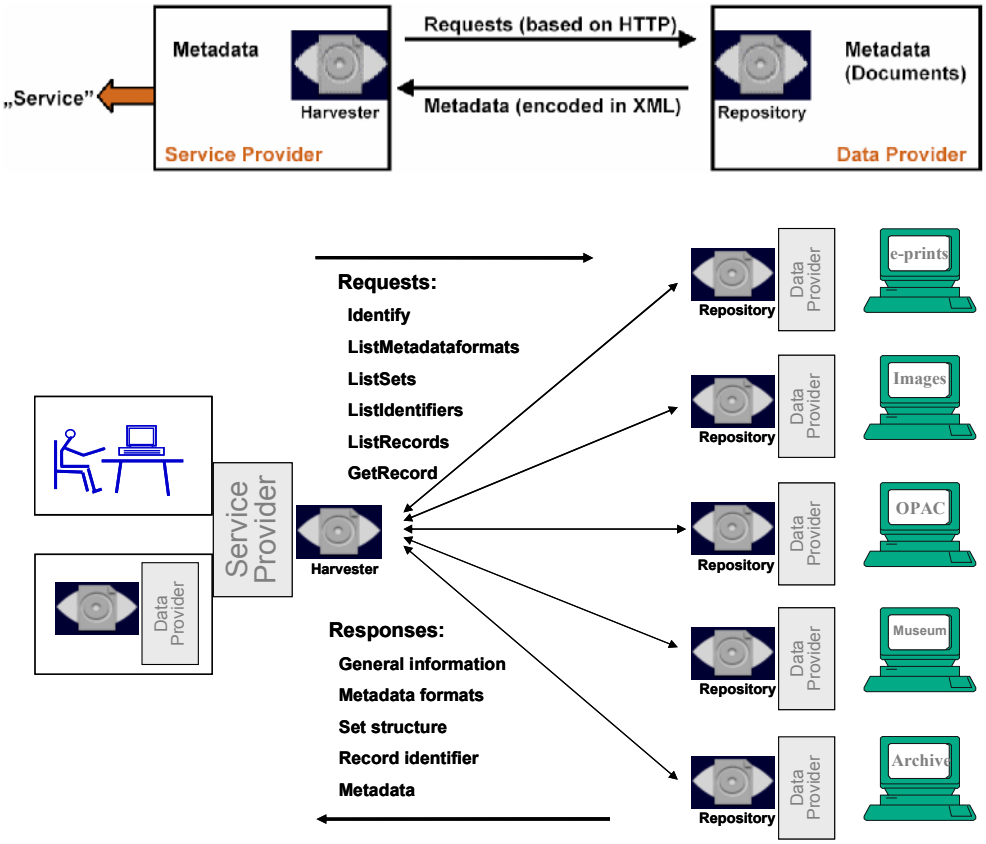


Abbildung 5: OAI Strukturmodell

Die Datenprovider, zu denen nach Fertigstellung dieser Diplomarbeit und des geplanten Dokumentenservers auch die Universitätsbibliothek Potsdam zählen wird, haben die Aufgabe, die in Form von *Repositories*[40] vorliegenden Inhalte aufzubereiten und für das OAI-Protokoll ‚faßbar' zu machen, so daß die Metadaten bei Bedarf von den *Harvestern* der Serviceprovider eingesammelt werden können. Dank der Nutzung von HTTP als Trägerprotokoll gestaltet sich

[39] einen guten Überblick gibt auch http://www.dlib.org/dlib/november02/liu/11liu.html
[40] „Lager", „Behälter", „Sammlung" - hier ist ein elektronisches Textarchiv gemeint, kann aber auch andere elektronische Datenformate enthalten.

dieser Datenaustausch sehr einfach: über normale GET- oder POST-Requests[41] und nur 6 mögliche Anfragearten (siehe Abbildung 5) spezifiziert der Client seine Wünsche und erhält vom Repository-Server die entsprechenden, dynamisch generierten und in XML-Tags eingebetteten Dublin Core-Rechercheergebnisse. Vereinfacht läßt sich folgender Pseudo-Code [Warn01] und nebenstehendes Strukturdiagramm (bei Nutzung einer SQL-Datenbank) angeben:

```
read GET, POST or command line request
check syntax of request
if syntax correct
{
    return XML-reply to request
}
else
{
    return error code and message
}
```

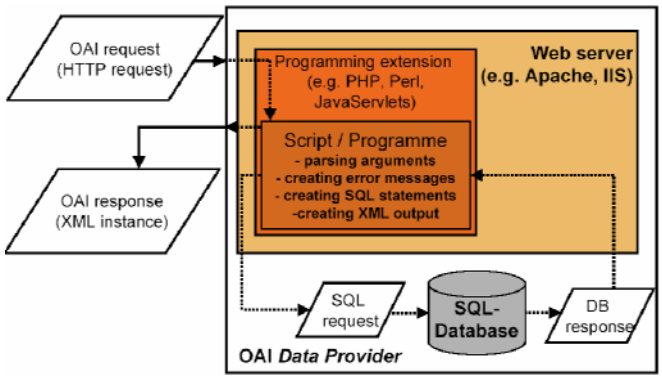


Abbildung 6: OAI Datenprovider Architektur

Die Requests bestehen immer aus der URL des Datenproviders (Adresse des Servers + Pfad zum Script) und einer Argumenten-Liste in Form von „Key=Value“-Paaren. Beispiel:

GET Request
http://www.archiv.de/cgi-bin/oai.pl?verb=ListIdentifiers&metadataPrefix=oai_dc

Der Parameter „verb=[Anfragetyp]“ muß in jedem Fall angegeben werden; Argumente, wie metadataPrefix=oai_rfc1807, identifier=4711 oder resumptionToken=weiter_ab_100, sind meist optional und dienen der Flußkontrolle und dem sogenannten „selective harvesting“. Welche „Key=Value“-Kombinationen für welche „verbs“ möglich sind, soll an dieser Stelle nicht weiter ausgeführt werden – umfangreiche Informationen finden sich in der offiziellen Protokollbeschreibung [OAI02]. Hier sollen lediglich einige wichtige Aspekte hervorgehoben werden, die für die konkrete Implementierung von Bedeutung sind. Die nachfolgende Grafik zeigt eine typische ‚Konversation‘ zwischen Service- und Datenprovider (siehe auch [MK03]):

Abbildung 7: OAI ResumptionToken

[41] mehr zu diesen Anfragearten im praktischen Teil dieser Arbeit

Die in Abbildung 7 dargestellte Beispielanfrage des Harvesters ist recht allgemein gehalten, schränkt die Ergebnismenge aber schon beträchtlich ein: zum einen sollen lediglich Dublin Core-codierte Metadaten zurückgeliefert werden („oai_dc“) und zum anderen sind für den Client offensichtlich nur Einträge seit Anfang 2003 interessant. Von den insgesamt 267 Treffern werden zunächst nur 100 herausgegeben, um die transferierte Datenmenge klein zu halten und um ein HTTP-Timeout zu vermeiden. Zusätzlich zu den Records (genauer Aufbau siehe unten) wird noch ein Token generiert, den der Harvester nutzen kann, um an die nächsten 100 Datensätze zu gelangen. Dies wiederholt sich solange, bis alle zur ursprünglichen Anfrage passenden Metadaten übertragen wurden. Wie die „resumptionToken“ dabei auszusehen haben, ist nicht festgelegt; und ob es eine Beschränkung der Listengröße gibt, bleibt ebenfalls dem Datenprovider überlassen.
Neben der Nutzung von Datumsangaben ist die Definition sogenannter „Sets“ eine weitere Möglichkeit des selektiven Harvestings. Die Metadaten werden dazu nach bestimmten inhaltlichen Kriterien – z.B. entsprechend der Zugehörigkeit des Volltextes zu einem bestimmten Fach – in (nicht notwendigerweise disjunkte) Mengen eingeordnet, die es dem Serviceprovider durch Angabe optionaler „set=[GuppenID]“-Parameter später erlauben, nur die für ihn relevanten Einträge herunterzuladen. Auch hierbei muß es zwischen beiden Partnern Absprachen bezüglich der genauen Syntax und Struktur geben – Festlegungen seitens OAI-PMH existieren nicht.

Nachdem in Grundzügen bekannt ist, wie Anfragen an einen Data Provider zu formulieren sind, sollte nun noch geklärt werden, wie die jeweiligen XML-Antworten auszusehen haben. Auf der nächsten Seite findet sich dazu ein kompletter Beispieldatensatz, der über die OAI-Schnittstelle des Anbieters BioMed Central unter angegebener URL[42] zu beziehen und von Browsern der neuesten Generation auch direkt darstellbar ist.

```
http://www.biomedcentral.com/oai/2.0/?verb=GetRecord
  &metadataPrefix=oai_dc&identifier=oai%3Abiomedcentral.com%3Abcr619
```

Die gestrichelten Linien sollen den grundlegenden Aufbau des dynamisch generierten Datensatzes besser veranschaulichen: zunächst wird ein Rootelement[43] „OAI-PMH“ samt Namespace und Verweis auf ein passendes XML-Schema deklariert. Dieser Stamm enthält immer drei Unterelemente:

- *responseDate*, welches ISO8601-codiert Tag und Uhrzeit der Anfrage angibt,
- *request*, welches als Inhalt die Basis-URL der Schnittstelle und als Argumente
- die übergebenen Parameter auflistet und
- entweder ein *error*-Element, falls Fehler aufgetreten sind oder ein Element mit dem gleichen Namen wie dem „*verb*“ der Anfrage (hier GetRecord).

In dieses letztgenannte Element werden alle zum Request passenden Records eingebettet, welche wiederum aus jeweils zwei bzw. drei Teilen bestehen: dem Header, den eigentlichen Metadaten und einem optionalen about-Feld. Im vorliegenden Fall enthält <GetRecord> – entsprechend der Anfrage – nur einen einzigen Datensatz, welcher über den identifier eindeutig spezifiziert wird.

[42] Die Zeichen / ? # & : ; und + innerhalb eines Parameters müssen als Escape-Sequenz codiert werden: oai%3Abiomed bedeutet daher oai:biomed

[43] zur allgemeinen Struktur einer XML-Datei siehe Abschnitt 3.2.8

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
 http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
  <responseDate>2003-08-01T20:15:00Z</responseDate>
  <request identifier="oai:biomedcentral.com:bcr619" metadataPrefix="oai_dc"
   verb="GetRecord">http://www.biomedcentral.com/oai/2.0/</request>
  <GetRecord>
    <record>
      <header>
        <identifier>oai:biomedcentral.com:bcr619</identifier>
        <datestamp>2003-07-01</datestamp>
        <setSpec>all</setSpec>
        <setSpec>articletype:research</setSpec>
        <setSpec>journalgroup:nonbmc</setSpec>
        <setSpec>journal:3003</setSpec>
      </header>
      <metadata>
        <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
         http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
          <dc:title>
           BRCA2mutation carriers, reproductive factors and breast cancer risk
          </dc:title>
          <dc:creator>Tryggvadottir, Laufey</dc:creator>
          <dc:creator>Olafsdottir, Elinborg J</dc:creator>
           [...]
          <dc:subject>
           BRCA2, breast cancer, cohort study, risk factors
          </dc:subject>
          <dc:description>
           Germline mutations in the BRCA genes dramatically increase
           the risk of breast cancer. [...]
          </dc:description>
          <dc:publisher>BioMed Central Ltd.</dc:publisher>
          <dc:date>2003-06-24</dc:date>
          <dc:type>Research article</dc:type>
          <dc:identifier>
           http://breast-cancer-research.com/content/5/5/R121
          </dc:identifier>
          <dc:language>en</dc:language>
          <dc:rights>Copyright 2003 Tryggvadottir et al.</dc:rights>
        </oai_dc:dc>
      </metadata>
      <about>
        <rights [...]>
         The metadata made available via this interface is made
         freely available for commercial or non-commercial use. [...]
        </rights>
      </about>
    </record>
  </GetRecord>
</OAI-PMH>
```

Der Header beinhaltet neben einem Datestamp auch Gruppierungsangaben in Form von setSpec-Tags, die ein *selective harvesting* ermöglichen (siehe weiter oben).
Im metadata-Bereich befindet sich die eigentliche Beschreibung des zugehörigen Volltextes, dessen URL im <dc:identifier>-Feld zu finden ist. Dieses dc: ist auch allen anderen Dublin Core-Metadatentags vorangestellt, um dem Anfrage-Parameter „`metadataPrefix=oai_dc`“ Rechnung zu tragen und um die Zugehörigkeit zum gesondert spezifizierten Namensraum bzw. zur Schema-Definition zu kennzeichnen.[44]
Der about-Teil schließlich enthält wichtige Copyright-Bemerkungen und sonstige eventuell bedeutsame Informationen zum Record, kann aber auch weggelassen werden.

Eine solche XML-Datei läßt sich verhältnismäßig einfach dynamisch generieren. Ein vom Webserver ausgeführtes CGI-Script muß die GET- bzw. POST-Parameter lediglich auswerten und die, z.B. in einer SQL-Datenbank vorgehaltenen Metadaten abhängig vom Request in die jeweiligen XML-Strukturen ‚pressen‘. Das Ganze wird dann via HTTP an den Client übertragen, dort gegebenenfalls unter Zuhilfenahme der entsprechenden DTD geparst und erneut in eine Datenbank überführt. Wie all das ganz genau zu realisieren ist, kann zusammenfassend beispielsweise sehr gut in [OAI01] nachgelesen werden und soll natürlich auch Gegenstand der praktischen Arbeit sein. An dieser Stelle sei abschließend noch auf zwei Punkte aufmerksam gemacht: zum einen muß der Datenprovider auch mit fehlerhaften Anfragen[45] umgehen können und zum anderen sollte das Repository nach einem ausgiebigen Test auf Korrektheit[46] auch bei der Open Archives Initiative registriert werden[47], um dem „Napster der Wissenschaft“ ein weiteres Stück näher zu kommen.

[44] XML Schema ist eine in XML-Syntax spezifizierte DTD. Möchte man keine *unqualified Dublin Core*-codierten, sondern z.B. „oai_marc“-codierte Metadaten austauschen, muß lediglich ein geeignetes XML-Container-Schema für die Syntaxprüfung vorliegen.
[45] Folgende Error-Codes können zusammen mit einer Klartextbeschreibung des Fehlers zurückgegeben werden: badArgument, badResumptionToken, badVerb, cannotDisseminateFormat, idDoesNotExist, noRecordsMatch, noMetaDataFormats, noSetHierarchy (siehe auch [OAI02]).
[46] hierfür bietet sich der Repository Explorer an: http://www.purl.org/NET/oai_explorer
[47] die offizielle Registrierungsseite ist http://www.openarchives.org/data/registerasprovider.html

4 Aspekte der Langzeitarchivierung

Die täglich weltweit von Menschen produzierten und verbreiteten Informationen – sei es nun wissenschaftlicher oder künstlerischer, alltäglicher oder fachspezifischer Art – erscheinen längst nicht mehr ausschließlich gedruckt auf Papier. Die einleitenden Kapitel haben versucht, die Gründe hierfür aufzuzeigen – zusammen mit den damit verbundenen Problemen.
Lösungsansätze die langfristige Archivierung betreffend wurden hingegen nur in Ansätzen beleuchtet und sollen daher nochmals Gegenstand der nachfolgenden Ausführungen sein.
Im Abschnitt 4.1 werden dazu zunächst einige grundsätzliche Verfahrensweisen für den Erhalt wichtiger Publikationen vorgestellt. Anschließend stehen konkrete Möglichkeiten der Sicherung von Authentizität und Integrität der gespeicherten Daten im Vordergrund – zusammen mit neuen Adressierungsarten für einen beständigeren Zugriff auf Dokumente im Internet. Kapitel 4.4 schließlich soll am Beispiel des OAIS-Referenzmodells einen kleinen Einblick in theoretische Überlegungen zum Aufbau eines Archivsystems geben.

4.1 Verfahren

Im Zusammenhang mit der Langzeiterhaltung bedeutender Werke ist es natürlich überaus wichtig, sich wie geschehen intensiver mit Dateiformaten auseinanderzusetzen – ebenso bedeutsam ist aber vor allem die Wahl geeigneter Speichermedien, die eine Aufbewahrung letztlich erst ermöglichen. Theoretisch eignet sich dafür jeder Datenträger, der in der Lage ist, eine Folge von Nullen und Einsen geordnet aufzunehmen. Voraussetzung ist aber auch, daß diese Abfolge unbeschädigt über einen langen Zeitraum hinweg erhalten bleibt. Ein schwieriges Unterfangen, kennt man die mit Jahren oder vielleicht Jahrzehnten als gering zu bezeichnende Lebensdauer aktuell verfügbarer Medien. Obwohl uns digitale Daten oft auf z.B. CD-ROM in physisch fester Gestalt und zumindest mit der Aussicht auf eine gewisse Haltbarkeit begegnen, ist die große Masse elektronischer Publikationen in höchst flüchtiger Form auf magnetisierbaren Speichermedien (Disketten, Magnetbänder- und kassetten, Festplatten, u.ä.) zu finden. Digitale Dokumente sind in ihrer Konsistenz instabil und in hohem Maße manipulierbar: sie können in einer Weise verändert, ergänzt, vervielfältigt, zerteilt und zerstört werden, wie es bei keinem anderen Informationsträger in der Vergangenheit der Fall war (siehe auch [Müll98]).

Aufgrund der begrenzten und mangels Langzeiterfahrung kaum abschätzbaren Lebenserwartung von Datenträgern muß von Zeit zu Zeit überprüft werden, ob die gespeicherten Informationen der untersten Ebene (Nullen und Einsen) noch lesbar sind. Noch vor Erreichen der angenommenen Lebensdauer muß der gesamte Datenbestand dann auf einen jüngeren Träger gleichen Typs umkopiert oder zu einer neuen Generation von Speichermedien gewechselt werden. Letzteres birgt dabei ein ungleich höheres Risiko, da eventuelle Veränderungen im Datenstrom und im Zeichenkodierungsschema nicht selten auch mit einem Datenverlust verbunden sind. Die Verfügbarkeit von Hard- und Software, die zur Erstellung, Bearbeitung und Wiedergabe elektronischer Publikationen benutzt wird, ist zu alledem sehr kurz. Dies führt zu Inkompatibilitäten älterer Dokumente mit neuen Systemen. Selbst wenn das digitale Dokument noch vorhanden und lesbar ist, wird es irgendwann an Rechnern und Programmen fehlen, um es verarbeiten zu können.

In Anlehnung an [Lieg01] sind für die tägliche Praxis der Bestandserhaltung in einer Archivbibliothek daher zusammenfassend folgende Punkte zu beachten:

- Schätzungen über die zu erwartende Lebensdauer von Speichermedien sollten nur als Richtwerte Verwendung finden. Sie entbinden nicht von der laufenden Überprüfung durch physikalische Methoden.
- Die Lebenserwartung von Datenträgern ist nach bisheriger Erfahrung größer als die Verfügbarkeitsspanne der notwendigen Hardwareausstattung auf dem Markt. Handlungsbedarf entsteht also z.T. bereits wesentlich früher als erwartet und ohne die Möglichkeit einer längerfristigen Vorplanung. Ein Frühwarnsystem (technology watch) sollte von den verantwortlichen Institutionen eingerichtet werden, um koordiniert Maßnahmen zum richtigen Zeitpunkt einleiten zu können.[48]
- Für die Durchführung von substanzerhaltenden Maßnahmen werden Metadaten neuer Qualität benötigt, die zur automatischen Prozeßsteuerung eingesetzt werden können. Dies sind z.B. strukturierte und maschinell interpretierbare Angaben über Datenträgertypen, Materialarten und Produktionszeitpunkte.
- Die Problematik der Substanzerhaltung muß in das Bewußtsein der Informationsproduzenten gebracht werden. Setzt man beispielsweise Kopierschutzverfahren ein, die digitale Objekte unlösbar mit spezifischen Trägermedien verbinden oder eine Datenextraktion verhindern, so wird die Information nur solange erhalten bleiben, wie ihr Datenträger.

Bibliotheken dienen dem Erhalt des „kulturellen und wissenschaftlichen Gedächtnisses“ der Gesellschaft. Die technischen Aspekte dieser Langzeitarchivierung sind jedoch – gerade für Medien in elektronischer Form und wie inzwischen mehrfach angedeutet – noch immer weitestgehend ungeklärt. Eine „zeitlich unbegrenzte Aufbewahrung“ bedeutet daher derzeit lediglich eine sichernde und bevorratende Zwischenspeicherung, um die Option aufrecht zu erhalten, die entsprechenden technologischen Fortschritte (Formatstrukturen, Speichermedien) mitvollziehen zu können. Trotz dieser in [AGVT98] zu findenden, recht pessimistischen Einschätzung der Möglichkeiten existieren einige Ansätze, um dem Verfall von Datenträgern und dem Veralten von Dokumentformaten dennoch entgegenzuwirken.
Neben der verlustbehafteten Speicherung der Daten auf nicht-maschinenlesbarer Materie (säurefreies Papier, Mikrofilm, ...) und dem bereits genannten „Copying“ auf stabilere Träger bestimmen vor allem zwei Strategien die Diskussion um die Erhaltung der Benutzbarkeit: Migration und Emulation.

4.1.1 Migration

„Migration benennt den Prozess, in dem ein Objekt durch äußere Einwirkung so modifiziert werden soll, dass es unter veränderten Umgebungsbedingungen [möglichst] ohne inhaltlichen oder strukturellen Informationsverlust weiterverwendet werden kann.“, so die kurze Zusammenfassung von [Lieg01]. Konkreter ausgedrückt ist die Migration ein Maßnahmenbündel, das eine periodische Übertragung digitaler Materialien von einer Hardware-/Software-Konfiguration auf eine andere oder von einer Computergeneration auf die nächste ermöglicht (vergl. [Leh96]). Der Kern der Migrationsstrategie besteht also darin, elektronisch vorliegende Dokumente rechtzeitig vor Veraltung der benutzten Dateiformate und Rechner-

[48] Meist ist bekannt, daß bestimmte Maßnahmen dringend erforderlich wären, aber leider fehlen häufig die Mittel, um diese Maßnahmen umzusetzen.

architekturen in Dateien moderneren Formats zu konvertieren, um sie mit jeweils aktuell verfügbaren Lesegeräten und Programmen betrachten zu können (für notwendige Voraussetzungen siehe auch Kapitel 3). Hauptwunsch dabei ist, daß der Informationsgehalt erhalten bleibt – interne Strukturierungen und Layoutangaben werden jedoch häufig an die neuen Hard- und Softwarekonventionen angepaßt, so daß die Datei dann nicht mehr unter den „alten" Systembedingungen zu verwenden ist. Als problematisch erweist sich dabei außerdem die oft große Menge von Objekten heterogenen Ursprungs und der kaum kalkulierbare Aufwand für deren Umwandlung: die Migration ist keine einmalig durchzuführende Aktion – jederzeit kann die Notwendigkeit zu erneuten Umsetzungen entstehen, weil sich die Umgebungsbedingungen wiederum geändert haben (siehe auch [Lieg01]). Verluste lassen sich dabei kaum vermeiden, so daß bei jeder Datenmigration nicht nur Kontrolläufe und Stichproben, sondern meistens auch manuelle Nachbearbeitungen – verbunden mit hohen Kosten – notwendig sind. Bei mehrfach wiederholten Konvertierungen können außerdem Verfälschungen auftreten, die sich schleichend und kaum bemerkbar vollziehen und letztlich zu einem Verlust der Identität der archivierten Daten führen können.
In der Arbeit von Dominik Bódi [Bódi00] werden mit *Formatvielfalt*, *Skalierbarkeit*, *Paradigmenwechsel* und *Aufschiebbarkeit* weitere wichtige Aspekte und Nachteile der Migration aufgezeigt, die hier nicht nochmals untersucht werden sollen. Festzuhalten bleibt, daß Konvertierungen nicht in jedem Fall für die Sicherstellung einer langfristigen Aufbewahrung geeignet sind:

> „Dort, wo Informationsgehalt und Präsentationssoftware unlösbar eng und individuell miteinander verbunden sind, entziehen sich Objekte der Behandlung durch Migration. Eine CD-ROM-Applikation, die für eine bestimmte Betriebssystemumgebung produziert wurde, kann mit dieser so verflochten sein, dass eine nachträgliche Umsetzung auf andere Systembedingungen mit vertretbarem Aufwand nicht möglich ist." [Lieg01]

Für Objekte diesen Typs wird eine andere Erhaltungsstrategie erwogen ...

4.1.2 Emulation

Bei der Emulation als Synonym für „Nachahmung" wird ein zwischenzeitlich veraltetes System in einer neuen Hard- und Softwareumgebung imitiert, so daß die unveränderte Originalsoftware genauso wie ursprünglich vorgesehen lauffähig ist. Die technischen Rahmenbedingungen müssen dazu möglichst so nachgebildet werden, daß im Programmverhalten kein Unterschied zwischen originalem und emuliertem System festgestellt werden kann. Die Idee, die insbesondere von Jeff Rothenberg, dem Hauptverfechter dieser Archivierungsstrategie, immer wieder propagiert wurde (siehe z.B. [Roth95B]), ist nicht neu: schon lange existieren Emulatoren, die z.B. auf einem Windows-Rechner eine Benutzung alter C64- oder Amiga-Software ermöglichen[49]. Größter Vorteil dabei: es sind keine verlustbehafteten Konvertierungen notwendig; Inhalt und Aussehen elektronischer Dokumente, um die es hier ja hauptsächlich geht, bleiben unverändert erhalten – und somit auch die Intention der Autoren. Anders als bei einem „Hardwaremuseum", in welchem der komplette Computer samt angeschlossener Peripherie archiviert wird, müssen für die Simulation ausrangierter Systeme nicht die Geräte selbst, sondern nur Informationen über die von ihnen bereitgestellten Funktionalitäten aufbewahrt werden. Das auf dieser Dokumentation aufbauende Emulationsprogramm braucht dann für jede neue Rechnergeneration nur *einmal* entwickelt

[49] Beispielhaft seien hier VICE (http://viceteam.bei.t-online.de) und WinUAE (http://www.winuae.net) genannt.

werden – funktionsfähig ist hingegen die *gesamte* auf der Ursprungsplattform basierende Software (siehe auch [Roth95B]). Leider ist dieser durchaus bemerkenswerte Ansatz recht schwer zu realisieren, denn mangels Herstellersupport sind vollständige Spezifikationen nur selten verfügbar. Technologien aus veralteten Systemen werden häufig in der aktuellen Produktpalette weiterverwendet – die Hardwareproduzenten haben somit kaum Interesse an der Veröffentlichung von Betriebsgeheimnissen, wie auch von [Grun01] festgestellt wird. Noch ist der Emulationsansatz im Zusammenhang mit der Bereitstellung elektronischer Publikationen eher theoretischer Natur und in der Praxis kaum verbreitet. Selbst nach Meinung von Jeff Rothenberg sind viele Grundlagen erst noch zu legen und weitere Forschungsarbeit zu leisten. Dennoch wird das Konzept von vielen als „bestechend" bezeichnet und hat langfristig durchaus das Potential, andere Strategien abzulösen oder zumindest (wie z.B. bei der Archivierung multimedialer Daten) zu unterstützen. In jedem Fall bedarf es laut [Roth99] folgender Schritte, die es zuvor zu lösen gilt (zit. n. [Bódi00]):

- Entwicklung von Methoden zur Spezifikation von Emulatoren. Diese Spezifikationen müssen so beschaffen sein, daß man auf zukünftigen Rechnern mit möglichst geringem Aufwand bzw. automatisch einen Emulator herstellen kann. Die Emulation muß so genau sein, daß die Attribute des digitalen Dokuments in der Emulation so weit wie gewünscht den Attributen auf dem ursprünglichen Rechner entsprechen.
- Entwicklung von Methoden zur Erstellung und Verarbeitung von Metadaten. Sie sollten auf jeden Fall einfacher verarbeitbar sein als die archivierten Dokumente („human readable"). Weiterhin müssen Informationen zur Suche, Zugriff und Wiederherstellung der Dokumente enthalten sein. Die Metadaten müssen so beschaffen sein, daß Emulation möglichst einfach wird.
- Entwicklung von Methoden zur Verkapselung von Daten. Eine solche Datenkapsel muß neben dem archivierten Dokument die zugehörigen Metadaten, die zur Emulation nötige Spezifikation und Originalsoftware (auch Betriebssystem) enthalten. Die Verkapselung muß natürlich auch Zusammenhalt der Daten und Schutz vor unerwünschter Veränderung sicher stellen.

4.1.3 Der ideale Ansatz

Keines der auf den letzten Seiten angesprochenen Verfahren ist für die Langzeiterhaltung jeglichen digitalen Materials uneingeschränkt geeignet – die einzelnen Ansätze decken zumeist nur spezifische Datenbestände ab und haben immer auch nicht zu unterschätzende Nachteile, seien es nun unverhältnismäßig hohe Kosten oder der Verlust von inhaltlichen und funktionellen Zusammenhängen. Es gilt daher, allgemeine Konservierungsstrategien und eine detaillierte Vorgehensweise zu entwickeln. Insbesondere muß die Ideallösung

- allgemeingültig sein, d.h. möglichst wenig verschiedene Teillösungen für möglichst viele Daten enthalten. Die Entwicklung eines einzelnen Lösungsansatzes für einen bestimmten Dokumenttyp sollte auf möglichst viele Dokumenttypen übertragbar sein.
- automatisierbar sein, d.h. mit möglichst wenig Benutzerinteraktion ablaufen.
- einfach verwaltbar sein. Insbesondere Metadaten müssen einfach zugänglich und ohne größeren Aufwand betrachtbar/durchsuchbar sein.
- den originalen Zustand der Dokumente nicht oder so geringfügig wie möglich verändern. Ideal wäre eine Lösung, bei der man direkt mit den Originaldaten arbeiten könnte. Wenn sich die Konvertierung von Daten (z.B. Metadaten) nicht vermeiden läßt, so sollte sie vollständig reversibel sein.

- flexibel in Bezug auf gewünschte Datensicherheit und Datenoriginalität, Einfachheit des Zugriffs und erlaubte Kosten sein. Für variierende Bedürfnisse sollten einfach auswählbare Alternativmethoden zur Verfügung stehen.
- auf jeder zukünftigen Rechnerarchitektur implementierbar sein. Die einzige Voraussetzung, die an zukünftige Rechnergenerationen gestellt werden darf, ist, daß sie jedes berechenbare Problem auch berechnen können.
- schon in der Gegenwart auf befriedigende Funktion und Zuverlässigkeit überprüfbar sein.

Erneut zeigt sich, daß all diese von [Bódi00] zusammengetragenen Anforderungen an einen idealen Archivierungsansatz gleichzeitig von keiner der erwähnten Methoden erfüllt werden. Die existierenden Strategien sind mittelfristige Lösungen, denn sie sind allesamt nicht vollständig zufriedenstellend. Der Trend geht hin zu großen Massenspeichern, die relativ einfach umzukopieren sind – an Bedeutung gewinnt aber vor allem die Migration zu Standardformaten, da diese gegenüber Hard- und Softwareveränderungen weniger anfällig sind. Die Machbarkeit der Emulationsstrategie im Kontext des elektronischen Publizierens ist hingegen noch immer recht umstritten, da Entwicklungsrichtung und -fortschritt entsprechender Werkzeuge nicht von Bibliotheken und Archiven bestimmt werden. Eine intensive Beobachtung des Marktes und weitere konkrete Investitionen in die Realisierung dieses Ansatzes sind daher erforderlich – und im Hinblick auf die gebotenen Vorteile auch durchaus empfehlenswert.

Der ideale Ansatz kann jedoch nicht ausschließlich Copying, Migration, Emulation oder z.B. analoges Speichern auf Mikrofilm heißen, sondern einer Kombination aller verfügbaren Archivierungsmethoden gehört die Zukunft. Vor allem aber gilt es, neue Speichertechniken und unzerstörbare, jederzeit lesbare Datenträger zu entwickeln, die eine Bewahrung wichtiger Zeugnisse der menschlichen Kultur in einer „Bibliothek für zehntausend Jahre“[50] ermöglichen. Als Stichwort sei hier der von Wissenschaftlern angestrebte Entwurf eines „Digital Rosetta Stones“ genannt, der als Schlüssel dienen soll, um heutige Dateiformate oder sogar Klassifikationen für künftige Generationen zugänglich zu machen. Die Idee geht zurück auf den im Jahre 1799 von napoleonischen Soldaten gefundenen „Stein von Rosette“, dessen Inschrift *gleichzeitig* in 2 Sprachen und mit 3 verschiedenen Schriftarten verfaßt war und welcher es der Nachwelt damit erstmals ermöglichte, die ägyptischen Hieroglyphen detailliert zu entziffern. Eine zeitgemäße, entsprechend als „Rosetta Disk“[51] bezeichnete und mit Nickel überzogene Variante soll bis zu 2000 Jahre haltbar sein. Beschrieben wird sie mit einem Ionenstrahl; ein Elektronenmikroskop – angeschlossen an eine Digitalkamera und einen Rechner – ermöglicht schließlich ein Auslesen der archivierten Informationen.
Weiterführende Informationen dazu und zu hybriden Lösungen, wie Hologrammen in Kristallspeichern oder sogenannten „Iridium-CDs“, finden sich z.B. in [Lupp01] – und zum Thema Langzeitarchivierung ganz allgemein und sehr umfangreich auch in [Asch01].

[50] dies ist z.B. Ziel der vor einigen Jahren in den USA gegründeten LongNow-Foundation: http://www.longnow.com
[51] quadratisch, 5cm Kantenlänge und entwickelt vom der Firma Norsam.com

4.2 Authentizität und Integrität

Der Betrieb eines Dokumentenservers als Plattform zur Aufbewahrung archivierungswürdiger Publikationen bringt zahlreiche Anforderungen mit sich – und dies vor allem auch für eine Universitätsbibliothek, welcher nicht mehr nur die Empfänger- und Verteilerrolle zukommt, sondern nun auch quasi die des „Verlegers". Neben der Dokumentenbearbeitung und deren Integration in den Geschäftsgang muß der sichere Datenaustausch zwischen Autor und Bibliothek auf der einen, und zwischen Bibliothek und Leser auf der anderen Seite gewährleistet werden. Wer elektronische Publikationen verwendet, „muss sich darauf verlassen können, dass diese vom darin bezeichneten Absender bzw. Autor stammen (*Authentizität*) und unverfälscht erhalten bleiben (*Integrität*). Vertrauen und Glaubwürdigkeit sind die beiden Ziele, die damit erreicht werden sollen", so [Scholze02]. Während die Wahrung von Authentizität und Integrität papiergebundener Dokumente relativ problemlos ist[52], müssen bei digital vorliegenden Veröffentlichungen besondere Maßnahmen getroffen werden, um mögliche Manipulationen und Verfälschungen zu verhindern – oder um diese zumindest nachweisen zu können. Zwei grundsätzliche Anforderungen lassen sich unterscheiden:

- Sicherung des Dokumentenservers
- Sicherung der einzelnen Dokumente

Der mit „*keine* Netzverbindung zum Server" wohl sicherste Ansatz ist in der Praxis nicht wirklich realisier- bzw. benutzbar und so sollte es zumindest eine Trennung in öffentlich zugänglichen Dokumentenserver und besonders geschützten, nicht oder stark eingeschränkt an Netzwerke angebundenen Archivserver geben. Die Deutsche Initiative für Netzwerkinformationen (DINI[53]) empfiehlt folgende Regelungen zur Sicherung des Zugangs:

- Administration des Servers ausschließlich durch einen autorisierten Personenkreis
- Nachweis der Administrationsaktivitäten
- physischer und softwaremäßiger Zugriffsschutz
- Registrierung und Kontrolle der Zugriffe
- regelmäßige Datensicherung und Konsistenzprüfung
- Sicherung der eindeutigen Identität des Dokumentenservers

Der letztgenannte Punkt läßt sich laut [DINI01] am ehesten erreichen, wenn der Server zertifiziert und in eine Public-Key-Infrastruktur (PKI) eingebunden ist.[54] Für den Leser muß erkennbar sein, daß die angeforderten Daten tatsächlich vom richtigen Rechner kommen und weder vor, noch während der Übertragung verändert wurden. Gleiches gilt natürlich auch für die Übermittlung einer Publikation an die Bibliothek und so bietet sich das standardisierte SSL-Protokoll[55] für die Verschlüsselung der Kommunikation an: Client, also der jeweilige Nutzer, *und* Server sollten dabei über gültige Signaturen von anerkannten Zertifizierungsstellen verfügen und sich so dem Partner gegenüber ausweisen können. Das gängigste Verfahren für eine SSL-Verbindung besteht laut [KS00] jedoch darin, „dass nur der Server eine gültige Signatur [...] besitzt. Der Client bleibt anonym und eine sichere Kommunikation kommt dann zustande, wenn der Client die vom Server übertragene Signatur akzeptiert." Für den Upload von Examensarbeiten auf einen Publikationsserver reicht dieser Kompromiß[56] meist aus, da vor der Bereitstellung des Dokuments im Internet in jedem Fall ein Vergleich

[52] von den Papierexemplaren lassen sich bei Bedarf z.B. Sicherheitsfilme oder Xerokopien herstellen
[53] http://www.dini.de
[54] Das Für- und Wider einer PKI an deutschen Universitäten wird einführend z.B. unter http://dissertationen.hu-berlin.de/e_rzm/24/bell-michael-2003-04-17/HTML/21.php dargestellt.
[55] Secure Sockets Layer
[56] für Gründe und Auswirkungen siehe ebenfalls [KS00]

mit dem zugehörigen Printexemplar erfolgt und spätestens dann Unstimmigkeiten auffallen würden. Außerdem ist es möglich, daß der Autor sein Werk vor dem Versand selbst verschlüsselt oder ‚manuell' mit einer Signatur versieht – entsprechende Absprachen mit der Bibliothek vorausgesetzt.
Wurde die Publikation dann erfolgreich übertragen, ist seitens der archivierenden Institution sicherzustellen, daß die volle Originalität auch langfristig bewahrt wird.

> „Die Authentizität und Integrität des digitalen Dokumentes muss stets nachweisbar sein. Das heißt, es muss nachvollziehbar bewiesen werden können, dass das veröffentlichte Dokument seit dem Tag der Bereitstellung nicht mehr verändert wurde, weder vom Autor, noch vom Systemadministrator oder gar einem Dritten." [DINI01]

Wenn dennoch Änderungen notwendig wurden, muß das geänderte Dokument als neue und ebenfalls signierte Version abgelegt werden. Wichtig hierbei ist ein Zeitstempeldienst, der die Signaturen nachprüfbar ihrem Erstellungsdatum zuordnet.[57] Aber was bedeutet Signatur, Zertifikat und Zeitstempel eigentlich? Das deutsche Signaturgesetz (SigG) definiert diese Begriffe wie folgt (auszugsweise zitiert nach [Scholze02]):

§1
(1) Eine digitale Signatur im Sinne dieses Gesetzes ist ein mit einem privaten Signaturschlüssel erzeugtes Siegel zu digitalen Daten, das mit Hilfe eines zugehörigen öffentlichen Schlüssels, der mit einem Signaturschlüssel-Zertifikat einer Zertifizierungsstelle oder der Behörde nach §3 versehen ist, den Inhaber des Signaturschlüssels und die Unverfälschtheit der Daten erkennen lässt.
(2) Eine Zertifizierungsstelle im Sinne dieses Gesetzes ist eine natürliche oder juristische Person, die die Zuordnung von öffentlichen Signaturschlüsseln zu natürlichen Personen bescheinigt und dafür eine Genehmigung gemäß §4 besitzt.
(3) Ein Zertifikat im Sinne dieses Gesetzes ist eine mit einer digitalen Signatur versehene digitale Bescheinigung über die Zuordnung eines öffentlichen Signaturschlüssels zu einer natürlichen Person (Signaturschlüssel-Zertifikat) oder eine gesonderte digitale Bescheinigung, die unter eindeutiger Bezugnahme auf ein Signaturschlüssel-Zertifikat weitere Angaben enthält (Attribut-Zertifikat).
(4) Ein Zeitstempel im Sinne dieses Gesetzes ist eine mit einer digitalen Signatur versehene digitale Bescheinigung einer Zertifizierungsstelle, dass ihr bestimmte digitale Daten zu einem bestimmten Zeitpunkt vorgelegen haben. […]

§7
Das Signaturschlüssel-Zertifikat muss folgende Angaben enthalten:
- den Namen des Signaturschlüssel-Inhabers, der im Falle einer Verwechslungsmöglichkeit mit einem Zusatz zu versehen ist, oder ein dem Signaturschlüssel-Inhaber zugeordnetes unverwechselbares Pseudonym, das als solches kenntlich sein muss,
- den zugeordneten öffentlichen Signaturschlüssel,
- die Bezeichnung der Algorithmen, mit denen der öffentliche Schlüssel des Signaturschlüssel-Inhabers sowie der öffentliche Schlüssel der Zertifizierungsstelle benutzt werden kann,
- die laufende Nummer des Zertifikates,
- Beginn und Ende der Gültigkeit des Zertifikates,
- den Namen der Zertifizierungsstelle und
- Angaben, ob die Nutzung des Signaturschlüssels auf bestimmte Anwendungen nach Art und Umfang beschränkt ist.

[57] Digital Timestamping and the Evaluation of Security Primitives: http://www.dice.ucl.ac.be/crypto/TIMESEC.html Secure Time/Date stamping in a Public Key Infrastructure: http://www.surety.com

Zur Erzeugung einer digitalen Signatur wird mit Hilfe einer mathematischen „Hashfunktion“ aus dem zu ‚unterzeichnenden‘ Dokument zunächst eine Zeichenkette fester Länge (z.B. 160 Bit) generiert, die entsprechend „Hashwert“ (Message Digest) genannt wird – oder auch „Fingerabdruck“, da zwei unterschiedliche Nachrichten nie (bzw. nur in sehr sehr seltenen Fällen, siehe unten) denselben Hashwert haben können. Dieser Fingerprint wird anschließend mit dem „Private Key“ verschlüsselt und abhängig vom Anwendungsfall entweder gemeinsam mit dem Originaldokument verschickt oder auf dem Server abgelegt. Hash-Funktionen bieten dabei folgende Eigenschaften (siehe [KS00]):

- Sie sind mit kurzer Bearbeitungszeit anwendbar.
- Die Hashfunktion ist eine sog. „Einweg-Funktion“, d.h. aus dem Hashwert kann das Ursprungsdokument nicht zurückberechnet werden.
- Nur mit extrem hohem Aufwand können zwei Nachrichten erzeugt werden, die denselben Hashwert besitzen.

Zur Sicherstellung der Unversehrtheit eines Dokuments genügt es also, anstelle der gesamten Datei nur einen Teil zu verschlüsseln. Soll die Authentizität später überprüft werden, ist z.B. wie folgt vorzugehen:

- Zuerst entschlüsselt der Empfänger den mitgelieferten Hashwert mit Hilfe des „Public Keys“.
- Anschließend wird der Hashwert des Originaldokuments neu berechnet und mit dem Entschlüsselten verglichen.
- Stimmen beide überein, ist die Datei authentisch, andernfalls wurde entweder die digitale Signatur verfälscht, oder das Dokument.

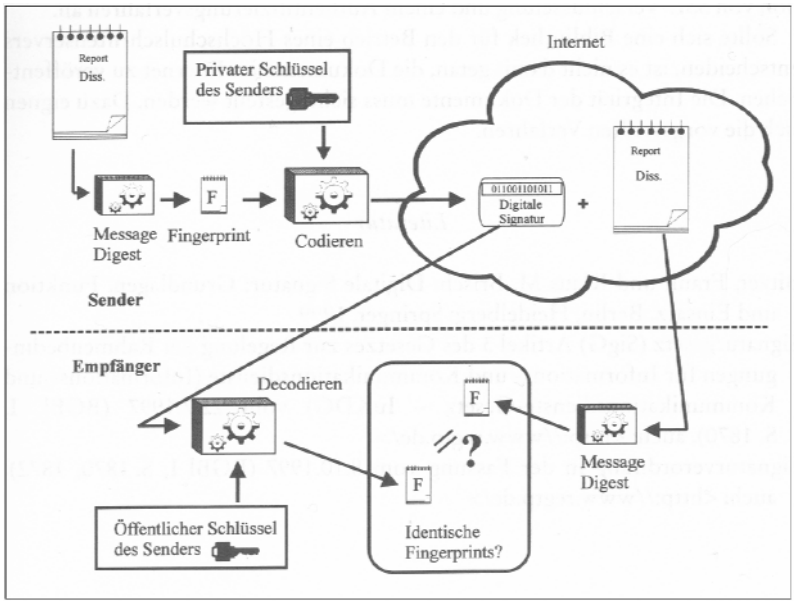


Abbildung 8: Digitale Signatur

Mit der Prüfung der digitalen Signatur kann zwar die Authentizität des gesendeten Dokuments festgestellt werden, nicht jedoch, ob der Absender tatsächlich existiert bzw. die Person ist, für die sie sich ausgibt. Hierfür sind Schlüsselzertifikate notwendig, ausgestellt durch anerkannte Trust Center, die eine zweifelsfreie Feststellung der Identität des Verfassers ermöglichen. Gesetz und entsprechende Verordnung zur digitalen Signatur sind bereits 1997 in Kraft getreten[58]. Dennoch existieren bisher nur wenige, im Sinne des SigG von der

[58] siehe auch http://www.regtp.de

Regulierungsbehörde für Telekommunikation und Post zertifizierte Trust Center, die Schlüsselpaare für die Erzeugung gesetzeskonformer Signaturen ausstellen. Die Nachfrage ist, trotz der gebotenen hohen Fälschungssicherheit, hinter den Erwartungen zurückgeblieben – und dies im universitären Umfeld nicht zuletzt wegen der zusätzlichen finanziellen Belastung. Die Zuteilung einer digitalen Unterschrift durch eine kommerzielle Zertifizierungsstelle kann – inklusive einer jährlichen Gebühr – durchaus einige hundert Euro kosten und so kann beispielsweise das Signieren einer Dissertation durch den Doktoranden nur auf freiwilliger Basis geschehen.
Zudem sind Zertifikate nur von befristeter Gültigkeit. Man muß sie fortlaufend erneuern, will man die zugehörigen Dokumente langfristig archivieren. Sollte die Schlüssellänge bzw. der verwendete Algorithmus z.B. aufgrund schnellerer Hardware nicht mehr genügend Sicherheit bieten oder das ursprüngliche Trustcenter irgendwann nicht mehr existieren, müssen die Daten mit neu zertifizierten, technisch sicheren Signaturen nachsigniert werden. Außerdem ergibt sich im Bereich der „dynamischen" oder „lebenden" Publikationen die Notwendigkeit, „jede Änderung zu dokumentieren und den resultierenden Zustand zu signieren (Version), um so eine lückenlose, authentische und integre Historie des Dokuments zu erhalten", wie von [Scholze02] festgestellt wird.
Das Sichern von Integrität und Authentizität aller Publikationen eines Dokumentenservers mit Hilfe digitaler Signaturen kann somit recht aufwendig – und im Falle der Zertifizierung durch kommerzielle Anbieter auch schnell recht teuer werden. Eine Alternative wäre hier der Einsatz der freien und auf vielen Plattformen verfügbaren Verschlüsselungssoftware PGP (Pretty Good Privacy[59]), die es neben der Signaturbildung beispielsweise auch erlaubt, Dateien zu komprimieren, als Ganzes zu verschlüsseln und anschließend base64-codiert via Mail zu verschicken. Auch hierbei kommen asymmetrische Verfahren mit öffentlichen und privaten Schlüsseln zum Einsatz: ein Autor könnte seine Dissertation z.B. mit dem Public Key der Bibliothek signieren oder eben vollständig verschlüsseln und auch ohne SSL-Verbindung sicher übertragen. Nur wenn die Entschlüsselung mit dem Private Key des Empfängers glückt, können Manipulationen ausgeschlossen werden. Einziger Nachteil der Nutzung von Software wie PGP: die damit erzeugten digitalen Signaturen wurden von keinem vertrauenswürdigen, signaturgesetzkonformen Trust Center zertifiziert – die Zuordnung eines bestimmten öffentlichen Schlüssels zu einer bestimmten natürlichen Person und der Nachweis, daß die Daten tatsächlich von dieser Person „unterzeichnet" wurden, wäre vor Gericht nicht möglich.

Der Einsatz rechtssicherer digitaler Signaturen hat also entscheidende Vorteile, ist für alle Beteiligten aber auch mit einem gewissen Mehraufwand (zeitlich und finanziell) verbunden und so müssen in einer Institution zunächst entsprechende Voraussetzungen für den Aufbau von PKIs geschaffen werden. Der Prototyp des hier implementierten Dokumentenservers wird die Integrität der gespeicherten Publikationen daher zunächst ‚lediglich' mit Hilfe von Hashwert-Berechnungen sicherstellen[60] – ein Kompromiß zwischen Aufwand und Nutzen, der nicht unbedingt schlecht sein muß und auch in vielen anderen existierenden Systemen zu finden ist: sofort nach Erhalt und Kontrolle eines eingegangenen Dokuments wird der zugehörige *Fingerprint* bestimmt und sicher verwahrt. Greift ein Nutzer später auf die separat archivierte Publikation zu, wird der Hashwert erneut berechnet und mit dem Hinterlegten verglichen. Hat sich inzwischen auch nur ein Byte im Dokument geändert, stimmen die Fingerabdrücke nicht mehr überein – der Zugriff kann nun entweder verwehrt oder mit einem Warnhinweis versehen werden (Einzelheiten dazu dann im praktischen Teil).

[59] http://www.pgpi.org
[60] für die Erzeugung der Hashwerte bietet sich z.B. der Message Digest 5 (MD5)-Algorithmus an

4.3 Beständige Identifikatoren

Ein Vorteil elektronischer Publikationen ist die im Vergleich zu Printmedien schnellere und einfachere Möglichkeit der Verbreitung über das Internet. Voraussetzung dafür ist jedoch die Festlegung des „digitalen Standorts“, über den das Dokument adressiert wird. Eine solche „elektronische Adresse“ ist zumeist als sogenannter „Hyperlink“ in Form eines Uniform Resource Locators realisiert. URLs sind weltweit eindeutig und werden deshalb sowohl für den Zugriff auf das Dokument, als auch als Identifier für das Zitieren von Publikationen verwendet. Problematisch wird es, wenn sich der elektronische Standort ändert: alle Referenzen auf das Dokument sind dann nicht mehr benutzbar – Hypertexte verlieren ihre Konsistenz und die Funktionalität der Navigation, wenn die Links nicht stabil sind.

> „[…] Damit die Nachteile von standortgebundenen Verweisen kompensiert werden können, existieren verschiedene Methoden, um eine zeitliche Stabilität der Verweise zu gewährleisten wie z.B. durch
> - URLs, aus denen der Server dynamisch den Speicherort ermittelt in Form von CGI-Skripten oder Datenbanken,
> - eine entsprechende Konfiguration von Web-Servern, welche die Umleitung von alten zu aktuellen Adressen in Form von "redirects" oder "aliases" erlaubt oder
> - die Anwendung entsprechender Vergabealgorithmen für die Bildung einer beständigen URL-Namensstruktur,
> - die Durchführung periodischer URL-Checks durch den Informationsprovider.“

Diese in [PersID] beschriebenen Methoden sind allerdings nur für bestimmte Situationen kurz- bis mittelfristige Lösungen. Bei einer Modifikation der Systemumgebung (z.B. Software- oder Datenbankwechsel) können sich die Adressierungsarten derart ändern, daß darauf auch mit dynamischen Redirects o.ä. vielleicht nicht mehr reagiert werden kann. URLs können außerdem temporär durch Netzwerkfehler und instabile Server-Verbindungen ausfallen. Informationen darüber, ob eventuell irgendwo Kopien des Dokuments existieren, auf die der Nutzer alternativ zugreifen kann, wären in diesem Fall wünschenswert, liegen jedoch außerhalb des Leistungsumfanges von Uniform Resource Names. Und die vorgeschlagenen URL-Checks sind letztlich nur im Zusammenhang mit einer konsequenten URL-Pflege sinnvoll. Der Arbeitsaufwand ist recht hoch, will man die Links in allen Nachweissystemen, wie Katalogen, Bibliographien oder Portalen, konsistent halten.

URLs sind für eine sichere formale Identifizierung elektronischer Dokumente also ungeeignet und so hat man schon frühzeitig speziell in Hypertextsystemen versucht, die Link-Zielpunkte von der direkten Referenz auf den Standort der Ressource zu trennen. Exemplarisch sei auf den ISO-Standard *HyTime – Hypermedia/Time-based Structuring Language*[61] verwiesen. Derartige Ansätze haben sich jedoch nicht durchsetzen können – und auch Standards, wie z.B. XLink[62] oder HLink[63], die den semantischen Verweis- und Funktionsumfang von HTML-Links erweitern und so eine standortunabhängige Adressierung zumindest teilweise ermöglichen, haben keine Breitenwirkung erzielt, da sie sehr anwendungsabhängig sind.

[61] http://xml.coverpages.org/hytime.html
[62] XML Linking Language, http://www.w3.org/TR/xlink
[63] Link recognition for the XHTML Family, http://www.w3.org/TR/hlink

Laut [PersID] wird ein Identifier benötigt,

„der in der Lage ist,
- ein digitales Objekt dauerhaft zu adressieren,
- gleichzeitig auf mehrere Speicherorte zu verweisen,
- ein digitales Objekt als Informationseinheit weltweit eindeutig, aber auch
- einzelne Teile zuverlässig zu identifizieren.

Daraus resultieren zwei Bereiche, die miteinander kombiniert werden müssen:
- die Identifizierung und
- Adressierung von digitalen Objekten."

Diese Anforderungen werden von *Persistent Identifiers* (PIs) erfüllt. Deren Grundidee, so [PersID] weiter, „[...] ist die strikte Trennung von Identifikation der Objekte durch eine eindeutige Zeichenkette und ihrer Standortreferenz. PIs werden angewendet, indem sie anstelle von URLs als Identifikatoren angegeben und anschließend über einen zwischengeschalteten Mechanismus (Resolving) in die zugehörigen URL(s) aufgelöst werden."

Da die URLs nur an einer einzigen Stelle – im Persistent Identifier-Dienst – (automatisiert) gepflegt werden, reduziert sich der Aufwand für die Aktualisierung der Verweise. Dies erinnert stark an die im Internet übliche Zuordnung eines Rechnernamens zu einer bestimmten IP-Adresse mit Hilfe des Domain Name Systems: ändert sich die IP, muß nur der DNS-Eintrag angepaßt werden – die Ressource bleibt weiterhin über den global eindeutigen Bezeichner zugänglich. In Analogie dazu verschleiert der PI die konkrete URL: der Resolving-Mechanismus analysiert den Identifier, prüft die Verfügbarkeit und gegebenenfalls die Integrität des zugehörigen Dokuments (beispielsweise via Hashwert-Berechnung, siehe letzter Abschnitt) und leitet zum entsprechenden Ziel weiter. Eine digitale Publikation wird dadurch zuverlässig zitierbar; vorausgesetzt natürlich, die Existenz des Dienstes ist auch langfristig sichergestellt.

Systeme, die beständige Identifikatoren bieten, gibt es viele.[64] Beispielhaft seien die insbesondere von großen Online-Verlagen zumeist kommerziell genutzten DOIs (Digital Object Identifiers) und die Entwicklungen des CNRI[65] (Handle System) bzw. des OCLC[66] (PURL) genannt, die hier allerdings nicht näher untersucht werden sollen. [PersID] und die CARMEN-AP4-Linkliste[67] geben einen guten Überblick über existierende Anbieter und die verschiedenen Möglichkeiten der Auflösung von PIs (z.B. mit Hilfe von Proxy-Servern, speziellen Browser-Plugins, Erweiterungen des DNS-Standards[68] u.ä.).

Im bibliothekarischen und nicht-kommerziellen Umfeld ist vor allem der *Uniform Resource Name* weit verbreitet und da dieser auch für die Übergabe von Online-Hochschulschriften an die mit der Langzeitarchivierung beauftragte Deutsche Bibliothek von Bedeutung ist, soll er nachfolgend etwas genauer vorgestellt werden.[69]

---

[64] Im Rahmen einer Belegarbeit hat der Autor ein solches System auch bereits selbst implementiert und insbesondere Erfahrungen mit „zertifizierten Links" sammeln können. [ON01]

[65] Corporation for National Research Initiatives - Handle-System: http://www.handle.net

[66] Online Computer Library Center - Persistent Uniform Resource Locator: http://www.purl.org

[67] http://www.bis.uni-oldenburg.de/carmen_ap4/link_collection.html - siehe auch Kapitel 5.2

[68] z.B. NAPTR: http://www.faqs.org/rfcs/rfc2915.html

[69] An dieser Stelle wird nur auf den grundlegenden Aufbau von URNs und deren Resolving eingegangen - die konkrete Umsetzung Der Deutschen Bibliothek findet sich im entsprechenden Kapitel)

Der URN[70] existiert seit 1992 und wird als Standard von der URN-Working Group der Internet Engineering Task Force[71] kontrolliert, die wiederum organisatorisch in die Internet Assigned Numbering Authority[72] eingegliedert ist. Erarbeitet und veröffentlicht wurde das System in Form von RFCs („Request for Comments“).[73]

RFC1737 stellt dabei folgende funktionale Anforderungen an URNs:

- ein URN soll standortunabhängig sein (global scope)
- ein URN muß weltweit einmalig sein (global uniqueness)
- ein URN muß beständig sein: er muß solange gültig sein, wie er explizit nicht gelöscht wird (persistence)
- ein URN muß so gestaltet sein, daß er nicht geändert werden muß, wenn die Anzahl der Ressourcen im Netz beträchtlich wächst (scalability)
- jedes URN-Schema muß so gestaltet sein, daß es leicht erweiterbar ist und ohne bestehende URNs zu tangieren (extensibility)
- in URNs müssen bestehende Ressourcenidentifikationssysteme (z.B. ISBN oder ISSN) integrierbar sein (legacy support)
- die URN-vergebenden Autoritäten sollen voneinander unabhängig sein: jede Institution bestimmt selbst, wem und wie sie URNs vergibt (independence)
- URNs müssen so gestaltet sein, daß ihnen aufgrund einer einheitlichen Vorgehensweise Adressen (URLs) zugeordnet werden können (resolution)

RFC2141 wiederum definiert die Syntax eines Uniform Resource Names:

URN:NID:SNID:NSS bzw. URN:NID:SNID:[Naming Authority]:[Opaque String]

URNs bestehen aus mehreren hierarchisch aufgebauten Teilbereichen. Dazu zählt der Namensraum (*Namespace Identifier*, NID), der den Standard bzw. die Normbezeichnung angibt (z.B. INET, ISBN, ...) und welcher sich aus weiteren Unternamensräumen (*Subnamespace Identifier*, SNID) zusammensetzen kann. Außerdem enthalten ist der *Namespace Specific String* (NSS), der den angegebenen Standard genauer charakterisiert und – erneut durch einen Doppelpunkt getrennt – zumeist die *Naming Authority* (also die für die URN-Vergabe verantwortliche Stelle) und einen *Opaque String* enthält. Dieser String kann aus beliebigen Zeichen, Ziffern und Buchstaben bestehen und spezifiziert letztlich das eigentliche Objekt. Ein gültiger URN könnte somit beispielsweise so aussehen:[74]

urn:inet:dstc.edu.au:dstc/tr017

Doch wie erfolgt die Zuordnung eines derartigen URN zu einem für den Zugriff notwendigen Uniform Resource Locator? Die RFCs 3401 bis 3404 (und weitere) erklären die möglichen Resolving-Mechanismen.

---

[70] URN war früher die Abkürzung für *Uniform Resource Number* - in der deutschsprachigen Literatur findet sich daher recht häufig auch der Ausdruck „*die* URN“.

[71] http://www.ietf.org

[72] http://www.iana.org

[73] URLs der hier genannten RFCs: http://www.faqs.org/rfcs/“rfc“+Nummer+“.html“, also z.B http://www.faqs.org/rfcs/rfc1737.html

[74] entnommen aus http://www.dstc.edu.au/RDU/TURNIP/burns.html, Groß-/Kleinschreibung ist normalerweise nicht relevant

[Pay99] faßt die grundsätzlichen Vorgänge bei der Auflösung eines kompletten URN anhand nebenstehender Abbildung wie folgt zusammen:

1. Der Client will eine Ressource mit einer bestimmten URN, aber unbekannter URL. Der Client sendet an einen Resolver Discovery Service (RDS) den Namespace Identifier (NID). Dieser hat gespeichert, welcher URN-Resolver für welche NIDs zuständig ist.[75]
2. Der RDS meldet dem Client die Adresse des zuständigen URN-Resolvers.
3. Der Client sendet an den URN-Resolver den Namespace Specific String (NSS). Der URN-Resolver bestimmt die URL(s) für die zugehörige Ressource.
4. Der URN-Resolver sendet diese URL an den Client.
5. Der Client fordert beim für diese URL zuständigen Server die Ressource an.
6. Der Server sendet an den Client die Ressource.

   Die Vorgänge 1 und 2 können auch entfallen, wenn der URN-Resolver bekannt ist (z.B. weil seine Adresse im Namespace Specific String genannt wird).

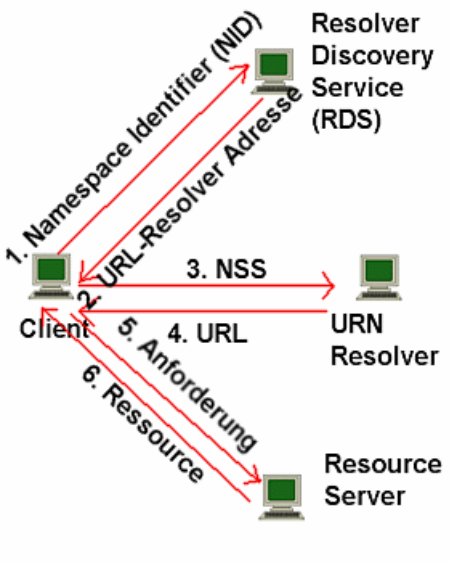


Abbildung 9: URN Resolving

Optimal wäre es, wenn ein Nutzer den eindeutigen Uniform Resource Name des gewünschten Dokuments ohne zusätzliche Angaben direkt in die Adreßzeile seines Browser eintragen könnte und beim Druck auf RETURN automatisch und ohne Umwege zum konkreten Server weitergeleitet werden würde. Die Realität sieht allerdings anders aus, da eine solche Funktionalität (derzeit) weder von existierenden Browsern, noch z.B. vom DNS geboten wird. Gegenwärtig muß man sich mit technischen Brückenlösungen, wie zwischengeschalteten Proxy-Servern oder lokal zu installierenden Plugins, weiterhelfen ... erst mit der bereits angesprochenen Erweiterung des Domain Name Systems um eine *Naming Authority PointR*-Komponente und einer globalen Akzeptanz des vielversprechenden Ansatzes könnte dieser zusätzliche Aufwand vermieden werden. Ein Kompromiß ist die Integration von bereits angewendeten Namensräumen bzw. Nummernsystemen in das URN-Schema und dessen Einbettung in gängige Protokolle und existierende Standards (HTTP, URL, ...). Für die Referenzierung einer URN kann dann z.B. einfach eine URL der Form

http://<resolver-adresse>/<Pfad zum Resolver-Script>/<Resolver-Script>?urn=urn:inet:dstc.edu.au:dstc/tr017[76]

benutzt werden, um (weitestgehend) transparent und ohne Plugins/Proxyserver auf die zugehörige Publikation zugreifen zu können. Fällt der Resolving-Server allerdings längerfristig aus, muß die URL in allen Katalogen und Nachweissystemen geändert werden – der Vorteil von URNs als „beständige Identifikatoren“ wird somit zunichte gemacht. Hier ist seitens der Anbieter also für eine hohe Verfügbarkeit und Ausfallsicherheit zu sorgen ... ... und langfristig gesehen eine „[...] aktive Mitarbeit an der Etablierung von Standards [wie

[75] Die Namensraum-Bezeichner und zuständigen Naming Authorities sind für die automatische Umsetzung URN→URL von entscheidender Bedeutung. Wie deren Registrierung allerdings genau zu erfolgen hat, soll an dieser Stelle ausgespart werden - entsprechende Informationen finden sich unter anderem in den RFCs 2288, 3187, 3188 und 3406.

[76] Doppelpunkt, Schrägstrich und einige weitere Zeichen sind innerhalb von URL-Parametern nicht zulässig und müßten eigentlich codiert angegeben werden („:“ → „%3A“ und „/“ → „%2F”).

dem DNS-NAPTR] zur technischen Realisierung eines globalen Resolving-Mechanismus auf Basis einer einheitlichen Nummernstruktur [...]“ anzustreben, wie auch in [PersID] festgestellt wird.

Trotz der Schwächen existierender Auflösungsmechanismen und einer eher verhaltenen internationalen Zusammenarbeit bleibt festzuhalten, daß Persistent Identifiers, z.B. in Form von URNs, für die Referenzierung elektronischer Dokumente weit besser geeignet sind, als die im Internet typischen URLs. Serverwechsel-bedingte Änderungen des digitalen Standorts wirken sich nicht auf die Zitierbarkeit aus – durch Aktualisierungen der Resolving-Datenbank und durch die Möglichkeit der Zuweisung eines URN zu mehreren alternativen URLs bleibt die Online-Publikation dauerhaft zugriffsfähig. Für den hier konzipierten Dokumentenserver ist die (optionale) Nutzung von PIs daher Pflicht. Als Grundlage sollen spezifische, von Der Deutschen Bibliothek administrierte Uniform Resource Names dienen und so werden diese einführenden Überlegungen zum allgemeinen Aufbau von URNs im Kapitel 5.2 nochmals aufgegriffen und konkretisiert.

4.4 Open Archival Information System

Wie ganz zu Beginn vorliegender Arbeit motivierend erwähnt wurde, kommen die Befürchtungen, ja gar Ängste bezüglich eines drohenden Datenverlustes nicht von ungefähr: die Vergangenheit hat gezeigt, wie schnell bedeutende wissenschaftliche Dokumente verloren gehen können – beispielhaft seien hier die Probleme der NASA ins Gedächtnis gebracht. Wenn man bedenkt, daß große Datenmengen vielleicht erst in Jahrzehnten oder Jahrhunderten auswertbar sind und auch erst dann eventuell überraschende Erkenntnisse liefern können, „dann begreift man, welche Bedeutung der langfristigen Aufbewahrung wichtiger Daten bei gleichzeitiger Wahrung ihrer Lesbarkeit zukommt.“ [Lupp00]
Und Karl-Ernst Lupprian führt weiter aus:

> „Die Probleme der langfristigen oder gar der unbefristeten Aufbewahrung digitaler Daten und ihrer ständigen Nutzbarmachung sind so schwerwiegend, dass sie nur in internationaler Kooperation gelöst werden können. Wer glaubt, diese immense Arbeit allein leisten zu können, wird über kurz oder lang vor den Kosten kapitulieren.“

Ein solches Modell eines offenen Archivsystems ist das mit OAIS abgekürzte, aber nicht mit dem bereits vorgestellten OAI-Standard zu verwechselnde, *Open Archival Information System*. Es wurde vom „Consultative Committee for Space Data Systems“ (CCSDS[77]) ursprünglich (und aus bekannten Gründen) für die Belange der NASA entwickelt, ist aber auch auf Bibliotheken anwendbar und dient als konzeptionelle Grundlage für den Aufbau digitaler Sammlungen. Ein erster Entwurf wurde im Mai 1999 veröffentlicht – inzwischen liegt die Spezifikation als ISO-Standard 14721:2002 vor und ist Grundlage für viele verschiedene Projekte, wie z.B. PANDORA[78] oder NEDLIB[79].

[77] Mitglieder sind verschiedene Weltraumforschungszentren, siehe auch http://www.ccsds.org
[78] http://pandora.nla.gov.au
[79] http://www.kb.nl/coop/nedlib

Das OAIS-Referenzmodell beschreibt digitale Informationen als Objekte, die in Form von Paketen spezifische Bereiche eines Archivs durchlaufen.

> Es identifiziert die zentralen Funktionen und Abläufe, bietet eine Terminologie und ein Strukturkonzept für Metadaten, ist neutral gegenüber unterschiedlichen Archivierungstechniken (Migration, Emulation, u.ä.) und ermöglicht aufgrund seiner Containerstruktur eine dezentrale Implementierung ...

... so eine kurze Zusammenfassung, wie sie in ähnlicher Form auch in [DT02] zu finden ist. Was das allerdings genau heißt, soll nachfolgend nochmals etwas ausführlicher geklärt werden, zumal das Modell Vorbild für ganz aktuelle Systeme ist[80] – und dies wegen seiner weiten Verbreitung und Akzeptanz möglichst auch für den eigenen Prototypen gelten sollte.

Das ISO-Referenzmodell eines OAIS

- ist strikt logisch strukturiert und damit unabhängig von jeder Implementation,
- lässt sich mit der Sprache UML (Unified Modeling Language) grafisch darstellen, und zwar von der Ebene einfachster Übersichten bis zum komplexen Detailplan,
- kann sowohl aus der Sicht der Funktionalität als auch des Informationsflusses dargestellt werden,
- betrachtet den Informationsfluss als Abfolge von aufgabenorientierten Datenpaketen,
- ist primär für die Verarbeitung digitaler Informationen gedacht, kann jedoch auch auf nichtdigitale Informationen angewendet werden und erlaubt damit die Verarbeitung hybrider Unterlagen
  [Lupp00]
- \+ spezifiziert die Schnittstellen der Funktionseinheiten zueinander und die Berührungspunkte des Archivsystems zu seiner Außenwelt.

Zentrale Einheiten des Modells sind Aufnahme, Speicherung, Datenmanagement, Langzeitarchivierung, Verwaltung und Zugriff, deren Zusammenspiel Abbildung 10 verdeutlicht:

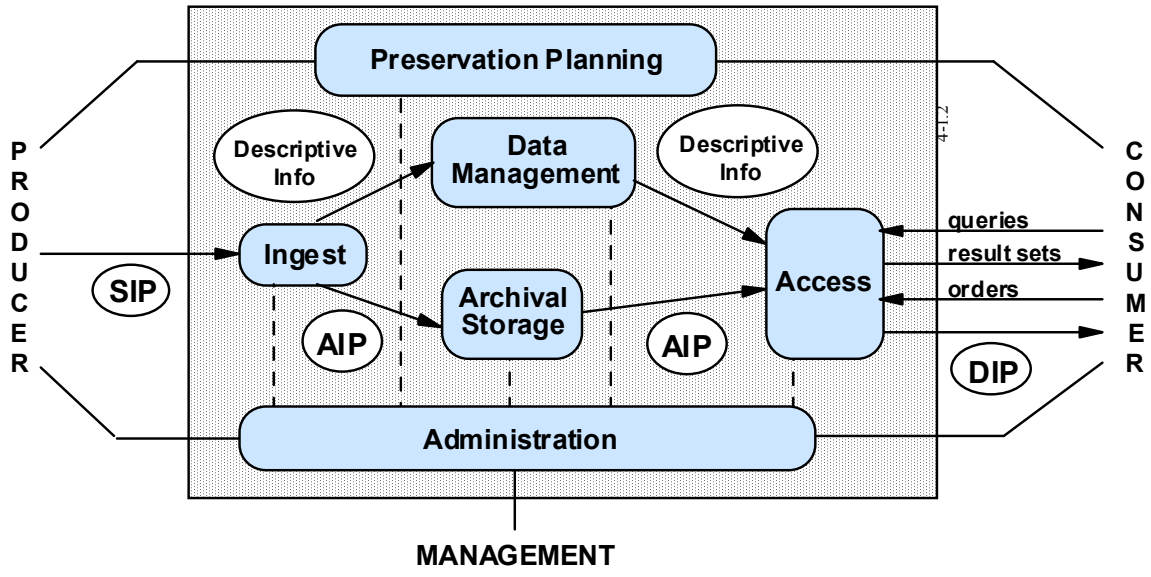


Abbildung 10: OAIS Referenzmodell

[80] dazu gehört auch DSpace - siehe Kapitel 6.2

Das OAIS fungiert als Bindeglied zwischen Produzenten und Konsumenten. Es nimmt die Informationen des Produzenten entgegen, bereitet diese für eine langfristige Speicherung auf und stellt Interfaces für den Zugriff und ein Retrieval bereit. Vereinfacht ausgedrückt nimmt es also gleichzeitig den Platz von Archiv, Bibliothek und Verlag ein.
Da die Informationen dabei allerdings nicht als beliebig lange Datenströme übernommen und nutzbar gemacht werden können, müssen sie in handhabbare Pakete aufgeteilt werden. Die eigentlichen Daten (*Content Information*) werden zusammen mit für die Erhaltung des Informationsinhalts notwendigen Angaben (*Preservation Description Information*, wie z.B. Codierung, Ursprungsplattform, Erstellungssoftware, ...) in eine *Packaging Information*-Hülle eingebettet. Zusätzlich dazu wird eine Beschreibung zum Paket selbst gespeichert, um nachvollziehen zu können, von wem es kommt, wie es technisch aufgebaut ist und in welcher Beziehung es zu anderen Paketen steht.

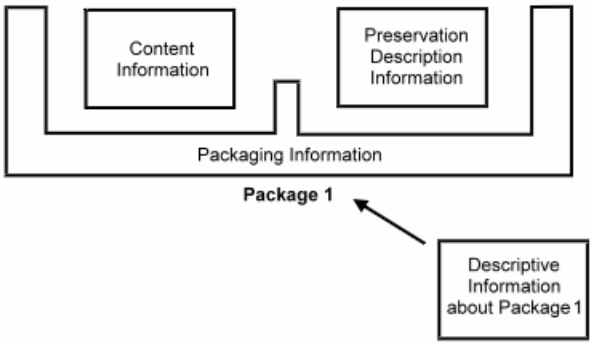


Abbildung 11: OAIS Information Package

Der Weg eines Objekts durch ein OAIS-konformes Archiv läßt sich insgesamt wie folgt skizzieren: die vom Autor angelieferten Daten müssen zunächst mit beschreibenden Metadaten versehen und in die vom System vorgegebene Paket-Form gebracht werden (siehe oben). Dieses standardisierte *Submission Information Package* (SIP, Einlieferungsbehälter) erlaubt es dem Archiv, die eingehenden Pakete mit Hilfe einer definierten Anzahl spezifischer Methoden zu ‚entpacken' und auf Vollständigkeit und Unversehrtheit zu überprüfen. Zuständig dafür ist die Funktionseinheit *Ingest* (Aufnahme), die SIPs anschließend entsprechend den internen Regeln (z.B. für Datenformat, Dokumentation, u.ä.) und getrennt von ihren *Descriptive Infos* in *Archival Information Packages* (AIP, Archivbehälter) überführt. Das *Archival Storage*-Modul sorgt als eigentlicher Archivspeicher für Aufbewahrung und Erhalt dieser AIPs. Hier werden Backups angelegt, die Daten auf Veränderungen hin überwacht und Wiederherstellungsmechanismen für Notfälle zur Verfügung gestellt. Eine Schnittstelle zum *Data Management* ermöglicht die Zuordnung der gespeicherten Informationen zu den entsprechenden Metadaten, die bei Bedarf gemeinsam an die *Access*-Einheit herausgegeben werden. Diese wiederum nimmt Anfragen des Nutzers entgegen, prüft Zugriffsrechte und erzeugt *Dissemination Information Packages* (DIP, Auslieferungsbehälter), deren Interpretation durch den *Consumer* natürlich gewährleistet sein muß. Die *Administration* steuert, von außen kontrolliert durch das Management, die Gesamtabläufe im OAIS und muß neben Konfiguration von Hard- und Software vor allem die Einhaltung der vorgegebenen Archivierungsstandards sicherstellen. Und die sechste, als *Preservation Planning* bezeichnete und in diesem Zusammenhang wohl wichtigste Funktionseinheit ist letztlich für die dauerhafte Speicherung der Daten verantwortlich. Hierzu werden System und Umgebung kontinuierlich überwacht und gegebenenfalls Maßnahmen vorgeschlagen, um eine Zugänglichkeit auch auf lange Sicht zu gewährleisten. Empfehlungen zur Migration, zur Änderung von Standards oder Regeln, und zur Anpassung von Dienstleistungen an neue Anforderungen der Nutzer sind nur einige Beispiele dafür.

Zusätzlich zu diesen, auch z.B. in [Lupp00] beschriebenen und in der Originaldokumentation[81] natürlich viel umfassender spezifizierten Bestandteilen sind noch einige weitere Dienste in das Modell integriert, die jedoch von der konkreten Implementierung abhängig sind und in Abbildung 10 daher fehlen (zit. n. [Schmitt03]):

- *Operating system services:*
  Hierunter werden alle Dienste zusammengefaßt, die den dauerhaften Betrieb der gewählten Hardwareplattform gewährleisten und die Schnittstellen zwischen den Applikationen und der Plattform bereitstellen.
- *Network services:*
  Alle Dienste, die erforderlich sind, um verschiedenen Applikationen aus einem externen heterogenen Netzwerk (z.B. das Internet) den Zugang zum OAIS zu ermöglichen.
- *Security services:*
  Unter diesem Oberbegriff werden einerseits alle Dienste zusammengefaßt, die die Authentizität und Integrität der Daten und andererseits die Kontrolle über die Zugriffsrechte sicherstellen. Dies meint sowohl die Authentifizierung der Nutzer, als auch eine etwaige Kostenabrechnung für den Abruf von Inhalten.

Das hier vorgestellte Referenzmodell definiert ganz bewußt keine technischen Standards, sondern nur den prinzipiellen Aufbau eines offenen Archivsystems, um auch langfristig als Grundlage für reale Umsetzungen dienen zu können. Die Literatur und die vielen auf OAIS basierenden Projekte zeigen, daß ein breiter Konsens hinsichtlich der Einschätzung der maßgeblichen Bedeutung dieses Modells herrscht, daß seine praktische Adaption allerdings in den verschiedenen Institutionen in durchaus unterschiedlicher Form geschieht (vergl. z.B. [Tapp01]). Aber diese Flexibilität ist auch genau der Vorteil von OAIS: es liefert ein konzeptionelles, implementationsunabhängiges Grundgerüst, welches sich problemlos erweitern und an die lokalen Bedürfnisse, Möglichkeiten und Absichten anpassen läßt. So geschehen z.B. beim bereits kurz erwähnten NEDLIB-Projekt[82], wo man die Arbeit an einem ureigenen funktionalen Konzept sogar zugunsten von OAIS aufgegeben und in Kooperation mit acht europäischen Nationalbibliotheken, einem Nationalarchiv, zwei Softwarefirmen und drei wissenschaftlichen Verlagen den konkreten Entwurf eines „Depotsystems für Elektronische Publikationen“ (DSEP) geliefert hat. Einzelheiten dazu finden sich in [Lieg01] bzw. in den vielen Projektberichten, die über die NEDLIB-Homepage[83] zu beziehen sind. Hier seien nochmals die wichtigsten Ergebnisse zusammengefaßt, die in nicht unerheblichem Maße auch die Arbeit deutscher Institutionen (wie z.B. Der Deutschen Bibliothek, siehe Kapitel 5.2) beeinflußt haben:

- Experiment zur Untersuchung von Emulationsverfahren als Methode zur Sicherung der Langzeitverfügbarkeit
- Zusammenstellung spezifischer Metadatenelemente für Zwecke der Langzeiterhaltung
- Prognose zur Marktentwicklung elektronischer Publikationen
- Diskussion marktrelevanter Standards für die Entwicklung eines Depotsystems
- Glossar des Begriffsumfeldes „Langzeiterhaltung elektronischer Publikationen“
- Zusammenstellung von Richtlinien für den Aufbau eines Depotsystems

[81] http://www.ccsds.org/documents/650x0b1.pdf
[82] Networked European Deposit Library, Laufzeit Januar 1998 - Dezember 2000
[83] http://www.konbib.nl/nedlib

5 Archivierungsbestrebungen am Beispiel von Dissertationen

Die „Konzeption von Dokumentenservern für Digitale Bibliotheken im Hinblick auf Langzeitarchivierung und Retrieval“ ist nicht nur Titel, sondern auch erklärtes Ziel der vorliegenden Arbeit und so wurde in den bisherigen Kapiteln versucht, einen allgemeinen und möglichst umfassenden Überblick über die wichtigsten in diesem Zusammenhang relevanten Aspekte zu geben. Hauptaugenmerk lag dabei auf der Untersuchung von Dateiformaten und deren Eignung für Erstellung, Präsentation und langfristiger Speicherung wissenschaftlicher Texte. Auch die Möglichkeiten einer strukturellen Recherche wurden beleuchtet und es hat sich gezeigt, daß ein Umdenken und die Entwicklung einer neuen „Kultur des elektronischen Publizierens“ [DINI01] erforderlich ist: zwar stellt die Nutzung und Verfügbarmachung digitaler Dokumente über das Internet im Vergleich zur konventionellen Publikation über Printmedien eine verbesserte Form der wissenschaftlichen Kommunikation dar (siehe Kapitel 2); allerdings ist hierfür auch eine Anpassung der Arbeitsweisen aller Beteiligten notwendig. Das beginnt beim Autor, der durch gezielte Strukturierungen wesentlich zu einem qualitativ verbesserten Retrieval beitragen kann und verlangt von den Institutionen eine Neuordnung der Arbeitsgänge und der Verantwortlichkeiten. Auch – und vor allem – Universitäts-bibliotheken sind ‚betroffen‘: die Verbreitung der im Rahmen von Forschung, Lehre und Studium anfallenden „grauen Literatur“ (Diplomarbeiten, Habilitationsschriften, Aufsätze, Reports, Preprints, Tagungsberichte, Vorlesungsskripte u.ä.) kann durch den Buchhandel zumeist nicht sinnvoll ermöglicht werden und so müssen (besser: können/sollten) Hochschul-publikationsserver diese Aufgabe übernehmen.[84] *Können* bzw. *sollten*, weil die Bereitstellung und Archivierung von Online-Publikationen nicht nur Vorteile (z.B. Platz- und Kosten-einsparungen), sondern eben auch neuartige Probleme und spezifische Anforderungen an die Infrastruktur mit sich bringt, wie insbesondere Kapitel 4 deutlich gemacht hat.

Grundsätzlich sind die Bibliotheken aber gewillt, diese Schwierigkeiten – in Kooperation mit Informatikern und Rechenzentren – auf sich zu nehmen, können mangels entsprechender Erfahrungen und personeller Kapazitäten allerdings nicht auf Anhieb alle Textarten gleichermaßen gut unterstützen. Sie möchten zunächst „[...] für einen klar definierten Veröffentlichungsbereich Kompetenz auf dem Gebiet des elektronischen Publizierens [erwerben]“ [Leh97A] und so haben sie sich insbesondere die *Dissertationen* als erstes Erprobungsfeld für den Umgang mit den neuen Medien ausgesucht. An ihnen läßt sich die

> „[...] tiefgreifende Umwandlung der Kommunikation in der Wissenschaft vom Austausch gedruckter papierener Dokumente zu digitalen (‚elektronischen‘) Dateien [...] besonders gut studieren: Dissertationen sind aktuelle, aber in sich abgeschlossene, besonders aufwendige und komplexe Dokumente, die intern an der Hochschule wissenschaftlich bewertet werden, aber öffentlich auch langfristig verfügbar sein sollen.“ [HZ00]

Weitere Gründe für die Spezialisierung der Universitätsbibliotheken (und des geplanten Publikationsservers) auf diesen Dokumenttyp wurden bereits in der Einleitung angegeben; hier sollen die Vorteile elektronischer Dissertationen nochmals explizit herausgestellt – und auch rechtliche Aspekte einer Veröffentlichung im Internet untersucht werden. Am Ende dieses einführenden Kapitels wird außerdem kurz auf das DFG-Projekt „Dissertationen Online“ eingegangen, welches sich bereits vor ca. 6 Jahren zur Aufgabe gemacht hatte,

[84] Die kommerziellen Verlage sind (nicht zuletzt wegen der hohen Kosten konventionellen Publizierens) vorrangig an qualitativ besonders wertvollen Materialien mit guten Absatzchancen interessiert, wie in [DINI01] festgestellt wird.

> „[...] ein möglichst einheitliches Konzept der Erstellung, des juristisch korrekten Umgangs mit Dissertationen als Examensarbeiten, der elektronischen Archivierung und des Retrievals von Dissertationen [...] zusammen mit den jeweiligen Hochschulbibliotheken [...]“ zu entwerfen, praktisch zu erproben und soweit anzupassen, daß das Konzept übertragbar ist. [DFG97]

Die ersten Dissertationen, die allgemein für die Erlangung des Doktorgrades erforderlich sind, erschienen bereits Ende des 16. Jahrhunderts. Ein Ministerialerlaß im Jahre 1913 machte den Druck der Arbeiten in Deutschland dann zur Pflicht und obwohl dieser Druckzwang in den Kriegs- und Nachkriegszeiten mehrere Male aufgehoben wurde, existiert er noch heute. In einem Beschluß der Kultusministerkonferenz der Länder[85] von 1977 werden von den Doktoranden noch 150 Exemplare ihrer Dissertation im Fotodruck verlangt (vergl. auch [Degen97]), die aktuell gültige Promotionsordnung der Humanwissenschaftlichen Fakultät der Universität Potsdam schreibt die Abgabe von 30 gebundenen Exemplaren vor, sofern die Verbreitung nicht durch einen gewerblichen Verleger erfolgen soll.[86] Aber auch in diesem Fall sind immer noch 10 Pflichtexemplare nötig, um die begehrte Promotionsurkunde überhaupt erhalten zu können. Für die Autoren bedeutet dies enorme, teilweise sogar drei-/vierstellige Druckkosten und auch für die jeweils zuständige Universitätsbibliothek stellen die eher selten benötigten Mehrfachexemplare einen nicht unerheblichen Arbeits- und Platzaufwand dar, müssen die Dissertationen doch erschlossen, gelagert und mit anderen Einrichtungen getauscht werden. Aus diesen und vielen weiteren Gründen, wie z.B. den längeren Bearbeitungszeiten oder der Belastung des Bibliotheksetats durch zusätzlich notwendige Erwerbungsmaßnahmen bei Verlagsdissertationen, hat man den entsprechenden Paragraphen der Promotionsordnungen mehr und mehr mit einem neuen Absatz digitale Dissertationen betreffend versehen.[87] Für die Mathematisch-Naturwissenschaftliche-Fakultät der Universität Potsdam lautet dieser beispielsweise:

> „Als Veröffentlichung gilt auch die Übergabe von vier vollständigen Exemplaren, die auf altersbeständigem, holz- und säurefreiem Papier gedruckt und dauerhaft haltbar gebunden sind, und einer elektronischen Version, deren Dateiformat und Datenträger mit der Universitätsbibliothek abzustimmen sind. [...]“[88]

Bei einer Parallelveröffentlichung verringert sich die Anzahl der auf Kosten des Doktoranden abzugebenden Pflichtexemplare also meist deutlich und entsprechend bereitwillig wird dieses Angebot auch genutzt. Der Trend geht damit auch hier weg von ausschließlich gedruckt vorliegenden Publikationen, hin zu Online-Dissertationen, deren wichtigste Vorteile von [Fritz99] wie folgt zusammengefaßt werden:

1. die Anzahl der abzugebenden Druckexemplare kann minimiert werden (oder, wie z.B. an der Virginia State University (Virginia Tech), vollkommen entfallen[89])
2. das aufwendige Tauschverfahren der Universitätsbibliotheken fällt weg und es kann Regalfläche eingespart werden

---

[85] www.kmk.org

[86] http://www.uni-potsdam.de/u/dekanat_philfak2/promotionsordnung/index.html

[87] In einer Neufassung des Beschlusses der KMK vom 30.10.1997 heißt es, daß eine Verbreitung der Arbeit zukünftig auch „durch die Ablieferung einer elektronischen Version, deren Datenformat und deren Datenträger mit der Hochschulbibliothek abzustimmen sind“, erlaubt ist (z.B. http://www.ub.uni-dortmund.de/Eldorado/kmk.html).

[88] http://www.uni-potsdam.de/u/ambek/ambek400.htm

[89] Dialog von [HZ00] mit E. Fox auf der IuK 99:
Frage: Wie werden bei Ihnen in VirginaTech die gedruckten Exemplare behandelt?
E. Fox: Das Einreichen von gedruckten Dissertationen ist bei uns seit zwei Jahren verboten.

3. die Erschließung der Dissertationen kann ohne großen Arbeitsaufwand in vielfältiger Art und Weise erfolgen, auch Volltextrecherchen werden möglich
4. lange Wartezeiten bei Fernleihbestellungen fallen weg, statt dessen kann die Dissertation international in Sekunden gefunden und abgerufen werden
5. das Interesse an den Dissertationen wird durch die automatisch protokollierten Zugriffe auf das Online-Dokument meßbar
6. der weltweite Bekanntheitsgrad und die weltweite Verbreitung von Forschungsergebnissen des Fachbereichs bzw. der Universität werden wesentlich verbessert
7. Forschungsergebnisse können in multimedialen Formaten dargestellt werden

Das Internet kann also genutzt werden, um die Dissertationen an den Hochschulen, an denen sie entstanden sind, verteilt zu archivieren und über das Netz anzubieten. „Web-Suchmaschinen ermöglichen das Retrieval, so dass die Mühe der Katalogisierung, der papierenen unvollständigen Verbreitung, der Aufstellung in Regalen, der Verschlagwortung, des Kampfes gegen Verlust, Beschädigung, Vergilbung entfällt." [HZ00] Und auch im Rahmen des DFG-Projekts „Dissertationen Online" (siehe nächste Seite) wurde festgestellt:

> „Eine mit einem Computer hergestellte Dissertation nur über Papier zu verbreiten, ist heutzutage wenig effektiv. Wäre sie im Netz abrufbar, könnte der interessierte Benutzer sie in Sekundenschnelle auf seinem Computer haben [… und] gezielt nach Titeln, Schlagwörtern, wichtigen Begriffen, zitierter Literatur usw. recherchieren; die wissenschaftlichen Ergebnisse wären über das Internet für die *Academic Community* weitaus leichter, früher und schneller erschließbar und zugänglich als im traditionellen Buchdruck und -vertrieb." [IuK98]

Projekte zum Aufbau von entsprechenden Archiven bergen dabei nicht nur Vorteile für die Nutzer, sondern sind durchaus auch als Weiterqualifikation der Bibliotheksmitarbeiter interessant, die in ihren Arbeitsbereichen neue Kompetenzen erwerben und so später umfangreichere und gegebenenfalls auch kompliziertere Aufgaben im Bereich des elektronischen Publizierens qualifiziert übernehmen können.
Insgesamt stellen elektronische Dissertationen also einen wirklichen Mehrwert gegenüber ihren Offline-Pendants dar und es ist eigentlich nur konsequent, die heutzutage sowieso computergestützt erstellten Examensarbeiten auch in dieser Form zu veröffentlichen.

Mit der Aufnahme einer Dissertation in den Bibliotheksbestand sind allerdings auch rechtliche Bestimmungen verbunden. Nach dem deutschen Urheberrechtsgesetz (UrhG[90]) erwirbt ein Autor mit der Fertigstellung seines Werkes automatisch alle Rechte daran. Diese sind z.B. das Veröffentlichungsrecht (§12 UrhG) und das Nutzungsrecht (§ 31 UrhG). Entsprechend § 2 Abs. 2 UrhG gelten als Werk dabei nur „persönliche geistige Schöpfungen". Diese sind bis 70 Jahre nach Tod des Verfassers gesetzlich geschützt und so muß der Autor bei Abgabe seiner Dissertation immer auch eine Erklärung unterzeichnen, in der er Teile seines Urheberrechts an die Universitätsbibliothek überträgt. Zum einen handelt es sich dabei um das Recht der Vervielfältigung (bis zu einer Höchstgesamtzahl von 150 Exemplaren, §16 UrhG) und zum anderen um das Recht der Verbreitung (§17 UrhG). Erst diese Einverständniskeitserklärung, die natürlich auch im Falle der immer weiter steigenden Zahl elektronischer Dissertationen zu unterschreiben ist, versetzt die Bibliothek in die Lage, den Volltext zusätzlich auch über das Internet anzubieten.

[90] http://www.urhg.de

Der derzeitige Publikationsserver der UB Potsdam sieht folgende Formulierung vor:

> „Hiermit übertrage ich der Universitätsbibliothek Potsdam - vertreten durch die Abt. Publikationen - das Recht, die oben genannte Publikation auf dem Publikationsserver der Universitätsbibliothek zu veröffentlichen und im Internet zu verbreiten. Zugleich gestatte ich der Deutschen Bibliothek und ggf. einer Sondersammelgebietsbibliothek, die Publikation öffentlich auf ihren Servern zur Benutzung bereitzustellen. Ich versichere, dass mit der Publikation dieses Dokumentes keine Rechte Dritter verletzt werden. Den für die Archivierung und Verbreitung ggf. notwendigen Konvertierungen des Dokumentes in andere Dateiformate stimme ich zu.“[91]

An anderen Universitäten finden sich ähnlich lautende Formulierungen, wie z.B.:

> „Der Doktorand überträgt das Recht an der Veröffentlichung seiner Dissertation an die UB. Die gedruckte Fassung muss einen Lebenslauf enthalten, in der elektronischen Version kann darauf verzichtet werden.“[92]

> „Ich überlasse der Universitätsbibliothek (UB) meine Dissertation in elektronischer Form. Ich versichere, daß die elektronische Fassung vollständig mit der Printversion übereinstimmt und übertrage der UB das Recht, die Dissertation auf ihrem Archivserver aufzulegen und über das Internet zugänglich zu machen. Rechte Dritter stehen der Veröffentlichung nicht entgegen. Mit später eventuell notwendigen Konvertierungen in andere Datenformate bin ich einverstanden.“[93]

Hervorgehoben wird meist, daß mit der Veröffentlichung keine Rechte Dritter verletzt werden dürfen, was neben einer Einbindung von z.B. Bildern oder sog. Großzitaten aus anderen Werken insbesondere immer dann problematisch wird, wenn eine Dissertation bereits als Verlagsausgabe publiziert wurde. Weiterführende Informationen zu derartigen Sonderfällen und allgemein zu rechtlichen Bestimmungen im Internet finden sich z.B. in [Bleu00] ...
... und sehr ausführlich auch in [Müll00], wo zusammenfassend festgestellt wird, daß sich die Hochschullandschaft trotz teilweise verwirrender Klauseln und fehlender fakultätsübergreifender Regelungen seit einiger Zeit „[...] mit Riesenschritten in Richtung digitale Dokumente und elektronisches Publizieren [bewegt]. Auf Servern von Universitätsbibliotheken oder Universitätsrechenzentren werden immer mehr Dissertationen, aber auch Diplomarbeiten und sonstige Forschungsergebnisse angeboten“.

Doktorarbeiten sind anderen Hochschulschriften dabei zahlenmäßig meist weit überlegen – insbesondere wegen der erwähnten Vorteile für Promovend und Bibliothek, nicht zuletzt aber auch wegen der vielen Projekte und Initiativen, die sich in den letzten Jahren intensivst mit diesem Dokumenttyp befaßt haben. Wegbereitend war dabei vor allem das Projekt „Dissertationen Online“, welches im Zeitraum 1998 bis 2000 von der DFG gefördert wurde und es sich zur Aufgabe gemacht hatte, „[…] den mit der elektronischen Veröffentlichung von Dissertationen verbundenen Fragenkomplex in technischer, bibliothekarischer und archivarischer Hinsicht aufzuarbeiten und in Zusammenarbeit von Wissenschaftlern, Informatikern und Bibliothekaren einer Lösung zuzuführen.“ [DM98]

---

[91] http://pub.ub.uni-potsdam.de/agree.htm
[92] http://darwin.inf.fu-berlin.de/help/german/DissAbgabe.html
[93] http://archiv.ub.uni-marburg.de/ubtexte/edissf1.pdf

Laut Homepage der Koordinierungsstelle „DissOnline“[94] waren die Ziele insbesondere:

- fachübergreifende und bundesweite Vorschläge zum elektronischen Publizieren von Dissertationen zu erarbeiten, nötige Standards zu entwickeln und mit den entsprechenden Gremien und Gruppen (Bibliotheken, Fachgesellschaften etc.) abzustimmen,
- für Bibliotheken den Einstieg ins elektronische Publizieren am Beispiel von Hochschulschriften zu geben,
- der Wissenschaft effektive Recherchetools in die Hand zu geben für eine qualitativ hochstehende, inhaltliche Suche, die über eine einfache Volltextsuche hinausgeht und Recherchen in Strukturelementen ermöglicht,
- Promovenden aller Fächer die Möglichkeit zu bieten, ihre Forschungsergebnisse schnell und kostengünstig zu veröffentlichen.

Entstanden war das Projekt bereits 1997 aus einer gleichnamigen IuK-Kommission[95], bestehend aus Vertretern von fünf wissenschaftlichen Fachgesellschaften (Chemie, Physik, Mathematik, Erziehungswissenschaften und Informatik) und angesiedelt an verschiedenen deutschen Universitäten sowie an Der Deutschen Bibliothek. Jeder Projektpartner hatte spezifische Aufgaben: während man in Duisburg und Frankfurt am Main Werkzeuge für die Verarbeitung dissertationsspezifischer Metadaten entwickelte, wurden von der Universität Oldenburg rechtliche Fragen geprüft und geeignete Retrieval-Möglichkeiten geschaffen. Die Chemiker in Erlangen wiederum beschäftigten sich mit der Einbettung multimedialer Elemente in digitale Hochschulschriften und die SUB Göttingen hatte schließlich die Tauglichkeit der erarbeiteten Vorschläge im bibliothekarischen Alltag erprobt (siehe auch [IuK98]).

Die wohl umfangreichsten Teilprojekte wurden allerdings an der HU Berlin durchgeführt: neben der Gesamtkoordination von *Dissertationen Online* standen hier vor allem die Wahl von Dateiformaten und die Durchführung von Schulungen im Vordergrund, wobei diese beiden Projekte in besonderem Maße voneinander abhängig waren: erst die Empfehlungen der „Formate“-Arbeitsgruppe haben eine umfassende Autorenbetreuung in Form von Beratungen und Kursen überhaupt notwendig gemacht. Der Grund hierfür soll im folgenden Abschnitt aufgezeigt werden – zusammen mit den nicht unwesentlichen Auswirkungen auf alle Beteiligten. Vor allem aber gilt es, den Dokumentenserver der Humboldt-Universität etwas genauer vorzustellen. Nicht nur, weil dieser letztlich das eigentliche Ergebnis des Gesamt-Projekts „Dissertationen Online“ darstellt, sondern weil er recht gut veranschaulicht, was ein modernes Archivsystem bieten sollte.

[94] Nach Abschluß der Projektförderung wurde zum 01.02.2001 eine Koordinierungsstelle für die Nutzung und Weiterentwicklung der Ergebnisse von „Dissertationen Online“ eingerichtet, die an Der Deutschen Bibliothek (siehe Abschnitt 5.2) angesiedelt ist. URL: http://www.dissonline.de

[95] http://www.iuk-initiative.org

5.1 Online-Dissertationen an der Humboldt-Universität Berlin

„Schnell, kostengünstig, up to date!“, so rief die HU Berlin ihre Doktoranden 1997 dazu auf, von den neuesten Entwicklungen Gebrauch zu machen und die Dissertationen elektronisch zu veröffentlichen.[96] Aufgrund geänderter Promotionsordnungen ein verlockendes Angebot – doch schon bald stellte sich heraus, daß es mit einer einfachen Abgabe des Word- oder Postscript-Dokuments und der ‚üblichen‘ Verlinkung auf einem Webserver nicht getan war: die Verwendung des präferierten Text- bzw. Satzsystems war zwar weiterhin erlaubt, allerdings mußte die Prüfungsarbeit *strukturiert* vorliegen und somit unter Nutzung spezifischer Dokumentvorlagen bzw. im Falle von z.B. TeX/LaTeX unter Befolgung ganz bestimmter Vorgaben erstellt werden. Der Grund: im Rahmen des DFG-Projekts „Dissertationen Online“ und des HU-internen Projekts „Digitale Dissertationen“ (DiDi[97]) hatte man 1997/98 – basierend auf den Untersuchungen von [Ohst98] – beschlossen, Publikationen nicht im Ursprungsformat, sondern in der Metasprache SGML zu archivieren und so neben einer verbesserten Konvertier- und Recherchierbarkeit auch eine Lesbarkeit über einen längeren Zeitraum hinweg zu gewährleisten (siehe auch [Dob98]).

Für den Promovenden ist diese noch immer geltende und durchaus zukunftsweisende Festlegung (siehe Kapitel 3.3) allerdings, wie angedeutet, mit einem nicht unerheblichen Mehraufwand verbunden: will er den Einsatz teurer, oft sehr komplizierter SGML-Editoren vermeiden, muß er mit Hilfe spezieller Werkzeuge und Formatvorlagen alle Elemente (Überschriften, Bilder, Zitate, usw.) explizit auszeichnen und Strukturinformationen in das Dokument einarbeiten, um eine automatische Überführung seines meist proprietären Dateiformats nach SGML (und inzwischen auch XML) zu ermöglichen. Für viele Autoren ein schwieriges Unterfangen, sind sie doch nur selten Spezialisten im Umgang mit Textverarbeitungsprogrammen und es zumeist gewohnt, „einfach drauf los zu tippen“, ohne die oftmals bereits integrierten Auszeichnungsfunktionalitäten zu benutzen: Überschriften werden so z.B. lediglich fett markiert, anstatt sie ausdrücklich als solche zu deklarieren. Die Auswertung entsprechender, von der HU verteilter Fragebögen bestätigt diese Vermutung:

> „Die Fähigkeiten, strukturierte Texte zu erstellen, haben [...] noch nicht in dem Maße Eingang in die wissenschaftliche Praxis gefunden, wie das aus unserer Sicht wohl wünschenswert wäre.“ [DK98]

Die zu Beginn des Projekts „Digitale Dissertationen“ durchgeführte Befragung potentieller Promovenden hatte aber vor allem ein Ziel: man wollte herausfinden, welche Text- und Satzsysteme bevorzugt verwendet werden, um programmspezifische Dokumentvorlagen und Stylevorgaben überhaupt entwerfen, und den Autoren bei deren Nutzung beratend zur Seite stehen zu können. Das Ergebnis verblüfft wenig: Microsoft Word war (und ist) mit Abstand die meistgenutzte Textverarbeitung und so war es nur logisch, sich bei den Entwicklungen insbesondere auf diese Software zu konzentrieren.[98] Die resultierende Formatvorlage „dissertation-97.dot“ läßt sich kostenlos vom Publikationsserver der HU herunterladen und problemlos in Word 97/2000/XP (bzw. Word 98 und Word 2001 für MacOS) integrieren.[99] Einmal installiert, ist das Schreiben oder Nachformatieren der eigenen Arbeit mit Hilfe dieser Vorlage Pflicht – und dies im doppelten Sinne: zum einen werden nur korrekt strukturierte

---

[96] siehe z.B. http://dissertationen.hu-berlin.de/e_rzm/24/dobratz-susanne-2003-04-17/HTML/14.php

[97] http://edoc.hu-berlin.de/epdiss

[98] Auf die Frage nach dem benutzten Textverarbeitungssystem antworteten laut [DK98] rund 85% mit WinWord, 11% mit LaTeX und 4% mit der Angabe anderer Programme.

[99] Es existiert auch eine selbst-extrahierende Version speziell für Word 2000/XP, ältere Versionen von Microsoft Word werden nicht unterstützt. URL: http://edoc.hu-berlin.de/e_autoren/vorlage.php?nav=diss

Word-Dokumente archiviert[100], und zum anderen dürfen für die Kennzeichnung der einzelnen Bestandteile nur die Funktionen der neu hinzugekommenen Menüleisten, Toolbars und Makros benutzt werden – auch für das Hervorheben einzelner Wörter. Soll ein Textbereich also beispielsweise kursiv erscheinen, ist er zunächst wie üblich zu markieren, dann aber mit dem entsprechenden Formatierungselement aus dem dissertationsspezifischen Menü auszuzeichnen. Gleiches gilt für das Rückgängigmachen einzelner Aktionen: hierfür ist unbedingt der ‚Radiergummi' aus der zusätzlichen Werkzeugleiste zu benutzen – nur so ist eine spätere, automatisierte Konvertierung nach SGML möglich.

Während die Hervorhebung von Wörtern und Textteilen noch trivial ist, wird es bei der Erzeugung von Tabellen, Querverweisen, Literaturverzeichnissen, Fußnoten, Aufzählungen u.ä. schon komplizierter – und entsprechend umfangreich ist das Handbuch, welches die Verwendung der Formatvorlage erläutert und anhand von Beispielen detailliert auf Besonderheiten eingeht.[101] An dieser Stelle soll auf eine genaue Beschreibung allerdings verzichtet und nur der grundlegende Aufbau eines „dissertation-97.dot"-konformen Dokuments vorgestellt werden.

Eine solche Datei gliedert sich immer in drei große Bereiche: das Deckblatt (front), den Hauptteil (body) und den Anhang (back). Das Deckblatt ist dabei ein äußerst wichtiger Bestandteil. Neben Titel, Autor, Datum, Fakultät und Gutachter enthält es z.B. auch Schlüsselwörter, Abstracts und weitere Angaben, nach denen später gesucht werden kann. Der verwendete Metadatensatz basiert auf Dublin Core (siehe Kapitel 3.4.1). Er wurde im Rahmen des bereits angesprochenen DFG-Projekts „Dissertationen Online" entwickelt und ist bundesweit gültig (vergl. auch [DS99]). Die Verwendung der Formatvorlage stellt nun sicher, daß alle Metaangaben vorhanden, und somit extrahierbar sind. Vor allem aber sorgt sie für ein einheitliches Layout: durch die immer gleiche Position der Elemente läßt sich das Deckblatt schneller lesen und erfassen – egal welche Arbeit man gerade vor sich hat.
Der eigentliche Inhalt der Dissertation verbirgt sich im Hauptteil. Hier sind neben Kapiteln, Absätzen, Überschriften, Abbildungsunterschriften und weiteren Strukturierungsmitteln auch Elemente für semantische Textcodierungen vorgesehen, die insbesondere für die Auszeichnung indexierungswürdiger Begriffe, sowie für Zitate, Definitionen oder Orts- und Personennamen verwendet werden können. Außerdem ist eine Einbindung von Grafiken, Diagrammen (z.B. aus Excel) und mathematischen sowie chemischen Formeln möglich, wobei die Objekte zumeist nicht direkt eingebettet, sondern nur mit dem Dokument verknüpft werden, um sie später in der SGML-Datei referenzieren zu können.
Den Abschluß bilden schließlich die Anhänge. Hier wird differenziert zwischen inhaltlichen Anhängen, die meist aus Tabellen, Quellcode von Programmen, Abbildungen oder z.B. Versuchsprotokollen bestehen, und typischen dissertationsspezifischen Anhängen, wie Lebenslauf, Danksagung, Selbständigkeitserklärung, Veröffentlichungslisten und Literatur. Anders als beim Titelblatt werden dabei keine sonderlich starken Strukturierungen vorgenommen, um die Formatvorlage möglichst allgemeingültig halten, und auch für andere Hochschulschriften benutzen zu können.
Für weiterführende Informationen sei erneut auf das Handbuch verwiesen – festzuhalten bleibt, daß die Erstellung einer Dissertation an der Humboldt-Universität aufwendiger ist, als

[100] dies betrifft auch Habilitationsschriften sowie Magister- und Diplomarbeiten: http://edoc.hu-berlin.de/e_autoren/was-diss.php?nav=diss
[101] eine etwa 40 seitige Version ist abrufbar unter http://edoc.hu-berlin.de/e_autoren/download/didi-man.pdf

an Einrichtungen, die keine derart spezifischen Anforderungen an die Struktur der Ursprungsdatei stellen. Doch dieser Mehraufwand (der übrigens nicht nur auf Word-Nutzer beschränkt ist[102]) bedeutet letztlich auch einen Mehrwert: dank der guten strukturellen Auszeichnung können die gelieferten Dokumente optimal durchsucht, und vor allem sehr viel einfacher in ein für die angestrebte Langzeitarchivierung besser geeignetes Format konvertiert werden.

Die HU Berlin hat sich dabei, wie eingangs erwähnt, für SGML(/XML) entschieden. Voraussetzung für die (kontrollierte) Speicherung in einer solchen Metasprache ist allerdings das Vorhandensein einer Dokumenttypdefinition, die ganz genau spezifiziert, welche Tags mit welchen Werten zu belegen sind. Aus diesem Grund wurde eine spezielle DTD für Dissertationen (DiML: „Dissertation Markup Language“) geschaffen, die zu einem großen Teil auf der ETD-ML von Virginia Tech beruht und nach und nach an deutsche Verhältnisse angepaßt wurde.[103] Die DiML-DTD bildet das exakte Gegenstück zur „dissertation-97.dot“-Formatvorlage: alle dort deklarierten Elemente finden sich in der SGML-basierten Markup Language wieder – und so auch die Dreiteilung in <front>, <body> und <back>. Hier ein mögliches Deckblatt, welches die Metadaten zur Hochschulschrift enthält (in Auszügen entnommen aus [Schulz99]):

```
<front>
  <school>Aus dem Institut für Pathologie [...]</school>
  <submission>Dissertation</submission>
  <title>Der Brustkrebs</title>
  <degree>zur Erlangung des akademischen Grades<br/>
  doctor medicinae (Dr. med.)</degree>
  <major>vorgelegt der Medizinischen Fakultät<br/></major>
  der Humboldt-Universität zu Berlin
  <author>von
    <given>Klaus</given> <surname>Muster</surname>
    <suffix>12.1.1963 in Frankfurt geboren</suffix>
  </author>
  <dean>Dekan: Prof. Dr. Mustermann</dean>
  <approvalls>
    <name>Prof. Dr. med. H. Musterfrau</name>
    <name>Prof. Dr. med. J. Mann</name>
  </approvalls>
  <date type="1">eingereicht: Juni 1997</date>
  [...]
  <keywords language="en">
    <keyword>cancer</keyword>
  </keywords>
  <abstract language="de">
    <p>(hier folgt nun der Text des deutschen Abstrakt)</p>
  </abstract>
  [...]
</front>
```

Die Reihenfolge der DiML-Tags ist nicht vorgeschrieben, wohl aber deren Vorhandensein sowie die Struktur von Schachtelungen und beschreibenden Attributen. Vor-/Nachname und Geburtsdatum/-ort gehören beispielsweise zwingend zum <author> und die ISO639-konforme

[102] Auch für WordPerfect und LaTeX existieren Styledateien bzw. Vorgaben, deren Anwendung zwingend erforderlich ist - andere Erstellungsformate werden derzeit nicht akzeptiert.
[103] Virginia Tech (Virginia Polytechnic Institute and State University) hatte ähnliche Ambitionen und so hatte man sich zu einer Zusammenarbeit entschlossen - siehe auch http://www.ndltd.org und http://etd.vt.edu

Angabe der Sprache von Schlüsselwörtern und Zusammenfassungen ist ebenfalls Pflicht. Geregelt wird all das mit Hilfe entsprechender Einträge in der DTD. Für einen Teil des Deckblattes sehen diese z.B. so aus:

```
<!ELEMENT front - -
  (school | submission | title | degree | major | author | dean |
   approvals | date | keywords | copyright? | abrstract | grant* |
   dedicaton? | acknowledgments? | declaration? | p)* >
[...]
<!ELEMENT author - -
  (#PCDATA | given | surname | suffix | organisation?)* >
<!ELEMENT given - -
  (#PCDATA | em | u | strong | tt | q | term | foreign |
   im | link | target | a | sup | sub | br)* >
[...]
<!ELEMENT keywords - - (keyword+) >
<!ATTLIST keywords
  %s.language;
  %s.id;
  %s.color;
  %type; >
[...]
```

Analog dazu werden auch die Strukturelemente des eigentlichen Textkörpers (<body>) und des abschließenden <back>-Bereichs definiert. Die erweiterte Backus-Naur-Form[104] dient dabei der genauen Spezifizierung der einzelnen Teile des zugehörigen DiML-Dokuments – angefangen bei Gliederungs-Tags (<chapter>, <section>, <subsection>, <block> usw.), über die Deklaration von Tabellen, Listen, Formeln und Bibliographien, bis hin zur korrekten Hervorhebung spezieller Zeichen und Wörter mittels <u> (unterstrichen), <sub> (tiefgestellt) oder z.B. (kursiver, blauer Text mit bestimmtem Schriftschnitt). Diese präzise Auszeichnung der Elemente macht es später möglich, lediglich in bestimmten Bereichen der Arbeit zu recherchieren (also beispielsweise nur im einleitenden Kapitel, in den Bildunterschriften oder in kursiv dargestellten Zitaten) und so die Treffermenge auf (hoffentlich) relevante Publikationen zu beschränken. Grafiken, Diagramme und sonstige nicht-textuelle Bestandteile bleiben von einer derartigen Suchanfrage allerdings ausgeschlossen: sie werden extern gespeichert und in der Dissertation Markup Language als „Multimedia Objects" (<mm>) entsprechend referenziert. Auch hierzu ein kurzes Beispiel, ebenfalls angelehnt an [Schulz99]:[105]

```
<!DOCTYPE ETD PUBLIC "-//HUB//DTD Electronic Theses and Dissertations
Version DiML 1.1//EN"
[ <!ENTITY obj.10 SYSTEM "grafik2.jpg" NDATA JPEG> ] >
<body>
  <chapter>
    <head>Beispiel Abbildungen</head>
    <mm entity="obj.10" type="CHEMISTRY">
      <caption>Abb. 1: Sicht in ein Kohlenstoffmolekül</caption>
    </mm>
[...]
```

[104] bei der (Extended)BNF handelt es sich um eine Metasprache zur Syntaxbeschreibung formaler Sprachen - siehe auch [Born01]

[105] Der Attributtyp ENTITY hat die Aufgabe, dem SGML-System ein Objekt mit einem bestimmten Namen bekanntzumachen. Im Beispiel verweist obj.10 auf eine JPEG-Datei; DiML erlaubt außerdem die Verwendung von EPS, CGM, TIFF, GIF, BMP und PNG als Grafikformat, sowie die Angabe eines konkreten Suchpfades.

Für die visuelle Gestaltung und Illustration eines Dissertationstextes sind Abbildungen natürlich unumgänglich, allerdings sollte man laut [Schulz99] nach Möglichkeit darauf verzichten, auch Tabellen und z.B. mathematische Formeln einfach als Bilddateien abzuspeichern. Im Falle des Einsatzes von Word läßt sich dies kaum vermeiden; SGML/ XML-Kenner können jedoch Standards wie CALS und MathML[106] nutzen, um Tabellen und Formeln direkt in das DiML-Dokument einzubetten und somit ebenfalls recherchierbar zu machen. Außerdem ist es möglich, an einigen Stellen TeX-Notation zu verwenden. Wie das allerdings genau auszusehen hat, soll hier nicht näher beschrieben werden – wichtiger erscheint es, den Gesamtworkflow zusammenzufassen und nochmals hervorzuheben, mit welchem Aufwand die Nutzung von SGML als Recherche- und Archivierungsformat verbunden ist. Dazu folgende (nicht mehr ganz aktuelle) Abbildung, die aus [Dob98] entnommen wurde und zusätzlich zeigt, daß die HU auch für die nötige Sicherheit bei der Bereitstellung sorgt:

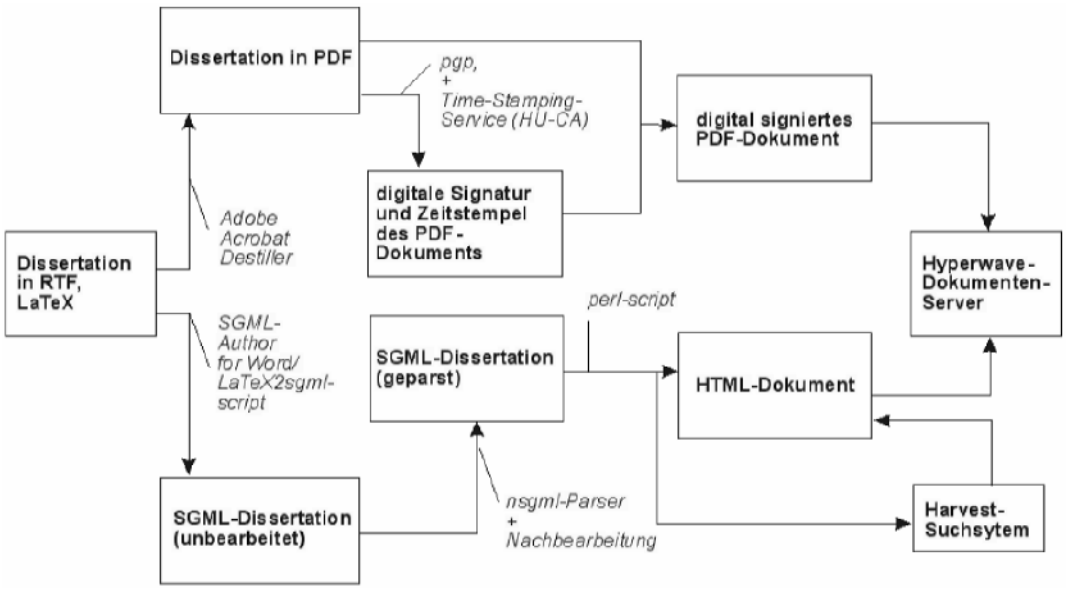


Abbildung 12: DiDi Workflow

Ausgangspunkt ist entweder ein mit der „dissertation-97.dot"-Dokumentvorlage erstelltes und im RTF-Format[107] abgespeichertes Word-Dokument, ein nach bestimmten Richtlinien layoutetes LaTex-File[108] oder eine (der Einfachheit halber hier nicht weiter berücksichtigte) WordPerfect-Datei, die spezifischen Style-Vorgaben entspricht.[109] Aus diesem Original-dokument ist vom Promovenden zunächst ein „digitales Druckexemplar" im PDF-Format zu erzeugen[110], anschließend sind beide Versionen möglichst auf elektronischem Wege an die für Konvertierung und Speicherung zuständige „AG Elektronisches Publizieren" – bestehend aus Bibliothekaren, Mitarbeitern des Rechenzentrums und stud. Hilfskräften – zu transferieren. Bevorzugt zu verwenden ist dabei ein spezielles Online-Formular, welches die Angabe von Metadaten und das Hochladen der Dissertation auf den Publikationsserver erlaubt.[111]

---

[106] http://www.oasis-open.org/specs/a502.htm bzw. http://www.w3.org/TR/REC-MathML

[107] Das Rich Text Format wurde von Microsoft als Austauschformat und für die Windows-interne Zwischen-ablage entwickelt und basiert im Gegensatz zum Word-Format auf 7-Bit-ASCII-Code - siehe auch [Born01].

[108] http://edoc.hu-berlin.de/e_autoren/latex

[109] http://edoc.hu-berlin.de/e_autoren/wpstyle.php?nav=diss

[110] dies kann mit einem speziellen Konverter oder beispielsweise mit Hilfe von Adobe Acrobat und dem integrierten Druckertreiber geschehen - siehe Kapitel 3.2

[111] https://edoc.hu-berlin.de/cgi/dokupload/dokupload.cgi

Nach erfolgreicher Übertragung werden alle weiteren Arbeitsschritte von einem eigens entwickelten Workflow-Management-System koordiniert und überwacht. Die nach und nach auflaufenden Dissertationen landen in einem gemeinsamen Taskpool. Sind personelle Kapazitäten frei, kann sich der jeweilige Mitarbeiter eine Publikation ‚herausfischen', diese kontrollieren, konvertieren, für die Öffentlichkeit freischalten – und deren Bearbeitung gegebenenfalls auch zurückstellen oder ablehnen, falls es Unklarheiten oder Probleme gibt.[112] Entsprechen Urdatei und PDF-Version den ebenfalls abzugebenden Papierexemplaren und wurden die Metaangaben korrekt übermittelt, wird die Publikation endlich in das angestrebte Archivierungsformat überführt. Dies geschieht entweder mit Hilfe spezieller Scripte oder bei vorliegendem RTF-Dokument durch Nutzung von *SGML Author*, einem von Microsoft kommerziell vertriebenen Add-On für Word. In beiden Fällen ist nach erfolgter Konvertierung zumeist noch Nacharbeit notwendig: zum einen muß die resultierende SGML-Datei noch kontrolliert, geparst und DiML-konform gemacht werden und zum anderen speichert *SGML Author for Word* die eingebetteten Bilder als externe WMF- bzw. CGM-Grafiken, die von Webbrowsern nicht dargestellt werden können. Hier ist also eine manuelle Konvertierung der Abbildungen z.B. ins GIF-Format und eine nachträgliche Anpassung der <mm>-Referenzen (siehe weiter oben) erforderlich.

Sind DiML-Dokument und zugehörige Grafiken schließlich erzeugt, werden in einem weiteren Konvertierungsschritt eine Reihe von HTML-Seiten generiert, die das Deckblatt, die einzelnen Kapitel und die jeweiligen Anhänge beinhalten und zusammen – korrekt untereinander verlinkt – das Präsentationsformat bilden. Vorteil dieser Unterteilung in mehrere Dateien: der Leser kann sich später sehr elegant durch die Arbeit klicken und z.B. nur die Kapitel herunterladen, die ihn wirklich interessieren. Die einzelnen Seiten eines Abschnitts werden dabei zusammenhängend gespeichert; durch horizontale Linien voneinander getrennt und mit der Seitenzahl des Ursprungsdokuments versehen, so daß die Zitierbarkeit gewährleistet bleibt. Die Umsetzung erfolgt vollautomatisch mittels eines von Virginia Tech zur Verfügung gestellten Perl-Scripts, welches die Gliederungselemente der SGML-Dissertation extrahiert, entsprechende Tags erzeugt und natürlich auch dafür sorgt, daß die Abbildungen an der richtigen Stelle ‚eingebunden' werden. Das Originallayout geht HTML-bedingt aber dennoch verloren: Bilder erscheinen z.B. nicht mehr „links unten und von Fließtext umgeben", sondern einfach zentriert auf der Seite. Da dies vielleicht nicht unbedingt der Intention des Autors entspricht, wird als weiteres Präsentationsformat immer auch die PDF-Version der Publikation angeboten, die es – als Abbild des Druckexemplars – allerdings besonders zu sichern gilt. Noch vor der Freischaltung für die Öffentlichkeit wird der Hashwert des PDF-Dokuments daher digital signiert und mit einem Zeitstempel versehen. Ziel ist es, „[...] Autor, Inhalt und Veröffentlichungszeitpunkt beweiskräftig vor Fälschung und Zweifel an der Authentizität [zu schützen]" [Dob98] (siehe auch Kapitel 4.2).

Zu guter letzt landen die beiden Versionen der Dissertation dann auf dem zertifizierten Dokumentenserver der HU. Während vor einigen Jahren noch das kommerzielle Hyperwave-System[113] für die Speicherung genutzt wurde (und daher auch in Abbildung 12 erscheint), kommen inzwischen Eigenentwicklungen und Open Source-Lösungen zum Einsatz. Ein normaler HTTP-Server liefert die statischen HTML- und PDF-Dokumente und ist mittels PHP auch in der Lage, dynamisch auf Nutzeranfragen (z.B. Browsing nach Fakultäten) zu reagieren. Die Hauptvorteile von Hyperwave (integrierte Verity Search Engine und konstante

[112] Bei einem persönlichen HU-Besuch konnte sich der Autor vor Ort ein Bild von den gebotenen Funktionalitäten machen und Anregungen für den eigenen Prototypen mit nach Hause nehmen.
[113] http:// www.hyperwave.com

URLs innerhalb des Systems, siehe auch [Dob98]), mußten allerdings auf geeignete Weise nachgebildet werden und so hat man sich zum einen für URNs Der Deutschen Bibliothek (siehe Abschnitt 5.2.2), und zum anderen für Harvest als Volltext-Suchmaschine entschieden. Letztere unterstützt bei entsprechender Konfiguration des Gatherers und des Brokers unterschiedliche DTDs von SGML und ermöglicht somit sogar strukturelle Recherchen. Zusätzlich dazu werden die Metadaten der Autoren aber auch in einer Datenbank vorgehalten und über ein entsprechendes Suchformular zugriffsfähig gemacht.

Auf weitere interessante Features (wie z.B. OAI-/Z39.50-Schnittstelle, Print-On-Demand/ ProPrint-Integration, Anbindung an das Bibliothekssystem für Katalogisierung und Nachweis der elektronischen Publikationen u.ä.) soll an dieser Stelle nicht näher eingegangen werden (einen guten Überblick gibt [Dob03]). Insgesamt kann der Publikationsserver der Humboldt-Universität als durchaus ausgereift und zukunftsweisend bezeichnet werden. Grund ist die Verwendung der für langfristige Speicherung und Retrieval bestens geeigneten Metasprache SGML(/XML) und der „Mehrzweck"-DTD DiML, mit der sich nicht nur Dissertationen (die hier ja im Vordergrund stehen), sondern beliebige wissenschaftliche Arbeiten darstellen lassen. Beweis dafür sind die über 1000 digital vorgehaltenen Dokumente, von denen derzeit ‚nur' ca. 560 Doktorarbeiten sind.

Die Ausführungen der letzten Seiten haben eins hoffentlich deutlich gemacht: um eine hohe Qualität der Recherche und der Langzeitarchivierung gewährleisten zu können, muß ein Teil der Verantwortung an die Autoren weitergegeben werden – sie müssen sich an Vorgaben halten und die Elemente ihrer Dissertationen bis ins Detail auszeichnen. Dies bedeutet also mehr Arbeit für die Promovenden der HU; paradoxerweise jedoch nicht gleichzeitig weniger Arbeit für Bibliothek und Rechenzentrum – im Gegenteil: im Rahmen von zeitlich und personell aufwendigen Schulungen, Kursen und Beratungsgesprächen muß dafür gesorgt werden, daß die Dokumente bei Ablieferung alle Anforderungen erfüllen und sich problemlos (aber leider ebenfalls recht aufwendig, siehe oben) in das DiML-Format (+DTD) überführen lassen. Es gilt „[...] die Autoren dahin zu führen, dass sie in die Lage versetzt werden, die für eine Konvertierung nach SGML/XML geeigneten Urformen der Dokumente in definierten Dateiformaten zu erstellen". Und [DK98] führt weiter aus:

> „Die Erfahrung [...] lehrt, daß die korrekte Benutzung der Formatvorlage den qualifizierten Umgang mit der Textverarbeitung voraussetzt. Hier liegen auch die Hauptprobleme: Da die informatorische Grundbildung der Studierenden und der Wissenschaftler nicht zum Lehrplan einer Universität gehört, sind hier sehr große Lücken in der Handhabung elektronischer Medien, besonders auch in den Textverarbeitungen zu erkennen."
> → „Die Integration einer Komponente *Autorenbetreuung* in den Geschäftsgang "Elektronische Publikationen" einer Universitätsbibliothek ist unserer Meinung nach zwingend",

so das abschließende Fazit.

5.2 Die Deutsche Bibliothek

Dokumentenserver zur Archivierung anfallender Dissertationen und sonstiger digitaler Hochschulschriften finden sich inzwischen an fast allen deutschen Universitäten.[114] Einige haben ihr „Institutional Repository“ selbst implementiert, andere nutzen vorgefertigte Lösungen und passen diese – entsprechende Kenntnisse vorausgesetzt – an die lokalen Bedürfnisse an.[115] Das Konzept der HU Berlin hebt sich aus dieser Masse insbesondere durch die Wahl von SGML als Archivierungsformat hervor und wurde deshalb etwas genauer vorgestellt. Eins hat die Humboldt-Universität mit anderen Einrichtungen aber dennoch gemeinsam: sie speichert die Dissertationen und Habilitationen nicht ausschließlich selbst, sondern vertraut in Sachen Langzeiterhaltung auf die Kompetenz einer anderen Institution: Der Deutschen Bibliothek.

In fast allen Ländern wurden im Laufe der Jahrhunderte gesetzliche Regelungen erlassen, die eine zwangsweise Verpflichtung zur Abgabe von Werken an bestimmte Bibliotheken vorsehen. Im Gegensatz zu historischen Motiven, zu denen durchaus auch hoheitliche Zensur und Begünstigungen einzelner Autoren und Buchhändler zählten[116], liegt diese Pflichtexemplargesetzgebung heutzutage vorrangig kulturpolitischen Überlegungen zugrunde:

> „Von jedem in einem Land erschienenen Werk müsse zumindest ein Exemplar an zentraler Stelle gesammelt, verzeichnet, aufbewahrt und für jeden Bürger zugänglich gemacht werden, um jedermann einen vollständigen Überblick über die geistig-kulturellen Erzeugnisse seiner Zeit zu verschaffen, und um die Werke für die Zukunft zu sichern.“ [Müll98]

Für Deutschland ist mit dieser Aufgabe Die Deutsche Bibliothek (DDB) betraut, die 1990 aus den Vorgängereinrichtungen *Deutsche Bücherei Leipzig* (gegründet 1912), *Deutsche Bibliothek Frankfurt am Main* (gegründet 1947) und *Deutsches Musikarchiv Berlin* hervorgegangen ist. Trotz räumlicher Trennung der drei Standorte ist sie die zentrale Archivbibliothek Deutschlands und als nationalbibliographisches Zentrum für die dauerhafte Aufbewahrung und Verfügbarmachung deutscher und deutschsprachiger Literatur ab Erscheinungsjahr 1913 zuständig.[117] Während von der Pflichtablieferung und Erschließung zunächst ausschließlich Druckwerke betroffen waren, hat sich der Zuständigkeitsbereich in den letzten 50 Jahren doch erheblich ausgeweitet: aufgrund des technologischen Fortschritts mußten zunehmend auch neu entstandene Medienarten, wie z.B. Bild- und Tonträger, mit einbezogen werden. Inzwischen beinhaltet der Sammelauftrag auch digitale Publikationen – allerdings nur solche, die auf physischen Trägern verbreitet werden (Disketten, CD-ROM, DVD, ...). Eine Speicherung von Netzpublikationen, wie E-Journals, elektronische Volltexte oder Websites, die online über Netzwerke transportiert werden, ist im Gesetz – nicht zuletzt wegen neuartiger, teilweise schwierig zu lösender Probleme – (noch) nicht vorgesehen (Stichworte z.B. „Transferfähigkeit“ und „Urheberrecht“, siehe [Schwen02]). Dennoch wurden in einer mehrjährigen Testphase und in Gesprächen mit Verlegern, Bibliothekaren, IT-Spezialisten und Regierungsvertretern die Bedingungen erprobt und ausgehandelt, unter denen Die Deutsche Bibliothek auch für Online-Publikationen Archivbibliothek sein kann.

---

[114] Die im Ausland eingesetzten Systeme sollen hier, mit Ausnahme von DSpace (siehe Kapitel 6.2), nicht näher untersucht werden, um den Umfang der Ausarbeitung zu beschränken.

[115] Beispielhaft sei das von der Universität Stuttgart entwickelte und weit verbreitete OPUS-System erwähnt.

[116] zur Geschichte des Pflichtexemplarrechts siehe z.B. [Kirch81]

[117] siehe ‚Gesetz über die Deutsche Bibliothek‘: http://bundesrecht.juris.de/bundesrecht/dbiblg/gesamt.pdf

Mit der Einrichtung eines entsprechenden Abgabeverfahrens beabsichtigte man,

- „einen weiteren Schritt auf dem Wege zur Bewältigung der Aufgabe Langzeiterhaltung elektronischer Publikationen zu gehen und die damit erzielten Ergebnisse und Erfahrungen in den internationalen Austausch einzubringen,
- für „transferfähige" Netzpublikationen geordnete und im Massenbetrieb anwendbare Übermittlungswege für Metadaten und Datenlieferungen zwischen Verlegern/ verlegenden Stellen und dem Archivsystem Der Deutschen Bibliothek einzurichten,
- die Konsistenz und Authentizität der archivierten Netzpublikationen zu sichern: nicht die Sammlung von Dateien ist das Ziel, sondern die Sammlung von Dokumenten als logische und konsistente Einheiten,
- gleichzeitig die Voraussetzungen für eine sofortige Benutzbarkeit der Publikationen im Archivsystem zu schaffen. Nur durch die Realisierung des Anspruchs auf umgehende und dauerhafte Benutzbarkeit wird die Situation eines „Frühwarnsystems" geschaffen, das auf Alterungsprozesse und entstehende Inkompatibilitäten aufmerksam macht und zum Handeln zwingt." [Schwen02]

Da man also grundsätzlich auch Netzpublikationen als ablieferungs- und sammelwürdig betrachtet, wurde zwischen Der Deutschen Bibliothek und dem Börsenverein des Deutschen Buchhandels eine Rahmenvereinbarung getroffen, die es Verlagen, Universitätsbibliotheken und sonstigen interessierten Einrichtungen auf freiwilliger Basis erlaubt, wichtige, jedoch nur im Internet publizierte Dokumente ebenfalls in das Depotsystem zu überführen. Eine Reihe von Paragraphen, die online einsehbar sind[118], regeln dabei ganz genau, wie Meldung, Übernahme und Zugriff zu erfolgen haben. Selektions- und Prioritätskriterien bestimmen außerdem, welche Publikationsformen in welchem Umfang und in welchem Format gespeichert werden. Faktoren sind dabei vor allem das Verlustrisiko für die Publikation (Vorhandensein eines analogen Äquivalents) und die technische Bewältigung des Transfers in das Archiv (siehe auch [Lieg01]).
Werden unterschiedliche Dateiformate angeboten, wird das funktional Umfassendste präferiert und zugriffsfähig gemacht (z.B. wegen zusätzlicher Layoutinformationen) – archiviert werden jedoch alle verfügbaren Versionen (also z.B. HTML und PDF). Um die Menge der abgelieferten Dokumente und den damit verbundenen Erschließungsaufwand dennoch in Grenzen zu halten, kommen für Die Deutsche Bibliothek nur zur öffentlichen Verbreitung bestimmte Werke, wie digital vorliegende Monographien oder Schriftenreihen, als „Sammelgut" in Frage: ausgenommen sind somit beispielsweise Akzidenzpublikationen, Patentschriften, öffentliche Kommunikation (eMails, Newsgroups), Werbung, Filme u.ä..

Für den Start des Depotsystems für Netzpublikationen am 1. Juli 1998 hatte man sogar ein noch engeres Auswahlspektrum festgelegt: *Online-Dissertationen* und digitale Publikationen von ausgewählten kommerziellen Verlagen. Speziell mit Hochschulschriften konnten die notwendigen Geschäftsgänge „hervorragend entwickelt und getestet werden, da es sich bei diesen im Verhältnis zur Gesamtzahl elektronischer Publikationen um eine kleinere Menge handelt, deren Datenformat- und Dateistrukturvielfalt noch überschaubar ist", wie von [Schwen00] festgestellt wird. Man wollte zunächst Erfahrungen mit einigen wenigen Publikationsformen sammeln – ohne sich jedoch langfristig auf deren Archivierung zu beschränken: Prämisse war, die gewonnenen Erkenntnisse später (vielleicht in modifizierter Form) auch auf andere Dokumenttypen übertragen zu können.

[118] http://deposit.ddb.de/netzpub/web_rahmenvereinbarung.htm

Für die Speicherung elektronisch vorliegender Dissertationen hat man sich allerdings auch deshalb entschieden, weil die Universitäten und Fachbereiche Anfang 1998 aufgrund des Beschlusses der Kultusministerkonferenz verstärkt damit begannen, ihre Promotionsordnungen um die Akzeptanz von Online-Hochschulschriften zu erweitern. Dieser bekanntermaßen durchaus positiven Entwicklung wollte Die Deutsche Bibliothek nicht entgegenstehen und so gingen dem auf der letzten Seite genannten Stichtag 1. Juli 1998 neben der Schaffung von technischen und organisatorischen Voraussetzungen eine Reihe weiterer notwendiger Vorbereitungen voraus:

- Ein Metadatensatz für die Beschreibung der Online-Dissertationen wurde festgelegt und auf dessen Basis ein HTML-Formular[119] auf die Homepage Der Deutschen Bibliothek gelegt, über das die Universitätsbibliotheken die Titel melden können.
- Die Recherche- und Darstellungsmöglichkeiten im DDB-OPAC wurden angepaßt und die für Offline-Publikationen bereits existierenden formalen Erschließungskriterien entsprechend erweitert.
- Ein eigener Dokumentenserver deposit.ddb.de wurde eingerichtet.
- Die Universitätsbibliotheken wurden über den Termin und die geplante Vorgehensweise informiert.
- (zusammengefaßt in Anlehnung an [Schwen00])

Die Resonanz war anfangs verhalten, da man in den Bibliotheken die notwendige Infrastruktur für die Aufnahme der Hochschulschriften zunächst erst selbst schaffen mußte. Inzwischen beteiligen sich aber fast alle deutschen Universitäten an dem freiwilligen Transfer der bei ihnen anfallenden Dissertationen ... und so auch die UB Potsdam. In den letzten Jahren wurden von hier aus ca. 100 Examensarbeiten an Die Deutsche Bibliothek geliefert – nicht viele, die Tendenz ist aber steigend.[120] Wie genau die Übergabe dabei erfolgt und welche neuen Entwicklungen es insbesondere in Sachen Metadatensatz gibt, soll nachfolgend kurz vorgestellt werden. Die Kenntnis des allgemeinen Workflows ist vor allem deshalb wichtig, um die zu implementierenden Funktionalitäten und die zusätzlichen Anforderungen an neuen Dokumentenserver besser abschätzen zu können.

Grundvoraussetzung für die Ablieferung von Netzpublikationen ist das Vorhandensein einer Anmelder-Identifikation und eines Paßwortes. Institutionen, die das Recht zur Verbreitung haben (Verlage, herausgebende Stellen), müssen daher einmalig einen entsprechenden Vordruck ausfüllen und an Die Deutsche Bibliothek schicken – erst dann ist die Nutzung der Übergabe-Schnittstelle möglich. Weiterhin dürfen nur Publikationen angemeldet werden, die mit speziell strukturierten Metadaten versehen wurden. Für deren Erzeugung und Versand können zwei Übertragungswege genutzt werden:
Wie bereits erwähnt, existiert zum einen ein interaktives Anmeldeformular, welches die Erfassung der Metadaten nach dem vereinbarten METADISS-Standard (siehe nächster Abschnitt) erlaubt. Und zum anderen kann seit März 1999 auch eine Mail an ep@ddb.de verschickt werden, die neben der ID der absendenden Institution alle notwendigen Angaben zur Publikation und zu dem/den Autor(en) beinhaltet. Zusätzlich zur Einbettung der Metadaten ist es dabei auch möglich, eine entsprechende HTML-Datei anzuhängen bzw. auf eine HTML-Seite zu verlinken, die Metaangaben enthält. HTML deshalb, weil das METADISS-Format zum gegenwärtigen Zeitpunkt auf der Hypertext Markup Language Version 4.0 basiert und nur Tags der Art <META NAME= ...> unterstützt. Vorteil des

[119] http://deposit.ddb.de/anmeldeformulare/dissmeld.htm
[120] wie der Transfer bisher erfolgte, wird im Kapitel 6.1 zusammengefaßt

letztgenannten Übergabeverfahrens: die Mail kann ggf. durch universitäre Verarbeitungssysteme automatisch erzeugt werden, so daß eine umständliche, manuelle und oft sogar doppelte Erfassung der Metadaten vermieden wird.

Unabhängig von der Wahl lösen beide Varianten anschließend innerhalb Der Deutschen Bibliothek einen Bearbeitungsablauf aus, in dessen Verlauf die jeweilige Online-Hochschulschrift von ihrem Ursprungsserver „abgeholt" und auf dem Archivserver „deposit.ddb.de" gespeichert wird. Bei Eindateiendokumenten ist dies problemlos möglich: existieren mehrere inhaltsgleiche Publikationen in unterschiedlichen Dateiformaten, wird entsprechend folgender Präferenzregelung das bevorzugte Archivierungsformat ausgewählt:[121]

1. XML/SGML 2. HTML 3. PDF 4. PS 5. Sonstige (.rtf, .doc, .tex, .dvi, .txt, etc.)	(Wegen der derzeit noch schlechten Präsentationsmöglichkeiten (Viewer-Verfügbarkeit) von XML- und SGML-Dokumenten sollten die gelieferten Metadaten nach Möglichkeit auch einen Verweis auf eine PDF-Version enthalten, so daß diese ebenfalls abgeholt und archiviert werden kann.)

Besteht eine Publikation hingegen aus mehreren Files (z.B. miteinander verlinkte HTML-Dateien), ist auf Seiten der abliefernden Bibliothek bzw. des Autors zunächst noch etwas Vorarbeit nötig: in Anlehnung an das OAIS-Modell ist ein Containerarchiv (SIP, Einlieferungsbehälter in den Formaten ZIP, TAR, GZ oder TGZ) zu erzeugen, welches die einzelnen Dateien und ein „_index.htm"-Navigationsfile enthält. Wie letzteres erstellt werden kann und genau auszusehen hat, ist sehr ausführlich auf den Webseiten Der Deutschen Bibliothek beschrieben und soll hier nicht weiter ausgeführt werden.[122]
Nach der Übernahme der Publikation in das Depotsystem findet ein Virencheck und die Berechnung eines MD5-Hashcodes statt, der es erlaubt, die Identität von Dokumentkopie in Relation zu der archivierten Referenzversion zu ermitteln.
Außerdem wird, basierend auf den gelieferten Metadaten, eine HTML-Frontpage generiert, die Titel, Verfasserangaben, Entstehungsort, Abstract und eine Übersicht der Formatvarianten beinhaltet (siehe auch Ende Abschnitt 5.2.2). Eventuell notwendige Viewer sowie der Authentisierungs-Code sind dort ebenfalls verankert – zusammen mit ‚versteckten' Dublin Core Metadaten, die eine bessere Indexierung durch Suchmaschinen ermöglichen.
Abschließend wird die Online-Hochschulschrift dann in der Reihe H der Deutschen Nationalbibliographie verzeichnet und ist sowohl über den DDB-OPAC[123], als auch über ein Z39.50-Interface recherchierbar.

Mit der Gestaltung der Eingangsschnittstelle zur Übernahme und Archivierung von digitalen Dissertationen und Habilitationen konnten wie gewünscht in einem abgegrenzten Bereich alle notwendigen Arbeitsschritte erprobt und ein funktionierender Geschäftsgang erarbeitet werden, der in ähnlicher Form inzwischen auch erfolgreich für andere Dokumenttypen, wie z.B. fortlaufend erscheinende Publikationen (elektronische Zeitschriften, Jahrbücher, Geschäftsbriefe u.ä.), angewandt wird. Für Einzelheiten die notwendigen ZIP-Container und den FTP-Transfer betreffend sei ein weiteres Mal auf die Webseiten Der Deutschen Bibliothek verwiesen.

---

[121] http://www.ddb.de/wir/pdf/praeferenzregelung.pdf
[122] http://deposit.ddb.de/meta_schnittstelle.htm - da von der UB Potsdam zunächst PDF als Archivierungsformat präferiert wird (für die Gründe siehe Kapitel 3.3), ist der Transfer von Containerdateien nur in Einzelfällen von Bedeutung
[123] http://dbf-opac.ddb.de

5.2.1 METADISS und METAPERS

Der bereits erwähnte und für die Übertragung der Hochschulschriften verbindliche Metadatensatz METADISS basiert auf dem Dublin Core Metadata Element Set und ist in Zusammenarbeit mit dem Projekt „Dissertationen Online“ entwickelt worden. Ziel des Teilprojekts „Metadaten“ war es dabei, den doch recht beschränkten Umfang des Dublin Core Standards (siehe Kapitel 3.4.1) so zu erweitern, daß er auch für die Erschließung von Dissertationen und Habilitationen genutzt werden kann. Ergebnis sind eine Reihe spezifischer Elementtypen, Qualifier und Attribute, die den Inhalt der Publikation genauer beschreiben und insbesondere für eine strukturierte Suche genutzt werden können. Zusätzlich zu Verfassername, Titel, Abstract u.ä. ist Die Deutsche Bibliothek für die Aktualisierung ihrer Personennamendatei (PND[124]) aber auch an ergänzenden Angaben zu den Autoren, wie Geburtsdatum oder Staatsangehörigkeit, interessiert. Sie möchte mehr über die Personen erfahren, deren Werke sie spiegelt und so wurde 2001 im Rahmen des Projekts Metalib[125] ein als METAPERS bezeichneter Personenmetadatensatz eingeführt, der zunächst separat auszufüllen war, inzwischen aber in das offizielle Anmeldeformular integriert wurde. Trotz dieser Zusammenführung beider Metadatensätze geschieht die Lieferung persönlicher Daten auf freiwilliger Basis und erfordert im Falle einer Eintragung in die PND natürlich das Einverständnis des Verfassers. Letzteres braucht dabei nicht explizit eingeholt werden: will der Autor beispielsweise seinen Geburtsort nicht preisgeben, läßt er das entsprechende Feld einfach leer. Die Unterscheidung zwischen obligatorischen und fakultativen Angaben findet sich auch in der Struktur-Definition der METAPERS- und METADISS-Elemente wieder. Hierzu ein Beispiel, entnommen aus der offiziellen Formatbeschreibung:[126]

Bezeichnung	DC.Description
Qualifier	LANG="[3-stelliger Sprachcode]"; ohne Angabe: LANG="ger"
Qualifier	SCHEME="freetext"; ohne Angabe: SCHEME="freetext"
Wiederholbar	Ja
Obligatorisch	Nein
Beschreibung	Es ist wünschenswert, eine Zusammenfassung der Hochschulschrift, möglichst in mehreren Sprachen anzugeben.
HTML-Syntax	<META NAME="DC.Description" LANG="[3-stelliger Sprachcode]" CONTENT="[Zusammenfassung der Hochschulschrift]">

Die Angabe eines Abstracts, obwohl sicherlich wünschenswert, ist offensichtlich keine Pflicht. Aber noch mehr läßt sich erkennen: das Basiselement mit dem Namen *Description* entstammt dem Dublin Core Set (DC) und wird über Qualifier näher spezifiziert. *Freetext* bedeutet hier, daß die Publikations-Beschreibung eingebettet vorliegt – ein SCHEME=“URL“ wiederum würde anzeigen, daß die Zusammenfassung als eigenständiges Online-Dokument abrufbar ist. Fehlen die Attribute, werden Standardwerte eingesetzt. Außerdem lassen sich mehrere Abstracts angeben; eines in deutsch und ein weiteres z.B. in französisch (LANG=“fre“). Die letzte Zeile zeigt schließlich die zu verwendende HTML-Syntax. Eine Definition in dieser Form ist immer dann notwendig, wenn die Metadaten nicht über das DDB-Meldeformular, sondern via Mail geliefert werden.

---

[124] Die PND wird als nationale Normdatei geführt und enthält alle Namen der Autoren aller deutschen Publikationen, die seit 1945 erschienen sind. Sie bietet damit ein Instrument, eindeutige Zuordnungen zwischen Personen und Publikationen herzustellen; Mehrfacharbeit bei der Ansetzung von Personennamen kann somit vermieden werden (siehe auch http://www.ddb.de/professionell/pnd.htm).

[125] http://www.ddb.de/professionell/projekte.htm#meta

[126] http://deposit.ddb.de/metadiss.htm

Wie allgemein üblich (siehe Kapitel 3.4.1), ermöglichen auch in METADISS und METAPERS Subelementnamen eine weitere Qualifizierung der Elemente:
<META NAME="DDB.Contact.ID" CONTENT="[Anmelder-Identifikation]"> beispielsweise gibt zusätzlich zu der in „DDB.Contact“-definierten Mail-Addresse des Ansprechpartners die zugewiesene Identifikationsnummer der abgebenden Institution an. Das vorangestellte DDB macht außerdem deutlich, daß es sich um Elementbezeichner handelt, die in Dublin Core nicht vorgesehen sind und für den Geschäftsgang extra vereinbart wurden.
Ähnlich verhält es sich im Metadatensatz für Personen. Auch hier ein Beispiel:

Bezeichnung	pc.name.alternative.official
Qualifier	scheme="ddb-mn-pns"
Wiederholbar	Nein
Obligatorisch	Nein
Beschreibung	Alle Bestandteile des offiziellen Namens des Autors werden angegeben.
HTML-Syntax	<meta name="pc.name.alternative.official" scheme="ddb-mn-pns" content="[Nachname, Vorname Präfix, Titel, Adelstitel]">
Hinweis	Der vollständige Name ist nur anzugeben, falls dieser vom Element "Name" abweichend ist. Die Einträge der Felder "Vorname", "Nachname", "Titel", "Adelstitel" und "Präfix" im Anmeldeformular werden gemäß der aufgeführten HTML-Syntax zusammengeführt.

Auffällig sofort (neben dem „pc“ als Elementtyp): Groß-/Kleinschreibung spielt für die Elementnamen und Attribute keine Rolle – sie ist lediglich für den Elementinhalt („content“-Feld) relevant.
Die Format-Spezifikation der hier vorgestellten Metadatensätze umfaßt in den derzeit aktuellen Versionen 1.5 (METADISS, Stand 05.05.2003) und 1.1 (METAPERS, Stand 07.07.2003) über 60 derartige Deklarationen. Die jeweiligen Eigenheiten aller verfügbaren Datenelemente aufzuzählen, würde den Rahmen der Ausarbeitung sprengen. Abschließend lediglich noch einige Festlegungen, die für eine konkrete Implementation von Bedeutung sind:[127]

Die Zeichencodierung erfolgt gemäß ISO-8859-1 (Latin-1). Erlaubt ist auch die nach HTML 4.0 zulässige Menge der „benannten Zeichen“ (z.B. Α), sowie Codierungen nach dem Universal Character Set UCS, ISO-10646 (z.B. ü). Datumsangaben müssen im ISO-8601-Format vorliegen („JJJJ-MM-TT“); der ISO-Standard 639-2 ist für die Codierung von Sprachangaben zuständig („russisch“ -> „rus“).
Zu beachten ist vor allem aber folgende Besonderheit: normalerweise kann die Reihenfolge der <META>-Tags beliebig gewählt werden – nicht so bei zusammengehörigen Element-gruppen, die als Ganzes wiederholbar sind!

> Der Name des Autors (DC.Creator.PersonalName), sowie dessen
> Geburtsdatum (DC.Creator.PersonalName.DateOfBirth),
> Geburtsort (DC.Creator.PersonalName.PlaceOfBirth) und
> Adresse (DC.Creator.PersonalName.Address)

bilden einen solchen, n-mal wiederholbaren Block: jedes erneute Auftreten des Verfasser-namens leitet eine neue Elementgruppe ein; die jeweils folgenden Angaben beziehen sich dabei auf das übergeordnete Element.

[127] Ziel ist, die Ablieferung der Metadaten durch Generierung von HTML-Mails so gut es geht zu automatisieren – mehr dazu in Kapitel 6.1

Der letztgenannte Punkt hat sich in der Vergangenheit als kritisch erwiesen: nicht selten kam es bei der Metadaten-Übermittlung via Mail zu Fehlern bei der Zuordnung einzelner Elemente ... URLs haben auf das falsche Dokument verwiesen; Autoren sind plötzlich in einem anderen Ort geboren worden. Anfragen über die Mailingliste[128] der Koordinierungsstelle DissOnline belegen dies – und über selbige kam Anfang August 2003 auch die Mitteilung, daß bis Ende des Jahres ein neues, auf XML basierendes Metadatenformat vorgelegt wird. Konkret heißt es

> „Für Zwecke des Datentransfers stellt die Einbettung der in METADISS definierten Elemente in HTML4 [...] nicht mehr den Stand der Technik dar. Deshalb setzen wir METADISS nach XML um." [Korb03]

Das Potential des XMetaDiss-genannten Formats besteht laut Ankündigung

- in der Nutzung von hierarchischen Strukturen und der damit verbundenen Vermeidung von Zuordnungsfehlern (bisher erfolgte die Zuordnung nur durch die Reihenfolge der Elemente),
- in Schaffung eines Formats für Online-Hochschulschriften, das per OAI-Protokoll ausgetauscht werden kann,
- in der Schaffung eines zum internationalen NDLTD-Set ETDMS[129] kompatiblen Formats; damit können deutsche Online-Hochschulschriften über internationale Metadatensuchmaschinen eingebunden werden,
- in der angestrebten gemeinsamen Festlegung eines XML Metadatensets für Online-Dissertationen mit der SLB (Schweizerische Landesbibliothek) und OeNB (Österreichische Nationalbibliothek)
- und in der einfachen Transformationsmöglichkeit mittels XSLT in andere Metadatenformate wie ETDMS und Dublin Core Simple

Grundsätzlich soll XMetaDiss alle inhaltsbeschreibenden und administrativen Datenelemente des METADISS-Formats enthalten und sich mittels XSLT-Transformation insbesondere auch in RDF-Strukturen einbetten lassen, so daß semantische Zusammenhänge noch besser wiedergegeben werden können. Alpha- und Beta-Versionen sollen jeweils als Diskussionsgrundlage noch bis November 2003 fertiggestellt werden – voraussichtlich im Mai 2004 wird METADISS dann von XMetaDiss abgelöst. Und obwohl es eine Übergangszeit geben wird, in der Publikationen weiterhin via HTML an Die Deutsche Bibliothek gemeldet werden können, ist es schon jetzt wichtig, die Entwicklung im Auge zu behalten, um rechtzeitig und möglichst problemlos auf XML umstellen zu können.

5.2.2 Uniform Resource Names

Die steigende Zahl elektronischer Publikationen, die wie beschrieben zu einer Erweiterung des Sammelspektrums Der Deutschen Bibliothek geführt hat, macht nicht nur die Anwendung substanzerhaltender Maßnahmen (Konvertierungen, Migrationen, ...) notwendig, sondern erfordert auch leistungsstarke Identifizierungs- und Adressierungsmechanismen, um eine Langzeitverfügbarkeit der archivierten Netzpublikationen zu gewährleisten. Bereits 1999 wurde im Rahmen der *Conference of Directors of National Libraries* (CDNL) die Empfehlung ausgesprochen, URLs durch stabilere Referenzierungen zu ersetzen und so hat sich Die Deutsche Bibliothek aktiv im Projekt CARMEN-AP4 „Persistent Identifiers and

[128] diss-online@ddb.de - Anmeldung unter http://www.ddb.de/professionell/dissonline_mail.htm
[129] NDLTD = Networked Digital Library of Theses and Dissertations - http://www.ndltd.org
ETDMS = Electronic Theses and Dissertations Metadata Standard

Metadata Management in Science“ (1999-2002) engagiert, um Erfahrungen in Bezug auf Verwaltung und Auflösung von beständigen Identifikatoren (siehe Kapitel 4.3) zu sammeln. Ergebnis ist ein auf Kooperation zwischen den Bibliotheksverbünden und Universitätsbibliotheken angelegtes, verteiltes URN-Management für Online-Dissertationen.
Entsprechend den Festlegungen der CDNL leiten sich die von Der Deutschen Bibliothek administrierten Uniform Resource Names dabei aus dem internationalen Namensraum „NBN“ (*National Bibliography Number*) ab und erhalten als Subnamespace-Bezeichner ein „DE“. Die hierarchische Vergabestruktur läßt sich allgemein wie folgt darstellen:

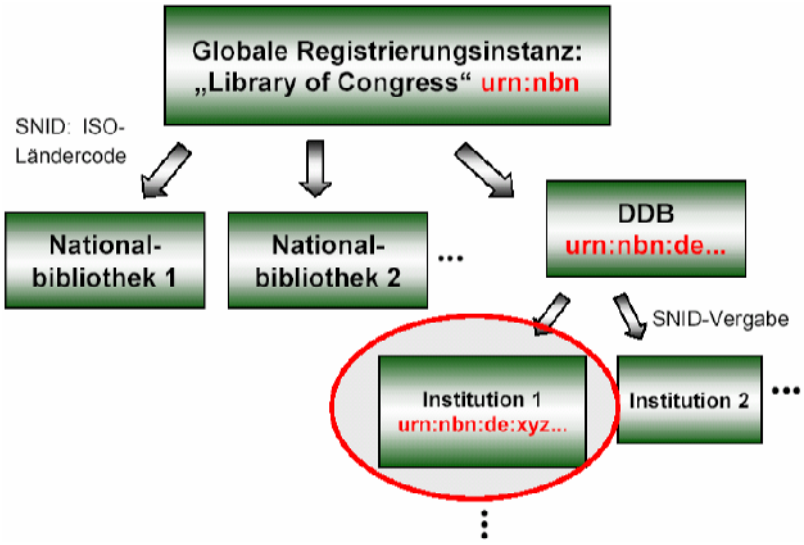


Abbildung 13: URN Namespace - National Bibliography Number

Das „xyz...“ in der aus [Pier02] entnommenen Abbildung wird wiederum nach folgendem Muster gebildet:

[Verbundabkürzung]:[Sigelnummer]-[Nummer][P]

[Verbundabkürzung] ist dabei ein Buchstabenkürzel des jeweiligen Bibliotheksverbunds[130], [Sigelnummer] steht für den Bibliothekssigel der entsprechenden Institution und [Nummer] muß die eindeutige ID eines bestimmten Dokuments sein. Erlaubt sind nur Kleinbuchstaben, Zahlen, Doppelpunkt und Bindestrich. Laut [PersID] dient dieser limitierte Zeichensatz „der Konfliktvermeidung bei Softwareanwendungen“. Aus gleichem Grund wird dem URN auch eine Prüfziffer [P] angehangen, die eventuelle Probleme bei der Transaktion (z.B. Tippfehler oder fehlende Teile) entlarven und eine Validierung des gesamten Strings ermöglichen soll. Ein Web-Formular[131] dient der Berechnung dieser Prüfziffer – erfreulicherweise ist der entsprechende Algorithmus auch verbal beschrieben[132] und kann somit selbst implementiert werden (siehe Kapitel 6.3.4).

Hier nun ein Beispiel für einen gültigen URN, (zukünftig) vergeben von der UB Potsdam:

urn:nbn:de:kobv:517-**0300001**5

Erläuterung: Die Universitätsbibliothek Potsdam ist Mitglied des Kooperativen Bibliotheksverbunds Berlin-Brandenburg (KOBV) und hat von Der Deutschen Bibliothek die Sigel-

[130] möglich ist hierbei „bsz“, „bvb“, „gbv“, „hebis“, „hbz“ und „kobv“
[131] http://nbn-resolving.de/nbnpruefziffer.php
[132] http://www.persistent-identifier.de/?link=316

nummer 517 zugewiesen bekommen. Die fett-markierte Zahl wurde hingegen frei gewählt und ist der **0001**ten im Jahre 20**03** gemeldeten Publikation (eindeutig) zugeordnet. Möglich wäre auch die Angabe einer laufenden Nummer, in jedem Fall ist aber die Institution selbst für die dublettenfreie Vergabe dieses *Namespace Specific Strings* verantwortlich. Durch diese „sich selbst organisatorisch tragende Nummernstruktur [...] entfallen ein zusätzlicher Verwaltungsaufwand und eine mögliche Zeitverzögerung [...], z.B. durch die Beantragung von Nummernkontingenten", so [PersID] zu den Gründen der dezentralen URN-Vergabe. Die 5 als letzte Ziffer der ID ist schließlich die ‚Checksum' über den gesamten unterstrichenen Bereich.

Ein solcher Uniform Resource Name soll als beständiger Identifikator eine bestimmte Hochschulschrift langfristig referenzieren und wird daher zusammen mit den Metadaten an Die Deutsche Bibliothek gemeldet. Möglich wird dies durch ein spezielles Feld im METADISS-Metadatensatz:

<meta name="DC.Identifier" scheme="URN:NBN:DE" content="urn:nbn:de:kobv:517-0300015">[133]

Außerdem notwendig ist natürlich die Angabe der Original-URL:

<meta name="DC.Identifier" scheme="URL"
content="[Protokoll://Servername/Unterverzeichnis 1/.../Unterverzeichnis n/Dateiname]">

Bei der Meldung von URN und URL(s) via Mail ist – ähnlich wie bei den Personendaten – darauf zu achten, daß zugehörige Zeilen in einem Block aufgeführt werden, um sie später einander zuordnen zu können. Gespeichert werden die gelieferten Identifier vom URN-Management-System, welches auch für das Resolving zuständig ist. Wie im Kapitel 4.3 bereits angedeutet, ist die URN-Darstellung hinsichtlich Skalierbarkeit und technischer Implementation zwar zukunftsweisend, derzeit aber noch nicht weltweit einsetzbar, da transparente, DNS-ähnliche Auflösungsmechanismen fehlen. Die Deutsche Bibliothek empfiehlt daher die Referenzierung eines URN als Hyperlink: [134]

http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Akobv%3A517-0300015

Die Zeichenkette %3A entspricht dabei der Codierung des Doppelpunkts, der in einer URL nicht vorkommen darf (siehe RFC1738). In einem Online-Katalog, aber auch in der Publikation selbst, sollten laut DDB nach Möglichkeit immer beide Darstellungsformen (also ‚reiner' URN *und* Resolver-URL) angeben werden, damit der Nutzer einen zitierbaren Identifier *und* die Kenntnis über einen möglichen Auflösungsdienst erhält.[135]
Wird ein solcher Hyperlink schließlich angeklickt, wird normalerweise direkt zur entsprechenden Original-URL (bzw. nacheinander zu alternativ angegebenen URLs) weitergeleitet.[136] Damit beim Resolving-Prozeß allerdings nur gültige Adressen ausgegeben werden, wird seitens Der Deutschen Bibliothek ein täglicher URL-Check durchgeführt. Bei Vorhandensein eines MD5-Hashwertes, der ebenfalls via Metadaten-Schnittstelle übermittelt werden kann, wird zusätzlich alle drei Monate die Integrität der zugehörigen Dokumente überprüft (siehe auch Kapitel 4.2). Wurden URLs bei derartigen Tests als verändert bzw. nicht erreichbar identifiziert, wird der Anwender beim Auflösen der URN automatisch auf

[133] durch Angabe von „DOI" als SCHEME und entsprechender Präfix/Suffix-Kombination ist auch die Verwendung *Digital Object Identifiers* möglich, soll hier aber nicht weiter vertieft werden

[134] neben der direkten Verlinkung ist auch die Nutzung eines Webformulars (http://nbn-resolving.de/Resolving.html) bzw. die Nutzung alternativer Resolving-Server möglich

[135] Auf den Frontdoor-Seiten Der Deutschen Bibliothek wird die Resolver-URL nicht (direkt) angezeigt, wie der Screenshot auf der nächsten Seite zeigt.

[136] Das Datenmodell des URN-Management-Systems erlaubt die Verwaltung und Auflösung einer 1:1 Beziehung (1 URN → 1 URL) und einer 1:n Beziehung (1 URN → mehrere URLs).

die Archiv-URL Der Deutschen Bibliothek verwiesen. Außerdem erhält die betroffene Universitätsbibliothek eine Mitteilung und die Möglichkeit, mit Hilfe entsprechender, paßwort-geschützter Webseiten und Formulare, (Link-)Korrekturen vorzunehmen.[137] Durch derartige Verfahren ist sichergestellt, daß keine URL-Adressenänderung unbemerkt bleibt. Die Langzeitverfügbarkeit und Referenzierbarkeit der elektronischen Ressource ist somit gewährleistet.

→ Und entsprechend intensiv werden die von Der Deutschen Bibliothek administrierten Persistent Identifiers auch genutzt: seit September 2001 wurden etwa 6.500 URNs für Online-Hochschulschriften mit über 60.000 Zugriffen registriert (Stand: 31.07.2003). Ca. 36% aller vergebenen URNs sind retrospektiv erfaßt worden und für knapp 2.300 Dokumente wurde außerdem die MD5-Prüfsumme vermerkt. 52 Universitätsbibliotheken haben sich bisher angemeldet, 35 davon nutzen das Registrierungsverfahren aktiv. Die UB Potsdam gehört auch dazu, hat laut Statistik[138] allerdings ‚nur' 67 mit einer URN versehene Publikationen liefern können, da es an einer entsprechenden Infrastruktur bisher weitestgehend gefehlt hat. Aber dies soll sich mit dem neu zu konzipierenden Dokumentenserver und den vielen geplanten Automatismen definitiv ändern.

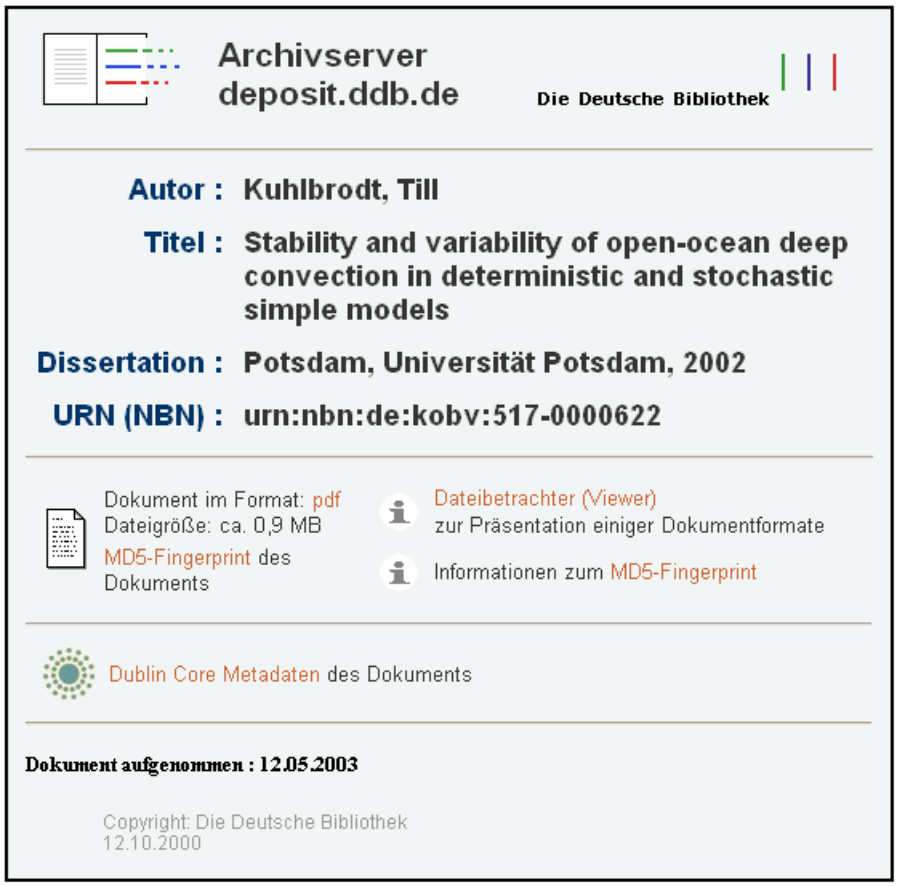


Abbildung 14: DDB Beispiel-Frontdoor-Seite

[137] Neue/retrospektive URN-Vergaben und nachträgliche Berichtigungen sonstiger Metaangaben sind natürlich ebenso möglich.

[138] http://www.persistent-identifier.de/?link=540

6 Konzeption des Dokumentenservers

Wie einleitend und auch an weiteren Stellen der vorliegenden Arbeit angedeutet, galt es nicht nur, die allgemeinen Aspekte der langfristigen Speicherung aufzuzeigen und anhand von Online-Dissertationen einen kleinen Einblick in konkrete Archivierungsbestrebungen zu geben, sondern das angesammelte Wissen auch selbst prototypisch umzusetzen. Ziel war die Schaffung einer „Digitalen Bibliothek“ in Form eines Dokumentenservers, der wissenschaftliche Publikationen der Universität Potsdam aufnehmen und geeignet zugriffsfähig machen kann. Keine leichte Aufgabe, wie die umfangreichen, aber bei weitem nicht allumfassenden Ausführungen der bisherigen Kapitel sicherlich deutlich gemacht haben. Die Thematik ist äußerst komplex – und die zur Verfügung stehende Zeit arg begrenzt, so daß leider nicht alle Ideen und Wünsche verwirklicht werden können. Dennoch wurde versucht, ein System zu schaffen, welches über einen „Proof Of Concept“ hinausgeht und stabil genug läuft, um auch tatsächlich produktiv eingesetzt werden zu können.

Schon frühzeitig wurde damit begonnen, alle Anforderungen auszuloten, die Nutzer und Bibliothek an einen modernen Publikationsserver stellen. Die nach und nach gewonnenen Erkenntnisse sind dabei zum einen in die eher theoretischen Betrachtungen der letzten Seiten, zum anderen aber natürlich auch in den praktischen Teil dieser Diplomarbeit eingeflossen: parallel zur schriftlichen Ausarbeitung wurde das jeweils aktuell ‚Gelernte‘ sofort programmtechnisch umgesetzt bzw. existierende Software auf entsprechende Potentiale hin untersucht – nur so konnte der zeitliche Rahmen eingehalten und das zusammengetragene Wissen optimal angewendet werden. Voraussetzung für diese Vorgehensweise war allerdings ein bestehendes Grundkonzept, in das die einzelnen Codefragmente nur noch eingebracht und zu einem Gesamtsystem vereint werden brauchten. Von Anfang an mußte also – zumindest in Grundzügen – bekannt sein, welche Funktionalitäten der zu implementierende Dokumentenserver später bieten sollte ... und ob nicht vielleicht sogar schon geeignete Dokument-Management-Systeme verfügbar sind, die eine Eigenentwicklung unnötig machen. Nachfolgend daher stichpunktartig nochmals alle Anforderungen, die mindestens erfüllt werden sollten:

- modernes Frontend, Zugriff via Internet und Browser
- Dublin Core-Metadaten
  - Anmelde-Formular für
    - autorenspezifische Metadaten
    - publikationsspezifische Metadaten
  - Speicherung in relationaler Datenbank
  - Metadaten-Suchmaschine
  - Browse-Funktionalität
    - Autorenliste
    - Titelliste
    - Fakultäts-/Institutsliste
  - OAI-Schnittstelle
  - Erzeugung von METADISS und METAPERS
    und Versand an DDB falls Dissertation/Habilitation
- Dokumente
  - Server nicht nur für Dissertationen
    - auch z.B. Preprints, Lehrmaterialien, Aufsätze, ...
  - Formatfestlegung
    - Urformat nach Möglichkeit strukturiert (HU-konform)

    - Präsentationsformat HTML/PDF(/PS)
    - Archivierungsformat = Urformat
      (langfristig SGML/XML für Migration)
  - Übertragung via
    - Browser-Upload
    - FTP
    - Download von anderem Server
    - Postversand (Disketten, CDROM)
  - Speicherung im Filesystem
    - Trennung in öffentlich zugänglichen Dokumentenserver
      und speziell geschütztes Archivsystem
  - Volltext-Suchmaschine (auch PDF)
  - automatische URN-Vergabe (DDB-konform)
  - automatische MD5-Hashwertbildung
  - Authentizitäts-/Integritätskontrolle
- Ingest-Prozeß/Archivierung in Anlehnung an OAIS
- Workflow-Management
  - paralleler Zugriff auf Anmeldungen
    - Kontroll-, Änderungsmöglichkeit
    - Ablehnung/Freischaltung
  - Mail-Versand (automatisch/manuell)
- Zugriffszähler

Auf eine Erläuterung der einzelnen Punkte kann an dieser Stelle weitestgehend verzichtet werden, da sie entweder trivial sind oder bereits implizit bzw. gar explizit in den entsprechenden Kapiteln behandelt wurden. Die vorgenommene Untergliederung ist außerdem eher kosmetischer Natur: natürlich lassen sich Dokumente und zugehörige Metadaten nicht voneinander trennen und z.B. losgelöst vom angedachten Workflow-Management[139] betrachten. Dennoch macht eine solche Aufteilung Sinn, um einen groben Überblick über zusammengehörige Funktionalitäten zu geben, die im konkreten System dann beispielsweise in gemeinsame Klassen oder Module integriert werden können (OAI-Schnittstelle z.B. basierend auf Metadaten-DB, Hashwert-Berechnung innerhalb der eigentlichen Dokumentverwaltung, u.ä.). Von entscheidender Bedeutung ist dabei natürlich die geeignete Zusammenführung aller der Komponenten, die für eine Benutzerinteraktion zuständig sind. Ein intuitiv zu bedienendes Webfrontend soll Anmeldeformulare und Recherchemöglichkeiten, aber vor allem auch workflow-relevante Features unter einer einheitlichen Oberfläche vereinen. Neben dieser browserbasierten und somit betriebssystem-unabhängigen Übergabe, Kontrolle und Veröffentlichung archivierungswürdiger Publikationen muß aber insbesondere auch für einen speziell geschützten Bereich gesorgt werden, der *nicht* via HTTP und nur mit bestimmten Tools zugänglich ist, um die Sicherheit der gespeicherten Dokumente nicht zu gefährden. Und schließlich sei noch ein letzter wichtiger Punkt erwähnt, der bisher noch gar nicht angesprochen wurde: die Zählung von Zugriffen. „Zu den Grundlagen bibliothekarischer Arbeit gehörte schon immer, Nutzung und damit auch Akzeptanz der bereitgestellten Medien zu analysieren [...]“, wie von [Berg00] festgestellt wird. Das zukünftige System sollte daher Protokollierungsmechanismen bieten und resultierende Statistiken derart aufbereiten, daß sie nicht nur von Bibliotheksmitarbeitern, sondern insbesondere auch von den Produzenten und Konsumenten der bereitgestellten Literatur öffentlich eingesehen werden können.

[139] hier ist eine Verwaltungskomponente gemeint, die den Bearbeiter bei der Kontrolle und Freischaltung neuer Publikationsanmeldungen durch Automatismen möglichst gut unterstützt - siehe auch Abschnitt 6.3.4

Alle in der obigen Liste aufgeführten Punkte sind Resultat einer umfangreichen Literaturrecherche und einer Evaluation real existierender Dokumentenserver anderer Universitäten – auch wenn in der vorliegenden Arbeit leider nur das Konzept der HU Berlin etwas genauer untersucht werden konnte. Die Idee war es, die Unterschiede des HU-Systems zu gängigen Lösungen in einem gesonderten Kapitel herauszuarbeiten, die vielen Gemeinsamkeiten aber lieber allgemein vorzustellen und die Erkenntnisse in den praktischen Teil einfließen zu lassen, ohne explizit auf das ein oder andere System eingehen zu müssen. Die jeweils gebotenen Funktionalitäten unterscheiden sich zumeist nur marginal voneinander, wie am Beispiel von OPUS[140], EVA[141] und MONARCH[142] unter anderem auch von [Kühn99] deutlich gemacht wird. Der angegebene Anforderungskatalog stellt somit eine Art Schnittmenge dar: im eigenen System sollte sich möglichst das wiederfinden, was auch von anderen Servern offeriert wird – nur so ist es den aktuellen Bedürfnissen gewachsen und gut für zukünftige Entwicklungen gerüstet. Aber wenn vielleicht auch nicht jedes Feature sofort umgesetzt werden kann, muß die Lösung zumindest einem Anspruch genügen: sie muß insgesamt eine Verbesserung im Vergleich zur bisherigen Praxis darstellen.

6.1 Bewertung der aktuellen Situation

Der angesprochene Wunsch nach Verbesserung kommt natürlich nicht von ungefähr: Mailkonversationen und persönliche Gespräche mit MitarbeiterInnen der UB Potsdam haben gezeigt, daß grundsätzlich Bedarf an einer ‚Modernisierung' der bestehenden technischen Infrastruktur besteht, daß diese mangels personeller Kapazitäten aber immer wieder aufgeschoben wurde. Der vor Jahren mit einfachsten Mitteln aufgesetzte Publikationsserver existiert daher noch heute – und dies fast unverändert: nach wie vor werden Dokumente mühsam per Hand auf den Rechner kopiert; und noch immer dient ein Text-Editor der manuellen Anpassung von Links und Layout aller HTML-Dateien. Keine wirklich befriedigende Lösung und so wurde zwischenzeitlich zumindest an einer Stelle für eine gewisse Automatisierung gesorgt: bei der Anmeldung neuer Publikationen.
Aufgrund geänderter Promotionsordnungen war abzusehen, daß insbesondere die Anzahl elektronischer Dissertationen beträchtlich zunehmen – und eine zeitnahe Bearbeitung auf herkömmlichem Wege kaum noch möglich sein würde. Aus diesem Grund wurde ein Webformular geschaffen, welches die Eingabe personen- und dokumentspezifischer Metadaten bereits durch den Autor erlaubt und so die sonst notwendige Erfassungsarbeit seitens der Mitarbeiter reduziert. Das dahinterliegende Perl-Script[143] bietet dabei schon erstaunlich viel: es generiert, basierend auf den gelieferten Metaangaben, zwei statische HTML-Seiten und verschickt diese via SMTP an die Abteilung Publikationen der Universitätsbibliothek. Soweit nichts besonderes, allerdings enthält die erste, „door.htm"-benannte Datei bereits den kompletten METADISS-Metadatensatz und kann somit direkt für die Meldung der Hochschulschrift an Die Deutsche Bibliothek genutzt werden (siehe Kapitel 5.2). Außerdem entspricht sie, von kleineren Nachbearbeitungen abgesehen, exakt der späteren Frontdoor-Seite – mit allen Angaben zur zugehörigen Publikation, wie Titel, Verfasser, Stichwörter und Abstract. Die zweite Datei („meta.htm") wiederum dient als sogenannter Laufzettel: auch dieser enthält die vom Autor vergebenen Metadaten, wird aber nicht online verlinkt, sondern

[140] Online Publikationsverbund der Universität Stuttgart - http://elib.uni-stuttgart.de/opus
[141] Elektronisches Volltextarchiv der Universität Karlsruhe - http://www.ubka.uni-karlsruhe.de/eva (früher VVV: Veröffentlichungs-Verzeichnis/Volltextarchiv)
[142] Multimedia Online Archiv Chemnitz - http://archiv.tu-chemnitz.de
[143] als stud. Hilfskraft der EDV-Abteilung der UB hat der Autor bei der Bereitstellung des Servers und bei allen Implementationen mitgewirkt

ausgedruckt an den zuständigen Fachreferenten verschickt und dort mit der entsprechenden RVK-Notation (RVK: Regensburger Verbundklassifikation) versehen.
Doch trotz dieser Arbeitserleichterung bleibt für die MitarbeiterInnen (neben der gleichwohl weiterlaufenden Bearbeitung analoger Medien) noch viel zu tun ... *zu viel*, wie folgende Workflow-Beschreibung leider eindrucksvoll verdeutlicht:[144]

0. Dokument und Metadaten entgegennehmen	Geliefert werden: 1. Dokument im Präsentationsformat (i.d.R. PDF) falls nicht: Konvertierung doc → PDF 2. Dokument im Urformat 3. Metadaten durch Ausfüllen des elektronischen Anmeldeformulars, daraus generiert: door.htm und meta.htm 4. Copyright-Erklärung Metadaten und Dokument auf temp-Pfad auf C:\ legen
1. Door-Seite editieren	
1.1 Metadaten bearbeiten	
1.1.1 notwendige Ergänzungen vornehmen	Daten ISO-gerecht eintragen: z.B. 2002-07-30
	Geburtstag, Geburtsort trennen
	Sprache ISO-gerecht eintragen: <META NAME="DC.Language" SCHEME="ISO639-2" CONTENT="ger"> bzw. „eng“
	Namen der Betreuer in der Form: <META NAME="DC.Contributor.Advisor" CONTENT="Willmitzer, Lothar; Prof. Dr."> Vornamen möglichst ermitteln
	Abstracts durchgehen, und <p> entfernen, Zeilen in Monitorbreite mit Entertaste auf „!“ am Zeilenende achten -> löschen
	Dateiformat eingeben: z.B. application/pdf <META NAME="DC.Format" SCHEME="IMT" CONTENT="application/pdf">
1.1.2 URL vergeben	Lfd. Nr. pro Jahr, z.B. <META NAME="DC.Identifier" SCHEME="URL" CONTENT="http://pub.ub.uni-potsdam.de/2002/0010/brandt.pdf">
1.1.3 URN vergeben	<META NAME="DC.Identifier" SCHEME="urn:nbn:de" CONTENT="urn:nbn:de:kobv:517-000041x"> Prüfziffer ermitteln: http://nbn-resolving.de/nbnpruefziffer.php Prüfziffer eintragen: <META NAME="DC.Identifier" SCHEME="urn:nbn:de" CONTENT="urn:nbn:de:kobv:517-0000410">
1.2 Frontdoor-Tabelle bearbeiten	URLs und URN ergänzen (Überschrift verlinken + Eintrag in entspr. Zeilen)
	Abstract durch Entertaste zeilenweise auf Monitorbreite bringen und <p> belassen; auf „!“ am Zeilenende achten, ggf. löschen!
	Bearbeitungsdatum im Fuß
1.3 Umlaute in den Header	Header kopieren → in ein „blank Dokument“ einfügen → maschinell umls ersetzen durch Umlaute → kopieren → Einsetzen des geänderten Headers in die Frontdoor
2. Dokument bearbeiten	
2.1 Lesezeichen erstellen	Navigationsfenster öffnen; i.d.R.: 1. Gliederungspunkte aus dem Inhaltsverzeichnis mit Textwerkzeug markieren → „neues Lesezeichen“ 2. Dokument durchgehen und Lesezeichen setzen („Lesezeichenziel festlegen“) 3. Offen lassen bis in die 3. Ebene (je nach Umfang auch nur 2. Ebene)
	Einstellen, daß Lesezeichen mit geöffnet wird: Datei → Dokumentinfo → öffnen → „Lesezeichen und Seite“
2.2 Sicherheits-Optionen festlegen	Speichern unter → Sicherheit: „Standard“ → Einstellungen: Kennwort für Ändern der Sicherheitsoptionen vergeben nicht zulässige Optionen ankreuzen: - Dokument ändern - Text/Grafik auswählen

[144] diese Original-Tabelle, ursprünglich als Gedächtnisstütze gedacht, wurde von Dagmar Schobert, Abt. Publikationen der UB Potsdam freundlicherweise zur Verfügung gestellt

	Titelblatt für spätere Prozesse 2x ausdrucken 1x in Kartei Dissertationen stellen (mit URL und URN)
2.3 MD5-Fingerprint berechnen	Start → DOS → Verzeichnis suchen → C:\WPWIN60\tempanmeldung\0010> Schreiben z.B.: C:\WPWIN60\tempanmeldung\0010>md5 brandt.pdf Ergebnis z.B.: 472defefd512debacc14f1767a8da81f brandt.pdf Kopieren in Header: <META NAME="DDB.Identifier.Fingerprint" SCHEME="md5" CONTENT=" 472defefd512debacc14f1767a8da81f brandt.pdf"> und in die Frontdoor-Tabelle
2.4 Door-Seite, Dokument und Originaldateien auf den Server kopieren	1. Verzeichnis suchen/anlegen: E:/htdocs/2003/lfd. Nr. PDF Datei hinkopieren 2. Metadatenpfad suchen/Verzeichnis anlegen: E:/htdocs/2003meta/lfd. Nr. door-Seite hinkopieren 3. Save-Pfad suchen/Verzeichnis anlegen: E:/save/2003/lfd.Nr. Originaldateien hinkopieren ggf. vorher verzippen
3. TESTEN!	Alle Funktionalitäten prüfen
4. Mail an Promovenden und an Dekanat	Mustermail verwenden e-mail-Adresse der Publikationsanmeldung entnehmen im Text Namen ersetzen, URL korrigieren, evtl. Besonderheiten einfügen Betreff: Ihre Dissertation auf dem Publikationsserver der Universitätsbibliothek Potsdam; Kopie an jeweilige Dekanatsassistentinnen: Dekanat Math.-Nat.: jszyska@rz.uni-potsdam.de Phil. Fak.: heisz@rz.uni-potsdam.de Humanwiss.: rudtke@rz.uni-potsdam.de Anhang PND-Fragebogen der DDB nicht vergessen!
5. Inventarisierung + Titelaufnahme in OPAC und Universitäts-bibliographie	1. Inventarisierung ohne Inventarnummer Gift / Medienf.: SO / Publ.-Art: M
	2. OPAC Exemplar nicht vergessen: LKZ: 9300 | ohne Bestandstyp | Standortsignatur = URL der Frontdoor
	3. Universitätsbibliographie Kopie aus OPAC durch Alt A → Shift C → ergänzen: #0c 2++, #68, #68a, #76u, #90b Link zum Volltext durch: #90b9300 = http://pub.ub.uni-potsdam.de/2003meta/0001/door.htm
6. Notationsvergabe	1. Dokument an Fachreferenten: Datei meta.htm ausdrucken (kam mit der Anmeldung), fehlende Daten eintragen Laufzettel „Neuzugang Einzelstück Elektronische Publikation“ ausfüllen; Markierung Express Laufzettel mit Titelblatt und meta.htm an Fachreferenten
	2. vom FR vergebene Notation ggf. in #30 der Titelaufnahme ergänzen Notation in der door.htm eintragen (1. in Metadaten, 2. in Tabelle)
7. Aktualisierung der Door-Seite	Door-Seite in den Metadatenpfad auf dem Server kopieren
8. Door-Seite verlinken	Wenn o.k. vom Promovenden: 1. Titel eintragen in alphabetische Liste (alphabet.htm) und in Liste der Fakultät beide Einträge mit door-Seite verlinken 2. alphabet.htm und die entspr. Fakultätsseite mittels Total Commander auf den Server schieben 3. testen
9. Meldung an die DDB	1. Wenn PND-Fragebogen zurück: http://deposit.ddb.de/cgi-bin/metapers.pl ausfüllen und absenden 2. Header in eine leere Mail kopieren an: ep@ddb.de Betreff: Melde-ID/Hochschulschrift/Name, Vorname/DNB z.B. L6000-0724/HS/Werner, Deljana/17 → absenden

Soweit der, durchaus als aufwendig zu bezeichnende, bisherige Workflow. Geht man die Liste Punkt für Punkt durch, wird schnell klar, daß alle Arbeitsschritte – insbesondere bei der Bereitstellung von Examensarbeiten – von Bedeutung sind und daher keinesfalls weggelassen werden dürfen. Ebenso fällt aber auf, daß sich vieles äußerst gut automatisieren und somit massiv vereinfachen ließe. Beispielhaft seien hier die eigentlich unnötigen Ergänzungen im generierten Metadatensatz erwähnt: der gesamte Punkt 1 ist problemlos durch geeigneten Programmcode ersetzbar; selbst der Uniform Resource Name kann automatisch vergeben werden, ist der Prüfziffer-Algorithmus doch allgemein bekannt.

Hauptziel der praktischen Arbeit war es also, derartige Optimierungsmöglichkeiten aufzudecken und die insgesamt notwendigen Workflow-Schritte auf ein erträgliches Maß zu reduzieren, so daß wieder mehr Zeit für die Bewältigung anderer Aufgaben bleibt. Hierzu sollte mittelfristig natürlich vor allem auch die lokale Umsetzung des HU-Konzepts zählen: nicht umsonst wurden entsprechende Projekte näher vorgestellt und gängige Dateiformate sehr ausführlich im Hinblick auf Langzeitarchivierung und Retrieval untersucht. Nur standardisierte, nicht-proprietäre Formate wie SGML und XML haben wirklich das Potential, auch dauerhaft einsetzbar zu sein und so müßten sie eigentlich die Grundlage für den neuen Dokumentenserver bilden. Daß dem nicht so ist – und zum gegenwärtigen Zeitpunkt auch leider nicht sein kann, wurde bereits in den Kapiteln 3.3 und 5.1 angedeutet: ein Einsatz der genannten Auszeichnungssprachen ist wegen unerläßlicher Strukturierungs- und Konvertierungsmaßnahmen mit einem erheblichen Mehraufwand für alle Beteiligten verbunden. Ohne zusätzliches Personal[145], umfangreiche Umstrukturierungsmaßnahmen und intensive Schulung der Autoren lassen sich die HU-internen Arbeitsabläufe nicht auf die UB Potsdam übertragen – Zielsetzung sollte es aber dennoch bleiben.

Dies zur aktuellen Situation; zur Bewertung des bisher eingesetzten Publikationsservers sei abschließend aber noch auf den wohl gravierendsten Mangel hingewiesen: der fehlenden Datenbankanbindung! Tatsächlich wurden bislang alle anfallenden Daten lediglich in den vielen statischen HTML-Seiten ‚gespeichert' – von einer Metadatenbank keine Spur. Auch dieser Umstand ist auf zeitliche und personelle Engpässe zurückzuführen, muß sich mit dem neuen System aber definitiv ändern, da nur so strukturelle Suchanfragen und z.B. eine dynamisch aktualisierte OAI-Schnittstelle möglich sind.

Trotz des leider nur eingeschränkt möglichen Supports seitens der EDV-Abteilung (andere wichtige Projekte standen im Vordergrund, wie z.B. die Einführung eines neuen Bibliotheks-systems) hat die Abteilung Publikationen in den letzten Jahren bemerkenswerte Arbeit geleistet: in mühevoller Handarbeit wurden – vorrangig durch nur eine Person – Dokumente verlinkt, Hashwerte berechnet, URNs vergeben, Mails verschickt und vor allem auch Inhaltsverzeichnisse erstellt, die letztlich ein Browsing nach Verfassern und Titeln erst ermöglichten. Hier war es ein persönliches Bedürfnis des Autors, der all dem beratend zur Seite stand, durch die Entwicklung eines einfach zu pflegenden Systems zumindest eine gewisse, aber doch spürbare Arbeitserleichterung zu schaffen.

[145] die Arbeitsgruppe „Elektronisches Publizieren" der Humboldt-Universität besteht gegenwärtig aus immerhin 14 Mitarbeitern - die Abt. Publikationen der Universitätsbibliothek Potsdam aus 3

6.2 Analyse existierender Systeme

Das Fazit des letztens Abschnitts und die auch sonst häufig erwähnten *Implementationen* haben die Entscheidung des Autors bereits vorweggenommen: nicht die Installation eines existierenden Dokument-Management-Systems, sondern die Entwicklung eigener Komponenten war Ziel der letzten Monate. Der Grund hierfür ist einfach: nur so konnte das System optimal an die lokalen Bedürfnisse angepaßt und in den bestehenden Workflow eingebunden werden. Ebenso wichtig war es dem Autor aber auch, noch mehr Erfahrung in Sachen Webprogrammierung zu sammeln und dabei Standards wie XML und SQL auch selbst praktisch anzuwenden. Der Einsatz fertiger Lösungen hat sicherlich seine Vorzüge, macht es aber unnötig, sich intensiver mit Speicherungs- und Zugriffsmechanismen auseinanderzusetzen, da entsprechende Funktionalitäten bereits implementiert sind. Der Wunsch war es, Theorie und Praxis bestmöglich miteinander zu verbinden und das angesammelte Wissen in eine *eigens* konzipierte OAI-Schnittstelle sowie z.B. in eine Metadaten-Suchmaschine zu überführen.

Doch trotz dieses Bedürfnisses, persönlich tätig zu werden, wurden zu Beginn der praktischen Arbeit bestehende Archivierungssysteme auf ihre Potentiale hin untersucht. Zum einen, um deren Arbeitsweise besser zu verstehen; vor allem aber, um Anregungen für den neuen Dokumentenserver zu sammeln. In die engere Auswahl kamen schließlich zwei Lösungen, die dem aufgestellten Anforderungskatalog am ehesten entsprechen: OPUS der Universität Stuttgart und DSpace des MIT[146]. Beide Produkte sind als Open Source (theoretisch) frei verfügbar und beliebig an die eigenen Bedürfnisse anpaßbar. Während dies auf DSpace uneingeschränkt zutrifft, wäre für das Herunterladen des OPUS-Quelltextes, der mit seiner DDB-URN- und Metadatensatz-Unterstützung natürlich bestens auf deutsche Verhältnisse zugeschnitten ist, eine „Schutzgebühr“ von 250 Euro fällig gewesen[147] – eine Investition, die sich im Hinblick auf die angestrebten Eigenentwicklungen nicht wirklich gelohnt hätte. Aus diesem Grund haben sich die Untersuchungen letztlich auf DSpace konzentriert, welches nachfolgend anhand von Grafiken und Screenshots kurz vorgestellt werden soll.

Wie überall auf der Welt hat man auch in den MIT Libraries „[...] the problem of storing and retrieving [...] intellectual work over the long term“[148] erkannt und seit Anfang 2000 verstärkt damit begonnen, ein geeignetes Institutional Repository aufzubauen. Ergebnis ist das in Kooperation mit Hewlett-Packard entwickelte und im November 2002 in der Version 1.0 fertiggestellte DSpace, welches – veröffentlicht unter BSD-Lizenz[149] – inzwischen auch an weiteren (insbesondere amerikanischen) Einrichtungen eingesetzt wird. Der Grund für diese Verbreitung: das im Quelltext kostenlos herunterladbare System enthält alle Komponenten, die ein moderner Dokumentenserver bieten sollte[150] und ist zu alledem relativ einfach zu installieren und zu handhaben. DSpace ist damit geradezu prädestiniert für Bibliotheken, die keine eigene Entwicklungsarbeit aufwenden wollen oder können. Hier die Features im Überblick:

- Es orientiert sich am OAIS-Referenzmodell: alle dort definierten Bestandteile wurden in DSpace umgesetzt und auch entsprechend betitelt. Eine *Ingest*-Komponente ist beispielsweise für die Annahme von *SIP*s zuständig und überführt diese, kontrolliert durch

[146] Masssachusetts Institute of Technology
[147] dieser Preis wurde bei einer OPUS-Präsentation am 15.07.2003 in der FU Berlin genannt; weiterführende Informationen zum Online Publikationsverbund der Universität Stuttgart siehe: http://elib.uni-stuttgart.de/opus
[148] siehe http://www.dspace.org/implement/case-study.pdf
[149] http://www.opensource.org/licenses/bsd-license.php
[150] zu vorhandenen Einschränkungen siehe weiter unten

Administration und *Management*, in *AIP*s und zugehörige Metadaten (*Descriptive Infos*, siehe auch Kapitel 4.4).

- Die teilweise von den Autoren, teilweise vom System vergebenen Metadaten werden Dublin Core-konform gespeichert, wobei die Qualifier beliebig vergeben werden können.
- Ein ausgefeiltes Datenmodell ermöglicht die Klassifizierung und hierarchische Gliederung der AIPs: auf unterster Ebene (und im Filesystem abgelegt) befinden sich die eigentlichen Dokumente in Form von *Bitstreams* eines bestimmten *Formats*. Diese sind in *Bundles* organisiert, welche wiederum als jeweils zusammengehörige *Items* inklusive *DC Records* in einer relationalen Datenbank gespeichert werden. Gleichartige Materialen, wie z.B. multimedial aufbereitete Dissertationen, werden als *Collection* aufgefaßt – und mehrere dieser Collections bilden schließlich eine *Community*. Die nachfolgende Abbildung verdeutlicht – ebenso wie das aus der DSpace-Dokumentation[151] entnommene Beispiel – das Zusammenspiel:

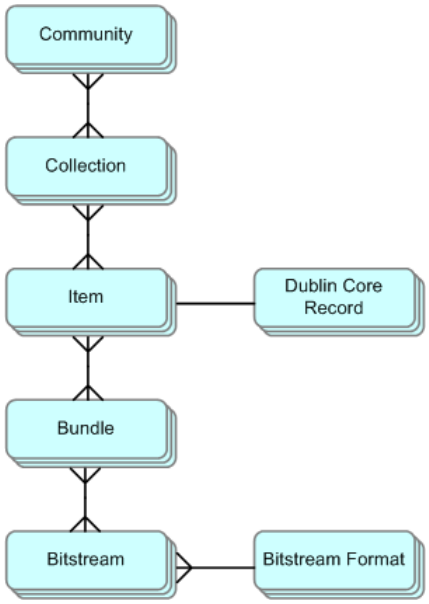


Abbildung 15: DSpace Datenmodell

Community	Laboratory of Computer Science; Oceanographic Research Center
Collection	LCS Technical Reports; ORC Statistical Data Sets
Item	A technical report; a data set with description; a video recording of a lecture
Bundle	A group of HTML and image bitstreams making up an HTML document
Bitstream	A single HTML file; a single image file; a source code file
Bitstream Format	Microsoft Word version 6.0; JPEG encoded image format

- Als beständige Identifikatoren werden sogenannte Handles eingesetzt, die für jedes Item lokal vom System vergeben und vom globalen CNRI Handle Service aufgelöst werden. Hier ein Beispiel für einen Identifier und die zugehörige Resolving-URL:

  hdl:1721.123/4567 → http://hdl.handle.net/1721.123/4567

- Die gespeicherten Metadaten werden über eine integrierte OAI-Schnittstelle[152] für Service Provider zugriffsfähig gemacht.

[151] http://dspace.org/technology/function.html
[152] hierbei kommt das OAICat-Framework von OCLC zum Einsatz: http://www.oclc.org/research/software/oai/cat.shtm

- Verfügt die jeweilige Institution über einen SFX-Server, kann DSpace OpenURLs erzeugen und auch auf OpenURL-Anfragen in Form von Suchergebnissen reagieren.[153]
- Eine History-Funktion protokolliert alle Veränderungen im System und erlaubt es später, Aktionen rückgängig zu machen. Der Zustand jedes einzelnen Objekts wird mit Hilfe von Harmony/ABC als RDF-Metadatensatz gespeichert.[154]
- Die integrierte Lucene-Searchengine[155] erlaubt eine Stichwortsuche in den archivierten Metadaten. Außerdem ist eine Browse-Funktionalität implementiert (Titel-, Verfasser- und Publikationsdatums-Liste).
- Während Suchanfragen (im Normalfall) anonym möglich sind, dürfen neue Publikationen nur von autorisierten Nutzern eingestellt werden. Diese müssen sich einmalig mit Mailadresse und Paßwort registrieren und haben dann Zugang zu einem personalisierten Bereich. Mehrere Nutzer sind in Gruppen zusammenfaßbar, deren Zugriffsmöglichkeiten durch die Vergabe spezifischer Rechte (READ, WRITE, ADD, REMOVE) auf bestimmte Collections, Items und sonstige Objekte beschränkt bzw. ausgeweitet werden kann.
- Registrierte User können sich für Collections ‚subscriben' und erhalten zukünftig Mails, sobald entsprechende Dokumente eingegangen sind.
- Neue Publikationsanmeldungen können mit Hilfe eines Workflow-Managements-Systems vor der Veröffentlichung kontrolliert und gegebenenfalls abgelehnt werden. Ähnlich wie bei der HU Berlin landen die Dokumente und zugehörigen Metadaten dazu (bei Bedarf) in einem gemeinsamen Pool und können von den Administratoren sowie speziell ausgezeichneten Benutzern abgearbeitet werden (siehe auch Kapitel 5.1).
- Als Client-Server-Applikation läßt sich DSpace vollständig browserbasiert bedienen und vereint alle Komponenten unter einer einheitlichen Oberfläche.
- Zusätzliche kommandozeilenorientierte Tools erlauben es, existierende Dokumente und Metadaten ohne Webfrontend in das Repository zu übernehmen, was die Migration aus anderen Systemen erleichtert. Ebenso ist ein Export von Items möglich, welcher z.B. für Backupzwecke genutzt werden kann.

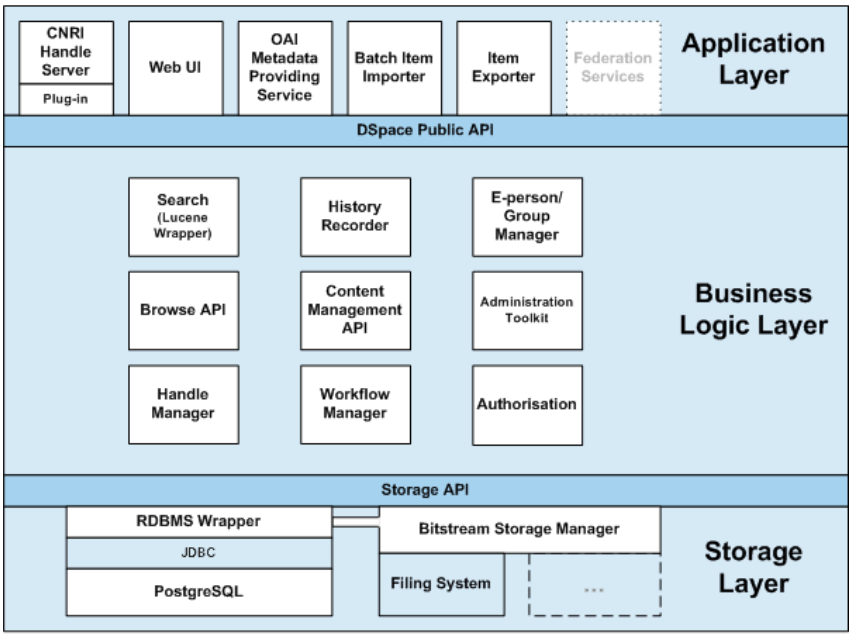


Abbildung 16: DSpace System Architektur

[153] „context sensitive reference linking"-System - http://www.sfxit.com/openurl/openurl.html
[154] weiterführende Informationen unter http://www.metadata.net/harmony
[155] http://jakarta.apache.org/lucene

Die im untersten Layer der Abbildung 16 angegebene JDBC-Kopplung des Datenbank-Wrappers an PostgreSQL läßt bereits erkennen, daß DSpace in Java implementiert ist und ein „UNIX-like OS“ wie Linux oder Solaris voraussetzt.[156] Neben dem Java-SDK und einigen speziellen Klassenbibliotheken wird vor allem die Servlet- und JSP-Engine benötigt, die später für die Generierung der dynamischen Webseiten zuständig ist.[157] „Ant“ hilft bei der Übersetzung des Quelltextes und als Webserver kommt standardmäßig Tomcat zum Einsatz, der mittels Apache bei Bedarf um eine SSL-Unterstützung erweitert werden kann. Genauere Angaben zu den jeweils ‚harmonierenden‘ Softwareversionen und zu den einzelnen Installationsschritten finden sich in der DSpace-Dokumentation.[158]

Nach korrekter Konfiguration und Einrichtung z.B. einer Community „Publikationsserver“ inklusive Collection „Dissertationen“ seitens des Administrators präsentiert sich das System wie folgt:

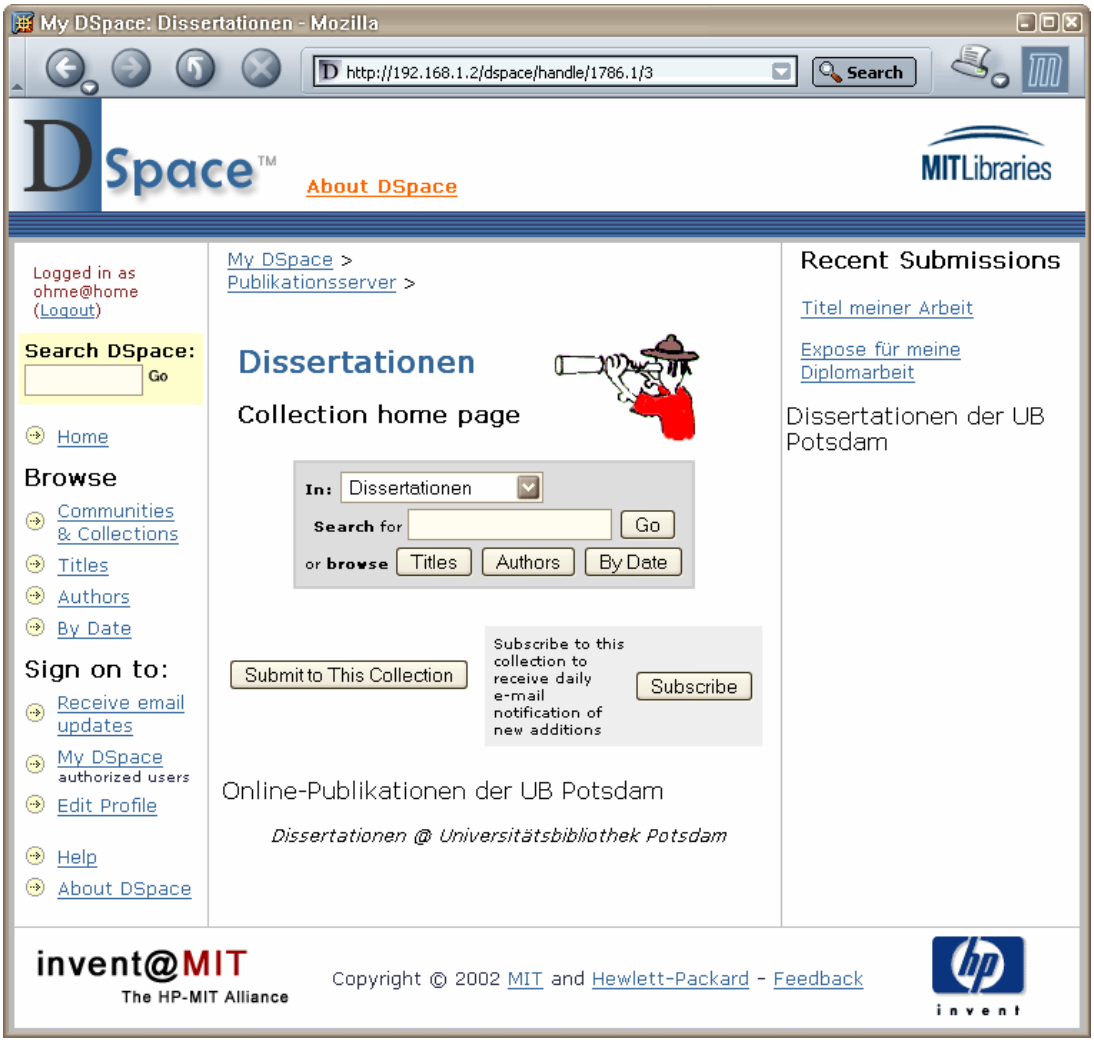


Abbildung 17: DSpace Collection-Einstiegsseite

[156] der Einsatz unter Windows ist zwar möglich, allerdings nicht unbedingt zu empfehlen, da das verwendete Datenbanksystem PostgreSQL dort nur mit Hilfe des ‚Unix-Emulators‘ CygWin läuft

[157] weiterführende Informationen zum Software Development Kit und zu Java Servlets bzw. Java Server Pages finden sich unter http://java.sun.com

[158] http://dspace.org/technology/system-docs

Die Oberfläche wirkt aufgeräumt, alle wichtigen Features sind sofort zugänglich: es kann gesucht und ‚gebrowset' werden, alle Neuzugänge sind sichtbar und es ist möglich, sich auch via Mail über neu eingestellte Dokumente informieren zu lassen. Der aktuell eingeloggte Nutzer hat neben der Änderung seiner persönlichen Daten außerdem die Möglichkeit, über den Submit-Button selbst Publikationen für die Veröffentlichung anzumelden. Das zugehörige Formular wurde dabei entsprechend des Inhalts und aus Gründen der Übersichtlichkeit auf mehrere Seiten verteilt: in sieben Schritten können personen- und dokumentspezifische Metadaten angegeben, Datei(en) auf den Server übertragen, die Eingaben überprüft, Lizenzbestimmungen akzeptiert und die Daten schließlich abgeschickt werden. Zwischen den einzelnen Seiten kann beliebig gewechselt werden (der Fortschritt wird am oberen Bildschirmrand angezeigt) – und auch ein Abbruch bzw. eine spätere Fortsetzung des Anmeldeprozesses ist möglich. Ähnlich gut gelöst ist das Hinzufügen weiterer Eingabefelder: existiert neben einer ISBN-Nummer beispielsweise auch eine ISSN, kann diese via „Add More" nachgetragen werden.

Der folgende Screenshot zeigt das zweite der sieben auszufüllenden Formulare:

Abbildung 18: DSpace Anmeldeformular

Publikationen und gemeldete Metadaten sind normalerweise nicht sofort für die Öffentlichkeit zugänglich. Eine zwischenschaltbare Workflow-Komponente erlaubt es bestimmten Nutzern, vor der Freischaltung einen Blick auf die Daten zu werfen und entweder ihr OK zu geben, oder die Aufnahme ins Repository abzulehnen. Im persönlichen Bereich des jeweiligen Bearbeiters[159] werden dazu – wie unten auszugsweise dargestellt – alle neuen und für ihn relevanten Dokumente aufgelistet, die er eingehend kontrollieren und gegebenenfalls auch wieder in den gemeinsamen Taskpool zurückstellen kann.

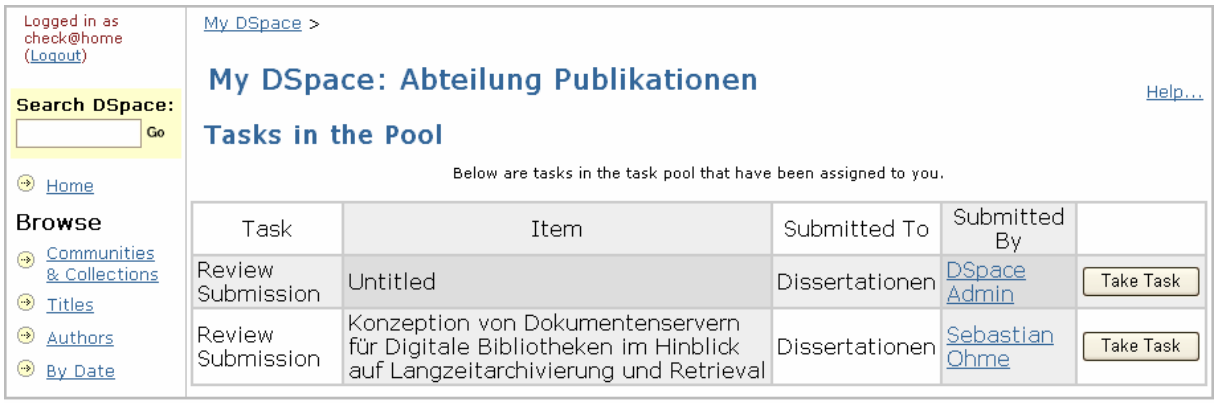


Abbildung 19: DSpace Taskpool

Sind schließlich alle Voraussetzung für die Veröffentlichung erfüllt, werden die bisher temporär vorgehaltenen Dokumente und Metadaten durch Klick auf einen entsprechenden „Approve"-Button in ein bestimmtes Verzeichnis bzw. in die Haupt-Datenbank kopiert und somit für alle Nutzer (mit entsprechenden Leserechten) zugänglich gemacht. Außerdem wird ein URN in Form eines eindeutigen Handles vergeben, der zur Frontdoor-Seite[160] führt und zukünftig anstelle der URL als Referenz verwendet werden sollte.

Während Userinterface, Anmeldeverfahren, Workflow-Management, Benutzerverwaltung und die hier nicht näher vorgestellten OAI-, OpenURL- und Browse-Funktionalitäten noch durchaus zu gefallen wissen, trübt insbesondere eine Komponente den guten Gesamteindruck: die Suchmaschine. Diese ist, insbesondere im Hinblick auf die z.B. am MIT anfallenden Datenmengen, eher rudimentär ausgefallen, erlaubt sie in Form eines kleinen Eingabefeldes (siehe Abbildung 20) doch gerade mal eine boolesche Stichwortsuche über alle Metadaten. Eine strukturelle Recherche in spezifischen – eigentlich genau für diesen Zweck erfaßten – Dublin Core-Einträgen ist leider ebenso wenig möglich, wie die Suche im gesamten Volltext.

Abbildung 20: DSpace "Suchmaschine"

[159] in der UB Potsdam wäre dies sicherlich ein Mitarbeiter der Abteilung Publikationen
[160] ein Beispiel für eine solche Frontdoor-Seite findet sich auf der nächsten Seite

Trotz dieses Mankos, welches sicherlich in einer der nächsten DSpace-Versionen ausgemerzt wird[161], ist das System – von kleineren Bugs abgesehen[162] – recht ausgereift und dank der Unterstützung mehrerer Communities und Collections speziell für große Bibliotheken und Einrichtungen interessant. Für den Einsatz an deutschen Universitäten ist DSpace jedoch weniger geeignet – zumindest nicht in der ursprünglichen Form: relativ viel Aufwand wäre nötig, um den Code an lokale Bedürfnisse anzupassen. Während der integrale Metadatensatz noch verhältnismäßig einfach an METADISS und METAPERS angleichbar ist, müßte die Handle-Komponente z.B. vollständig durch eigene Routinen ersetzt werden, um DDB-konforme URNs generieren zu können. Auch das Weglassen oder der Austausch der auf Mailadressen basierenden Registrierungs- und Login-Prozedur gegen ein alternatives System wäre mit viel Arbeit verbunden. Und all dies, um beim nächsten Update dann mit Erschrecken feststellen zu müssen, daß die eigenen Java-Klassen nicht mehr compilierbar sind, daß die speziell entwickelte Suchmaschine nicht mehr funktioniert oder daß die mühsam eingedeutschte Oberfläche nicht mehr verwendbar ist.

Diese Unsicherheit bezüglich des Fortbestands selbst eingebrachter Komponenten beim nächsten Versionssprung war es auch, die den Autor – zumindest im Rahmen vorliegender Diplomarbeit und zusätzlich zu den bereits genannten Gründen – von DSpace-Anpassungen abgehalten und zu einer Eigenimplementation bewogen haben. Mit seiner Benutzer-verwaltung, den Subscribe-Mechanismen und den Communities/Collections ist das MIT-Komplettsystem zudem auch ein wenig over-sized und für die aktuellen Bedürfnisse der UB Potsdam eher ungeeignet ... nichtsdestotrotz lohnt es sich, die Entwicklungen im Auge zu behalten: der Funktionsumfang ist beachtlich und die (dank Open Source) steigende Zahl interessierter Anwender läßt vermuten, daß es schon bald auch fertige, speziell auf deutsche Universitäten zugeschnittene DSpace-Versionen geben wird.

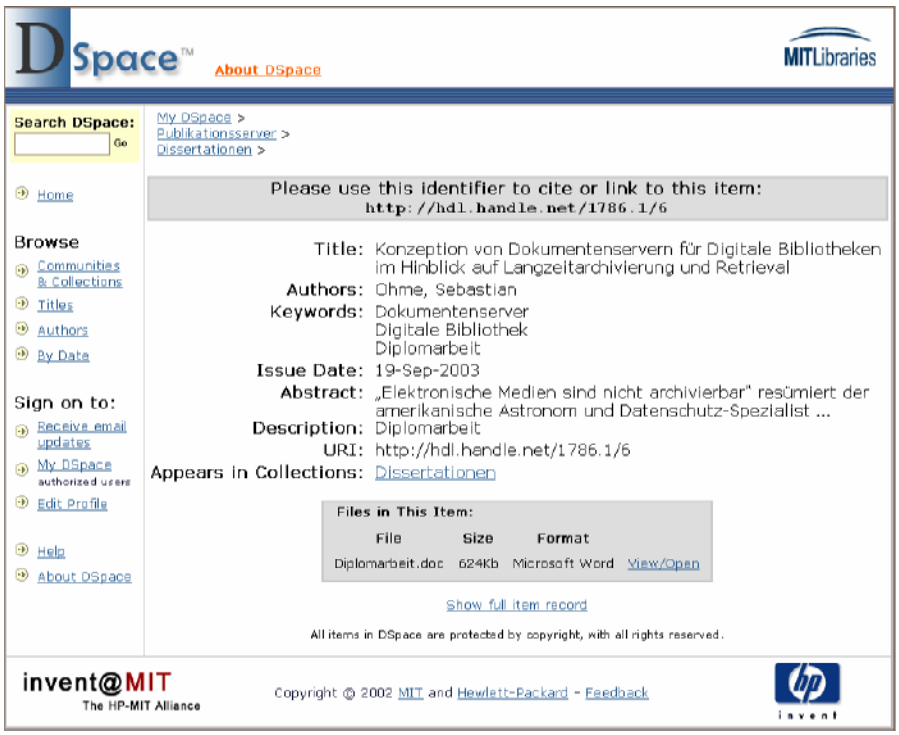


Abbildung 21: DSpace Frontdoor-Seite

[161] einige Institutionen haben die verwendete Lucene-Searchengine bereits durch andere Lösungen ersetzt und sind nun z.B. in der Lage, explizit in den Abstracts oder innerhalb von PDF-Dokumenten zu recherchieren

[162] Meldungen über die offizielle Mailingliste bestätigen dies: http://lists.sourceforge.net/lists/listinfo/dspace-tech

6.3 Implementation des Dokumentenservers

Die Evaluation existierender Lösungen – seien es nun DSpace des MIT, DiDi der HU Berlin oder OPUS, MONARCH und z.B. ELDORADO[163], auf die hier aus Gründen der Umfangsbeschränkung nicht näher eingegangen wurde – hat sich auf jeden Fall gelohnt: viele Ideen und Anregungen konnten gesammelt werden, die sich in entsprechender Form letztlich auch im eigenen System wiederfinden. Dieses Kapitel soll den in ca. zwei Monaten konzipierten Dokumentenserver nun genauer vorstellen und vor allem verdeutlichen, daß viele der einführend herausgearbeiteten Aspekte auch tatsächlich in die Praxis umgesetzt wurden.

Entwickelt wurde der Prototyp mit Hilfe von Perl[164], einer Script-Sprache, die sich trotz ihrer einfachen Erlernbarkeit insbesondere durch mächtige Stringverarbeitungsfunktionen auszeichnet und als Webserver-‚AddOn' unter Nutzung des Common Gateway Interfaces zumeist für die Generierung dynamischer Inhalte zuständig ist. Perl ist frei für alle gängigen Plattformen verfügbar und wird hier in der von ActiveState[165] bereitgestellten Windows-Version 5.8.0 eingesetzt, die durch zusätzliche Module um wichtige Funktionalitäten erweitert wurde; dazu später mehr. Ebenfalls kostenlos herunterladbar ist der nur wenige Kilobyte große und speziell für Microsoft-Systeme optimierte HTTP-Server Xitami, welcher nicht nur äußerst schnell und stabil läuft, sondern (neben vielen weiteren Standards, wie Server Side Includes und LRWP) auch SSL unterstützt. Außerdem bietet Xitami einen integrierten FTP-Server, der zur Laufzeit(!) konfigurierbar und somit optimal für den Upload von Dokumenten in temporäre Verzeichnisse geeignet ist (siehe auch Abschnitt 6.3.2). Die Festlegung auf Windows als Serverbetriebssystem und das Vorhandensein entsprechender universitärer Lizenzen hat schließlich den Einsatz von Microsoft Access als Datenbanksystem möglich gemacht. Im Rahmen vorliegender Diplomarbeit ist dies sicherlich eine legitime Wahl, langfristig sollte die Speicherung der Metadaten aber besser mittels freier und plattformunabhängiger Systeme, wie Mini- oder MySQL, erfolgen. Vorteile von Access: die Desktop-Datenbank läßt sich sehr einfach grafisch administrieren und ist via ODBC und SQL auch problemlos von Perl aus zugänglich.

Weitere Komponenten sind – von einer bestehenden Netzwerkverbindung abgesehen – für den Betrieb des Dokumentenservers nicht notwendig. Client-seitig ist lediglich ein Browser der neueren Generation (Netscape ab Version 7, Opera ab Version 6, Internet Explorer ab Version 4, Mozilla, Phoenix, Konqueror, usw.) erforderlich – und gegebenenfalls ein FTP-Client für die Übertragung großer Dateien. Zusätzliche Software wird nicht benötigt, auch nicht seitens der Systemverwalter: Xitami läßt sich (falls überhaupt notwendig) vollständig via Webfrontend konfigurieren und auch sonst ist im Normalfall kein direkter Zugang zu Server und Datenbank vonnöten. Alle Bestandteile – und so auch das weiter unten beschriebene Workflow-Modul – wurden unter einer einheitlichen Oberfläche vereint, um eine (paßwortgeschützte) Administration des Systems von jedem internetfähigen Rechner aus zu ermöglichen.

[163] http://eldorado.uni-dortmund.de:8080 - hier kommt Hyperwave zum Einsatz
[164] Practical Extraction and Report Language
[165] http://www.activestate.com

Der implementierte Prototyp besteht aus einer Reihe von Konfigurationsdateien, statischen HTML-Seiten und Perl-Programmen, deren Zusammenspiel im folgenden verdeutlicht werden soll. Ziel ist es dabei nicht, den gesamten Quellcode zu kommentieren – wichtiger erscheint es, auf algorithmische Besonderheiten einzugehen und anhand von geeigneten Screenshots die einzelnen Komponenten etwas genauer vorzustellen. Unter der temporären URL http://pub.ub.uni-potsdam.de/new kann der Dokumentenserver außerdem auch live erlebt und bereits vor einer eventuellen Überführung in den produktiven Betrieb von Nutzern bzw. Bibliotheksmitarbeitern ausgiebig gestestet werden.[166]

6.3.1 Benutzerschnittstelle

Noch bevor spezifische Funktionalitäten (Metadaten-/Volltext-Suchmaschine, Dokumentannahme/-Management usw.) implementiert wurden, galt es, eine moderne grafische Oberfläche zu kreieren, die problemlos alle geplanten Komponenten aufnehmen und von gängigen Webbrowsern dargestellt werden kann. Das Userinterface sollte dabei intuitiv zu bedienen sein, sich z.B. mit Buttonleisten am linken bzw. oberen Rand, einer dezenten Farbwahl, deutlich abgegrenzten Strukturelementen und klar erkennbaren Hyperlinks an gängige Konventionen halten und es insbesondere den Nutzern des alten Systems leicht machen, sich auch auf den neuen Seiten sofort zurechtzufinden. Inhaltlich sollte sich durch Übernahme aller Bezeichnungen und Erläuterungstexte nicht viel ändern – wohl aber im Design. Unschön gelöst war bisher beispielsweise die Aufsplittung der einzelnen Teilbereiche (Informationen zum Urheberrecht, Anmeldeformular, Autorenliste, u.ä.) auf mehrere Seiten: jeder Klick auf einen Link hat entweder ein neues Browserfenster geöffnet oder die aktuellen Angaben überschrieben – und somit ein mühsames Zurückschalten zur Startseite notwendig gemacht. Hier wäre der Einsatz eines Framesets von Vorteil gewesen; oder aber die dynamische Generierung einer Webseite, die neben dem jeweiligen Inhalt immer auch das gemeinsame Navigationsmenü enthält.

Sehr elegant ließe sich diese Idee mit Hilfe von PHP, ASP oder JSP umsetzen, die es erlauben, in statische HTML-Seiten Programmcode einzubetten, der dann z.B. die gewünschte Menüleiste generiert und den aktivierten Eintrag hervorhebt. Die Festlegung auf Perl hat den umgekehrten Weg notwendig gemacht: ein CGI-Script erzeugt den dynamischen Anteil der Webseite und baut zusätzlich an bestimmten Stellen statischen HTML-Code ein, der in Form vorgefertigter Dateien im Filesystem abgelegt ist. Der Vorteil dabei: die externen Dateien können ohne Rücksicht auf eventuell eingebettete Programmzeilen mit einem beliebigen HTML-Editor bearbeitet werden – und dies auch von MitarbeiterInnen der Abteilung Publikationen, die vielleicht schnell mal irgendwo ein Bild, eine Tabelle oder weiteren Text einfügen wollen.

Zuständig für den grundlegenden Aufbau der Benutzerschnittstelle ist das Script show.pl: es erzeugt den aus Logo, Menüleisten und Sprachauswahl bestehenden Korpus jeder Webseite und lädt entsprechend der via URL übergebenen Parameter die jeweils passende HTML-Datei – oder aber weitere Perl-Module nach. Im ersten, einfacheren Fall sieht das dann wie folgt aus (siehe nächste Seite):

[166] Sollte das neue System akzeptiert werden, wird es nach einer Übergangszeit den alten Publikationsserver ablösen. Aufgrund von existierenden Suchmaschinen-Einträgen werden die ursprünglichen Seiten allerdings nicht gelöscht - ein direkter Zugriff ist über http://pub.ub.uni-potsdam.de/old weiterhin möglich.

Abbildung 22: Nutzerschnittstelle

Nur die Adreßzeile des Browsers läßt vermuten, daß die Oberfläche dynamisch aus mehreren Komponenten zusammengesetzt wurde. „/show“ ist dabei ein Alias für „/cgi-bin/show.pl“ und der Parameter „Informationen“ gibt an, daß zum einen der entsprechende Menüeintrag zu markieren, und zum anderen im sonst leeren mittleren Bereich die statische „Informationen.htm“-Seite einzubetten ist. Intern realisiert wurde der Zusammenbau mittels mehrfach verschachtelter <table>‘s; Framesets kommen nicht zum Einsatz.

Wie angedeutet existiert aber auch ein komplizierterer Fall – dann nämlich, wenn der dezent eingerahmte mittlere Bereich auch selbst dynamischen Content, wie z.B. personalisierte Überschriften oder vorbelegte Formularfelder, enthält. Um den Programmcode übersichtlich zu halten und eine gute Austauschbarkeit einzelner Komponenten zu gewährleisten, wurde das Gesamtsystem modular aufgebaut: zusammengehörige Funktionalitäten (also z.B. die Anzeige aller Verfasser eines Instituts oder die Annahme und Überprüfung neuer Dokumente) wurden ausgelagert und in weitestgehend eigenständige Programme überführt.
show.pl kommt dabei eine Verteilerrolle zu: alle Anfragen landen zunächst bei diesem Script, welches nach Generierung des Korpus‘ entscheidet, welches Modul den weiteren Aufbau der Nutzerschnittstelle übernimmt. Ermöglicht wird dies durch die Definition zusätzlicher Aliases: „/browse“, „/search“ und „/manage“ leiten einfach zu „c:\Xitami\cgi-bin\show.pl“ weiter, das abhängig von Alias-Name und Parameter dann für die Einbindung weiterer Scripte sorgt.

Ein Beispiel: beim Aufruf von „http://pub.ub.uni-potsdam.de/manage?Anmeldungen“ wird das initiale show-Programm gestartet, welches durch Auswertung der Umgebungsvariable $ENV{'SCRIPT_NAME'} einerseits weiß, daß es die spezifische Management-Oberfläche generieren soll, und über $ENV{'QUERY_STRING'} anderseits erfährt, daß insbesondere neu eingegangene Publikationsanmeldungen zu kontrollieren sind. Hierfür ist das Script „register.pl“ zuständig und so wird es eingebunden (1), initialisiert (2) und nach Erzeugung der speziell angepaßten Buttonleiste (3) auch gestartet (4).

```
require("$cgidir\\register.pl");    (1)
&FormHandler;                       (2)
[...]                               (3)
&FormGenerator($zeile);             (4)
```

Die Unterroutinen FormHandler und FormGenerator finden sich dabei nicht nur in der Verwaltungskomponente, sondern auch in allen anderen Scripten, die für den Aufbau des Userinterfaces unterstützend notwendig sind (browse.pl, metasearch.pl, ...). Das show-Programm muß somit nur das jeweilige Modul mittels „require“ im Namespace verfügbar machen und kann sich dann darauf verlassen, daß der passende Code ausgeführt wird. Die Verteilerrolle macht show.pl zum Herzstück des Systems: hier kann wunderbar protokolliert werden, welche Bereiche des Dokumentenservers am häufigsten besucht werden. Ebenso geeignet ist das initiale Script aber auch für die Deklaration aller globalen Variablen und sonstigen Unterroutinen, die für den reibungslosen Betrieb notwendig sind – und was auszugsweise z.B. so aussieht:

```
$xitami = "c:\\Xitami";  $wwwdir = "$xitami\\webpages";
$cgidir = "$xitami\\cgi-bin";  $uploaddir = "c:\\upload";
$ftpserver = "pub.ub.uni-potsdam.de";
$adminmail = 'ohme@rz.uni-potsdam.de';
[...]
sub toHTML
{
  ($_) = @_;
  s/ä/&auml;/sg; s/ö/&ouml;/sg; s/ü/&uuml;/sg;
  s/Ä/&Auml;/sg; s/Ö/&Ouml;/sg; s/Ü/&Uuml;/sg;
  s/ß/&szlig;/sg; s/</&lt;/sg; s/>/&gt;/sg; s/\n/<br>/sg;
  return $_;
}
sub Datum
{
  local($sec,$min,$hour,$mday,$mon,$year,$wday) = localtime();
  $year = $year + 1900;
  sprintf(qq/%s%s%02d_%02d%02d%02d/,$year,
    ("01","02","03","04","05","06","07","08","09","10","11","12")[$mon],
    $mday,$hour,$min,$sec);
}
```

Ändert sich irgendwann das Verzeichnis der statischen Webseiten oder die Mailadresse des Administrators, muß nur dieses eine Script angepaßt werden. Außerdem wurde auf eine einfache Erweiterbarkeit geachtet: ist das in Abbildung 22 dargestellte Auswahlmenü beispielsweise um einen Eintrag zu erweitern, muß nicht der gesamte Programmcode umgeschrieben, sondern lediglich das zugehörige Array ergänzt werden (siehe nächste Seite).

```
@eintraege = (
 "#Startseite#|#Startpage#",
 "#Informationen#|#Information#",
 "#Dissertationen#|#Dissertations#",
 "-#Suche#...|-#Search#...",
 "...entsprechend der #Uni-Struktur#|...by Faculties and #Institutions#",
    [...]
 "#Kataloge#|#Catalogs#",
 "-#Management#|-#Management#" );
```

Eine Schleife sorgt dafür, daß die einzelnen Elemente geparst und alle notwendigen Informationen extrahiert werden. Die optionale englischsprachige Benutzerschnittstelle wird dabei ebenso berücksichtigt, wie Trennlinien zwischen einigen Einträgen (angegeben durch führende „–“). Die Doppelkreuze „#“ rahmen schließlich den Dateinamen der zu ‚verlinkenden‘ Webseite ein: „... entsprechend der #Uni-Struktur#“ ergibt somit einen gleichnamigen Menüeintrag (ohne „#“), der auf „/show?Uni-Struktur“ verweist und bei Auswahl dann die Fakultäts- und Institutsliste erzeugt.
Abschließend sei nochmals kurz auf das weiter oben angegebene „&FormGenerator($zeile)“ eingegangen. Um eine möglichst gute Trennung des überwiegend statischen HTML-Codes von seinen dynamischen Anteilen zu ermöglichen, wurden an den jeweils zu generierenden Stellen eindeutige Kommentarzeilen eingebaut, die von den Perl-Scripten dann durch entsprechende Angaben ersetzt werden. Zum besseren Verständnis hier zunächst ein kurzer Ausschnitt der „Autorenangaben.htm“-Datei sowie einige Programmzeilen aus show.pl:

```
 [...]
</td></tr>
<!-- ERROR -->
</tbody></table>
</td></tr>
</tbody></table>
<div class=text align=justify>
<!-- EINLEITUNG -->
</div>
<table border=0 cellspacing=0 cellpadding=0 width="100%">
<tr><td align=center valign=top>
<table border=0 cellspacing=10 cellpadding=0 width=490>
<!-- AUTOREN -->
<tr><td colspan=2>
 [...]
```

```
[...]
if (!open(CONTENTS,"< $htmldir\\$page.htm")) { [Fehlermeldung] }
else
{
  while(<CONTENTS>)
  {
    $zeile = $_;
    if (($anmeldung) || ($suche) || ($browse))          (1)
    {
      &FormGenerator($zeile);
    }
    elsif ($publikationsid)
    {
      $zeile =~ s/_PUBLIKATIONSID_/$publikationsid/;
      if ($zeile =~ /<!-- DATEN -->/i)                  (2)
      {
        [...]
```

Das Programm versucht die Datei zu öffnen (der Name wurde als Parameter übergeben und in $page gespeichert) und lädt sie bei Erfolg mittels einer while-Schleife zeilenweise ein. Nun wird überprüft, ob ein externes Modul aufzurufen ist (1) oder ob show.pl den Content-Bereich selbst aufbauen soll. Da letzteres jedoch nur für die Frontdoor-Seiten der Fall ist (2) und wir uns im Anmeldeformular befinden, wird die weitere (zeilenweise) Bearbeitung an register.pl abgegeben. Im dortigen FormGenerator befinden sich schließlich die zu den HTML-Kommentaren passenden Abfragen – und natürlich entsprechende print-Anweisungen, die an vorgegebener Stelle Texte und Tags in die sonst statischen Seiten einbauen.

```
  [...]
}
elsif ($zeile =~ /<!-- EINLEITUNG -->/i)
{
  print "Um ";
  print "eine " if ($form{'Anmeldeformular_Publikationsart'} ne "Lehrunterlagen");
  print "$form{'Anmeldeformular_Publikationsart'} zur Ver&ouml;ffentlichung ";
  print "anzumelden, sind die orange betitelten Felder auszuf&uuml;llen.\n";
}
elsif ([...])
{
  [...]
```

Auf den ersten Blick mag all das kompliziert und umständlich erscheinen. Die vom Autor bevorzugte Nutzung der Scriptsprache Perl hat diese Vorgehensweise jedoch notwendig gemacht – und sich im nachhinein als durchaus praktikabel erwiesen: Programm und Layout werden so recht gut voneinander getrennt. Durch die Vergabe eindeutiger Kommentar-bezeichner können statischer und dynamischer Code dennoch schnell zugeordnet und bei Bedarf einfach verändert werden ... und dies wie eingangs erwähnt gerne auch mittels WYSIWYG-Editor durch MitarbeiterInnen der Bibliothek.

6.3.2 Publikationsanmeldung

Nachdem nun bekannt ist, wie sich die Nutzerschnittstelle grundsätzlich zusammensetzt, sollen im folgenden die einzelnen bereits angesprochenen Module etwas genauer vorgestellt werden. Die mit über 2000 Programmzeilen wohl umfangreichste Komponente ist dabei das register.pl-Script, welches alle für die Publikationsanmeldung und Freischaltung notwendigen Funktionalitäten enthält. Der doch beachtliche Umfang (das bisher eingesetzte Meldescript besteht aus 238 Zeilen) zeigt, daß hier besonders viel Arbeit investiert wurde.
Aus gutem Grund: die Formulare sollten komfortabel bedienbar und gleichzeitig optisch ansprechend ausfallen. Vor allem aber galt es, die sonst getrennten Schritte des Anmelde-prozesses unter einer gemeinsamen Oberfläche zu vereinen – was es den Autoren letztlich nicht nur erlaubt, sich und ihre Publikation bis ins kleinste Detail zu beschreiben, sondern die Dokumente z.B. auch direkt via Browser auf den Server zu transferieren. Ziel war es ja, den bisherigen Workflow innerhalb der Abteilung Publikationen so gut es geht zu automatisieren und ein in sich geschlossenes System zu schaffen, welches Interaktionen seitens der MitarbeiterInnen weitestgehend unnötig macht. Während früher die für Die Deutsche Bibliothek interessanten zusätzlichen Angaben zur Person (Geburtsort, Staatsangehörigkeit, Studienfach, ... siehe auch Kapitel 5.2.1) beispielsweise mit Hilfe einer manuell erstellten Mail abgefragt wurden, sollte das neue System diesen zusätzlichen Aufwand durch Integration optionaler METAPERS-Formularfelder überflüssig machen.

Wie im letzten Abschnitt kann auch hier nur auf einige wenige Aspekte der Programmierung näher eingegangen werden – auf alle Fälle zu erwähnen ist, daß natürlich auch (und gerade) das Anmeldemodul einen wichtigen Teil des Userinterfaces ausmacht und daß die vorgenommene Gliederung des Kapitels 6.3 daher eher Script-bezogen zu verstehen ist.

Als Bestandteil der Benutzerschnittstelle wird register.pl vom initialen show-Script aus gestartet; allerdings mit dem Unterschied, daß die Parameter – anders als in den Beispielen der letzten Seiten – hier nicht innerhalb der URL, sondern gesondert übergeben werden. Das HTTP-Protokoll, welches für die Übertragung der Seiten zwischen Browser und Webserver/ CGI-Script zuständig ist, kennt (unter anderem) zwei Modi: GET und POST.
Während die Argumentlänge bei den allgemein üblichen GET-Requests stark eingeschränkt ist, kann mittels POST theoretisch eine unbegrenzte Anzahl an Parametern verschickt werden – was es somit zum präferierten Modus für die Übergabe der Formularinhalte macht. Jede Seite des Anmeldemoduls beinhaltet daher neben den eigentlichen Eingabefeldern auch das HTML-Tag `<form action="/show" method="post">`. Im FormHandler wiederum befindet sich das entsprechende Gegenstück:

```
use CGI qw(:standard);
$CGI::POST_MAX = $maxsize+($maxsize/2);
$q = new CGI;
%form = $q->Vars;
[...]
```

Alle via POST übertragenen Formulardaten werden vom Standard-„CGI“-Modul entgegengenommen und für eine spätere Auswertung im assoziativen Array „%form“ hinterlegt. Genaueres zu diesem Zusammenspiel von Webseite und Perl-Script vielleicht an einem Beispiel:

Abbildung 23: Publikationsart-Auswahlformular

Der Screenshot zeigt das erste einer ganzen Reihe von Formularen, die für die Publikationsanmeldung von Bedeutung sind.[167] Hier ist es mittels einer Auswahlliste und eines Submit-Buttons lediglich möglich, die Publikationsart auszuwählen. Aber gerade diese Einfachheit

[167] Logo, Menüleiste und der restliche umgebende Bereich sind sonst natürlich vorhanden, wurden in Abbildung 23 aus Platzgründen aber weggelassen.

des Formulars erlaubt es, die internen Programmabläufe – die natürlich auch auf die anderen, weiter unten beschriebenen und sehr viel komplexeren Anmeldeschritte zutreffen – etwas besser vorstellen zu können.
Ein Blick in die statische HTML-Seite zeigt, daß die Einträge des Pull-Down-Menüs (wie z.B. `<option value="Dissertation">Dissertation</option>`) dynamisch eingebunden werden:

```
[...]
<select class=compact name="Anmeldeformular_Publikationsart" size=5>
<!-- ARTEN -->
</select> [...]
<input class=button type=submit value="Publikation anmelden">
[...]
```

Außerdem ist der Name des Formularfelds zu erkennen, dessen Syntax für das System von entscheidender Bedeutung ist. Dazu gleich mehr, zunächst aber zurück zum FormHandler und zum assoziativen Array „%form“: wählt der Nutzer „Dissertation“ und klickt anschließend auf den Submit-Button, werden Schlüssel (name="...") und Wert (value="...") im POST-Modus[168] an show.pl/register.pl geschickt und dort in einen korrespondierenden Array-Eintrag überführt (`"Anmeldeformular_Publikationsart" => "Dissertation"`), dessen Inhalt natürlich auch ausgelesen werden kann:

```
$art = $form{'Anmeldeformular_Publikationsart'};
```

Dies wiederum macht es möglich, Listen und Eingabefelder vorzubelegen und es dem Nutzer zu ersparen, einmal ausgewählte Einträge oder mühsam beschriebene Textfelder bei seiner eventuellen ‚Rückkehr‘ zum entsprechenden Formular nochmals anklicken bzw. ausfüllen zu müssen (auch dazu gleich mehr). Hier der Vollständigkeit halber noch der Programmcode, der für die Generierung und Vorauswahl der Publikations-Auswahlliste zuständig ist – und der mittels require auch eine wichtige Konfigurationsdatei einbindet:

```
[...]
elsif ($zeile =~ /<!-- ARTEN -->/i)
{
  require "$cgidir\\auswahl.pl";
  foreach $eintrag (@arten)
  {
    ($key,$value) = split(/\|/,$eintrag);
    print "<option value=\"$key\"";
    print " selected" if ($key eq $form{"Anmeldeformular_Publikationsart"});
    print ">" . &toHTML($value) . "</option>\n";
  }
}
[...]
```

Auszug aus "auswahl.pl":

```
@arten = (
   "Dissertation|Dissertation",
   "Habilitation|Habilitation",
   "Aufsatz|Aufsatz",
   "Monographie|Monographie (Buch)",
   [...]
   "Lehrmaterialien|Lehrmaterialien mit Lehrbuchcharakter",
   "Dokument|Sonstiges" );
```

[168] bei so wenigen Daten wäre natürlich auch GET möglich, Vorteil von POST allerdings: die Parameter werden nicht in der Adreßzeile des Browsers angezeigt

Neben dem Array @arten, welches alle aktuell von der UB Potsdam akzeptierten (und in „Key|Value“-gesplitteten) Publikationsarten umfaßt, enthält auswahl.pl auch eine Liste aller Fakultäten, Institute, Länder, Sprachen, Dateiformate und aller sonstigen Angaben, die für die Erfassung der Metadaten von Bedeutung sind. Kommt irgendwann ein Institut hinzu, braucht lediglich ein entsprechender Eintrag – gerne auch durch Bibliotheksmitarbeiter – ergänzt werden ... an den Scripten ändert sich nichts.

Für das erwähnte Vorbelegen einzelner Felder und das ‚Durchreichen‘ der Eingaben von Formular zu Formular ist es wichtig zu wissen, von wo man kommt und wo man sich gerade befindet. Ersteres wird durch Angabe eines versteckten und passenderweise „Formular“ genannten Feldes erreicht, welches als Wert immer die Bezeichnung des jeweiligen Anmeldeschritts enthält:

```
<input type=hidden name="Formular" value="Anmeldeformular">
```

Beim Klick auf „Publikation anmelden“ wird dieser Wert (zusammen mit allen sichtbaren Feldinhalten) an das CGI-Script übertragen, das nun den Ursprung der Daten kennt und durch Auswertung der Button-Beschriftung zusätzlich weiß, welches Formular als nächstes zu generieren ist. Im vorliegenden Fall ist dies das Formular zur Erfassung der Autorenangaben:

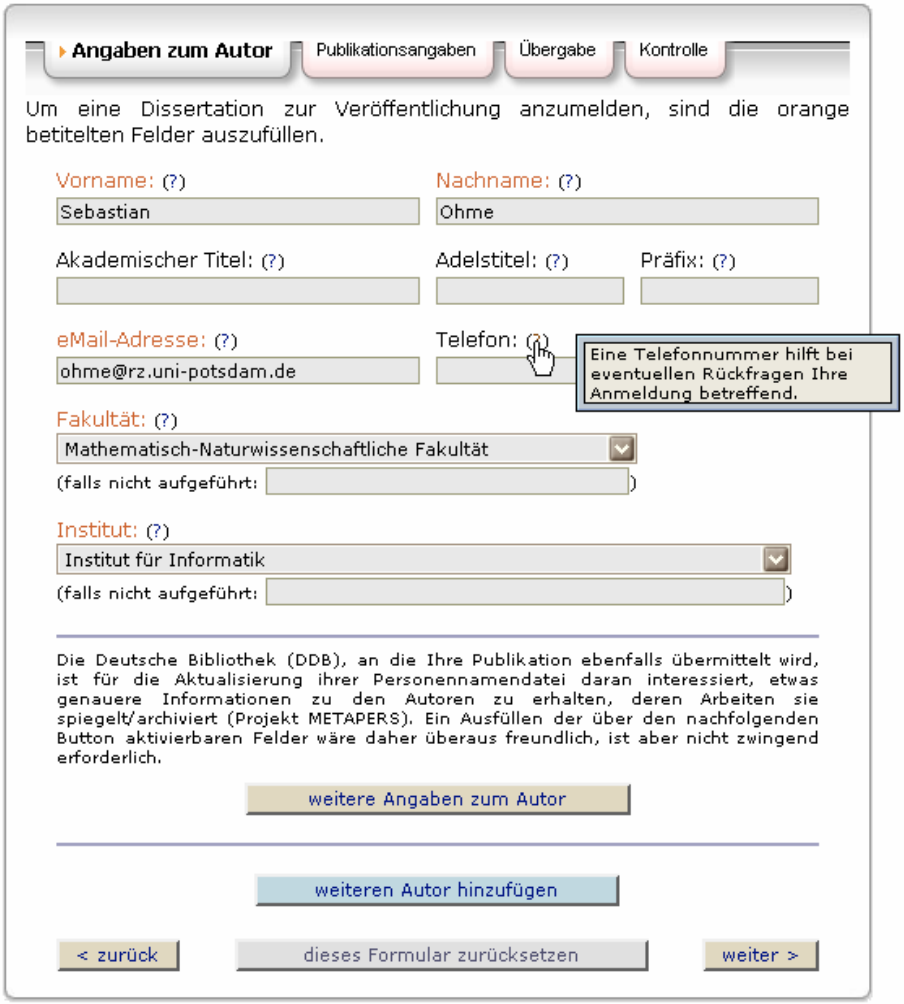


Abbildung 24: Autorenangaben-Formular

Bevor näher auf die Besonderheiten dieses nächsten (und ersten großen) Anmeldeschritts eingegangen wird, sollen die Abhängigkeiten zwischen den einzelnen Seiten nochmals konkretisiert werden.

Aufgrund der erwähnten eindeutigen Bezeichnung aller Felder und Formulare kann im register.pl-FormHandler überprüft werden, ob der Nutzer auch wirklich eine Publikationsart ausgewählt und nicht einfach nur auf den Knopf geklickt hat. Fehlen obligatorische Angaben, wird eine entsprechende Fehlermeldung generiert (1) und das gleiche Formular nochmals angezeigt (2):

```
 [...]
 elsif ($form{"Formular"} =~ /Anmeldeformular/i)
 {
   if ($form{"Anmeldeformular_Publikationsart"} eq '')
   {
     $error =<<"ERROR";                              (1)
[...]
<font color=#b00000>
Bitte w&auml;hlen Sie eine Publikationsart aus !
</font>
[...]
ERROR
     $page = "anmeldeformular";                      (2)
   }
   else { $page = "autorenangaben"; }                (3)
  }
  [...]
```

War alles in Ordnung, wird das nächste (und auf der letzten Seite dargestellte) Formular erzeugt (3). Dabei ist es allerdings notwendig, die Daten des vorherigen Anmeldeschritts nicht nur wie in Abbildung 24 gezeigt auszugeben („Um eine *Dissertation* zur Veröffentlichung anzumelden, [...]“), sondern auch irgendwo zu speichern, so daß sie auch über mehrere Formulare hinweg verfügbar sind. Für dieses ‚Weiterreichen‘ der Nutzereingaben kommen erneut „hidden“-Felder zum Einsatz. Nicht alle Daten dürfen dabei jedoch versteckt in den HTML-Code eingebettet werden: hat sich der Anwender mit „weiter >“ zum nächsten Formular durchgeklickt und kommt später wieder „< zurück“, will er – wie auch weiter oben schon erwähnt – seine Eintragungen nicht versteckt, sondern in den zugehörigen *sichtbaren* Feldern vorfinden. Und hier kommt nun die Syntax der Feldnamen ins Spiel, deren Bedeutsamkeit bereits ebenfalls kurz angedeutet wurde: den Elementbezeichnern des jeweils aktuell vorliegenden Formulars wird immer der Name des Anmeldeschritts – getrennt durch einen Unterstrich – vorangestellt, wie auch der fett markierte Bereich im obigen Programmausschnitt zeigt. Für die mit „Autorenangaben“ betitelte METAPERS-Erfassungsseite bedeutet dies also die Deklaration eines solchen Formularfelds

```
<input class=compact type=text name="Autorenangaben_Vorname" value="*" size=35>
```

wobei der Stern im value normalerweise dynamisch vorbelegt wird. Alle Eingaben, die von anderen Seiten stammen, werden mit Hilfe folgender Zeilen unsichtbar und eventuell gekürzt (mehr als maximal 4096 Zeichen können pro Feld nicht übertragen werden) in die HTML-Seiten integriert:

```
 [...]
 elsif ($zeile =~ /<!-- HIDDEN -->/i)
 {
   while (($key,$value) = each %form)
   {
     if (($key =~ /_/) && ($key !~ /^$page/i))
     {
       $value = substr($value,0,4000) . " [...] (Eintrag wurde gekürzt)"
         if (length($value) > 4096);
       print "<input type=hidden name=\"$key\" value=\"$value\">\n";
     }
   }
 }
```

Soviel zu dem doch recht komplexen Zusammenspiel zwischen den einzelnen Formularen ... und zurück zu Abbildung 24. Sofort auffällig ist sicherlich die Anlehnung an den DSpace-Anmeldeprozeß: in vier Schritten kann der Autor sich und seine Publikation beschreiben, die Dokumente via Browser (oder FTP) auf den Server übertragen und abschließend alle Angaben kontrollieren. Vor- und Zurück-Buttons erlauben dabei das komfortable Umschalten zwischen den jeweiligen Seiten (siehe oben) und ein „zurücksetz"-Knopf ermöglicht das schnelle Löschen aller bisherigen Eingaben. Durch Klick auf einen Eintrag in der (nicht dargestellten) Menüleiste oder durch Schließen des Browserfenster kann die Anmeldung problemlos abgebrochen werden – temporäre Datenbankeinträge, wie bei DSpace, werden nicht erzeugt. Und dies ist auch ein wichtiger Unterschied zum MIT-System: bis zum Abschluß der Anmeldung werden alle Nutzereingaben lediglich client-seitig innerhalb der angesprochenen „hidden"-Felder gespeichert. Dies schließt Datenbank-Inkonsistenzen aus, falls die Internet-Verbindung während der Übertragung zusammenbricht oder das Fenster voreilig ohne „Save" und „Logout" geschlossen wird (siehe Abschnitt 6.2). Auf eine Autorisierungs-Komponente wurde verzichtet, da ein Mißbrauch des Systems eher unwahrscheinlich ist.[169]

Der dezente Einsatz bandbreite-schonender Grafiken dient der optischen Aufbereitung des Interfaces – ebenso wie die Formatierung der einzelnen Elemente mittels CSS. Außerdem gut in Abbildung 24 zu erkennen sind die über Fragezeichen erreichbaren Erläuterungstexte: fährt der Anwender mit der Maus über ein solches Symbol, öffnet sich mit Hilfe einer JavaScript-Funktion ein kleines DHTML-Fenster, welches nähere Informationen zum jeweiligen Formularfeld enthält. Ist JavaScript im Browser deaktiviert, werden auch die „(?)" nicht angezeigt.

Wurden alle obligatorischen Felder ausgefüllt, ist es möglich, die Metadaten weiterer Autoren hinzuzufügen. Außerdem – und das war eine Forderung an das System – können ergänzende Angaben zum Verfasser gemacht werden, die einerseits für die UB Potsdam, vor allem aber für die Personennamendatei Der Deutschen Bibliothek von Interesse sind. Da diese Angaben freiwilliger Natur sind, werden die optionalen Formularfelder auch erst nach einem Klick auf den entsprechenden Button sichtbar.

[169] Das ebenfalls ‚ungeschützte' Anmeldeformular des bisherigen Publikationsservers wurde in den letzten Jahren kein einziges Mal mißbräuchlich verwendet.

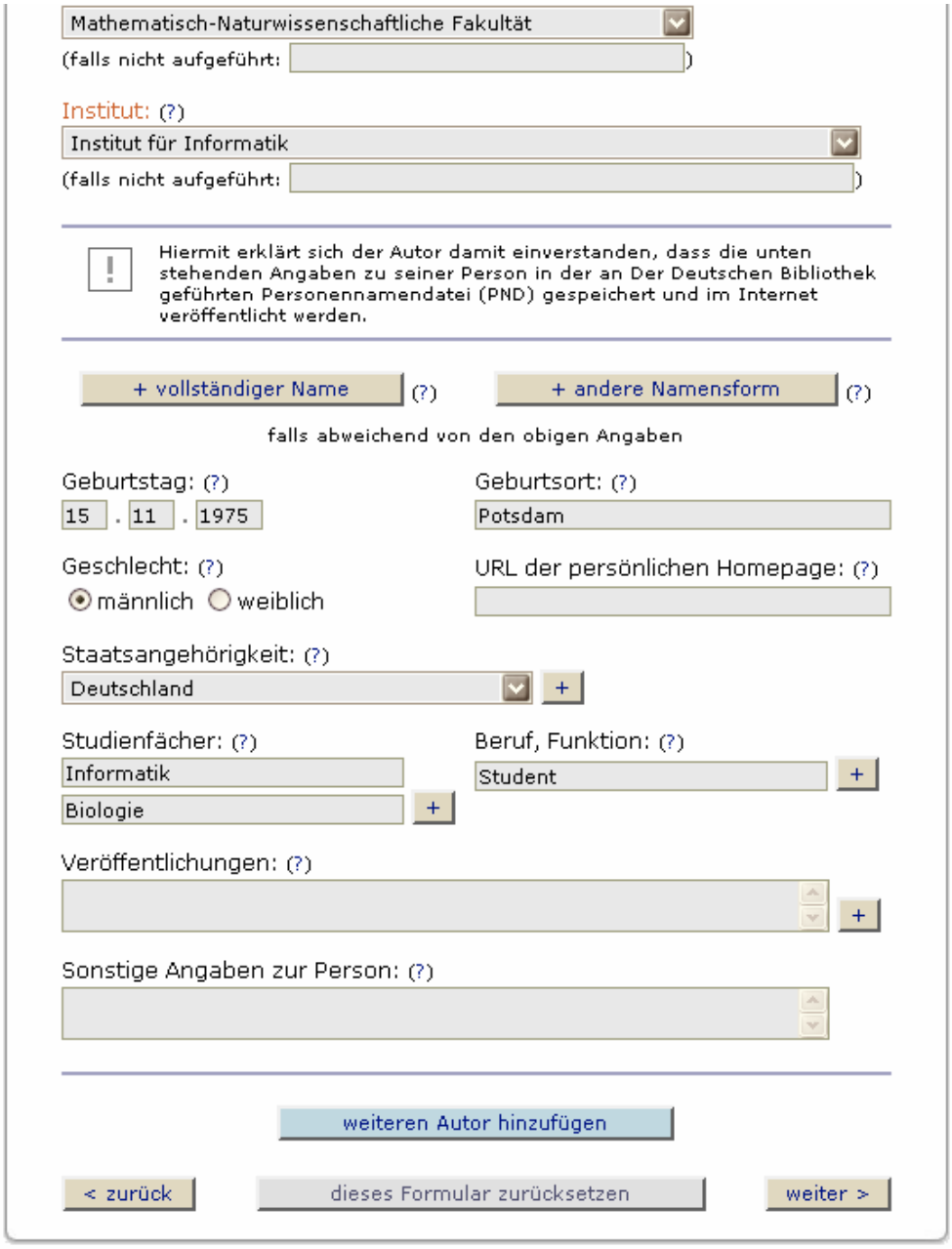


Abbildung 25: METAPERS-Erfassungsformular

Der Screenshot zeigt, daß das „Autorenangaben“-Formular dynamisch um weitere Eingabemöglichkeiten ergänzt wurde. Aber auch diese neue Webseite ist (fast) beliebig erweiterbar: die „+“-Buttons erlauben es – erneut in Anlehnung an DSpace – zusätzliche Felder hinzuzufügen; beispielsweise, um eine zweite Staatsangehörigkeit oder wie geschehen ein zweites Studienfach anzugeben.
Alle Elemente entsprechen dabei den Vorgaben Der Deutschen Bibliothek. Der METAPERS-Metadatensatz gestattet es z.B., mehrere Veröffentlichungen (pc.relation.haswritten) eines Autors (pc.name) zu speichern – und so auch der hier implementierte Dokumentenserver.

Wurden schließlich alle autorenspezifischen Metadaten eingetragen (Vorname, Nachname, Mailadresse, Fakultät und Institut *müssen* angegeben werden, der Rest ist fakultativ), kann mit dem „weiter“-Button zum nächsten Formular gewechselt werden.

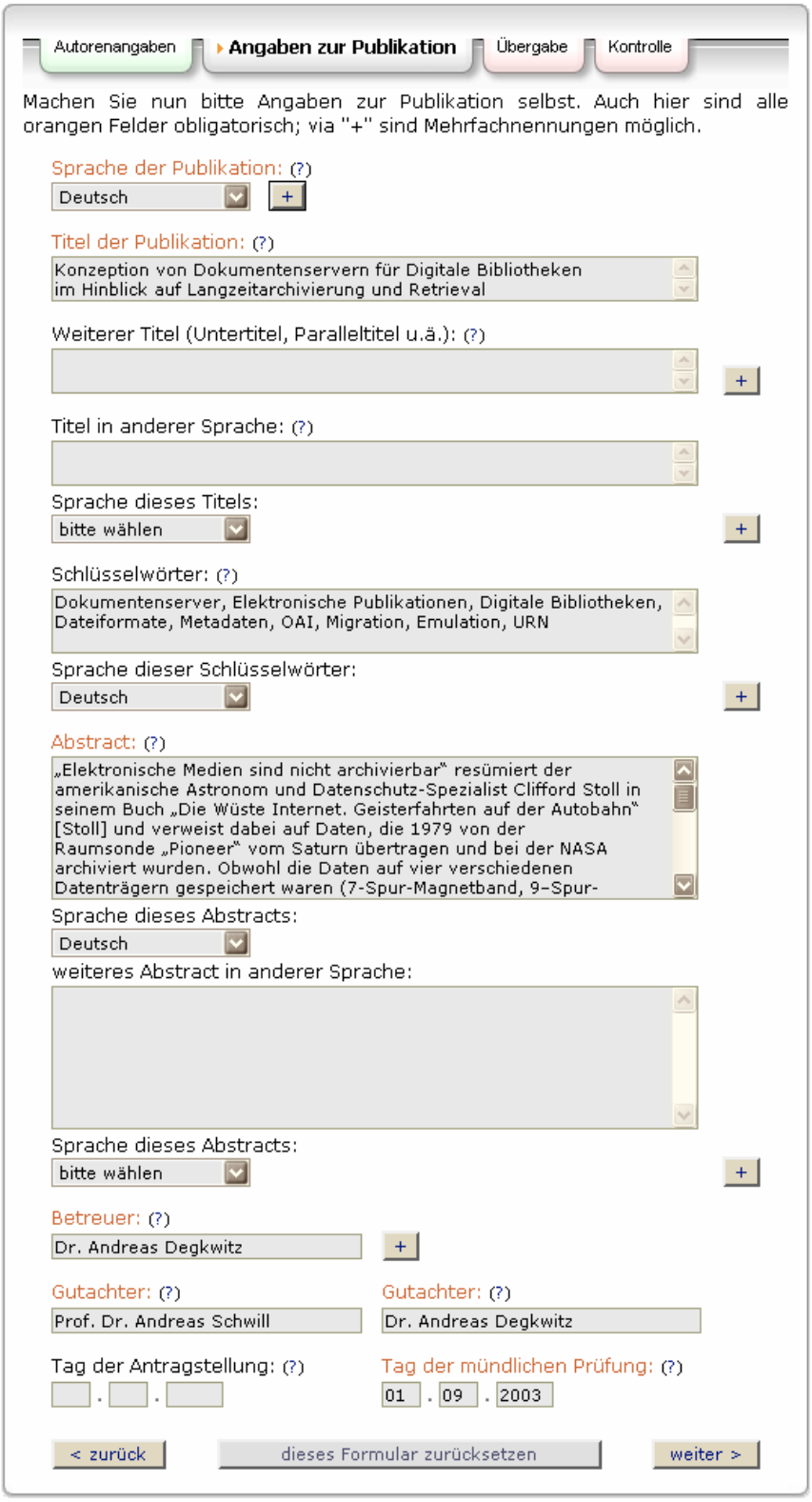


Abbildung 26: Publikationsangaben-Formular

Viel muß zu diesem Screenshot eigentlich nicht gesagt werden. Der Fortschritt des Anmeldeprozesses wird auch hier am oberen Bildschirmrand deutlich gemacht: grünlich hinterlegt sind die Tabs bzw. Bereiche, die bereits abgehakt sind. Große, dicke Schrift gibt den Inhalt der aktuellen Seite an und rötlich markiert sind schließlich die Schritte, die es noch abzuarbeiten gilt. Dieses zweite wichtige, und natürlich METADISS-konforme Formular ermöglicht es, die

Publikation genauer zu beschreiben. Zu jedem Element des DDB-Metadatensatzes gibt es hier ein entsprechendes Pendant. Generiert, vorbelegt und gegebenenfalls mit einem „+“ versehen werden die Eingabefelder dabei wie alle anderen. Die Inhalte der Sprachauswahl-Listen wurden aus der weiter oben erwähnten Konfigurationsdatei „auswahl.pl“ entnommen, der gesamte sonstige HTML-Code liegt statisch in der Datei „Publikationsangaben.htm“ vor.

Programmintern passiert bei der Weiterschaltung von Formular zu Formular und bei deren Erzeugung immer das gleiche: zunächst wird ermittelt, von welcher Seite der Zugriff kam und welche neue Seite angefordert wird. Abhängig von der anfangs gewählten Publikationsart kann dann geprüft werden, welche Angaben obligatorisch sind und welche Felder gegebenenfalls gar nicht angezeigt werden müssen.[170] Wurden wichtige Felder vom Nutzer fehlerhaft oder gar nicht ausgefüllt, wird das Ursprungsformular zusammen mit einer entsprechenden Bemerkung erneut angezeigt – andernfalls wird die angeforderte Seite nachgeladen. In beiden Fällen werden an bestimmten Stellen des Formulars zusätzliche Texte und HTML-Tags eingebaut, um z.B. Felder mit Werten vorzubelegen, spezielle Bereiche hervorzuheben – oder um die in anderen Formularen getätigten Eingaben versteckt zwischenzuspeichern.
Bei all dem kommen, unabhängig vom aktuellen Anmeldeschritt, immer dieselben Codefragmente zum Einsatz – eine gute Erweiterbarkeit und Austauschbarkeit von Formulareinträgen bzw. Programmteilen ist somit gewährleistet.

Wurden schließlich auch die publikationsspezifischen Metadaten vergeben, können endlich auch die Dokumente selbst auf den Server transferiert werden. Zuständig dafür sind zwei (oder besser drei) Formulare, die allesamt unter dem Tab „Übergabe“ vereint wurden. Die ersten beiden Formulare erlauben dabei den bequemen Upload von Präsentations- und Archivierungsformat via Browser. Kann diese bevorzugte Übertragungsmöglichkeit nicht genutzt werden (z.B. aufgrund von Dateigröße und langsamer Internet-Verbindung), ist auch ein FTP-Transfer oder ein Versand auf herkömmlichem Wege (Diskette, CDROM, Mail-Attachment) möglich, wobei die Wahl des Nutzers dann im dritten Formular anzugeben ist.

... Doch eins nach dem anderen: hier zunächst die erste Upload-Seite, die im Normalfall für die Übertragung des PDF-Dokuments genutzt werden kann.
Eine kurze Anleitung, wie der Nutzer vorzugehen hat, befindet sich bereits im Screenshot.
Die dort angegebene maximale Dateigröße von 20MB ist eher als Richtwert zu verstehen: auch 25MB sind, eine entsprechende Netzanbindung vorausgesetzt, problemlos hochladbar – „HTTP POST“ und ein “enctype=multipart/form-data“ machen es möglich.

Abbildung 27: Upload-Formular für Präsentationsformat

[170] Die Angabe eines Gutachters ist z.B. nur bei Examensarbeiten sinnvoll, für die Anmeldung von Vorträgen oder Lehrmaterialen ist dieses Feld nicht erforderlich.

Über die im FormHandler vorgenommene Zuweisung

`$CGI::POST_MAX = $maxsize+($maxsize/2);` (siehe auch Anfang des Abschnitts)

wird die Maximalgröße aus Sicherheitsgründen jedoch auf 30MB beschränkt ($maxsize = 20*1024*1024) – ein Wert, der auch für anspruchsvolle Dokumente ausreichen sollte.[171] Dank der Nutzung des im Umfang der Perl-Distribution enthaltenen „CGI"-Moduls gestaltet sich die Übertragung der Publikation vom Rechner/Browser zum Perlscript/Server überaus einfach.

`$q->param("Uebergabe_PDF")` liefert den im „`<input type=file name=Uebergabe_PDF>`"-Feld angegebenen Namen der Datei, der gleichzeitig als Filehandle für die Leseschleife dient:

```
$size = 0;
while ($bytesread=read($dateiname,$buffer,1024))
{
  last if ($size > $maxsize);
  $size += $bytesread;
  print OUTFILE $buffer;
}
```

Anders als bei allen bisher transferierten Formulardaten können 30MB natürlich nicht innerhalb von „hidden"-Feldern gespeichert werden. Aus diesem Grund wird zuvor (und zwar zu Beginn des gesamten Anmeldeprozesses) ein temporäres Verzeichnis auf dem Server eingerichtet, dessen eindeutiger Name sich aus Jahr, Monat, Tag, Stunde, Minute und Sekunde zusammensetzt und gleichzeitig die sogenannte „Session_ID" darstellt, welche – zusammen mit den anderen Eingaben – ebenfalls von Formular zu Formular durchgereicht wird:

```
<input type=hidden name="Session_ID" value="20030815_204711">
```

Aus dem übergebenen Dateinamen und der Sitzungs-ID wird eine Pfadangabe generiert (`$datei = "$uploaddir\\$form{'Session_ID'}\\$datei"`) und die Datei dort schließlich temporär gespeichert (`open(OUTFILE,"> $datei")`)[172]

War das Hochladen des Präsentationsformats erfolgreich (client-seitiger Pfad zur Datei war korrekt, Maximalgröße wurde nicht überschritten, ...), wird eine Bestätigungsmeldung und das optisch und funktionell ähnlich aufgebaute zweite Formular ausgegeben, welches die Übertragung aller weiteren Dateien (z.B. Originaldokument im DOC-Format, ZIP-Archiv mit LaTeX- und DIV-Files, u.ä.) erlaubt.

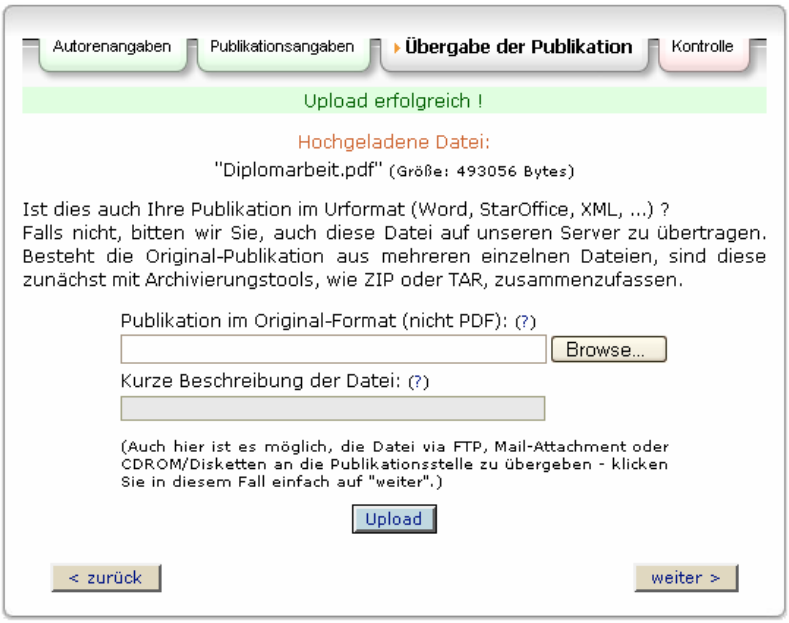


Abbildung 28: Upload-Formular für Archivierungsformat

[171] Ausnahmen bestätigen die Regel: auf dem alten Publikationsserver wurden Dateien entdeckt, die sogar 80MB noch übersteigen.

[172] Wird die Anmeldung nach einer gewissen Zeitspanne nicht abgeschlossen, werden die abgelegte(n) Datei(en) und das erzeugte Verzeichnis automatisch wieder gelöscht.

Zur Kontrolle wird nochmals die Dateigröße der auf dem Server gespeicherten Version angezeigt. Sollte es hier eine Unstimmigkeit geben, kann diese im letzten Anmeldeschritt (siehe weiter unten) wieder ausgemerzt werden. Wie erwähnt ist es allerdings keine Pflicht, die Upload-Formulare zu benutzen. Als weiteres –automatisches– Übergabeverfahren bietet sich FTP an und so kann es im dritten Formular als alternativer Lieferweg angegeben werden.

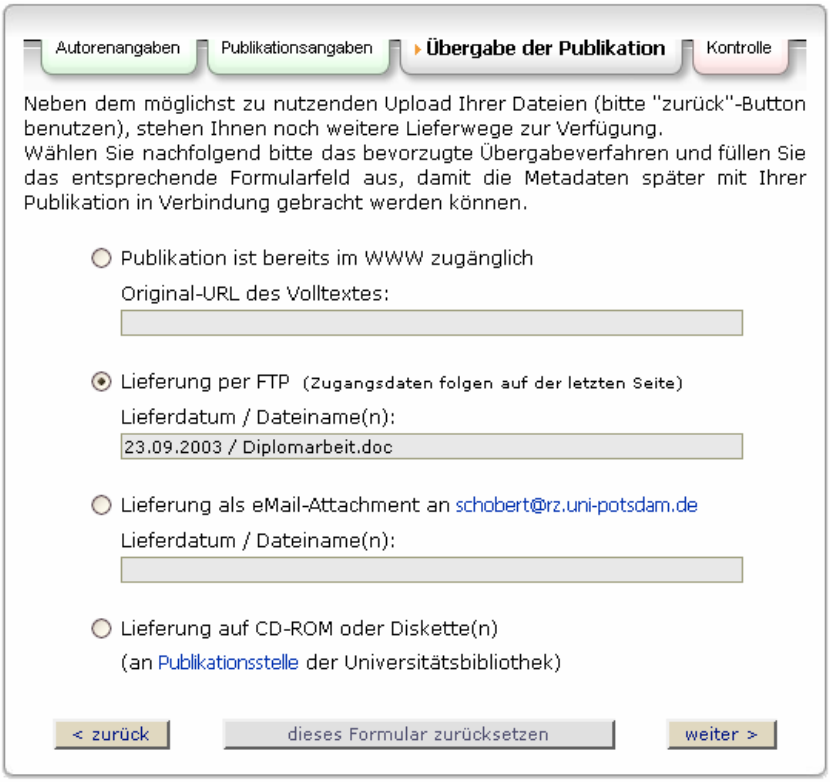


Abbildung 29: Angabe eines alternativen Übergabeverfahrens

Die Verwendung von FTP war auch bisher schon möglich, allerdings mußten Interessenten zunächst die Abteilung Publikationen kontaktieren und Username+Paßwort für den Server erfragen. Wurden die Dokumente dann übertragen, war ein manueller Kopiervorgang aus dem Upload-Verzeichnis in das eigentlich vorgesehene Directory erforderlich – ein nicht unerheblicher Mehraufwand für alle Beteiligten. Das neue System ist komfortabler ausgefallen: wählt der Nutzer – wie im Screenshot dargestellt – „FTP“ als Transfer-Variante, erhält er ganz am Schluß der Anmeldung *persönliche*, aus der Session_ID erzeugte Zugangsdaten, die er *sofort* für den Versand seiner Dateien nutzen kann. Username und Paßwort verfallen automatisch nach einer Woche – die innerhalb dieser Zeitspanne eingegangenen Dokumente können vom Bearbeiter direkt im Management-Bereich verwaltet werden. Näheres dazu an entsprechender Stelle und am Ende dieses Abschnitts.

Letzter Schritt einer jeden Publikationsanmeldung ist die Kontrolle der Metadaten und die Überprüfung der via Browser hochgeladenen Dateien. Die Generierung einer visuell ansprechenden Übersicht hat – im Vergleich zu allen anderen bisher vorgestellten Programmteilen – die meiste Mühe gemacht: die abschließende Präsentation der Nutzer-eingaben sollte schließlich über eine einfache Auflistung hinausgehen; gleichartige Formularfelder sollten akkumuliert werden, Abstracts sollten sich in einem extra Fenster öffnen, nur wirklich vorhandene Angaben sollten sichtbar sein – insgesamt sollte die Kontrollseite so ‚schick‘ aussehen, daß sie ohne Bedenken auch ausgedruckt und zu den Akten gelegt werden kann.

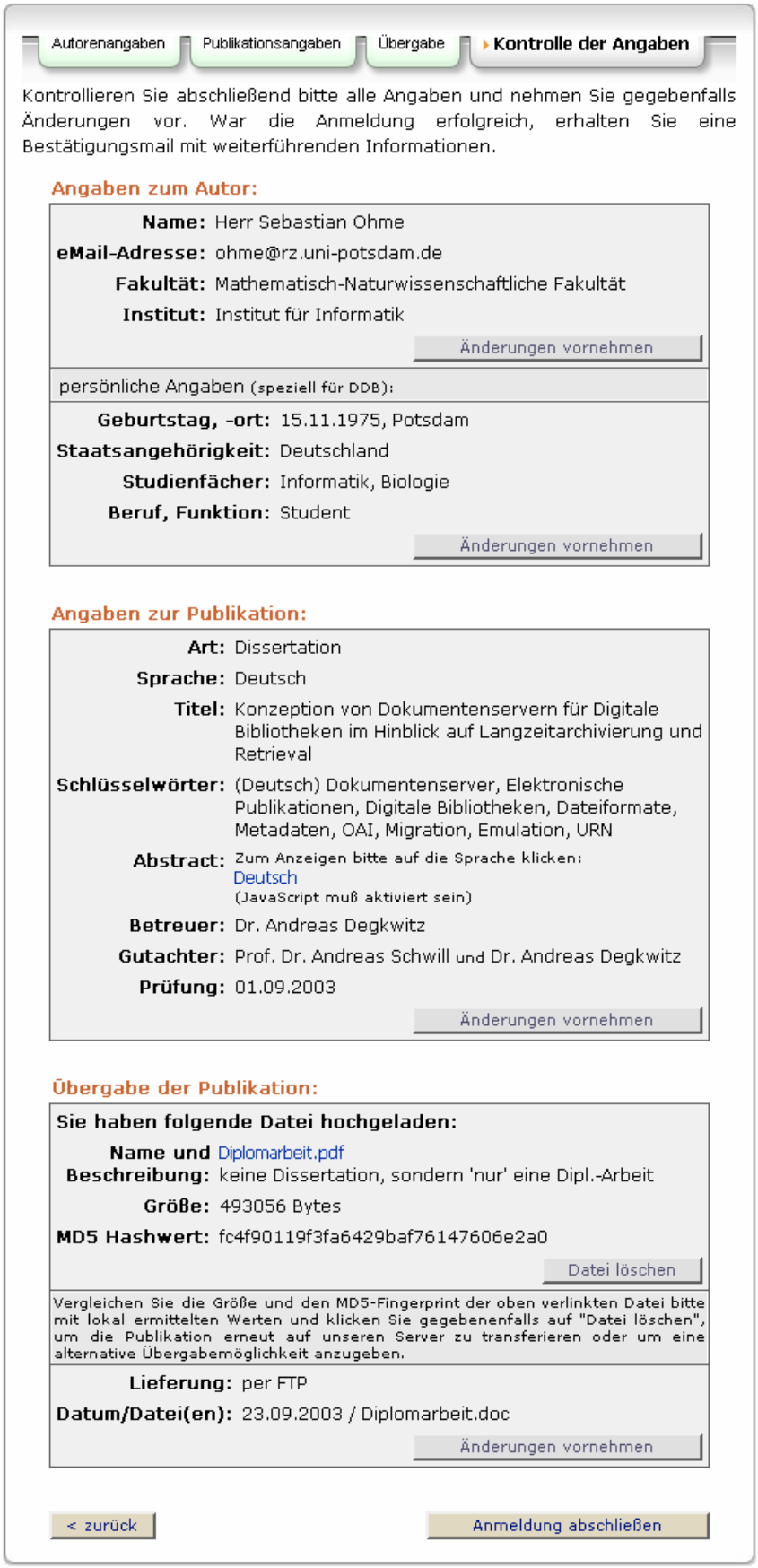


Abbildung 30: Kontrollformular

Der angesprochene Aufwand beim Design der Ausgabe läßt vermuten, daß der Programmcode vollkommen unflexibel gegenüber Veränderungen ist und es z.B. schwierig wird, weitere Formularfelder in die Übersicht aufzunehmen. Das Gegenteil ist der Fall: wie im gesamten restlichen System wurde auch hier extrem viel Wert auf eine einfache Erweiterbarkeit gelegt. Nicht kompliziert verschachtelte (und somit schlecht wartbare) if/else-Konstrukte geben an, wie die Kontrollseite auszusehen hat, sondern ein einfach strukturiertes Array, welches wie alle konfigurierenden Variablen in auswahl.pl abgelegt ist und auch von BibliotheksmitarbeiterInnen an sich verändernde Bedürfnisse angepaßt werden kann.

```
@felder = (
"-1-",
"Name|Autorenangaben_Geschlecht Autorenangaben_Adel Autorenangaben_Anrede
   Autorenangaben_Vorname Autorenangaben_Praefix Autorenangaben_Nachname",
"eMail-Adresse|Autorenangaben_Mail",
"?Telefon|Autorenangaben_Telefon",
"?Fakult&auml;t|Autorenangaben_Fakultaet",
"?(Bereich)|Autorenangaben_AlternativFakultaet",
"?Institut|Autorenangaben_Institut",
"?(Institut)|Autorenangaben_AlternativInstitut",
"-2-",
"vollst&auml;ndiger Name|Autorenangaben_vollerAdel
Autorenangaben_volleAnrede Autorenangaben_vollerVorname
Autorenangaben_vollesPraefix Autorenangaben_vollerNachname",
"andere Namensform|Autorenangaben_andererAdel Autorenangaben_andereAnrede
   Autorenangaben_andererVorname Autorenangaben_anderesPraefix
   Autorenangaben_andererNachname",
"GeburtsTagOrt",
"Laender",
"?URL der Homepage|Autorenangaben_Homepage",
"Faecher",
"Berufe",
"Werke",
"?Sonstige Angaben|Autorenangaben_Sonstiges",
"-3-",
"Art|Anmeldeformular_Publikationsart",
"Sprachen",
"Titel|Publikationsangaben_Titel",
[...]
```

Die Elemente des auszugsweise dargestellten Arrays haben eine definierte Struktur: der String vor dem zumeist vorhandenen „|“ entspricht dem auszugebenden Feldnamen. Dieser kann zusätzliche Zeichen beinhalten, die dem Parser angeben, ob es sich um ein optionales Feld handelt („?“) oder ob Abhängigkeiten zu beachten sind („(...)“). Fehlt der senkrechte Strich, wie z.B. in „Faecher“, sind alle gleichnamigen Formularfelder zusammenzufassen, was im Screenshot u.a. zu der Ausgabe „Informatik, Biologie“ führt („Faecher1=Informatik“, „Faecher2=Biologie“). Im Normalfall werden aber einfach alle hinter „|“ angegebenen Elemente dargestellt – der Verfassername könnte somit z.B. so aussehen: „Herr Graf Sebastian von und zu Ohme“ (ist ein Feld nicht belegt, wird es weggelassen). Trifft der Parser auf zwei Bindestriche und eine eingeschlossene Zahl, werden alle folgenden Einträge zu einer umrahmten Einheit zusammengefaßt. Abschluß eines solchen Blocks bildet schließlich ein Button, der zum zugehörigen Formular führt und nachträgliche Änderungen möglich macht. Dies gilt auch für den Knopf „Datei löschen“, der vor Weiterleitung zum „Übergabe“-Bereich allerdings noch die temporär abgelegte Datei aus dem „Session_ID“-Verzeichnis entfernt.

Ein Löschen ist natürlich nur notwendig, wenn es beim Hochladen des Dokuments Probleme gab und die auf dem Server abgelegte Version nicht der des Nutzers entspricht. Dieser hat im letzten Block daher die Möglichkeit, Inhalt und Größe der verlinkten Datei(en) zu kontrollieren. Ist auf dem Rechner des Anwenders außerdem ein Tool zur Berechnung des „Message Digest“ installiert, kann der lokal ermittelte Wert mit dem in der Übersicht angegebenen MD5-Hashwert verglichen werden. Der für die spätere Überprüfung der Integrität notwendige „Fingerabdruck“ der Publikation wird äußerst einfach mittels des Perl-internen „Digest“-Moduls erzeugt, welches neben MD5 auch weitere Algorithmen unterstützt (MD2, SHA1, ...).

```
require Digest::MD5;
if (open(MD5,"< $uploaddir\\$form{'Session_ID'}\\$form{$schluessel}"))
{
     binmode(MD5);
     $checksum = Digest::MD5->new->addfile(*MD5)->hexdigest;
     close(MD5);
}
else { $checksum = "Checksum-Berechnung fehlgeschlagen !?"; }
```

Die Berechnung des 32 Zeichen langen HEX-Strings dauert auch bei sehr großen Dateien nur wenige Sekunden und kann daher auch „on-the-fly“ erfolgen, um die Publikation zukünftig z.B. nur ‚herauszugeben‘, wenn die Integrität zwischenzeitlich nicht verletzt wurde.

Nach erfolgter Kontrolle und gegebenenfalls Korrektur der Angaben (hier kommt die mehrfach angesprochene Vorbelegung der Eingabefelder ins Spiel), kann die Anmeldung durch Klick auf den entsprechenden Button abgeschlossen werden. Dies ist auch der Zeitpunkt, an dem die bisher nur innerhalb der „hidden“-Felder vorgehaltenen Metadaten persistent auf dem Server abgelegt werden. Die Access-Datenbank (siehe nächster Abschnitt) kommt dafür aber noch immer nicht in Frage: aufgrund der fehlenden Autorisierungs-Komponente ist ein Mißbrauch des Systems (im Sinne einer nicht ernstgemeinten Publikationsanmeldung) nie ganz auszuschließen und so werden die Nutzereingaben zunächst in einer einfachen ASCII-Textdatei abgespeichert. Die Struktur ist dabei äußerst simpel: die Elementbezeichner werden in eckige Klammern eingefaßt; der zugehörige Feldinhalt wird in der bzw. den darauffolgenden Zeile(n) aufgeführt – Umbrüche in mehrzeiligen Eingabenfeldern (Veröffentlichungen des Autors, Abstracts, Schlüsselwörter, usw) bleiben somit erhalten.

```
[Session_ID]
20030815_204711
[Anmeldeformular_Publikationsart]
Dissertation
[Autorenangaben_Vorname]
Sebastian
[Autorenangaben_Nachname]
Ohme
[...]
[Publikationsangaben_Sprache1]
ger
[Publikationsangaben_Titel]
Konzeption von Dokumentenservern für Digitale Bibliotheken
im Hinblick auf Langzeitarchivierung und Retrieval
[...]
[Uebergabe_PDF]
Diplomarbeit.pdf
[...]
```

Zusätzlich zur Erzeugung der „_metadaten_.txt“-Datei im „Session_ID“-Verzeichnis werden auch zwei Benachrichtigungs-Mails verschickt: eine an den Nutzer, dessen Mailadresse ja bekannt ist – und die andere an alle potentiellen Bearbeiter:

```
use Mail::Sendmail;
unshift @{$Mail::Sendmail::mailcfg{'smtp'}},$mailserver;
%mail = (
 From => $systemmail,
 To => $adminmail,
 Subject => '- Publikationsanmeldung -',
 Message => "Hallo,\nsoeben wurde$art1 zur Veroeffentlichung angemeldet!\n
   (ID: $form{'Session_ID'}, eMail: $adressen)\n\n
   Unter $httpserver/manage?Anmeldungen koennen Sie alle\n
   Angaben kontrollieren und gegebenenfalls Aenderungen vornehmen.\n
   Dort finden Sie auch sonstige eventuell noch abzuarbeitene\n
   Anmeldungen und Informationen, wie Sie nun weiter vorgehen muessen.\n\n
   Schoene Gruesse, Sebastian Ohme\n"
);
sendmail %mail;
```

Das verwendete (Open Source) „Mail“-Modul ist leider kein offizieller Bestandteil der Perl-Distribution und so wurde es nachträglich ins System integriert.

Wurden die Mails erfolgreich verschickt und die Textdatei fehlerfrei erzeugt, wird eine abschließende Bestätigungsseite präsentiert, welche u.a. zum (weitestgehend vorausgefüllten) Copyright-Formular führt. Erst wenn dieses ausgedruckt und unterschrieben in der Abteilung Publikationen eingeht, werden die Metaangaben durch die jeweiligen Mitarbeiter kontrolliert, Datenbankeinträge erstellt und die Publikation schließlich für die Öffentlichkeit freigeschaltet (siehe Abschnitt 6.3.4).

Abbildung 31: Bestätigungsseite

Hatte der Autor „FTP“ als alternative Übergabemöglichkeit gewählt (also so wie im Beispiel geschehen), werden auf der letzten Seite die entsprechenden –personalisierten– Zugangsdaten angezeigt. Grundlage für deren Generierung bildet die „Session_ID“, die ja immer den Beginn der Publikationsanmeldung angibt und – zusammengesetzt aus Datum und Uhrzeit – eindeutig ist.[173] Mittels einfacher Regeln werden jeweils zwei Ziffern in ein Zeichen (Ziffer bzw. Klein-/Großbuchstabe) überführt:

```
00 - 09  ->  0 - 9
10 - 35  ->  a - z  (o -> Y)
36 - 59  ->  A - X  (O -> Z)
```

Für die Session_ID „20030622_015636“ ergibt sich also beispielsweise

```
"20030622" -> "k36m"
  "015636" -> "1UA"
```

und der zugehörige Perl-Codierungsalgorithmus sieht so aus:

```
($pass,$user) = split(/_/,$form{'Session_ID'});
$user = &code($user); $pass = &code($pass);

sub code
{
  my ($string) = shift(@_);
  my $part; my $result = "";
  for($i=0;$i<length($string);$i=$i+2)
  {
    $part = substr($string,$i,2);
    if ($part < 10) { $result .= chr($part+48); }
    elsif ($part < 36) { $result .= chr($part+87); }
    else { $result .= chr($part+29); }
  }
  $result =~ s/o/Y/g; $result =~ s/O/Z/g;
  return $result;
}
```

Die so erzeugte Username/Paßwort-Kombination wird – zusammen mit der Pfadangabe des zuvor erstellten FTP-Unterverzeichnisses – in die Xitami-Konfigurationsdatei eingetragen und ist sofort und *ohne* Web-/FTP-Server-Restart verwendbar.[174] Ein Upload von mehr als 50MB ist dabei allerdings nicht erlaubt, wie der nachfolgende „ftpusers.aut“-Auszug zeigt – und welcher gleichzeitig als Abschluß für diese doch recht umfangreiche Beschreibung der Anmeldekomponente dient:

```
[kLb]
    Root = "c:\\upload\\20030815_204711\\ftp"
    Password = "k38f"
    Access = "GPUD"
    Aliases = "0"
    Pipe = ""
    Use-quotas = "1"
    Soft-quota = "30.0"
    Hard-quota = "50.0"
```

[173] Die Vergabe gleicher Session_IDs ist nicht möglich: wird von zwei Nutzern gleichzeitig bis auf die Sekunde genau auf das Anmeldeformular zugegriffen, wird das Script Zufallsgenerator-bedingt so lange angehalten, bis eine neue ID frei ist.

[174] Mit Hilfe eines kleinen Perl-Scripts und eines Schedulers werden täglich eventuell überflüssige Einträge aus der Konfigurationsdatei entfernt.

6.3.3 Datenbankstruktur

Während die Vergabe und temporäre Speicherung der Metadaten noch ganz ohne Access auskommt, wird für ein performantes Retrieval und für die Bereitstellung der autoren- und publikationsspezifischen Angaben via OAI eine Datenbankanbindung benötigt. Und diese soll – anders als bei DSpace – auch tatsächlich zum Einsatz kommen: nicht eine gleichzeitige Stichwortsuche über alle Daten (dafür würden die Text-Dateien ausreichen), sondern eine strukturierte Recherche innerhalb einzelner Einträge sollte möglich sein. In diesem Abschnitt wird daher kurz auf den internen Aufbau der verwendeten Access-Datenbank eingegangen, die Dank ODBC und eines entsprechenden Moduls natürlich auch von Perl aus zugänglich ist.

```
require Win32::ODBC;
$db = new Win32::ODBC("DSN=Publikationen");
&ShowError("could not connect to database") if (!defined $db);
[...]
```

Aufgrund der besonderen Ausrichtung des Dokumentenservers auf die Entgegennahme und Archivierung von Examensarbeiten und des damit verbundenen Wunsches nach ‚Kompatibilität' zur Übergabeschnittstelle Der Deutschen Bibliothek war es notwendig, deren Metadatensätze genauer zu analysieren und die eigenen Strukturen entsprechend anzupassen. Erfreulicherweise sind METADISS und METAPERS sehr allgemein gehalten und so können die darin definierten Elemente auch problemlos auf andere Dokumenttypen übertragen werden. Auch Aufsätze haben beispielsweise einen Titel, einen (oder mehrere) Verfasser und lassen sich kurz in Form eines Abstracts beschreiben. Sonderfälle mußten bei der Modellierung der Tabellen und Felder somit nicht beachtet werden: Ziel war eine möglichst umfassende Datenbankstruktur, die alle für Such-, Browse- und OAI-Funktionalität relevanten Elemente aufnehmen kann und zudem DDB-konform ist. Letzteres bedeutet jedoch nicht, daß auch wirklich jedes als „obligatorisch" angegebene Feld[175] via Access vorgehalten werden muß. Es macht wenig Sinn, später in Angaben wie

```
<meta name="DC.Publisher.CorporateName.Address"
      content="Am Neuen Palais 10, 14469 Potsdam">
```

oder

```
<meta name="DDB.Contact.ID" content="L60000724">
```

zu suchen und so werden derartige Felder lediglich im Header der Frontdoor-Seiten aufgeführt (genaueres dazu im nächsten Abschnitt).

Die umfangreiche Übersicht auf den folgenden Seiten zeigt das Ergebnis der METADISS- und METAPERS-Analysen: Schritt für Schritt und in Absprache mit der Abteilung Publikationen wurden die insgesamt über 60 in der aktuellen Formatbeschreibung zu findenden Elemente durchgegangen und diejenigen extrahiert, die für die Charakterisierung der unterstützten Publikationsarten von besonderer Bedeutung sind. Schnell wurde dabei klar, daß die Datenbank aus mehr als nur einer großen Tabelle bestehen muß, da auch Inklusionen und Wiederholungen möglich sind. Einige Beispiele: jede Publikation hat einen Haupttitel, kann aber auch über weitere Untertitel und sogar Titel in einer anderen Sprache verfügen. Ein Autor wiederum könnte – sofern er persönliche Angaben preisgeben will – durchaus zwei Staatsangehörigkeiten besitzen, die natürlich irgendwo (ohne Redundanzen) gespeichert werden müssen. Und schließlich ist es möglich, daß es mehr als nur einen Verfasser gibt. Bei Dissertationen kommt dies zwar eher selten vor, aus oben genannten Gründen mußte dieser Fall aber dennoch berücksichtigt werden.

[175] siehe http://deposit.ddb.de/metadiss.htm

Die Übersicht ist wie folgt zu lesen: die dunkelgrau hinterlegten Zeilen bezeichnen einen zusammengehörigen Block, der meist n mal wiederholbar ist und die eingerückt dargestellten, hellgrau hinterlegten Elemente enthält. Außerdem kann ein solcher Block auch weitere Blöcke beinhalten, die später zu eigenständigen Datenbank-Tabellen werden. In der linken Spalte sind jeweils die METADISS- bzw. METAPERS-Bezeichner aufgeführt, die mittels eventueller Qualifier näher spezifiziert werden (rechte Spalte).

autorenspezifische Angaben:

Autoren (n mal wiederholbar)	
Vorname	
pc.name, DC.Creator.PersonalName	scheme="ddb-mn-pns" [Nachname,_*Vorname*_Präfix,_Titel,_Adelstitel]
Nachname	
pc.name, DC.Creator.PersonalName	scheme="ddb-mn-pns" [*Nachname*,_Vorname_Präfix,_Titel,_Adelstitel]
AkademischerTitel	
pc.name, DC.Creator.PersonalName	scheme="ddb-mn-pns" [Nachname,_Vorname_Präfix,_*Titel*,_Adelstitel]
Adelstitel	
pc.name, DC.Creator.PersonalName	scheme="ddb-mn-pns" [Nachname,_Vorname_Präfix,_Titel,_*Adelstitel*]
Praefix	
pc.name, DC.Creator.PersonalName	scheme="ddb-mn-pns" [Nachname,_Vorname_*Präfix*,_Titel,_Adelstitel]
EMail	
pc.address.email	
Institut	
pc.relation.isaffiliatedto	[Hochschule, Ort, *Fachbereich*]
VollstaendigerName	
pc.name.alternative.official	scheme="ddb-mn-pns" [Nachname,_Vorname_Präfix,_Titel,_Adelstitel]
AndereNamensform	
pc.name.alternative	scheme="ddb-mn-pns" [Nachname,_Vorname_Präfix,_Titel,_Adelstitel]
Geburtstag	
pc.description.dateofbirth, DC.Creator.PersonalName.DateOfBirth	scheme="w3cdtf"
Geburtsort	
pc.description.placeofbirth, DC.Creator.PersonalName.PlaceOfBirth	

Geschlecht	
pc.description.gender	

Homepage	
pc.relation.hashomepage	

Autoren_Staatsangehoerigkeiten (2 mal wiederholbar)

Land	
pc.description.nationality	

Autoren_Studienfaecher (n mal wiederholbar)

Fach	
pc.subject.topic	

Autoren_Berufe (n mal wiederholbar)

Beruf	
pc.subject.profession	

Autoren_Veröffentlichungen (n mal wiederholbar)

Werk	
pc.relation.haswritten	

Sonstiges	
pc.description.note	

---

publikationsspezifische Angaben:

Publikationen

Art	
DC.Type	

Publikationen_Sprachen (n mal wiederholbar)

Sprache	
DC.Language	SCHEME="ISO639-2"

Titel	
DC.Title	LANG="[3-stelliger Sprachcode]"

Publikationen_Untertitel (n mal wiederholbar)	
WeitererTitel	
DC.Title.Alternative	LANG="[3-stelliger Sprachcode]"

Publikationen_TitelAndereSprache (n mal wiederholbar)	
AndererTitel	
DC.Title.Translated	LANG="[3-stelliger Sprachcode]" → AndereSprache

Publikationen_Schluesselwoerter (n mal wiederholbar)	
Schluesselwoerter	
DC.Subject	SCHEME="freetext" → SchluesselSprache

DNB (Sachgruppe)	
DC.Subject	SCHEME="DNB-Sachgruppe"

RVK (Klassifikation)	
DC.Subject	SCHEME="RVK"

SWD (Schlagworte)	
DC.Subject	SCHEME="SWD"

Publikationen_Abstracts (n mal wiederholbar)	
Abstract	
DC.Description	LANG="[3-stelliger Sprachcode]" → AbsSprache

Publikationen_Betreuer (n mal wiederholbar)	
Betreuer	
DC.Contributor.Advisor	

Gutachter1	
DC.Contributor.Referee	

Gutachter2	
DC.Contributor.Referee	

Antragstellung	
DC.Date.Submitted	SCHEME="W3CDTF"

Pruefung	
DC.Date.Accepted	SCHEME="W3CDTF"

PublikationsID	
DC.Identifier	SCHEME="URL"

PublikationsID + Pruefziffer	
DC.Identifier	SCHEME="URN:NBN:DE"

MD5	
DDB.Identifier.Fingerprint	SCHEME="MD5"

Diese Liste wurde schließlich in eine korrespondierende Datenbankstruktur überführt. Die Blöcke und Unterblöcke bilden dabei eigenständige Tabellen, die über entsprechende Schlüssel referenzier- und zuordbar sind. Primärschlüssel ist die PublikationsID, die sich aus der zweistelligen Jahreszahl und einer laufenden Nummer zusammensetzt. Die Abbildung der nächsten Seite zeigt alle Tabellen, die darin enthaltenen Felder und die Beziehungen der Elemente zueinander. Auf ein vollständiges ER-Modell[176] wurde – ebenso wie auf die Angabe der Datentypen – der Einfachheit halber verzichtet; zur Visualisierung wurden Access-interne Funktionalitäten genutzt. Neben den in der Übersicht angegebenen DDB-konformen Feldern werden auch system-interne Daten, wie z.B. die Dateinamen, das Publikationsdatum (= Freischaltungsdatum, siehe nächster Abschnitt) oder der Tag der FTP-Lieferung gespeichert. Außerdem gut zu erkennen sind die zusätzlichen IDs innerhalb der einzelnen Tabellen, die es beispielsweise möglich machen, einer Publikation mehrere Autoren und diesen wiederum mehrere Berufe zuzuordnen.

[176] das „Entity-Relationship Model“ dient der grafischen Darstellung von Datenbankstrukturen

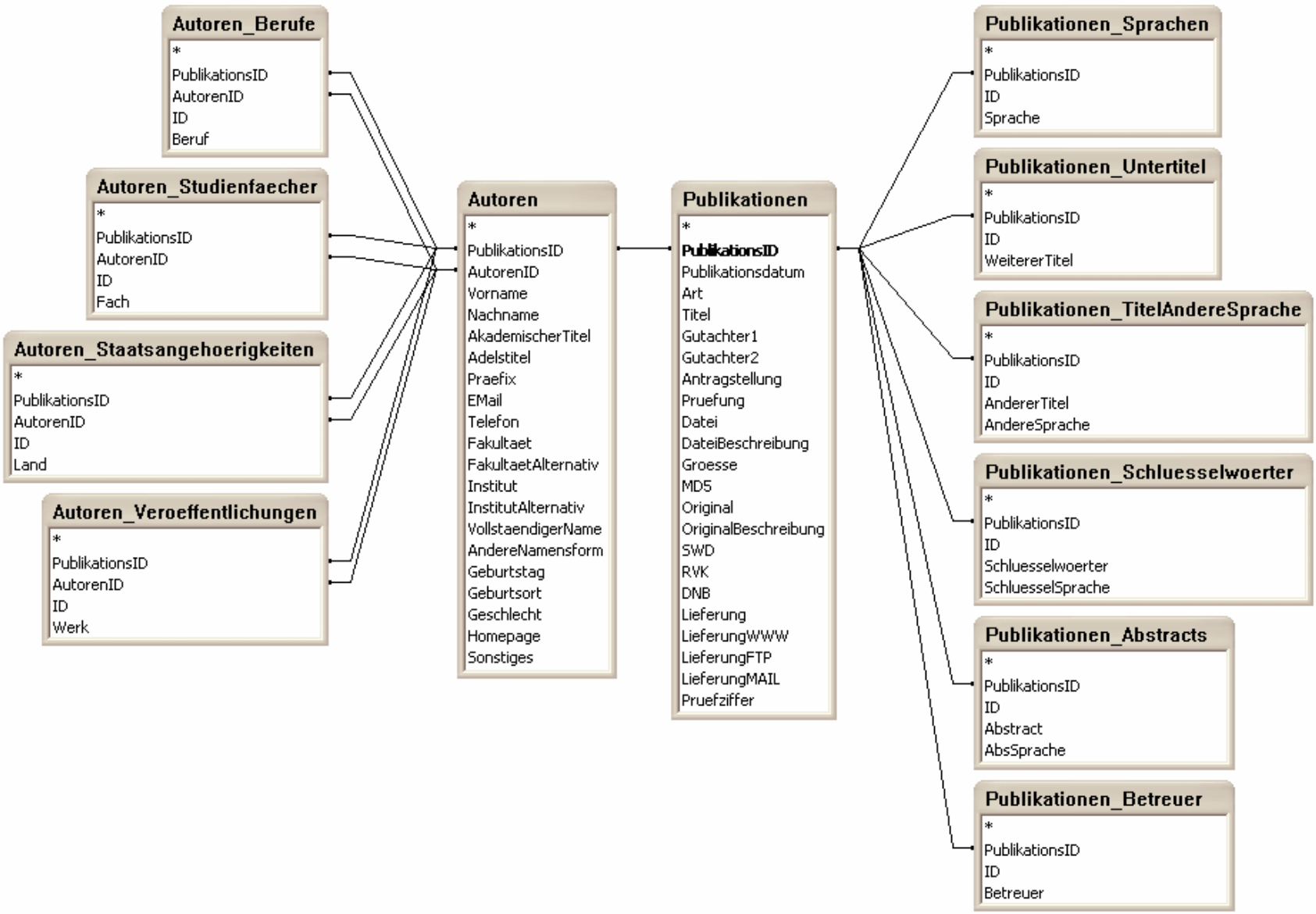


Abbildung 32: Datenbankstruktur

6.3.4 Anmeldungsmanagement

Zur Verwaltungskomponente, die paßwortgeschützt über „/manage?Anmeldungen“ zu erreichen ist, braucht eigentlich nicht mehr viel gesagt werden; zumindest was die Oberfläche betrifft. Diese entspricht nämlich bis auf wenige Erweiterungen dem bereits sehr ausführlich beschriebenen Anmelde-Frontend – und zwar aus gutem Grund: auch (und gerade) die MitarbeiterInnen der Abteilung Publikationen sollten die zwischenzeitlich eingegangenen Metadaten nicht nur kontrollieren, sondern gegebenenfalls auch komfortabel editieren können, falls sich irgendwo ein Buchstabendreher eingeschlichen hat. Es lag daher nahe, die recht ausgereiften und eigentlich für die Anmeldung vorgesehenen Formulare auch für die Überprüfung und Freischaltung der Publikationen zu verwenden. Um dies zu erreichen, mußte zunächst eine Möglichkeit geschaffen werden, die in den „_metadaten_.txt“-Dateien vorliegenden Formulareingaben wieder in den „Durchreichprozeß“ einzuspeisen. Zuständig dafür ist ein Formular, welches der bereits bekannten Kontrollseite vorgeschaltet ist und die jeweiligen Dateiinhalte in Form von „hidden“-Feldern enthält.

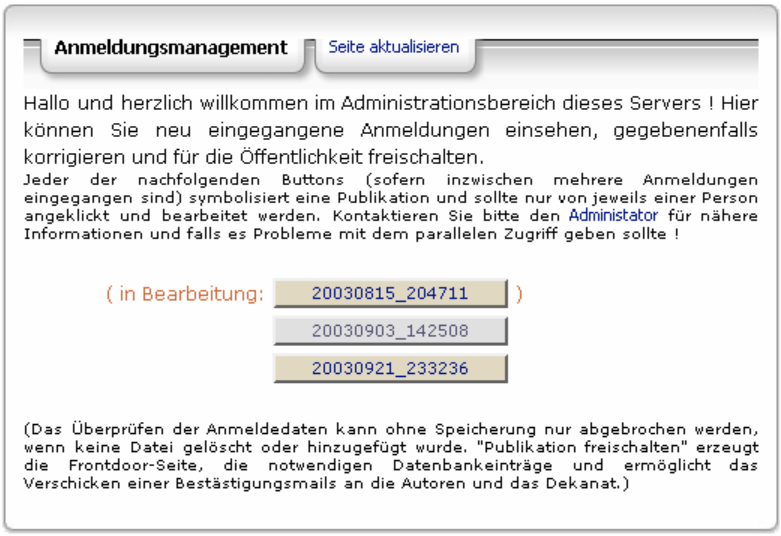


Abbildung 33: Taskpool

Sicherlich sofort wiederzuerkennen sind die „Session_IDs“, die hier gleichzeitig Button-Bezeichner sind und jeweils eine noch nicht (vollständig) abgearbeitete Publikations-anmeldung repräsentieren. Bei der Generierung dieser Übersicht wird wie folgt vorgegangen: in einer Schleife werden alle im Upload-Directory vorhandenen „Session_ID“-Unter-verzeichnisse nach „_metadaten_.txt“-Dateien abgesucht. Wird das Script fündig (auch hier kommt register.pl zum Einsatz), wird zunächst ein neuer „`<form action="/manage" method="post">`“-Block aufgemacht und dann jeder Eintrag in einen entsprechenden „hidden“-Tag transformiert; Umbrüche bleiben dabei erhalten:

```
...
[Publikationsangaben_Abstract1]
dies ist
eine kurze Zusammenfassung
[Publikationsangaben_AbsSprache1]
ger
[Publikationsangaben_Abstract2]
...
```

```
...
<input type=hidden name="Publikationsangaben_Abstract1" value="dies ist
eine kurze Zusammenfassung">
<input type=hidden name="Publikationsangaben_AbsSprache1" value="ger">
...
```

Abschluß eines solchen Blocks bildet der Submit-Knopf – und natürlich ein schließendes `</form>`. Klickt ein Bearbeiter nun auf eine derart präparierte Session_ID, wird einfach zur Kontrollseite weitergeleitet, die nun annimmt, daß es sich um eine ‚normale' Anmeldung handelt ... lediglich das „/manage" in der URL verrät den Ursprung der Anfrage.
Natürlich ist dabei auch ein paralleler Zugriff durch zwei oder mehr Mitarbeiter möglich. Diese können sich zur gleichen Zeit den selben Metadatensatz anschauen und sich z.B. telefonisch über Aufnahme oder Ablehnung austauschen. Nur einer (der Schnellere) kann die Publikation aber schließlich freischalten – außerdem sollte auch nur ein Bearbeiter Änderungen vornehmen. Sinnvoller ist daher die arbeitsteilige Kontrolle unterschiedlicher Anmeldungen. Hat sich einer der Administratoren für eine Session_ID entschieden, wird eine „_metadaten_.lock"-Datei im entsprechenden Verzeichnis erzeugt bzw. der darin enthaltene Counter hochgezählt. Im Formular wird dies durch ein „in Bearbeitung" deutlich gemacht und auch erst wenn alle parallelen Zugriffe durch ein „Cancel" (dazu gleich mehr) abgebrochen wurden, ist eine Freischaltung wieder ‚gefahrlos' möglich.

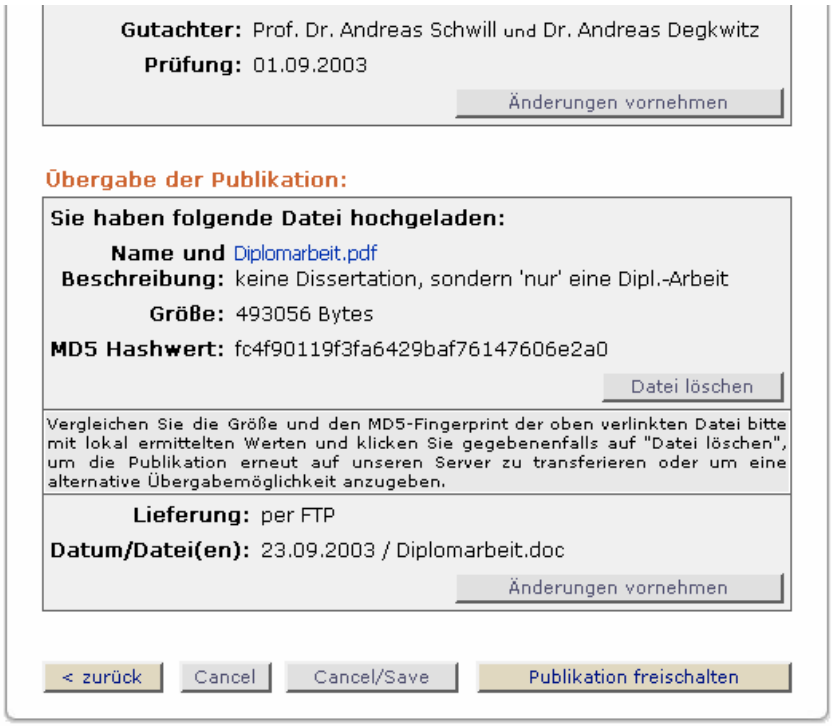


Abbildung 34: Kontrollformular (Management)

Die im Screenshot nur auszugsweise dargestellte Seite entspricht (fast) exakt dem Kontrollformular, welches der anmeldende Nutzer relativ zum Schluß zu Gesicht bekommen hatte. Einziger Unterschied (neben einem „Management"-Eintrag in der weggelassenen Menüleiste): statt „Anmeldung abschließen" (siehe auch Abschnitt 6.3.2) existiert nun ein Button „Publikation freischalten". Außerdem sind zwei neue Knöpfe hinzugekommen. Zum einen „Cancel/Save", der auf allen Management-Seiten zu finden ist, und „Cancel", der nur solange existiert, solange keine Datei gelöscht oder neu hochgeladen wurde. Der Grund leuchtet sicherlich ein: während „Cancel/Save" die eventuell veränderten Metadaten vor dem Abbruch speichert, wird bei „Cancel" umgehend zur Übersichtsseite zurückgekehrt. Dies würde im Falle neuer oder gelöschter Dokumente jedoch zu Inkonsistenzen führen, da auf dem Server dann Dateien vorhanden/nicht mehr vorhanden sind, die in „_metadaten_.txt" noch bzw. noch nicht aufgeführt sind. Weiterer Unterschied zwischen „Cancel" und „Cancel/Save": ersteres zählt den angesprochenen Counter herunter und löscht die Lock-Datei, wenn dies 0 ergibt.

Beim Speichern wird die Lock-Datei hingegen grundsätzlich gelöscht, was es unter anderem möglich macht, den Status einer Bearbeitung nach einem Browserabsturz oder einer versehentlichen „Back“-Button-Benutzung zurückzusetzen. Insgesamt kann das Management neu eingegangener Anmeldungen als recht durchdacht bezeichnet werden und braucht sich nach Meinung des Autors auch nicht vor den von DSpace gebotenen und doch sehr ähnlichen Funktionalitäten zu verstecken.
Besser als im Referenz-System DSpace gelöst (da dort nicht vorhanden), ist die komfortable Dateiübernahme vom FTP-Server. Hatte der Nutzer diese alternative Transfer-Variante gewählt, mußte er ja auch das voraussichtliche Datum der Übergabe angeben. Der Bearbeiter kennt dieses Datum (siehe Abbildung 34) und so kann er zu gegebener Zeit mit „Änderungen vornehmen“ den Upload-Bereich aufrufen und die dort aufgeführte Datei in das „Session_ID“-Verzeichnis übernehmen.

Abbildung 35: Upload-Formular (Management)

Da im Beispiel nur eine Datei („Diplomarbeit.doc“) hochgeladen wurde, wird auch nur ein zusätzlicher Button angezeigt. Grundsätzlich möglich wäre auch eine gezielte Auswahl aus allen vorhandenen Dateien – und natürlich der ‚normale‘ Upload via Browser, falls das Dokument die Abteilung Publikationen auf herkömmlichem Wege erreicht hat.

Wurden schließlich alle Angaben überprüft, eventuelle Fehler behoben und fehlende Dateien transferiert, kann die Publikation endlich für die Öffentlichkeit freigeschaltet werden. Dies ist wie erwähnt von der abschließenden Kontrollseite aus möglich, die nun optional auch nur noch mittels „Cancel/Save“ verlassen werden kann.

Abbildung 36: Kontrollseite nach Upload (Management)

Erster Schritt bei der Freischaltung ist die Speicherung der Metadaten in der Text-Datei. Geht anschließend etwas schief, sind die Änderungen zumindest nicht verloren. Zweiter Schritt ist die Ermittlung der höchsten PublikationsID des aktuellen Jahres und deren Erhöhung um 1. Hier kommt zum ersten Mal auch die Access-Datenbank ins Spiel, deren Struktur im letzten Abschnitt etwas genauer vorgestellt wurde.

```
(!$db->Sql("SELECT Max(Publikationen.PublikationsID) AS hoechsteID
            FROM Publikationen WHERE (PublikationsID Like '$jahr%');"))
  || &ShowError("[...]" . $db->error);
if (!$db->FetchRow()) { $id = $jahr . "0001"; }
else
{
  $id = $db->Data("hoechsteID");
  if (substr($id,0,2) ne $jahr) { $id = $jahr . "0001"; }
  else { $id = $jahr . sprintf("%04d",substr($id,2,4)+1); }
}
```

Auf Syntax und Semantik der SQL-Abfrage und der if/else-Zweige soll an dieser Stelle nicht näher eingegangen werden. Wichtig zu wissen ist, daß die zurückgelieferte Ergebnismenge (falls nicht leer) mittels $db->FetchRow traversiert werden kann und mit Hilfe von $db->Data bzw. $db->DataHash ein Zugriff auf die Elemente der aktuellen Zeile möglich ist.
Die so erzeugte eindeutige PublikationsID dient zum einen als Primärschlüssel für die Datenbank, vor allem aber ist sie für die URL- und URN-Vergabe von Bedeutung. Wie eine DDB-konforme URN dabei auszusehen hat, wurde bereits ausführlich in Kapitel 5.2.2 beschrieben – hier soll (für den interessierten Leser) lediglich der Prüfziffer-Algorithmus angegeben werden, der natürlich der verbalen Beschreibung[177] entspricht und selbst implementiert wurde, um (dem eigenen Anspruch treu bleibend) einen weiteren der vielen theoretischen Aspekte auch einmal selbst praktisch umzusetzen.

```
sub Pruefziffer
{
  my($urn) = shift(@_); my $string = ""; my $pruefziffer = 0;
  my %conversion = (
    "0" => 1, "1" => 2, "2" => 3, "3" => 4, "4" => 5, "5" => 6,
    "6" => 7, "7" => 8, "8" => 9, "9" => 41, "a" => 18, "b" => 14,
    "c" => 19, "d" => 15, "e" => 16, "f" => 21, "g" => 22, "h" => 23,
    "i" => 24, "j" => 25, "k" => 42, "l" => 26, "m" => 27, "n" => 13,
    "o" => 28, "p" => 29, "q" => 31, "r" => 12, "s" => 32, "t" => 33,
    "u" => 11, "v" => 34, "w" => 35, "x" => 36, "y" => 37, "z" => 38,
    "-" => 39, ":" => 17 );
  for($i=0;$i<length($urn);$i++)
  {
    $string .= $conversion{substr($urn,$i,1)};
  }
  for($i=0;$i<length($string);$i++)
  {
    $pruefziffer = $pruefziffer + (($i+1)*substr($string,$i,1));
  }
  $pruefziffer = int($pruefziffer/substr($string,length($string)-1,1));
  return substr($pruefziffer,length($pruefziffer)-1,1);
}
```

[177] http://www.persistent-identifier.de/?link=316

Nachdem diese und einige weitere Vorbereitungen getroffen wurden, werden die bisher in den versteckten Feldern und im assoziativen Array „%form“ vorgehaltenen Formulareinträge schließlich mittels „INSERT INTO [...]“-SQL-Anweisungen in die Datenbank überführt – und natürlich kommen auch hierfür wieder Konfigurationsdateien zum Einsatz, die eine überaus einfache Erweiterung oder Anpassung des Systems ermöglichen.

```
@zuordnungpublikation = (
  "Anmeldeformular_Publikationsart|Art",
  "Publikationsangaben_Titel|Titel",
  [...]
  "Schluesselwoerter+SchluesselSprache|Publikationen_Schluesselwoerter",
  "Abstract+AbsSprache|Publikationen_Abstracts",
  [...]
  "Kontrolle_Datum|Publikationsdatum" );
```

```
$db->Sql("INSERT INTO $value (PublikationsID,ID,$key)
          VALUES ('$id','$i','$string');");
```

Das im Kästchen auszugsweise dargestellte Array und eine der vielen zugehörigen Programmzeilen läßt erahnen, wie der Parser vorgeht: er ermittelt den Namen der Zieltabelle und fügt dort nacheinander die Inhalte der jeweiligen Variablen zusammen mit den internen IDs ein. %form{‘Abstract‘} und %form{‘AbsSprache‘} landen so z.B. in der Tabelle „Publikationen_Abstracts“.

Nächster und vorletzter großer Schritt ist die Generierung statischer HTML-Seiten, die für die spätere (eventuelle) Lieferung der Metadaten an Die Deutsche Bibliothek notwendig sind. Voraussetzung dafür ist das Vorhandensein eines Verzeichnisses im „webpages“-Directory des Xitami-Servers, in dem auch das Präsentationsformat der Publikation abgelegt wird. Pfad und somit URL der zukünftigen Frontdoor-Seite leitet sich aus der PublikationsID ab: für „030006“ wird beispielsweise im Verzeichnis 03 ein Verzeichnis 0006 erzeugt, so daß Metadaten und Volltext später unter der einprägsamen URL http://pub.ub.uni-potsdam.de/03/0006 abrufbar sind.

Die Erzeugung der METADISS- und METAPERS-Header übernimmt – wie sollte es anders sein – erneut (und ein letztes Mal) ein eigener Parser, der anpaßbare Konfigurationsdateien einliest und die von Der Deutschen Bibliothek geforderten Metadatensätze erzeugt.

```
DC.Identifier
     scheme="URL"
     content=">URL<"
DC.Identifier
     scheme="URN:NBN:DE"
     content="urn:nbn:de:kobv:517->ID<>Uebergabe_Pruefziffer<"
[...]
DC.Type
     content=">Anmeldeformular_Publikationsart<"
{ Publikationsangaben_Sprache?
DC.Language
     scheme="ISO639-2"
     content=">Publikationsangaben_Sprache?<"
}
[...]
```

```
<meta name="DC.Identifier" scheme="URL"
      content="http://pub.ub.uni-potsdam.de/03/0006">
<meta name="DC.Identifier" scheme="URN:NBN:DE"
      content="urn:nbn:de:kobv:517-0000016">
[...]
<meta name="DC.Type" content="Text.Thesis.Doctoral">
<meta name="DC.Language" scheme="ISO639-2" content="ger">
[...]
```

Die Syntax der angegebenen metadiss.txt-Beispieleinträge (siehe letzte Seite) ist sehr einfach gehalten: ein am Zeilenanfang beginnender Dublin Core-Bezeichner charakterisiert jeweils einen zugehörigen „<meta [...]>“-Tag, dessen Argumente eingerückt darunter aufgeführt sind. Platzhalter wie „>URL<“ werden durch die entsprechenden Variableninhalte ersetzt (also z.B. %form{‘URL‘}); Klammerungen („{ ... }“) und Fragezeichen verdeutlichen einen wiederholbaren Block. Ist die Publikation beispielsweise zweisprachig, werden auch zwei "DC.Language"-Metatags erzeugt.
Die METAPERS-Datensätze landen in metapers.htm, die METADISS-Einträge wiederum in der eigentlichen index.htm-Datei, welche außerdem ein Frameset enthält, das bei Aufruf zu „/show“ + PublikationsID weiterleitet. Dank der Ablage beider Dateien unterhalb des webpages-Directories kann die Publikation sehr einfach durch Angabe der URLs an Die Deutsche Bibliothek gemeldet werden – früher war hierfür ein mühsames Copy/Paste der <meta>-Tags in die Anmelde-Mail erforderlich (siehe Abschnitt 6.1).

Nachdem das Präsentationsformat in das zugehörige Xitami-Unterverzeichnis und alle weiteren noch im Session_ID-Ordner verbliebenen Dateien (_metadaten_.txt, Archivierungsformat der Publikation, auf den FTP-Server transferierte Files, usw.) über eine geschützte Netzwerkverbindung auf einen ebenso geschützten anderen Server verschoben wurden, wird als endgültig letzter Schritt der Freischaltung noch eine abschließende Bestätigungsseite angezeigt. Diese erlaubt es zum einen, umgehend die neue Frontdoor-Seite zu begutachten (und dort nach Fehlern Ausschau zu halten), und zum anderen gestattet sie den bequemen Versand einer Benachrichtigungsmail an den oder die Autor(en) der Publikation und an das zuständige Dekanat, falls es sich um eine Examensarbeit handelt. Schön im nebenstehenden letzten Screenshot dieses Abschnitts zu erkennen ist der dynamisch erzeugte Inhalt dieser Mail ... und die Möglichkeit, sofort mit der Bearbeitung der nächsten Anmeldung fortzufahren.

Abbildung 37: Bestätigungsseite (Management)

6.3.5 Dokumentrecherche

Während Anmelde- und Managementkomponente (inklusive automatischer URN-Vergabe, Hashwert-Berechnung und DDB-Metadatensatz-Erzeugung) ‚nur' für die publizierenden Autoren und für den internen Workflow innerhalb der Universitätsbibliothek von Bedeutung sind, steht für den recherchierenden Nutzer insbesondere der schnelle und komfortable Zugriff auf die archivierten Dokumente im Vordergrund. Dazu gehört – neben der Möglichkeit des ‚Browsens' nach Titeln, Verfassern, Fakultäten, Instituten, ... und der Suche in den Metadaten und Volltexten – vor allem auch eine geeignete Anzeige der autoren- und publikationsspezifischen Angaben. Zuständig für deren Präsentation sind auch hier die Frontdoor-Seiten, die den (PDF-)Dokumenten vorgeschaltet sind und einen ersten Überblick über den Inhalt der jeweiligen Publikation geben.

Die Generierung übernimmt das bereits bestens bekannte show.pl-Script, welches die ID als Parameter übergeben bekommt und anschließend mit Hilfe mehrerer SQL-„SELECT"-Abfragen alle relevanten Datenbankeinträge extrahiert und – hübsch formatiert – ausgibt. Der nachfolgende Screenshot zeigt eine solche Frontdoor-Seite, die über „/show?030006" zu erreichen und im Normalfall natürlich auch von Menüleiste und Logo umgeben ist:[178]

Abbildung 38: Frontdoor-Seite (Teil 1)

Titel und „/fulltext"-Link führen zum Dokument selbst – allerdings nicht direkt, sondern über ein zwischengeschaltetes Script (erneut show.pl), welches die Datei vor Herausgabe auf Integrität prüft. Dazu wird ein frisch berechneter MD5-Fingerprint mit dem in der Datenbank gespeicherten Hashwert verglichen und nur wenn beide identisch sind, ist ein Download der Publikation möglich.

[178] Das auf der letzten Seite angesprochene Frameset macht auch eine URL wie „http://pub.ub.uni-potsdam.de/03/0006" möglich.

Weiterer Vorteil der Zwischenschaltung von show.pl („/show?030006“ bzw. „/show?030006_Fulltext“): so kann wunderbar protokolliert werden, wie oft auf ein bestimmtes Dokument oder aber auf die zugehörige Frontdoor-Seite zugegriffen wurde, was letztlich die geforderten Statistiken möglich macht (siehe Kapitel 6).

Die Präsentation der Metadaten wurde der Übersichtlichkeit halber auf zwei Seiten verteilt. Während der erste und in Abbildung 38 dargestellte Teil eher inhaltliche Angaben (Titel, Schlüsselwörter, Abstracts, usw.) enthält, werden auf der zweiten Seite zusätzliche autoren-, dokument- und dissertations-spezifische Daten angezeigt. Weggelassen werden dabei persönliche Angaben, wie Staatsangehörigkeit oder Geburtsort, die für die Dokumentbeschreibung nicht relevant sind. Wie all dies in Kombination mit den restlichen Elementen der Benutzerschnittstelle aussieht, soll der nächste Screenshot verdeutlichen, welcher wegen der dargestellten Menüleiste auch die Überleitung zu den Such- und Browsekomponenten einfacher macht.

Abbildung 39: Frontdoor-Seite (Teil 2)

Ein Dokumentenserver ist nichts ohne umfassende Recherchemöglichkeiten und so bietet das hier implementierte System gleich eine ganze Reihe entsprechender Funktionalitäten. Der Nutzer hat die Wahl zwischen einer strukturellen Suche in der Metadatenbank, einer Volltextsuche sowohl innerhalb der Webpräsenz als auch in den Publikationen (derzeit wird PDF und HTML unterstützt, dazu später mehr) und einer einfachen Auflistung aller Titel, Verfasser und Fakultäten/Institute, wobei letzteres zuerst vorgestellt werden soll.

Der Browse-Bereich wird parametrisiert gestartet: „/browse?Uni-Struktur“ beispielsweise ruft show.pl/browse.pl auf und generiert eine Übersicht aller Fakultäten und Einrichtungen der Universität Potsdam. Dazu wird einfach die „@fakultaeten“-Liste der Konfigurationsdatei auswahl.pl durchgegangen und in der Access-Datenbank nachgeschaut, ob entsprechende Publikationen existieren. Wenn ja, wird ein Hyperlink mit dem zugehörigen Fakultätsschlüssel (also z.B. „/browse?MatNat“) erzeugt und außerdem deutlich gemacht, wie viele Dokumente dort ansässiger Autoren vorrätig sind. Existieren keine Datenbankeinträge, so erfolgt auch keine Verlinkung und es wird eine (0) ausgegeben. Klickt ein Nutzer anschließend auf einen vorhandenen Link, wird die gleiche Seite nochmals aufgebaut, allerdings werden zusätzlich auch die jeweiligen Institute samt zugehöriger Dokumentenanzahl angezeigt, was für das MatNat-Beispiel schließlich wie folgt aussieht:

Abbildung 40: Browsing (Teil 1)

```
$query = "SELECT";
$query .= " Autoren.Fakultaet, Autoren.Institut,
  Publikationen.PublikationsID,";
$query .= " Publikationen.Art, Autoren.Vorname, Autoren.Nachname,
  Publikationen_Sprachen.Sprache,";
[...]
$query .= " FROM";
$query .= " ((((Publikationen INNER JOIN Autoren ON
  Publikationen.PublikationsID = Autoren.PublikationsID)";
$query .= " LEFT JOIN Publikationen_Abstracts ON
  Publikationen.PublikationsID = Publikationen_Abstracts.PublikationsID)";
[...]
$query .= " WHERE";
$query .= " (Autoren.Fakultaet like '$fakultaet') AND
  (Autoren.Institut like '$institut')";
$query .= " ORDER BY Publikationen.PublikationsID,
  Publikationen_Sprachen.ID,Publikationen_Untertitel.ID,Autoren.AutorenID,
  Publikationen_Abstracts.ID,Publikationen.Publikationsdatum DESC;";
```

Programmtechnisch realisiert wurde die Suche und spätere Anzeige der jeweils zum übergebenen Parameter passenden Einträge mit Hilfe einer dynamisch zusammengebauten SQL-Abfrage (siehe Kästchen auf der letzten Seite), die die gesamte Arbeit der Verknüpfung der Tabellen und der Sortierung der Treffermenge an das Datenbanksystem weiterreicht. Das Ergebnis der anschließenden Auswertung aller zurückgelieferten Datensätze (%feld = $db->DataHash();) ist im nächsten Screenshot dargestellt; das Script wurde dabei mit „/browse?MatNat_Mathe“ aufgerufen:

Abbildung 41: Browsing (Teil 2)

Die Anzeige der Metadaten übernimmt eine Unterroutine, die aus Gründen der Nachnutzbarkeit in das initiale show-Programm aus- bzw. eingelagert wurde: auch die zugehörige Metadaten-Suchmaschine, die im folgenden kurz vorgestellt wird, nutzt diesen Programmcode für die Auflistung aller gefundenen Datenbankeinträge.[179]

Für ein besseres Verständnis der Ausführungen sei schon jetzt auf Abbildung 42 verwiesen. Dort wurden Metadaten- und Volltext-Suchmaschine gegenübergestellt und zudem mit Beispieleingaben ‚gefüttert‘, die die gebotenen Funktionalitäten recht gut erkennen lassen.

Zur metasearch.pl-benannten ersten Suchkomponente (/search) muß eigentlich nicht mehr viel gesagt werden: die Formularfelder werden ähnlich wie im Anmeldemanagement generiert und vorbelegt, der Zugriff auf die Datenbank erfolgt mittels dynamisch generierter SQL-Queries und die Treffermenge wird mit Hilfe der bereits angesprochenen ShowTitle-Subroutine ausgegeben. Auf zwei wichtige Unterschiede sollte aber explizit eingegangen werden: zum einen nimmt die aus den Nutzereingaben erzeugte SQL-Abfrage im Vergleich zur Browsekomponente gigantische Ausmaße an, da es möglich ist, so gut wie in jedem Datenbankfeld zu suchen – und dies unter Zuhilfenahme boolescher Verknüpfungen.

[179] Auf eine Beschreibung des Abstract-Anzeigescripts und der Titel-/Verfasser-Browsefunktion soll an dieser Stelle verzichtet werden.

Ein Beispiel für eine solche SELECT-Anweisung ist ebenfalls in Abbildung 42 zu finden und veranschaulicht schon recht gut, wie der String zusammengesetzt wird: die Wörter im Titel/ Untertitel-Feld („Lehr und Sprach“), die laut Pull-Down-Menü ja alle vorkommen müssen, werden z.B. über mehrere Split- und Konkatenations-Operationen in folgende (zusammenhängende) Zeichenkette überführt:

```
((Publikationen.Titel Like '%Lehr%') OR (Publikationen_Untertitel.WeitererTitel Like '%Lehr%'))
 AND
((Publikationen.Titel Like '%und%') OR (Publikationen_Untertitel.WeitererTitel Like '%und%'))
 AND
((Publikationen.Titel Like '%Sprach%') OR (Publikationen_Untertitel.WeitererTitel Like '%Sprach%'))
```

Bei einer Phrasensuche nach dem Betreuer/Gutachter „Strohe, Gerhard“ würde das Splitten an den Leerzeichen wiederum entfallen und folgender Teilstring entstehen:

```
((Publikationen_Betreuer.Betreuer Like '%Strohe, Gerhard%') OR (Publikationen.Gutachter1
  Like '%Strohe, Gerhard%') OR (Publikationen.Gutachter2 Like '%Strohe, Gerhard%'))
```

Insgesamt also ein recht ausgefeiltes Suchsystem; nichts ist hart codiert – kommt ein weiteres Datenbankfeld hinzu, kann (nach Anpassung der Konfiguration und somit des Frontends) sofort danach gesucht werden. Und die Eingaben des Anwenders wirken sich dabei sogar auf die Anzeige aus: erst wenn auch explizit nach einem Betreuer gesucht wird, erscheint ein entsprechender Eintrag in der Treffermenge.

Die besonders intensive Nutzung von Datenbankfunktionalitäten ist einer der angesprochenen Unterschiede – der andere betrifft die Übergabe der Formularfelder an das CGI-Script. Zwar kommen auch in metasearch.pl ein FormHandler und ein FormGenerator zum Einsatz, allerdings wird nicht „POST“, sondern „GET“ für die Übertragung genutzt. Der Grund: werden mehr Datensätze gefunden, als auf eine Seite ‚passen‘ (voreingestellt sind derzeit 5), muß ‚geblättert‘ werden, was im System dann z.B. so aussieht:

Treffer 6 - 10 von insgesamt **13** (Seite 2 von 3)

< zurückblättern [1 **2** 3] weiterblättern >

Die Unterstreichungen deuten an, daß wie üblich Hyperlinks für die Umschaltung genutzt werden, was aber wiederum eine Verwendung von „HTTP POST“ unmöglich macht. Im vorliegenden Fall ist dies jedoch nicht weiter tragisch, da die Argumentlänge relativ beschränkt ist. Und die Verwendung von Links hat sogar einen Vorteil: man kann direkt via Browser-Adreßzeile nach deutschen Dissertationen des letztes Jahres suchen und auf gut Glück die dritte Ergebnisseite ansteuern:[180]

```
/search?Feld1=Titel&Text1=&Modus1=AND&Feld2=Autor&Text2=&Modus2=AND&Feld3=Schluess
elwoerter&Text3=&Modus3=AND&Feld4=Beschreibung&Text4=&Modus4=OR&Feld5=RVK&Text5
=&Modus5=EXACT&Art=Dissertation&Institut=&Sprache=ger&von=2002&bis=2002&Seite=3
```

[180] Die anderen Parameter wurden zum besseren Verständnis mit angegeben und können auch weggelassen werden.

Soviel zur eigens implementierten Metadaten-Suchmaschine. Nicht selbst programmiert wurde hingegen die Volltext-Suchmaschine. Hier kommt der Einfachheit halber die Open Source Software „Perlfect Search“[181] zum Einsatz, die - wie der Name zurecht vermuten läßt - ebenfalls auf Perl basiert und sich somit optimal in das Gesamtsystem integrieren ließ. Viele Anpassungen waren dabei nicht notwendig: die Searchengine funktioniert normalerweise „out-of-the-box“, lediglich ein zusätzlichen Perl-Modul (DB_File) mußte nachinstalliert werden, welches für die Persistenzhaltung der sonst temporären Indexierungsdatenbank zuständig ist. Der Indexer kann dabei in zwei Modi betrieben werden: zum einen ist eine Indexierung aller Dateien des lokalen Filesystems möglich, zum anderen kann das Programm via HTTP auch ‚remote‘ auf Webseiten zugreifen und deren Inhalte in der internen Datenbank verzeichnen. Für den Webindexer macht es dabei keinen Unterschied, ob er statische Seiten vom Server herunterläd, oder ob diese dynamisch generiert wurden. Und genau deshalb ist der zweite Modus wie geschaffen für den hier implementierten Dokumentenserver: alle HTML-Seiten werden ja erst mit Hilfe des show.pl-Scripts aus mehreren Komponenten zusammengesetzt – im Filesystem gäbe es nichts ‚zu holen‘ ... bis auf die Publikationen! Viel wichtiger als eine Suche in den Frontdoor-Seiten und Erläuterungstexten ist natürlich eine Recherche in den Dokumenten selbst. Da diese aber zumeist im derzeit noch präferierten PDF-Format vorliegen und somit nicht direkt durchsuchbar sind, muß das Freeware-Tool „xpdf“[182] zunächst den gesamten Text extrahieren, der anschließend dann indexiert werden kann. Zum Glück erlaubt Perlfect die Einbindung externer Programme und so ist es ein leichtes, zukünftig (einen entsprechenden Konverter vorausgesetzt) auch andere Dokumentformate via Webfrontend recherchierbar zu machen.

```
%EXT_FILTER = ("pdf" => "c:\\Xitami\\search\\xpdf\\pdftotext FILENAME")
@HTTP_CONTENT_TYPES = ('text/html', 'text/plain', 'application/pdf');
```

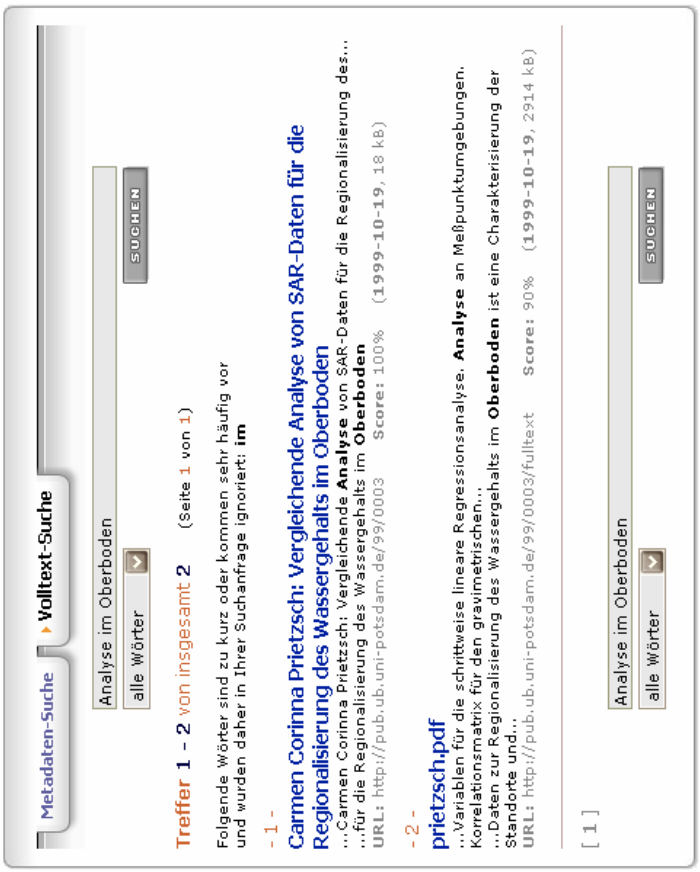
Hat der automatisch jede Nacht gestartete Indexer schließlich die gesamte Webpräsenz und alle (z.B. im Browse-Bereich) verlinkten Publikationen indexiert, kann mittels eines kleinen Eingabefeldes und eines UND/ODER-Auswahlmenüs natürlich auch gesucht werden. Wie der Screenshot auf der nächsten Seite zeigt, läßt sich das Design der Suchseite dabei überaus gut an die eigenen Bedürfnisse anpassen: (fast) nichts läßt darauf schließen, daß ein fremdes Script zum Einsatz kommt. Grund dafür sind Template-Dateien, die beliebigen HTML-Code beinhalten können. In diesen wiederum lassen sich an geeigneten Stellen Kommentare einbauen, die dann von fullsearch.pl (ganz ähnlich wie im eigenen System) ausgewertet und durch die entsprechenden Werte ersetzt werden. Hier ein Beispiel:

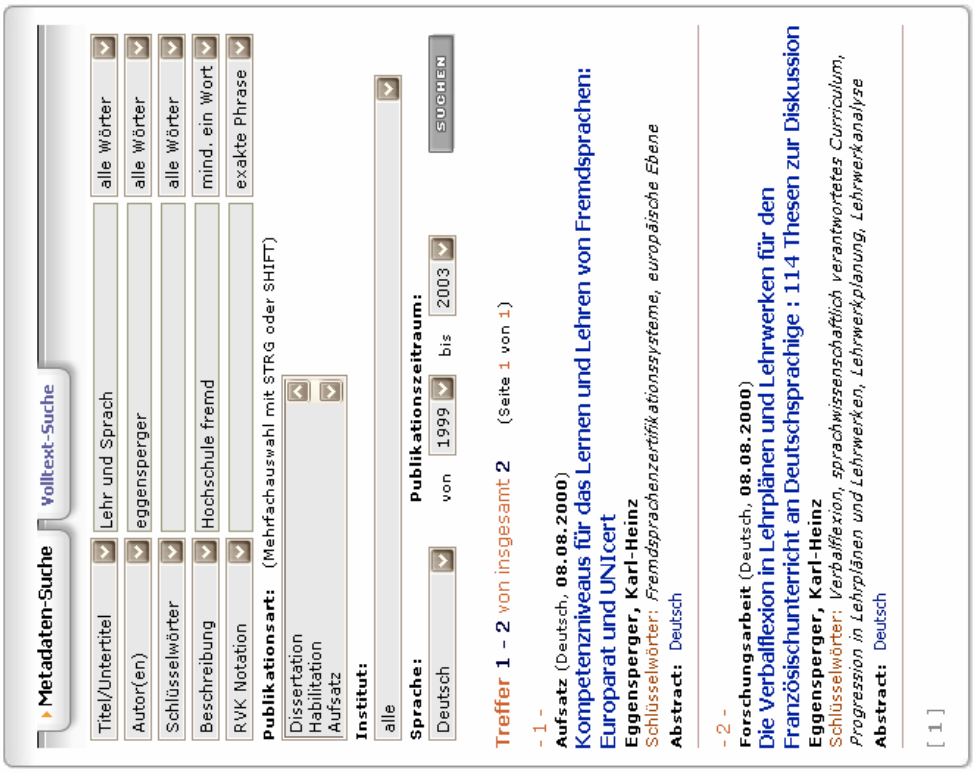
```
<b>Treffer <font color="#000040">
<!--cgi: first_number--> - <!--cgi: last_number-->
</font></b>
[...]
<!--loop: results-->
  [...]
  <!--item: title-->
  [...]
<!--end: results-->
```

Noch sind die Suchmöglichkeiten von Perlfect eher bescheiden: zwar kann die Treffermenge durch das Voranstellen von „+“ und „– “ beeinflußt werden (also z.B. „+ohme –Sebastian“: „liefere alle Dokumente, die Ohme, aber nicht Sebastian enthalten“), Wildcards werden jedoch

[181] http://www.perlfect.com/freescripts/search/
[182] http://www.foolabs.com/xpdf/

nicht unterstützt und auch eine Phrasensuche ist (noch) nicht möglich. Dennoch macht die Perl-Suchmaschine einen vielversprechenden Eindruck ... und Dank der offenen Architektur und der freien Verfügbarkeit kann sie auch selbst um gewünschte Funktionalitäten erweitert werden.






```
SELECT Publikationen.PublikationsID, Publikationen.Art, Autoren.Vorname, Autoren.Nachname,
Publikationen_Sprachen.Sprache, Publikationen.Titel, Publikationen_Untertitel.WeitererTitel,
Publikationen_Schluesselwoerter.Schluesselwoerter,
Publikationen_Schluesselwoerter.SchluesselSprache, Publikationen.SWD,
Publikationen_Abstracts.AbsSprache, Publikationen_Betreuer.Betreuer, Publikationen.Gutachter1,
Publikationen.Gutachter2, Publikationen.RVK, Publikationen.PublikationsDatum FROM
((((Publikationen INNER JOIN Autoren ON Publikationen.PublikationsID = Autoren.PublikationsID)
LEFT JOIN Publikationen_Abstracts ON Publikationen.PublikationsID =
Publikationen_Abstracts.PublikationsID) LEFT JOIN Publikationen_Schluesselwoerter ON
Publikationen.PublikationsID = Publikationen_Schluesselwoerter.PublikationsID) INNER JOIN
Publikationen_Sprachen ON Publikationen.PublikationsID = Publikationen_Sprachen.PublikationsID)
LEFT JOIN Publikationen_Betreuer ON Publikationen.PublikationsID =
Publikationen_Betreuer.PublikationsID) LEFT JOIN Publikationen_Untertitel ON
Publikationen.PublikationsID = Publikationen_Untertitel.PublikationsID WHERE
(Publikationen.PublikationsDatum Between #1/1/1999# AND #12/31/2003#) AND
(Publikationen_Sprachen.Sprache = 'ger') AND ((Publikationen.Titel Like '%Lehr%') OR
(Publikationen_Untertitel.WeitererTitel Like '%Lehr%')) AND ((Publikationen.Titel Like '%und%')
OR (Publikationen_Untertitel.WeitererTitel Like '%und%')) AND ((Publikationen.Titel Like
'%Sprach%') OR (Publikationen_Untertitel.WeitererTitel Like '%Sprach%')) AND ((Autoren.Vorname
Like '%eggensperger%') OR (Autoren.Nachname Like '%eggensperger%')) AND
((Publikationen_Abstracts.Abstract Like '%Hochschule%') OR (Publikationen_Abstracts.Abstract
Like '%fremd%')) AND (Publikationen_Abstracts.AbsSprache = 'ger') ORDER BY
Publikationen.PublikationsID,Publikationen_Sprachen.ID,Publikationen_Untertitel.ID,
Autoren.AutorenID,Publikationen_Abstracts.ID,Publikationen.PublikationsDatum DESC;
```

Abbildung 42: Metadaten und Volltextsuchmaschine

6.3.6 OAI-Schnittstelle

In diesem letzten Abschnitt soll nun noch kurz auf die OAI-Schnittstelle eingegangen werden, die ja ein wichtiger Grund für die Erweiterung des alten Publikationsservers um eine Datenbankanbindung war (siehe Abschnitt 6.1). Das oai.pl-Script ist – in Analogie zum Protokoll selbst – äußerst einfach gehalten: mit Hilfe eines split-Befehls werden die einzelnen via URL übergebenen Parameter zunächst in einem Array zwischengespeichert:

```
@parameter = split(/\&/,$ENV{'QUERY_STRING'});
```

Eine Unterroutine „Header" generiert dann den typischen XML-‚Kopf', der in allen Antworten gleich aussieht und neben der Definition des Stammelements OAI-PMH auch das responseDate und den request-String enthält (siehe Kapitel 3.4.2 und Kästchen).
Nun wird das @parameter-Array nach gültigen Schlüsselwörtern (verb, metadataPrefix, resumptionToken, ...) durchsucht – wird es fündig, werden die jeweiligen values (also z.B. verb=Identify, oder from=1975-11-15) in entsprechenden Variablen vermerkt und anschließend in die entsprechende Subroutine verzweigt. Im Falle eines fehlerhaften Query-Strings (falsches „verb", nicht-unterstütztes Metadatenformat[183], anderes Datumsformat, Syntax-Fehler im resumptionToken usw.) wird in den XML-Code eine entsprechende Meldung eingebettet und ein abschließendes </OAI-PMH> ausgegeben, was bei der Eingabe von http://pub.ub.uni-potsdam.de/oai?verb=ListRecords dann z.B. so aussieht:[184]

```
<?xml version="1.0" encoding="iso-8859-1"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
    http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
  <responseDate>2003-10-01T01:23:45Z</responseDate>
  <request verb="ListRecords">http://pub.ub.uni-potsdam.de/oai</request>
  <error code="badArgument">Missing or wrong defined argument:
  'metadataPrefix'</error>
</OAI-PMH>
```

Wurden alle obligatorischen Parameter (eventuell zusammen mit optionalen Angaben, wie „from=..." oder „until=...") korrekt übergeben (Groß-/Kleinschreibung spielt nur für die values eine Rolle), wird wie erwähnt eine zugehörige Unterroutine angesprungen, die die weitere Generierung der jeweiligen Rückgabemenge übernimmt. Hier soll beispielhaft auf das wohl meistgenutzte ListRecords-„verb" eingegangen werden; alle anderen Anfragen werden ähnlich ausgewertet:
Fragt ein Service-Provider also nach allen Einträgen im Dublin Core-Format

(http://pub.ub.uni-potsdam.de/oai?verb=ListRecords&metadataPrefix=oai_dc),

werden mit Hilfe einer SQL-Abfrage alle relevanten Datenbankeinträge extrahiert und nacheinander in die bereits im Kapitel 3.4.2 beschriebene XML-Struktur ‚gepreßt'. In einem header-Bereich werden die PublikationsID samt vorangestelltem oai:pub.ub.uni-potsdam.de: sowie set-Zugehörigkeit und Publikationsdatum angegeben. Der metadata-Bereich enthält wiederum die Angaben zu dem oder den Autoren(en) und natürlich zur Publikation selbst (Schlüsselwörter, Abstracts, usw.). In einer Schleife werden so alle zur Anfrage passenden

[183] derzeit ist nur oai_dc zulässig

[184] hier fehlt die Angabe von metadataPrefix=oai_dc

Metadaten ausgegeben, bis die maximal mögliche Anzahl an Datensätzen pro Durchlauf erreicht wurde (aus Test-Zwecken sind dies derzeit 10). In diesem Fall wird ein eindeutiger resumptionToken generiert, der für die nächste Abfrage genutzt werden kann und von oai.pl natürlich auch entsprechend ausgewertet wird. Abschließend wird auch hier ein </OAI-PMH> ausgegeben, was die Ergebnismenge komplettiert.

Alle Funktionalitäten wurden nur ein einziges Mal implementiert. ListRecords ruft so beispielsweise eine Subroutine &Record auf, die jeweils nur einen Datensatz ausgibt und natürlich auch für das „verb“ GetRecord verwendet werden kann. Der nachfolgende Internet Explorer-Screenshot zeigt einen solchen Datensatz (allerdings passend zum obigen Beispiel via ListRecords und Schleifenabbruch erzeugt) und soll gleichzeitig als Abschluß für die Beschreibung der praktischen Arbeit dienen.

Abbildung 43: OAI-Datensatz

7 Zusammenfassung

„Elektronische Medien sind nicht archivierbar“, so die einleitenden Worte der vorliegenden Diplomarbeit, die gleichzeitig den Abschluß bilden sollen. Als Fazit sind sie jedoch nicht zu verstehen, denn es hat sich gezeigt, daß eine längerfristige Aufbewahrung auch digitaler Daten durchaus möglich ist – umfangreiche und vor allem *rechtzeitige* Archivierungsbestrebungen vorausgesetzt. Ganz unrecht hat C. Stoll mit seiner Aussage aber dennoch nicht: viele Aspekte der Langzeiterhaltung sind nach wie vor ungeklärt. Dies betrifft sowohl die Sicherung der Datenspeicherung (*Trägermedium*) als auch den zukünftigen Zugriff auf die enthaltenen Informationen (*Datenformate*) und deren dauerhaften Nutzbarkeit (*Erschließung*). Digitale Dokumente haben Einzug gehalten in alle Bereiche des öffentlichen Lebens und schon bald liegt das Wissen unserer Zivilisation nur noch in elektronischer Form vor – Buchstaben, Texte, Bilder reduziert auf Nullen und Einsen. Doch wie lange lassen sich Disketten und Festplatten aufbewahren, ohne vorher ihr magnetisches Gedächtnis verloren zu haben? Wie behutsam muß mit den Daten umgegangen werden? Sind all unsere Medien für die Ewigkeit gemacht? Oder müssen wir uns tatsächlich mit dem Gedanken abfinden, daß auch die digitale Welt vergänglich ist?
Ob das Informationszeitalter sein Gedächtnis verlieren wird, hängt einzig und alleine davon ab, wie wir die Relikte unserer Vergangenheit pflegen. Zuständig hierfür sind Archive und Bibliotheken, die dafür sorgen müssen, daß auch der digitale Anteil der kulturellen Überlieferung für die Nachwelt erhalten bleibt. Keine einfache Aufgabe, stehen sie nun doch vor zusätzlichen, grundsätzlich neuen Herausforderungen, deren Umfang und Bedeutung zudem noch in das Bewußtsein der allgemeinen Öffentlichkeit und der Produzenten digitaler Informationen gebracht werden muß. Die Problematik der Langzeitarchivierung ist äußerst komplex und die Angabe detaillierter, endgültiger Lösungsstrategien – insbesondere im Rahmen dieser Diplomarbeit – unmöglich. Ziel der Ausarbeitung war es daher vielmehr, einen kleinen Einblick in entsprechende Bemühungen und Projekte zu geben und aufzuzeigen, wie wichtig es ist, sich schon jetzt mit den Problemen der Zukunft auseinanderzusetzen. Noch können Dokumente (relativ) problemlos von einem Datenträger auf einen anderen kopiert und alte Dateiformate mit aktuellen Programmversionen geöffnet werden. Mit hoher Wahrscheinlichkeit ist dies aber irgendwann nicht mehr möglich ... spätestens dann zahlt sich die Wahl eines „archivierungsfreundlichen“ Dokumentformats aus, welches Migrationen erleichtert und die konvertierungsbedingte Gefahr einer Informations- und Authentizitätsverfälschung minimiert. Die „Suche“ nach einem solchen Format hat auch das relativ umfangreiche Kapitel 3 notwendig gemacht, in dem – von den allgemeinen Anforderungen ausgehend – schließlich SGML bzw. XML als am besten geeignet herausgestellt wurden. Außerdem wurde hier auch das Dublin Core Metadata Element Set genauer untersucht, welches nicht nur für die inhaltliche Erfassung der Dokumente und für ein verbessertes Retrieval von Bedeutung ist, sondern u.a. auch innerhalb des OAI-Protokolls und der Transferschnittstelle Der Deutschen Bibliothek zum Einsatz kommt. Im 4. Kapitel standen einige ausgewählte Aspekte der Langzeitarchivierung im Vordergrund. Neben Migration und Emulation gehören hierzu natürlich insbesondere Verfahren, die eine Unverfälschtheit der Daten sicherstellen („Authentizität und Integrität) und gleichzeitig einen langfristigen Zugriff ermöglichen (Stichwort „beständige Identifikatoren“). Noch bevor im 5. Kapitel dann auf konkrete, dissertationsspezifische Archivierungsbestrebungen eingegangen wurde, galt es, auf eher theoretischer Ebene die notwendigen Komponenten eines Depotsystems zu identifizieren. Hierfür hat sich das OAIS-Referenzmodell als geeignet erwiesen, welches Grundlage vieler real existierender Repositories ist (NEDLIB, DDB, DSpace, ...) und daher etwas genauer vorgestellt wurde.

Viele, wenn auch nicht alle, der hier nur in äußerster Kürze zusammengefaßten Aspekte wurden schließlich in einen eigenen Dokumentenserver überführt. Ziel war es, selbst einen kleinen Beitrag zum Erhalt wichtiger wissenschaftlicher Erkenntnisse zu leisten und ein System zu schaffen, welches Dank einer offenen Architektur und einer guten Erweiterbarkeit auch zukünftigen Anforderungen gewachsen ist. Alle grundlegenden Funktionalitäten sind vorhanden: Dokumente können komfortabel mit Metadaten versehen und auf den Server transferiert werden, eine Management-Komponente erlaubt die bequeme Kontrolle der Anmeldungen und Suchmaschinen ermöglichen den schnellen Zugang zu den archivierten Publikationen. Für eine Sicherung von Server und Dateien wurde ebenfalls gesorgt: nur wenige Nutzer haben Zugriff auf das System, Hashwert-Berechnungen gestatten die Überprüfung der Integrität und umfangreiche Backup-Maßnahmen wirken einem eventuellen Datenverlust entgegen. An dieser Stelle soll nicht nochmals auf die vielen Features des neuen Dokumentenservers eingegangen werden – hierfür sei auf die recht umfassenden Ausführungen der einzelnen Kapitel verwiesen. Festzuhalten bleibt, daß der hier implementierte Prototyp eine deutliche Verbesserung im Vergleich zur bisherigen Praxis darstellt: war früher noch viel Handarbeit bei der Überprüfung und Freischaltung eingegangener Publikationen erforderlich (siehe Workflow-Beschreibung in Kapitel 6.1), läuft nun vieles automatisch ab ... manuelle, zeitlich und personell aufwendige Eingriffe (Fingerprint-Bestimmung, URN-Vergabe, DDB-Meldung, Mail-Versand, Webseiten-Anpassungen, usw.) können somit entfallen – vorausgesetzt natürlich, das entwickelte System wird auch tatsächlich produktiv eingesetzt. Wünschenswert wäre es – viel Arbeit wurde investiert, um einen für Bibliotheksmitarbeiter und Nutzer gleichermaßen ansprechenden Dokumentenserver zur Verfügung zu stellen, der dank des konsequenten Einsatzes anerkannter Standards (DC, OAI, URN, ...) auch national und international ‚wettbewerbsfähig' ist.

Nachdem die programmtechnischen Grundlagen geschaffen wurden, sollte die UB Potsdam nun noch die Verwendung zukunftsweisender Dokumentformate forcieren und vielleicht sogar – in Anlehnung an das DiDi-Projekt der HU Berlin – die Verwendung geeigneter Formatvorlagen explizit vorschreiben. Nur so macht ein Publikationsserver, der auf eine Langzeiterhaltung ausgelegt ist, auch wirklich Sinn. Niemand weiß, wie lange das bisher für die Archivierung noch zulässige Word-Format gelesen und konvertiert werden kann. SGML und XML sind diesbezüglich, aber auch für Recherchezwecke, weitaus besser geeignet und so wäre es an der Zeit, auch an der Universität Potsdam entsprechende Regelungen zu erlassen, die den Einsatz von „dissertation-97.dot", DiML und Co. (trotz der damit verbundenen „Nachteile", siehe Kapitel 5.1) möglich machen. Nur wenige Anpassungen wären am eigenen System notwendig: auch mit Formatvorlagen oder Styledateien ausgezeichnete Dokumente müssen irgendwie mit Metadaten versehen und auf den Server transferiert werden. Dort angekommen sind sie zu kontrollieren und durch Eintragung in die Datenbank recherchierbar zu machen. Die Hashwert-Berechnung bleibt die Gleiche, ebenso die Generierung der DDB-konformen Metadatensätze und URNs. Einziger Unterschied: eine zusätzlich zwischengeschaltete Konvertierungskomponente müßte die entgegengenommenen Dokumente parsen, DiML-gerecht aufbereiten und aus dem erzeugten SGML/XML-Archivierungsformat zusätzliche Präsentationsformate generieren, die dann wie gewohnt auf den Frontdoor-Seiten verlinkt werden. Noch ist dies Zukunftsmusik – derartige Erweiterungen des implementierten Systems sind dank der guten Modularisierung aber jederzeit problemlos möglich.[185]

[185] Bereits angedacht ist z.B. eine Anbindung an das lokale Bibliothekssystem,
um Datensätze zwischen den Systemen austauschen zu können ... aber dazu an anderer Stelle mehr :)

Abbildungsverzeichnis

Literaturverzeichnis

[AGVT98] Arbeitsgruppe Volltexte und Hochschulpublikationen des Ministeriums für Wissenschaft, Forschung und Kunst Baden-Württemberg: Empfehlungen zum Aufbau eines Servernetzwerkes für elektronische Hochschulpublikationen. Konstanz 1998 http://elib.uni-stuttgart.de/opus/doku/AGVT_Empf.pdf

[Arch99] Archimedes Online-Artikel: Wir verlieren unser Gedächtnis. (04.05.1999) http://www.arte-tv.com/hebdo/archimed/19990504/dtext/sujet1.html

[Arm00] Arms, William Y. (Herausgeber): Digital Libraries. MIT Press, 2000

[Asch01] Aschenbrenner, Andreas: Long-Term Preservation of Digital Material - Building an Archive to Preserve Digital Cultural Heritage from the Internet. Diplomarbeit, TU Wien, Dezember 2001 http://www.ifs.tuwien.ac.at/~aola/publications/thesis-ando.pdf

[Aud02] Audersch, Stefan: Metadatenverwaltung für Multimedia-Content-Management mit OLAP-Funktionalität. Diplomarbeit, Rostock, Januar 2002 http://e-lib.informatik.uni-rostock.de/fulltext/2002/diploma/AuderschStefan-2002.pdf

[Ball00] Ball, Rafael: Wissenschaft und Bibliotheken : Das aktive Engagement im Kontext elektronisches Publizierens. In: Wissenschaft Online Nr. 80, Herausgegeben von Beate Tröger, 2000, ISBN 3-465-03081-8

[Bern97] Berners-Lee, Tim: Metadata Architecture. W3C, 1997 http://www.w3.org/DesignIssues/Metadata.html

[Berg00] Berg, Heinz-Peter: Nutzungsuntersuchungen für elektronische Publikationen. In: Wissenschaft Online Nr. 80 (2000)

[Bilo00] Bilo, Albert: „Anpassungen oder Strukturwandel“ - Elektronische Publikationen und digitale Bibliotheken aus der Sicht bibliothekarischer Praxis. In: Wissenschaft Online Nr. 80 (2000)

[Blank98] Blankenburg, Ralf: Implementierung von Dokumentenarchivierungssystemen - eine betriebswirtschaftliche Untersuchung. Lohmar, Köln 1998, ISBN 3-89012-610-3

[Bleu00] Bleuel, Jens: Online publizieren im Internet. Elektronische Zeitschriften und Bücher. Ursprünglich: Pfungstadt und Bensheim, Edition Ergon, 2000 http://www.bleuel.com/ip-wel.pdf

[Bódi00] Bódi, Dominik: Emulation als Methode zur Langzeitarchivierung digitaler Dokumente. Vortragsausarbeitung, Universität der Bundeswehr München, Mai 2000 http://ist.unibw-muenchen.de/Lectures/FT2000/Digitale-Bibliotheken/handout5.pdf

[Born01] Born, Günter: Dateiformate - Eine Referenz. Datenbanken, Tabellenkalkulationen, Textdarstellung, Grafik, Multimedia, Sound, Internet. (Galileo Computing) Galileo Press, 2001, ISBN 3-934358-83-7

[Chip03] Artikel in Computer-Zeitschrift CHIP: Kein Einheitssprache in Sicht. Ausgabe 6/2003, S. 98

[DC97] The Dublin Core: A Simple Content Description Model for Electronic Resources. (1997) http://purl.oclc.org/docs/metadata/dublin_core/main.html

[Degen97] Degenhardt, Eileen: „Elektronische Dissertationen“ in Bibliotheken. Diplomarbeit, Fachhochschule Hannover, November 1997 http://www.ik.fh-hannover.de/ik/personen/bock/degenhardt/ediss.pdf

[DFG97] Diepold, Peter u.a.: Projekt „Dissertationen Online“. Erstantrag an die DFG 1997
Originaldokument (http://www.educat.hu-berlin.de/diss_online/antrag.pdf)
nicht mehr verfügbar, daher zitiert nach URL
http://deposit.ddb.de/netzpub/web_online-hochschulschriften.htm

[DFG02] Deutsche Forschungsgemeinschaft: Leistungszentren für Forschungsinformation:
eine Förderinitiative der Deutschen Forschungsgemeinschaft (DFG) zur Stärkung der
Informations-Infrastrukturen an deutschen Hochschulen und Forschungseinrichtungen.
Erster Aufruf zur Einreichung von Projektanträgen.
In: Bibliotheksdienst 36 (2002), Nr. 8/9, S. 1096–1104

[Diep01] Diepold, Peter: Das interdisziplinäre DFG-Projekt „Dissertationen Online“ -
Ergebnisse und Ausblick. Januar 2001
http://www.dissonline.de/tagungen/abschlusstagung_2000_12_13/ergebnis.ps

[DINI01] Deutsche Initiative für Netzwerkinformation: Elektronisches Publizieren an Hochschulen -
Empfehlungen. Dezember 2001
http://www.dini.de/veranstaltung/jahres/2001/DINI-EPub-2001-11-28.pdf

[DK98] Dobratz, Susanne; Kamke, Hans-Ulrich: Geschäftsgang unter besonderer Berücksichtigung
der Autorenbetreuung oder „Wo fängt elektronisches Publizieren an?“. November 1998
http://eldorado.uni-dortmund.de:8080/bib/98/workshop/dobratz/pdf.pdf

[DM98] Dobratz, Susanne; Martin, Norbert: „Schnell - kostengünstig - up-to-date“ -
DiDi: Digitale Dissertationen an der Humboldt-Universität. September 1998
http://dochost.rz.hu-berlin.de/epdiss/humboldt.html

[Dob98] Dobratz, Susanne: Die „Humboldt-Dissertationen“ im Internet - Erste Ergebnisse
und Erfahrungen aus dem Projekt DiDi. In: HU-RZ-Mitteilungen Nr. 16, Juni 1998
http://edoc.hu-berlin.de/e_rzm/16/dobratz-susanne-1998-06-01/PDF/4.pdf

[Dob99] Dobratz, Susanne: Strukturierte digitale Dissertationen als Beispiel für qualitatives
Informationsmanagement und Information-Retrieval in wissenschaftlichen Bibliotheken.
Vortrag auf der DGI-Online Tagung (19. Mai 1999) in Frankfurt
http://dochost.rz.hu-berlin.de/epdiss/dgi.pdf

[Dob03] Dobratz, Susanne: Elektronisches Publizieren - Etablierung eines neuen Service
für die Universität durch UB und CMS. In: cms-journal Nr. 24, April 2003
http://www.hu-berlin.de/rz/rzmit/cmsj24/38-42.pdf

[DS99] Dobratz, Susanne; Schulz, Matthias: Verbessertes Wissensmanagement durch Nutzung
SGML/XML-basierter Technologien am Beispiel der elektronischen Publikationen
an Hochschulen, September 1999
http://dochost.rz.hu-berlin.de/projekte/epdiss/publications/knowtech99/Knowtech99.pdf

[DT02] Dobratz, Susanne; Tappenbeck, Inka: Thesen zur Zukunft der digitalen Langzeitarchivierung
in Deutschland. Zeitschrift Bibliothek 26 (2002), Nr. 3

[EU99] Ewert, Gisela; Umstätter, Walther: Die Definition der Bibliothek.
In: Bibliotheksdienst Heft 6 (1999)
http://deposit.ddb.de/ep/netpub/89/96/96/967969689/_data_stat/www.dbi-berlin.de/dbi_pub/bd_art/bd_99/99_06_03.htm

[Fritz99] Fritz, Tobias: Bereitstellung und Erschließung von elektronischen Dissertationen -
theoretische Ansätze und praktische Umsetzungen unter besonderer Berücksichtigung
des Projekts „Online-Dissertationen“ am Fachbereich Veterinärmedizin der FU Berlin.
Magisterarbeit, August 1999
http://dochost.rz.hu-berlin.de/diplom/phil/fritz-tobias/PDF/Fritz.pdf

[Germ00] Germain, Carol Anne: URLs: Uniform resource locators or unreliable resource locators. College & Research Libraries, 61(4), Juli 2000

[GR00] Goossens, Michel; Rahtz, Sebastian u.a.: Mit LaTeX ins Web - Elektronisches Publizieren mit TeX, HTML und XML. Addison-Wesley, 2000, ISBN 3-8273-1629-4

[Grun01] Gruner, Simone; Möller, Katrin: Archivierungsverfahren digitaler Publikationen. Vortragsausarbeitung, Fachhochschule Potsdam, 2001
http://hera.rz.hu-berlin.de/lehre/fhp/ss2001/referate/gruner.pdf

[Hilb95] Hilberer, Thomas in http://www.uni-duesseldorf.de/WWW/ulb/virtdef.html

[Hilb01] Hilberer, Thomas: Gründung eines elektronischen Hochschulverlages auf Verbund-Ebene: Vorüberlegungen und Thesen. In: Bibliotheksdienst 35 (2001), Nr. 12, S. 1629–1632

[HS97] Henze, Volker; Schefczik, Michael: Metadaten - Beziehungen zwischen Dublin Core Set, Warwick Framework und Datenformaten. Bibliotheksdienst 31 (1997), Nr. 3, S. 413-419
http://deposit.ddb.de/ep/netpub/89/96/96/967969689/_data_stat/www.dbi-berlin.de/dbi_pub/bd_art/97_03_05.htm

[HZ00] Hilf, Eberhard R., Zimmermann, Kerstin: Dissertationen via Internet - Voraussetzungen, Verfahren, Verträge. In: Wissenschaft Online Nr. 80 (2000)

[IuK98] Grötschel, Martin u.a.: Entwicklung von Konzepten zur Neugestaltung der elektronischen Information und Kommunikation in Wissenschaft und Technik - Arbeitskreis „Dissertationen Online“. Projekt-Schlußbericht, 1998
http://www.iuk-initiative.org/doc/IuK-Schlussbericht.htm

[Keitz97] von Keitz, Wolfgang: Vom Buch zum Bit?? - Analoges zur Digitalen Bibliothek; Begriffe, Konzepte, Projekte. Stuttgart, November 1997
http://v.hdm-stuttgart.de/~keitz/digilib.html

[Kell01] Keller, Alice: Zeitschriften in der Krise: Entwicklung und Zukunft elektronischer Zeitschriften. Berlin, Humboldt-Universität, Dissertation, Januar 2001
http://e-collection.ethbib.ethz.ch/show?type=extdiss&nr=1

[Kell02] Keller, Alice: Elektronische Zeitschriften: was sagen Nutzungsstatistiken aus? In: B.I.T. online 5 (2002), Nr. 3, S. 213–232

[Kirch81] Kirchner, Hildebert: Bibliotheks- und Dokumentationsrecht. Frankfurt 1981, S. 179-180

[Kist88] Kist, J.: Elektronisches Publizieren - Übersicht, Grundlagen, Konzepte. Stuttgart 1988

[Korb03] Korb, Nicola; Die Deutsche Bibliothek: Neues Datenformat XMetaDiss. Ankündigung eines neuen Metadatenformats für Hochschulschriften über Mailing-Liste diss-online@ddb.de

[KS00] Klotz-Berendes, Bruno; Schönfelder, Gabriele: Sicherungsverfahren für den Betrieb eines Dokumentenservers - Anforderungen, kryptographische Grundlagen, Zertifizierung und digitale Signatur. In: Wissenschaft Online Nr. 80 (2000)

[Kuhl95] Kuhlen, Rainer: Informationsmarkt: Chancen und Risiken der Kommerzialisierung von Wissen. – Konstanz: UVK, 1995

[Kühn99] Kühn, Armin: Bibliotheken und elektronische Publikationen - Analyse, Konzeption am Beispiel des Südwestdeutschen Bibliotheksverbunds. Konstanz 1999
http://www.ub.uni-konstanz.de/kops/volltexte/1999/39/pdf/39_1.pdf

[LB98] Lyman, Peter; Besser, Howard: Time and Bits - Managing Digital Continuity. Conference background paper. The Long Now Foundation, Februar 1998
http://www.longnow.com/10klibrary/TimeBitsDisc/tbpaper.html

[Leh96] Lehmann, Klaus-Dieter: Das kurze Gedächtnis digitaler Publikationen.
In: Zeitschrift für Bibliothekswesen und Bibliographie 43 (1996) Nr. 3, S. 219-224

[Leh97A] Lehmann, Klaus-Dieter: Dissertationen Online. In: Bibliotheksdienst Heft 4, 1997
http://deposit.ddb.de/ep/netpub/89/96/96/967969689/_data_stat/www.dbi-berlin.de/dbi_pub/bd_art/97_04_10.htm

[Leh97B] Lehmann, Klaus-Dieter: Die Mühen der Ebenen: Regelwerke - Datenformate - Kommunikationsschnittstellen; erschienen in: Zeitschrift für Bibliothekswesen und Bibliographie 44 (1997), Nr. 3, S. 229-240

[Lieg01] Liegmann, Hans: Langzeitverfügbarkeit digitaler Publikationen - Zusammenfassung, Erkenntnisse und Pläne. Frankfurt am Main, 2001
http://www.uni-muenster.de/Forum-Bestandserhaltung/konversion/digi-liegmann.shtml

[Lind00] Lindner, Bernd: Vergleichende Untersuchungen zur XML-Repräsentation von Verkehrs-telematik-Daten in Client-Server-Anwendungen und deren multimedialer Aufbereitung. Diplomarbeit, Ulm, 2000

[Lupp00] Lupprian, Karl-Ernst: Open Archival Information System. Beitrag zu den EDV-Tagen 2000
http://www.museumtheuern.de/edvtage/g/g11_inh.htm

[Lupp01] Lupprian, Karl-Ernst: Ein Archiv für 1000 Jahre? - Wege zu einer dauerhaften Archivierung digitaler Unterlagen. Beitrag zu den EDV-Tagen 2001
http://www.museumtheuern.de/edvtage/h/h07_inh.shtml

[Lux95] Lux, Claudia: Vom Bibliothekar zum Cybrarian - die Zukunft des Berufs in der virtuellen Bibliothek (1995)
http://www.ifla.org/IV/ifla61/61-luxc.htm

[MK03] Müller, Uwe; Klotz-Berendes, Bruno: Die Open Archive Initiative.
Vortrag zum DINI-Workshop OAI am 08.05.2003 in Köln
http://www.dini.de/veranstaltung/tutorial/koeln_tutorial_gesamt.pdf

[Mönn00] Mönnich, Michael: Elektronisches Publizieren von Hochschulschriften - Formate und Datenbanken. In: Wissenschaft Online Nr. 80 (2000)

[Müll98] Müller, Harald: Elektronisches Pflichtexemplarrecht oder das Recht des Bürgers auf ungehinderten Zugriff zu elektronisch gespeicherten Informationen. Heidelberg, 1998
http://www.eblida.org/eblida/meetings/events/copenhagen/mullerpa.htm

[Müll00] Müller, Harald: Die rechtlichen Zusammenhänge im Rahmen des elektronischen Publizierens. In: Wissenschaft Online Nr. 80 (2000)

[OAI01] Lagoze, Carl; Van de Sompel, Herbert: The Open Archives Initiative - Building a low-barrier interoperability framework. August 2001
http://www.openarchives.org/documents/oai.pdf

[OAI02] Lagoze, Carl; Van de Sompel, Herbert u.a.: The Open Archives Initiative Protocol for Metadata Harvesting. Version 2.0 vom 14.06.2002
http://www.openarchives.org/OAI/openarchivesprotocol.html

[Odly95] Odlyzko, Andrew: Tragic loss or good riddance? The impending demise of traditional scholarly journals. In: Intern. J. Human-Computer Studies 42 (1995), S. 71–122
http://www.dtc.umn.edu/~odlyzko/doc/tragic.loss.txt

[Ohst98] Ohst, Daniel: Dateiformate für das elektronische Publizieren.
Studienarbeit, HU Berlin, März 1998
http://edoc.hu-berlin.de/buecher/ohst-daniel/HTML

[ON01] Ohme, Sebastian; Nelaimischkies, Andreas: eLink - Linkkonsistenz für Literaturreferenzen. Großer Beleg. Universität Potsdam, Institut für Informatik, Juli 2001

[Pay99] Payer, Margarete: Computervermittelte Kommunikation. Kapitel 13, 2,2,1: OSI-7 - Application Layer. Teil 2, 2: Die Anwendungsschicht im Internet: WWW - World Wide Web. 1. HTTP und URI. Fassung vom 8.7.1999 http://www.payer.de/cmc/cmcs13221.htm

[PersID] Die Deutsche Bibliothek u.a.: „Persistent Identifier … eindeutige Bezeichner für digitale Inhalte“. Web-Präsentation der Projektergebnisse von CARMEN-AP4 und EPICUR http://persistent-identifier.de

[Pier02] Pieruschka, Thomas; Korb, Nikola: „XML - Die Lösung?“ - Konvertierungstools und DissOnline Aktivitäten. Vortragsfolien für INETBIB-Tagung 2002, Göttingen http://eldorado.uni-dortmund.de:8080/bib/2002/inetbib2002/volltexte/korb/korb.pdf

[RBW86] Riehm, Ulrich; Böhle, Knut; Wingert-Afass, Bernd u.a.: Begleit- und Wirkungsuntersuchungen zum Elektronischen Publizieren. Ergebnisse aus Phase I. Karlsruhe: Kernforschungszentrum 1986

[RBW88] Riehm, Ulrich; Böhle, Knut; Wingert-Afass, Bernd u.a.: Aspekte der Autoren-Verlagsbeziehungen beim Elektronischen Publizieren. Ergebnisse aus Expertengesprächen. Mit zwei Bereichsstudien zur Norm- und Rechtsinformation, KfK 4436. Karlsruhe 1988

[Reil02] Reil, Dennis: Konzeption Digitaler Bibliotheksdienste auf Basis von Webservices. Diplomarbeit, Oldenburg, Juni 2002 http://www-is.informatik.uni-oldenburg.de/~reil/privat/data/DigBib-WebServices.pdf

[Rein99] Reinhardt, Werner: Zeitschriftenpreise 1999: ein Zwischenbericht über die Reaktionen auf den Offenen Brief der Kommission. In: Bibliotheksdienst 33 (1999), Nr. 5, S. 804-809

[Riehm92] Riehm, Ulrich u.a.: Elektronisches Publizieren – Eine kritische Bestandsaufnahme. Springer-Verlag, 1992, ISBN 3-540-54159-4

[Roth95A] Jeff Rothenberg: Die Konservierung digitaler Dokumente. Spektrum der Wissenschaft, 1995

[Roth95B] Jeff Rothenberg: Ensuring the Longevity of Digital Documents. Scientific American, Vol. 272, No. 1, Januar 1995, S. 42-47

[Roth99] Jeff Rothenberg: Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation. Council on Library and Information Resources, 1999 http://www.clir.org/pubs/reports/rothenberg/contents.html

[Rusch97] Rusch-Feja, Diann: Dublin Core Metadata - Auf dem Weg zur Entwicklung eines Internet-Standards - mit einem Bericht über den 4. Dublin Core Metadata Workshop in Canberra. In: Bibliotheksdienst, 32 (1997) Nr. 4, S. 622-639 http://deposit.ddb.de/ep/netpub/89/96/96/967969689/_data_stat/www.dbi-berlin.de/dbi_pub/bd_art/97_04_08.htm

[Rusch99] Rusch-Feja, Diann: Digital Libraries - Informationsform der Zukunft für die Informationsversorgung und Informationsbereitstellung, In: B.I.T. online, 1999, S. 143-156

[SB96] Schmundt, Hilmar; Bock, Patrick: No Future. In: Die Woche, 1996

[Schirm98] Schirmbacher, Peter: Dateiformate : ein Kernstück des elektronischen Publizierens. Vortrag, gehalten auf dem Kolloquium „Neue Organisationsformen elektronischer Veröffentlichungen“, Dortmund, 23.11.1998 http://eldorado.uni-dortmund.de:8080/bib/98/workshop/schirmbacher

[Schmitt96] Schmitt, P. H.; Jakob, Adelheid: Formate für elektronische Dissertationen. (1996)
http://i12www.ira.uka.de/dissonline/tp3/tp3.html

[Schmitt03] Schmitt, Lutz: Archivierung digitaler Daten - Die Probleme im Umgang mit digitalen Daten. Persönlicher Mailkontakt mit dem Autor dieser Vordiploms-Arbeit im April 2003

[Scholze02] Scholze, Frank: Authentizität und Langzeitarchivierung. In: Workshop-Arbeitspapier „Langzeitverfügbarkeit digitaler Dokumente - Erarbeitung eines ersten kooperativen Konzepts für Deutschland“. Oktober 2002
http://www.dl-forum.de/Foren/Langzeitverfuegbarkeit/Arbeitspapier2.pdf

[Schulz99] Schulz, Matthias: Dissertation Markup Language (DiML) -
Archivierungs- und Rechercheformat für Dissertationen nach dem SGML-Standard. 1999
http://edoc.hu-berlin.de/e_autoren/software/dimldoc.pdf

[Schwen00] Schwens, Ute: Die Rolle der Deutschen Bibliothek. In: Wissenschaft Online Nr. 80 (2000)

[Schwen02] Schwens, Ute u.a.: Die Deutsch Bibliothek -
Sammlung, Verzeichnung und Archivierung von Netzpublikationen. Oktober 2002
http://www.ddb.de/wir/netzpubl.htm

[Somp00] Van De Sompel, Herbert: The Santa Fe Convention of the Open Archives Initiative.
In: D-Lib Magazine 6 (2000), February, Nr. 2. ISSN 1082-9873
http://www.dlib.org/dlib/february00/vandesompel-oai/02vandesompel-oai.html

[Stoja00] Stojanow, Alexander: Untersuchung zu den Potenzen von XML für das Bereitstellen und das Management von Dokumenten in Informationssystemen. Diplomarbeit, TU Ilmenau, 2000
http://www-ia.tu-ilmenau.de/IPI/FGT/diplomarbeiten/stojanow.pdf

[Stoll96] Clifford Stoll: Die Wüste Internet. Geisterfahrten auf der Autobahn.
Frankfurt am Main, 1996, S.263

[Tapp01] Tappenbeck, Inka: Infrastrukturen für die Archivierung digitaler Dokumente -
Ein Tagungsbericht. In Bibliotheksdienst - Heft 2 (2001)
http://bibliotheksdienst.zlb.de/2001/01_02_05.htm

[Warn01] Warner, Simeon: Exposing and Harvesting Metadata Using the OAI
Metadata Harvesting Protocol. In: HEP Libraries Webzine Issue 5, Juni 2001

[Weiß00] Weiß, Berthold: Dublin Core : Metadaten als Verzeichnungsform elektronischer Publikationen. In: Wissenschaft Online Nr. 80, Herausgegeben von Beate Tröger, 2000

[WL02] Wilde, Erik; Lowe, David: XPath, XLink, XPointer, and XML - A Practical Guide to Web Hyperlinking and Transclusion. Addison Wesley, Juni 2002, ISBN 0-201-70344-0

[WR01] Wissenschaftsrat: Empfehlungen zur digitalen Informationsversorgung
durch Hochschulbibliotheken. Greifswald, 2001
http://www.wissenschaftsrat.de/texte/4935-01.pdf

[Zimm02] Zimmel, Daniel: Wissenschaftliche Informationsversorgung im Umbruch -
die neuen Publikationsmodelle und die Rolle der Bibliotheken.
Diplomarbeit, Fachhochschule Stuttgart, Oktober 2002
http://schnorchelfabrik.de/diplomarbeit/dipl_voll.pdf

Erklärung

Hiermit erkläre ich, daß ich die vorliegende Diplomarbeit selbständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich genannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht.

Potsdam, den 01.10.2003 — S. Olle

Ort und Datum — Unterschrift