Projekt Meta-Akad
Abschlussbericht

Inhaltsverzeichnis

I Einleitung....3

I.1. Aufgabenstellung....3

I.2. Voraussetzungen....3

I.3. Planung und Ablauf....3

I.4. Wissenschaftlicher und technischer Stand....6

I.5. Zusammenarbeit mit anderen Stellen....6

II Ergebnisse....6

II.1. Arbeitspaket 1: Sammlung des Lehr- und Lernmaterials....6

II.1.1. Automatisches Sammeln....6

II.1.2. Intellektuelles Sammeln in Kaiserslautern....8

II.1.3. Intellektuelles Sammeln in Regensburg....11

II.2. Arbeitspaket 2: Definition von Metadaten....13

II.3. Arbeitspaket 3: Bewertung des Lehr- und Lernmaterials....13

II.3.1. Inhaltliche Qualität....13

II.3.2. Didaktische Qualität....14

II.3.3. Benutzerfreundlichkeit....14

II.4. Arbeitspaket 4: Entwicklung und prototypische Realisierung eines DB-gestützten, Web-basierten Informationssystems....15

II.4.1. Dokumentenverwaltung....15

II.4.2. WebDAV-basierte prototypische Realisierung mit Slide....20

II.4.3. Produktionssystem....22

II.4.4. Realisierung der Anfrageverarbeitung....28

II.4.5. Realisierung der Ergebnisverarbeitung....32

II.4.6. Realisierung der Ergebnisverarbeitung....35

II.4.7. Semi-automatische Erschließung....35

II.5. Arbeitspaket 5: Verbesserung der Qualität der Erschließung bei schlecht erschlossenem Material....36

II.6. Arbeitspaket 6: Gestaltung der Benutzerschnittstelle....36

II.6.1. Konzept und Design....36

II.6.2. Prototyping....37

II.6.3. Benutzertests....38

II.6.4. Implementierung und Test....38

II.7. Arbeitspaket 7: Evaluation von META-AKAD durch die Benutzer....39

III Literatur....40

III.1. Weitere Literatur....41

IV Anhang.....42

IV.1. META- AKAD Metadata Element Set and Structure.....42

IV.2. CCG Benutzeranleitung.....42

IV.3. Kontrollierte Vokabularien der Metadatenelemente.....42

IV.4. Sammeln von Lerndokumenten: Strategie und Softwareunterstützung.....42

IV.5. Pony Bedienungsanleitung.....42

IV.6. Qualitätsbeurteilung von Web-Lernobjekten: Erweiterung der Metadatendefinition.... 42

IV.7. Neugestaltete Architektur des META-AKAD-Systems mit Java 2 Enterprise Edition… 42

IV.8. Benutzerschnittstelle für das kollaborative Erschließen von Dokumenten: Ein Rahmenkonzept.....42

IV.9. Versuchsanleitung der Usability Studie.....42

IV.10. Verwaltungssysteme für Online-Lernmaterial. Ein benutzerorientierter Vergleich .42

IV.11. Gestaltung der Schnittstelle für den Endbenutzer.....42

I Einleitung

I.1. Aufgabenstellung

Um elektronische Lehr-/Lernmaterialien für Lehrzwecke im Rahmen von Lehrveranstaltungen effizient einsetzen oder für Lernzwecke im Selbststudium zielorientiert nutzen zu können, sollten Lehrende wie Lernende einen schnellen, möglichst umfassenden und bedarfsgerechten Zugriff auf die im Internet verfügbaren Dokumente in einem auf diese Anforderungen zugeschnittenen integrierten Informationssystem haben. META-AKAD hatte zum Ziel, solch einen innovativen Pilot-Service durch den Einsatz und die Entwicklung geeigneter technischer Mittel und den Aufbau funktionierender Organisationsstrukturen zu entwickeln. Dieser neue Nutzer-Service sollte exemplarisch für ausgewählte Fächer aufgebaut werden. Die Konzeption und Realisierung dieses Dienstes sollte allerdings so ausgelegt werden, dass er prinzipiell auf alle Disziplinen erweitert werden kann.

Zur Realisierung dieses Dienstes sollten Online-Lehr- bzw. Lernmaterial auf kooperativer Basis gesammelt, durch standardisierte und materialspezifische Meta-Daten erschlossen, nach inhaltlichen und didaktischen Kritierien bewertet und in einer einheitlichen Nutzeroberfläche zugänglich gemacht werden.

Das Portal ist unter dem Namen AKLEON (Akademisches Lernmaterial Online) und der Domain <http://www.akleon.de/> online.

I.2. Voraussetzungen

I.3. Planung und Ablauf

Das gesamte Unterfangen wurde inhaltlich in sieben Arbeitspakete untergliedert, deren Bearbeitungsreihenfolge und Abhängigkeit durch einen Meilensteinplan festgelegt war.

Die Arbeitspakete waren:

1. Arbeitspaket 1: Sammlung des Lehr- und Lernmaterials
   - MS 1: Sichtung und Vergleich der zur Verfügung stehenden Werkzeuge, Implementation einer vorläufigen Datenbank. Erste Phase der Sammlung.
   - MS2: Grobkonzept für Verfahren zur Beschreibung und Klassifikation.
   - MS3: Verfeinerung der Konzepte, Prototypische Sammlung
   - MS4: Abschluss der Sammlung von Dokumenten
2. Arbeitspaket 2: Definition von Metadaten
   - MS1: Definition des Metadatensets
   - MS3: Evaluation des Prototyps, internationale Abstimmung
3. Arbeitspaket 3: Bewertung des Lehr- und Lernmaterials
   - MS2: Computerunterstütztes System zur Bewertung
   - MS4: Sammlung bewerteten Materials
4. Arbeitspaket 4: Entwicklung und prototypische Realisierung eines DB-gestützten, Web-basierten Informationssystems
   - MS1: Die zur Realisierung in Frage kommenden Werkzeuge und Systeme wurden gesichtet, untersucht und miteinander verglichen.

- MS2: Anhand der Anforderungen an die eigentliche Dokumentenverwaltung wurde ein entsprechendes System ausgewählt und die notwendigen Datenstrukturen (i.a. in Form eines DB-Schemas) zur Verwaltung von Primär- und Sekundärdaten wurden festgelegt. Weiterhin sind die Konzepte für die verschiedenen Aspekte der Dokumentenverwaltung ausgearbeitet.
- MS3: Ein erster, lauffähiger Prototyp steht zur Verfügung, der die bis zum MS2 entwickelten Konzepte für Erschließung und Bewertung von Dokumenten berücksichtigt und die ebenfalls bis zum MS2 im Arbeitsschwerpunkt 6 gestaltete Benutzerschnittstelle implementiert.
- MS4: Ein Prototyp der sowohl die verfeinerten Konzepte zur Erschließung und Bewertung von Dokumenten als auch die Ergebnisse der internen Testphase reflektiert, steht zur Verfügung.

5. Arbeitspaket 5: Verbesserung der Qualität der Erschließung bei schlecht erschlossenem Material
   - MS1: In Abstimmung mit Arbeitspaket 2 Entscheidung darüber, welche Metadaten automatisch extrahiert werden sollen; vergleichender Test der möglichen Softwaresysteme bzgl. Anwendbarkeit im Rahmen des neu zu entwickelnden Systems; Entscheidung darüber, welche Systeme zum Einsatz kommen sollen;
   - MS2: Adaption der Verfahren der ausgewählten Systeme an die eigenen Anforderungen, insbesondere: Übertragung auf neue Anwendungsbereiche, Integration in das Datenbanksystem in Zusammenarbeit mit Arbeitspaket 4;
   - MS3: In Zusammenarbeit mit Arbeitspaket 1 Anwendung der Indexierungswerkzeuge für die Erschließung und Sammlung von Netzpublikationen; verbesserte Qualität der Ergebnisse durch intellektuelle Nachbearbeitung der automatisch vergebenen Metadaten;
   - MS4: Abschließende Evaluierung der automatisch erzeugten Metadaten durch Vergleich mit den Ergebnissen einer intellektuellen Erschließung;
6. Arbeitspaket 6: Gestaltung der Benutzerschnittstelle
   - MS1: Sichtung und Evaluierung der möglichen Mechanismen zur Realisierung der Benutzerfunktionen (z.B. XML, HTML, HTTP, Java Applets und Servlets, Web-Server, CGI, etc.);
   - MS2: Beschreibung der einzelnen an der Systemschnittstelle anzubietenden Funktionen;
   - MS3: Realisierung der Benutzerfunktionen im Rahmen von Arbeitsschwerpunkt 4.
7. Arbeitspaket 7: Evaluation von META-AKAD durch die Benutzer
   - MS1: Sichtung und Evaluierung der möglichen Mechanismen zur Realisierung der Benutzerfunktionen (z.B. XML, HTML, HTTP, Java Applets und Servlets, Web-Server, CGI, etc.);
   - MS2: Beschreibung der einzelnen an der Systemschnittstelle anzubietenden Funktionen;
   - MS3: Realisierung der Benutzerfunktionen im Rahmen von Arbeitsschwerpunkt 4.

Die Abb. 1 zeigt den ursprünglichen Ablaufplan des Projekts.

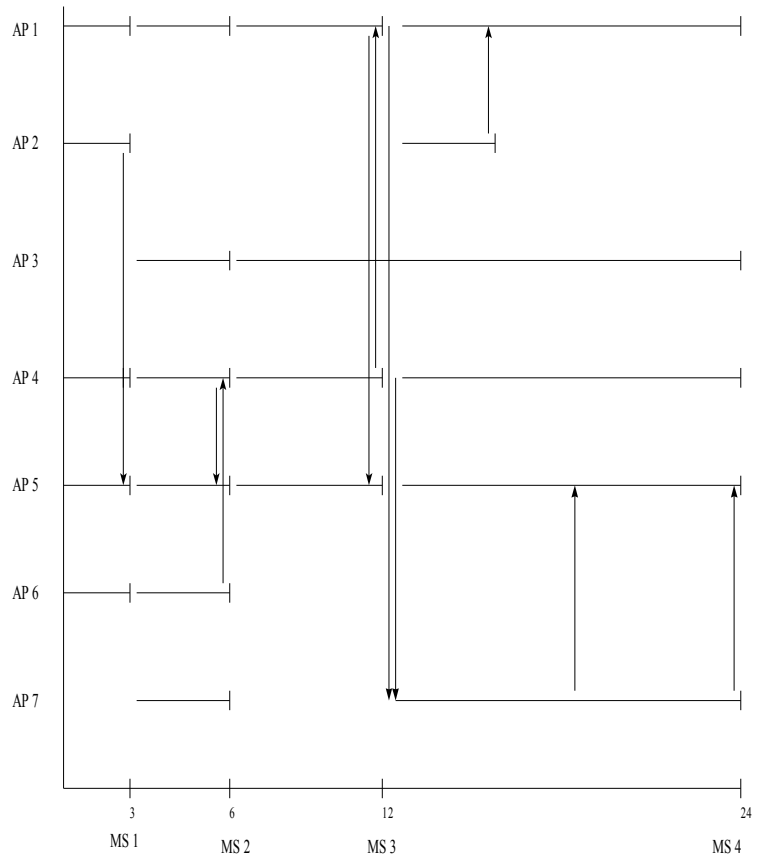

Abbildung 1: Grafische Darstellung des Projektablaufs

Der Projektverlauf entsprach in überwiegendem Maße dem ursprünglichen Plan, wobei einige Änderungen aufgrund verzögerter Arbeitsschritte, aber auch inhaltlicher Erwägungen notwendig wurden:

- Das Redaktionssystem für die Qualitätsbeurteilung wurde in technischer Hinsicht in das Teilsystem zur Vergabe von Metadaten integriert, mit dem Ziel eines einheitlichen redaktionellem Workflow. Diese zusätzlichen Anforderungen führten in AP4 zu weiterer Verzögerung.
- Nach dem ersten Systementwurf musste eine weitere funktionale Anforderung an das Redaktionssystem aufgenommen werden, was eine Überarbeitung des Entwurfs notwendig machte.
- Das Arbeitspaket „Evaluation“ wurde inhaltlich zu einer insgesamt benutzerorientierten Entwicklung erweitert. Die damit zusammenhängenden Aufgaben wurden parallel zur Systementwicklung durchgeführt.
- Aufgrund der Verzögerungen konnten folgende Ziele bisher nicht erreicht werden:
  - Vollständige technische Umsetzung des Redaktionssystems
  - Vollständige Metadatenerschließung aller Dokumente
  - Testbetrieb und abschließende Evaluation

Teilweise konnten die nicht erreichten Ziele aber anderweitig kompensiert werden. Die Metadatenerschließung wurde statt mit dem Redaktionssystem weiterhin mit dem Eingabeformular der Linksammlung durchgeführt. Für die Qualitätsbegutachtung wurde auf eine Erhebung per Fragebogen ausgewichen. In gewissem Maße wurde eine Evaluation schon durch die Benutzertests der Oberfläche abgedeckt.

I.4. Wissenschaftlicher und technischer Stand

I.4.1. Metadatenstandards

Bemühungen, Metadaten zu Lehr- und Lerndokumenten auszutauschen, erlangen zunehmende Bedeutung. Es existieren mehrere, teilweise sehr aktive internationale Initiativen, die semantische und syntaktische Standards für den Metadatenaustausch erarbeiten.

Das Resource Description Framework RDF[1] integriert eine breitgestreute Anzahl von Anwendungen, von Bibliothekskatalogen und weltweiten Verzeichnissen über die Sammlung und Weiterverteilung von Nachrichten, Software und anderen Informationen bis zu persönlichen Sammlungen von Musik, Photographien und Ereignissen, wobei die Extensible Markup Language XML[2] als verbindende Syntax für den Datenaustausch eingesetzt wird. Die RDF-Spezifikationen stellen ein leichtgewichtiges Ontologiesystem zur Verfügung, daß den Austausch von verschiedensten Informationen über das WWW unterstützt. RDF wird in unterschiedlichen Ausprägungen von einer Vielzahl von Organisationen und Firmen eingesetzt.

Die Dublin Core Metadata Initiative[3] entwickelt eine Reihe von interoperablen Metadaten-Standards für Online-Material, die ein breites Feld von Verwendungen und Geschäftsmodellen abdecken. Die gemeinsame Basis dieser Standards ist der Metadatensatz Dublin Core Element Set[4], der zur Minimalbeschreibung fast aller Ressourcen geeignet ist. Um genauere, domänenspezifische Beschreibungen zu ermöglichen existieren standardisierte Qualifizierer des Metadatensatzes[5]. Gleichzeitig ist der Dublin-Core-Metadatensatz auch offen für anwendungsspezifische, semantische Verfeinerungen. Eine Interoperabilität bleibt dennoch erhalten, da Verfeinerungen von Metadatenelementen, die einem System unbekannt sind gemäß dem Standard ohne Verfälschung auf die übergeordenten Elemente abgebildet werden können. Eine eigene Arbeitsgruppe DC Education entwickelt lehr- und lernmaterialspezifische Qualifizierer. Nachdem eines der zentralen Anliegen von Dublin Core ist, eine möglichste weite Verbreitung durch die Einfachheit des Metadatensatzes zu fördern, sind die Beschreibungsmöglichkeiten aber auch mit DC Education noch sehr eingeschränkt.

Aufgrund seiner Universalität und leichten Implementierbarkeit fand der Dublin-Core-Metadatenstandard sehr weite Verbreitung. Der Dublin-Core-Metadatensatz ohne Qualifizierer ist deshalb auch das vorgeschriebene Minimum beim Austausch von Metadaten nach dem Open Archives Metadata Harvesting Protocol. Dieses leicht zu implementierende Protokoll wurde von der Open Archives Initiative (OAI)[6] ursprünglich für den XML-kodierten Austausch von Metadaten zu elektronischen Fachartikeln (Eprints) entwickelt, findet aber inzwischen allgemeine Anwendung für die Verbreitung und Sammlung (Harvesting) von Metadaten über verschiedenste Ressourcen. Die Deutsche Initiative für NetzwerkInformation (DINI)[7] fordert die Einrichtung von OAI-Schnittstellen für alle deutschen Hochschularchive, um den gezielten Austausch von Metadaten nach festgelegten formalen und inhaltlichen Kriterien zu ermöglichen.

Speziell für den Austausch von Metadaten über Lehr- und Lernmaterial wurde unter Beteiligung internationaler Firmen und Organisationen ein sehr elaborierter Metadatenstandard (kurz: LOM-Standard) entwickelt, der in zwei minimal unterschiedlichen Versionen existiert, die IMS Learning

1 <http://www.w3.org/RDF/>

2 <http://www.w3.org/XML/>

3 <http://dublincore.org/>

4 <http://dublincore.org/documents/dces/>

5 <http://dublincore.org/documents/dcmes-qualifiers/>

6 <http://www.openarchives.org/>

7 <http://www.dini.de/>

Resource Meta-data Specification des IMS Global Learning Consortium[8] und der IEEE Learning Object Metadata Standard[9]. Der LOM-Standard bietet vielfältige detaillierte Beschreibungsmöglichkeiten für Lehr- und Lernmaterial, besonders für lehr-/lernmaterialspezifische Aspekte, aber auch für technische Informationen, Rechtsfragen, Anmerkungen und Klassifikationen. Für den LOM-Standard stehen XML-Bindungen incl. eines XML-Schemas zu Verfügung, die die Implementierung und den tatsächlichen Austausch sehr erleichtern[10].

I.4.2. Methoden zum automatischen Sammeln und Erschließen

Die untersuchten und realisierten Methoden zum automatischen Sammeln werden in Kap.II.2.1 auf Seite 10ff erläutert, die zum automatischen Erschließen in Kap.II.6 auf Seite 47ff.

I.4.3. Systemarchitektur

Auf die Architektur des Meta-Akad-Systems (MAS) wird ausführlich in Kap. II.5 auf Seite 22ff eingegangen.

I.5. Zusammenarbeit mit anderen Stellen

Mit der Virtuellen Hochschule Bayern[11] wurde im Bereich Metadatenstrukturen zusammengearbeitet. Im nächsten Schritt (vgl. unter Kap. III) sollen ein Datenaustausch und eine weitergehende Integration der Projekte besprochen werden. Mit dem Virtuellen Campus Rheinland-Pfalz[12] wird kontinuierlich zusammengearbeitet.

In Planung befinden sich derzeit die Kooperation mit der Virtuellen Universität Regensburg (Lehrstuhl für Wirtschaftsinformatik 3, Prof. Lehner, Uni Regensburg)[13] und der studentischen Initiative Knowledgebay[14].

Bei der VUR handelt es sich um ein kursbasiertes System zur Unterstützung der Lehre. Sie umfasst u.a. Funktionen zum Contentmanagement, Kommunikationsfunktionen und Media Streaming. Geplant ist eine bilaterale Anbindung zwischen AKLEON und der VUR, indem die in der VUR bereitgestellten Lernmaterialien in AKLEON verzeichnet werden und Suchfunktionen von AKLEON in die VUR integriert werden.

Knowledgebay ist eine recht junge studentische Initiative an der Universität Regensburg mit dem Ziel multimediales Lernmaterial selbst zu produzieren und über ein Webportal bereitzustellen. Hier ist eine ähnliche Anbindung, wie zur VUR geplant. Im Projekt Knowledgebay wird bereits aktiv daran gearbeitet, die Metadaten an die Vorgaben von Meta-Akad anzupassen und in einer Form bereitzustellen, die eine automatische Erfassung erlaubt. Im Gegenzug kann die Sacherschließung und Begutachtung der Materialien an der UB Regensburg geleistet werden. Ferner soll in Zukunft in Knowledgebay eine Suchfunktion von AKLEON integriert werden.

8 <http://www.imsproject.org/specifications.cfm>

9 <http://ltsc.ieee.org/wg12/index.html>

10 <http://www.imsproject.org/specifications.cfm>

11 <http://www.vhb.org/>

12 <http://www.vcrp.de/>

13 <http://vur.uni-regensburg.de/>

14 <http://www.knowledgebay.de/>

II Ergebnisse

II.1. Allgemeines

Die Ziele des Projektes Meta-Akad waren

- der Aufbau einer Sammlung von Lehr- und Lernmaterial ausgewählter Fächer basierend auf
- einem datenbankgestützten Informationssystem,
- die Definition von Metadaten und
- die Verbesserung der Erschließung des Materials durch intellektuelle und semi-automatische Vergabe von Metadaten,
- die Entwicklung eines Systems zur Qualitätskontrolle des Materials,
- die Präsentation des gesammelten und erschlossenen Materials und
- die Entwicklung einer Benutzeroberfläche, die durch Benutzer evaluiert wurde.

Diese Ziele wurden im wesentlichen erreicht. Aus dem Projekt Meta-Akad entstand der Dienst AKLEON – **Ak**ademisches **Le**rnmaterial **On**line[15]. In AKLEON ist bisher eine umfangreiche Sammlung von ca. 6200, im WWW verfügbaren Lehr- und Lerndokumenten in den Fächern Biologie, Germanistik, Mathematik, Physik und Psychologie zusammengestellt worden. Darüberhinaus sind auch (zum Teil interdisziplinäre) Lerndokumente anderer Fächer in einem geringeren Umfang im System vorhanden. Diese Lehr- und Lerndokumente können von den Benutzern, Studenten und Hochschullehrer, anhand der erhobenen Metadaten und in Volltextauszügen bequem in einer Weboberfläche recherchiert werden. Vor allem die Dokumente in den Fächern Mathematik und Physik sind nicht nur formal sondern auch inhaltlich erschlossen worden.

Neben diesen bereits stark mit Metadaten angereicherten, veröffentlichten Dokumenten befinden sich noch ca. 800 intellektuell gesammelte Lerndokumente auf einer früheren Erschließungsstufe in der Datenbank des Sammelsystems Pony, von wo sie im Laufe der weiteren Erschließung in die Datenbank von AKLEON übergeführt werden. In Pony sind außerdem gut 1500 Webseiten mit Linklisten mutmaßlicher Lerndokumente nachgewiesen, die fortlaufend auf geeignete Lerndokumente ausgewertet werden.

Die Sammlung und Erschließung von Lehr- und Lernmaterial incl. der Pflege („tote Links“) des Bestandes in AKLEON sind Teil eines während des Weiterbetriebs von AKLEON andauernden Prozesses.

Grundlage für die Lehr- und Lernmaterial-gerechte Recherche in AKLEON sind vor allem die während des Projekts definierten und zu den Dokumenten intellektuell und semi-automatisch erhobenen Metadaten. Daneben wird der erfreulich große Umfang des Korpus von Lernmaterial als ausschlaggebend für den zukünftigen Erfolg von AKLEON angesehen. Im Rahmen des Workflows findet eine mehrstufige Qualitätskontrolle der nachgewiesenen Dokumente statt.

Die im Rahmen des Projekts entwickelte Software für die manuelle und semi-automatische Sammlung und Erschließung von Lehr- und Lernmaterial, für die Verwaltung der Metadaten und für die Präsentation in der Endbenutzerschnittstelle steht für die weitere Nutzung zur Verfügung.

15 <http://www.akleon.de>

II.2. Arbeitspaket 1: Sammlung des Lehr- und Lernmaterials

II.2.1. Automatisches Sammeln

Seit dem letzten Zwischenbericht ist das in Kaiserslautern entwickelte Programm CCG (Collect and Classify GUI) fertig gestellt worden.

Hauptaufgabe des Programms CCG ist es, Dokumente aus dem Internet zu sammeln und diese auf vorher erlernte Eigenschaften hin zu überprüfen. Das bedeutet, bevor man überhaupt die eigentliche Aufgabe erledigen kann, muss CCG erlernen, welche Eigenschaften ein Dokument aufweisen soll. Da hier Lehrmaterial gesucht werden soll, muss CCG erlernen, welche Eigenschaften Lehrmaterial im allgemeinen aufweist. Um diese Eigenschaften zu erlernen, muss man dem Programm eine Menge von Lehrmaterialien (Trainingsmenge) zur Verfügung stellen, an Hand derer CCG die Eigenschaften ermitteln kann, die Lehrmaterial auszeichnen.

Um diese komplexe Aufgabe zu erledigen, geht CCG in mehreren Schritten vor. Zunächst werden Dokumente aus dem Internet gesammelt und in einem Datenbanksystem gespeichert. In CCG wird hier immer der Begriff Indexieren verwendet, da nicht nur das Dokument gesammelt wird, sondern zu jedem Dokument auch noch Metadaten im Datenbanksystem abgelegt. (Rein technisch gesehen macht dieser Teil des Programms nichts anderes als jede Suchmaschine im Internet, die Dokumente indexiert, um später für Suchanfragen passende Ergebnisse liefern zu können).

Nachdem die Dokumente indexiert wurden, können diese dann klassifiziert (getestet) werden. Unter Klassifizieren wird im allgemeinen die Aufteilung von Dokumenten auf Grund verschiedener Dokumenteigenschaften verstanden. Bei CCG wird zum Klassifizieren immer nur eine Eigenschaft betrachtet, d.h. die gesammelten Dokumente werden in zwei Mengen aufgeteilt. In eine, für die die betrachtete Eigenschaft gilt und in eine, für die diese Eigenschaft nicht zutrifft. Da neben der Eigenschaft ´Ist Lehrmaterial´ auch noch festgestellt werden soll zu welchem Fachbereich sich ein Dokument zuordnen lässt, ist es sinnvoll auf die Eigenschaft ´Ist Lehrmaterial für Fachbereich XY´ hin zu klassifizieren. Der Vorgang des Klassifizierens wird in CCG auch als Testen bezeichnet, d.h. es wird für eine Menge von Dokumenten getestet, ob eine bestimmte Eigenschaft für die Dokumente dieser Menge gilt.

Damit CCG überhaupt klassifizieren kann, muss das Programm zunächst auf die gewünschte Eigenschaft hin trainiert werden. Dazu benötigt das Programm eine Menge von Dokumenten die bereits klassifiziert ist. Um solch eine Menge zu erhalten ist der Benutzer gefordert, denn dieser muss eine solche Menge zusammenstellen. Dazu indexiert man einige Dokumente von denen bekannt ist, dass sie die gewünschte Eigenschaft erfüllen und markiert diese als positiv. Ebenso sollte der Benutzer einige Beispiele indexieren, die völlig unzutreffend im Hinblick auf die gewünschte Eigenschaft sind und diese dann als negativ markieren. Zusätzlich kann man auch noch Dokumente in die Trainingsmenge aufnehmen, die man eher als neutral einstuft und diese dann auch als solche markieren. Je besser und größer die Trainingsmenge ist, umso besser kann das Programm auch trainieren, d.h. umso besser sind die später erreichten Ergebnisse.

Ist eine Trainingsmenge zusammengestellt, kann CCG das Training beginnen. Zum Trainieren verwendet CCG eine Support Vector Machine (SVM). Dazu werden die Dokumente zunächst in Vektoren umgewandelt und von der SVM analysiert. Das bei der Analyse erworbene Wissen wird hier als Modell bezeichnet. Wenn also in CCG ein Modell erzeugt wird, dann wird eine Trainingmenge analysiert und das Wissen in einem Modell gespeichert. Dieses Modell wird dann beim Testen (Klassifizieren) von Dokumenten herangezogen, um entscheiden zu können, ob für ein Dokument die im Modell gelernte Eigenschaft zutrifft oder nicht. Da bei einem Trainingsvorgang immer nur trainiert wird, eine Eigenschaft zu erkennen, d.h. ein Modell erzeugt wird, muss für jede gewünschte Eigenschaft ein eigenes Modell erzeugt werden.

Wurde eine Testmenge klassifiziert (getestet) so entsteht ein Testergebnis, welches aus der Ergebnismenge besteht. Diese Ergebnismenge beinhaltet alle Dokumente aus der Testmenge, die die im Modell erlernte Eigenschaft(en) aufweisen. Diese Ergebnismenge kann schließlich nochmals an Hand vom Benutzer festgelegter Begriffe gefiltert werden, um so ein exakteres bzw. besseres Ergebnis zu erhalten.

Insgesamt ergibt sich für die Überprüfung auf Lehrmaterial das in Abb. 2 unten dargestellte Bild.

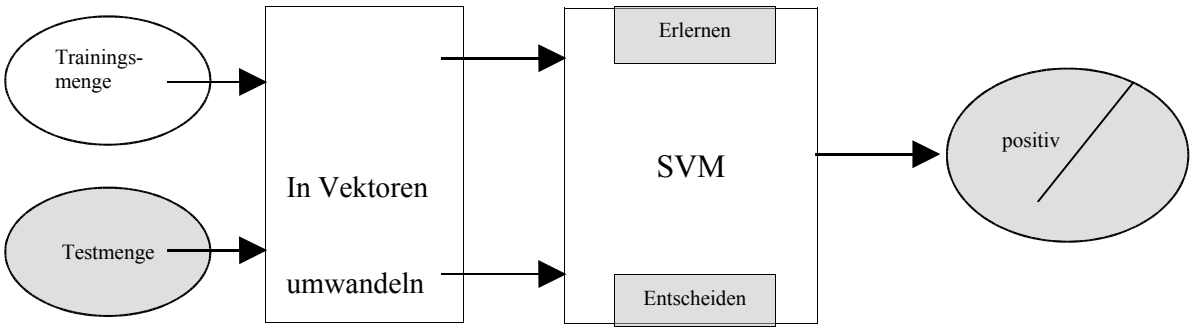

Abbildung 2: Überprüfung auf Lernmaterial mit der SVM

II.2.2. Intellektuelles Sammeln in Kaiserslautern

Schon seit Beginn des Projektes wird in Kaiserslautern das Sammeln, Erschließen und Klassifizieren von Lehr- und Lernmaterialien für die Fächer Physik und Mathematik kontinuierlich betrieben. Dabei handelte es sich anfangs beim Sammeln um eine rein intellektuelle Arbeit. Zum manuellen Suchen wurden unter anderem folgende Quellen benutzt:

- Verzeichnisse von Suchmaschinen wie zum Beispiel das Physik- und Mathematikverzeichnis von galaxy: <http://www.galaxy.com/>
- Die Webseiten von Universitäts- und Forschungsinstituten. Hier beschränkte sich die Suche hauptsächlich auf deutsch- und englischsprachige Institutionen. Für die deutschen Universitäten stand uns die ausführliche Liste „Multimedia-Aktivitäten in der Physiklehre an deutschen Hochschulen“ zur Verfügung. Diese Liste wurde im Rahmen des Projekts Physics Education Network des Fachbereichs Physik der Universität Kaiserslautern und des (ehemaligen) rheinland-pfälzischen Ministerium für Bildung, Wissenschaft und Weiterbildung erstellt: <http://pen.physik.uni-kl.de/mmphysik/>
- Bereits existierende Portale, die Lern- und Lehrmaterialien in Form von Vorlesungsskripten, Animationen, Applets und Videos anbieten. Hier gibt es beispielsweise:
    - The Virtual Library <http://vlib.org/>
    - World Lecture Hall <http://www.utexas.edu/world/lecture/>
    - MERLOT <http://www.merlot.org/Home.po>
    - The Math Forum <http://mathforum.org/library/>
- Einige wenige amerikanische Verlage, die Lehrbücher und Vorlesungsskripte online frei anbieten wie beispielsweise:
    - Light and Matter <http://www.lightandmatter.com/>
    - Samizdat Press <http://samizdat.mines.edu/>
    - Private Anbieter von Skriptsammlungen wie z.B. „Skripten und Vorlesungsmitschriften“ von R. Wagner <http://www.physik.tu-muenchen.de/~rwagner/physik/skripten.html>

Im Laufe der Zeit wurde jedoch die Suche nach Dokumenten mehr und mehr mit Hilfe des Programms CCG durchgeführt, das im Rahmen dieses Projektes entwickelt wurde und das im vorhergehenden Paragraf dieses Berichtes beschrieben wird.

Zurzeit findet man in der Datenbank folgende Anzahl an Dokumenten, die in der Tabelle nach Medientyp geordnet sind:

Medientyp	**Physik**	**Mathematik**
Applet	383	116
Audio	0	0
Bild	10	1
Multimedia	268	193
Software	3	2
Sonstige Anwendung	8	9
Text	816	987
Video	12	2
Gesamt	**1490**	**1309**

Zum Vergleich betrug der Datenbankinhalt für beide Fächer im Februar 2002 1545 Dokumente, im April 2002 2076, im Oktober 2002 2535 und im Juli 2003 2799. Diese 2799 Dokumente wurden inzwischen in die endgültige Datenbank importiert.

Dass die Dokumentenanzahl in diesen beiden Fächern seit dem letzten Zwischenbericht nicht so schnell gewachsen ist wie im ersten Projektjahr, hat mehrere Ursachen: Einmal wurde zu Beginn des zweiten Projektjahres nachträglich mit der Klassifikation nach dem System des Regensburger Verbundes (RVK) aller - inzwischen waren es ca. 2000 - schon gesammelten Dokumente begonnen. Ein weiterer Grund ist, dass das neue Programm zum automatischen Sammeln von Lehrmaterialien CCG getestet und optimiert werden musste. Zu diesem Programm wurde außerdem eine ausführliche Benutzeranleitung verfasst, die im Anhang zu diesem Bericht zu finden ist. Auch die regelmäßige Linkpflege nimmt viel Zeit in Anspruch. Außerdem wurde festgestellt, dass mit wachsendem Bestand der Datenbank die Materialien, die in Web-Sammlungen aufgelistet sind, zum großen Teil schon in die Datenbank erfasst sind.

Die Linkpflege, die oben erwähnt wurde, besteht darin, dass in regelmäßigen Abständen Linkchecks durchgeführt werden, um zu überprüfen, ob die Links noch zu den Materialien führen. Es hat sich bei den letzten vierteljährlichen Checks herausgestellt, dass etwa 5% der Dokumente nicht mehr erreichbar sind. Die Hälfte davon wird durch manuelle Suche unter einer geänderten URL gefunden. Die restlichen bleiben in der Datenbank, sind jedoch für den Endbenutzer unsichtbar.

Auch die Begutachtung der Materialien durch Experten hat schon begonnen. So wurden die multimedialen Materialien zur Quantenmechanik (Applets, Lernprogramme etc.) im Rahmen des Projektes „European Physics Education Network“ (EUPEN) evaluiert [MB], [HJJ]. Zurzeit werden im Rahmen des gleichen Projektes die Materialien zur Optik und im Jahr 2004 werden diejenige über Mechanik bewertet.

Der Einsatz besonders wertvoller Materialien findet weiterhin statt: Vor allem Animationen zu physikalischen Phänomenen werden im Rahmen des Physik-Fernstudiums FiPS (**F**rüheinstieg **i**ns **P**hysik**s**tudium)[16] verwendet.

In den beiden Fächern Mathematik und Physik gibt es zurzeit 1191 Dokumente in deutscher Sprache und 1587 Dokumente in englischer Sprache.

16 <http://fips-server.physik.uni-kl.de/index.html>

Die Suche in den beiden Fächern hat bis zum heutigen Tage zu einer flächendeckenden Sammlung an Lehr- und Lernmaterialien geführt, wie man anhand der folgenden Abbildungen sehen kann. Die Verteilung der Lehrmaterialien zur Mathematik ist in den Abb. 3 und 4 nach den verschiedenen Gebieten aufgeschlüsselt dargestellt. In der Abb. 5 sieht man die Aufschlüsselung der Lehrmaterialien für das Fach Physik.

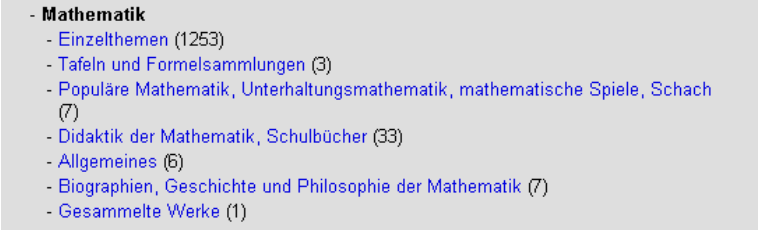

Abbildung 3: Lehr- und Lernmaterialien nach mathematischen Gebieten aufgeschlüsselt

Abbildung 4: Weitere Aufschlüsselung der 1253 Einzelthemen im Fach Mathematik

- **Physik**
  - Mechanik, Kontinuumsmechanik, Hydrodynamik, Schwingungslehre (283)
  - Thermodynamik, statistische Physik, Quantenstatistik (165)
  - Physikalische Optik (115)
  - Quantentheorie, Quantenmechanik (129)
  - Enzyklopädien und Lehrbücher (197)
  - Klassische Felder, Elektrizitätslehre, Optik, Quantenoptik, Laser, Relativitätstheorie (107)
  - Festkörperphysik (62)
  - Quantenoptik, Laser- und Masertheorie, Quantenelektronik (110)
  - Allgemeine Nachschlagewerke, Bibliographien, Geschichte und Philosophie der Physik (475)
  - Elementarteilchenphysik, Physik der Felder und Hochenergiephysik (123)
  - Atom- und Molekülphysik (45)
  - Struktur der Flüssigkeiten, Physik der Gase, Plasmaphysik (16)
  - Experimentelle Methoden der Physik und physikalische Technologie (38)
  - Kernphysik (43)
  - Astronomie, Astrophysik (125)
  - Geophysik (18)
  - Physik und Chemie der Luftverunreinigungen und Aerosole (7)
  - Meteorologie (4)
  - Materialwissenschaft (6)
  - Polymerphysik (1)
  - Vielteilchentheorie (2)
  - Mineralogie (1)

Abbildung 5: Lehr- und Lernmaterialien nach physikalischen Gebieten aufgeschlüsselt

Nach Dokumenttypen geordnet sieht die Verteilung für beide Fächer wie in Abb. 6 aus.

In Zusammenhang mit diesem Projekt sind zwei Artikel zur Veröffentlichung eingereicht worden [GW1], [GW2].

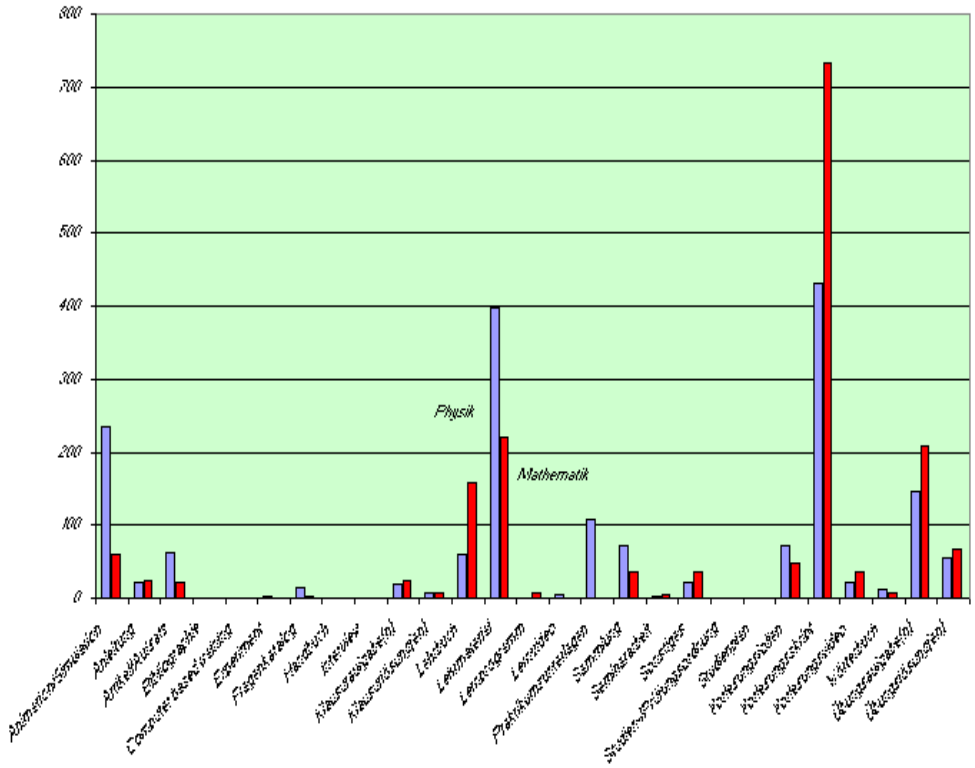

Abbildung 6: Dokumenttypen nach Fächern getrennt

II.2.3. Intellektuelles Sammeln in Regensburg

Entwicklung eines Unterstützungssystems zur Webrecherche

Beim intellektuellen Sammeln von Lernmaterial in Regensburg wurde eine andere Strategie verfolgt als in Kaiserslautern. Eine Aufgabenanalyse des Sammelns von Webdokumenten zeigte erhebliche ergonomische Schwierigkeiten und einen Mangel an adäquaten Werkzeugen. Die intellektuelle Webrecherche stellt aber einen neuralgischen Punkt dar, da

- es für den Erfolg des Dienstes entscheidend ist eine große Menge an Dokumenten zu erfassen und so mit dem Angebot eine kritische Masse zu erreichen
- Webressourcen eine geringe Stabilität aufweisen, was die Suche zu einem permanenten Prozess macht
- die automatischen Verfahren fast immer auf eine intellektuell erstellte Lernmenge angewiesen sind.

Um das Suchen und Sammeln von Lerndokumenten dauerhaft effizient betreiben zu können, wurde eine softwareunterstützte Suchstrategie entwickelt.

Ergebnisse der Aufgabenanalyse

Das Erstellen eines Katalogs fachlich geordneter Lerndokumente aus dem Web erfordert eine Webrecherche mit bestimmten Eigenschaften. Eine heuristische Evaluation dieser Aufgabe nach ergonomischen Kriterien kam zu folgenden Ergebnissen:

1. Die gängigen Suchmaschinen liefern nur selten Dokumente, wie sie in META-AKAD gesammelt werden müssen. Das erfordert vom Webrechercheur ein hohes Maß an Recherche-Kompetenz und manuelles Durchsuchen des Web.
2. Die manuelle Suche geht häufig von thematisch geordneten Linksammlungen aus. Diese Links führen jedoch eher zu allgemeinen Websites zu diesen Fächern. Dies erfordert vom Webrechercheur rekursives Durchsuchen vieler Websites mit z.T. komplexen Verzeichnisstrukturen. Diese Suche ist äußerst anstrengend, zeitaufwändig und fehleranfällig.
3. Mit dem Fortschreiten der Recherche kommt der Webrechercheur immer häufiger in die Situation, entscheiden zu müssen, ob er ein Dokument schon erfasst hat oder nicht. Auch das ist fehleranfällig und kostet Zeit.
4. Für die Erfassung der gesammelten Ressourcen standen zwei Alternativen zur Verfügung, die vorläufige Datenbank in Kaiserslautern und die Bookmarkverwaltung des Browsers. Die Bookmarkverwaltung erwies sich aus verschiedenen naheliegenden Gründen als vollkommen ungeeignet. Bei der Eingabe in die vorläufige Datenbank ist ein umfangreiches Formular auszufüllen. Diese Aufgabe interferiert mit der ohnehin belastenden Webrecherche.

Konzeption und Entwicklung des Unterstützungssystems

Ausgehend von der Aufgabenanalyse muss ein System zur Unterstützung der Webrecherche folgende Eigenschaften besitzen:

- Es muss über eine eigene Suchmaschine verfügen, die auch Seiten indiziert, die von allgemeinen Suchmaschinen nicht erfasst werden.
- Es muss eine Datenbankanbindung haben, mit der sich sehr schnell neue Einträge machen lassen
- Es muss dem Benutzer online zeigen, welche Dokumente bereits erfasst sind

Ein solches System wurde an der Universitätsbibliothek Regensburg unter dem Namen *Pony* entwickelt und ist seit Oktober 2001 einsatzbereit. Es wurde in Form eines CGI Proxy verwirklicht, der den Benutzer beim Browsen „begleitet“. Als Datenbanksystem wird MySQL eingesetzt. Die Suchmaschinenkomponente basierte zunächst auf Harvest, wurde jedoch später auf das leistungsfähigere ASPSeek umgestellt.

Eine genauere Beschreibung befindet sich im Anhang unter „Sammeln von Lerndokumenten: Strategie und Softwareunterstützung“ und „Pony Bedienungsanleitung“.

Um möglichst viele und technisch weniger versierte Benutzer mit Pony beim Sammeln von Lernmaterial zu unterstützen, wurde das System einer Usability Studie unterzogen. Dies geschah in Zusammenarbeit mit dem Lehrstuhl für Angewandte Psychologie in Regensburg (Prof. Dr. Alf Zimmer). Die Ergebnisse ermöglichten es, Pony in punkto Benutzerfreundlichkeit entscheidend zu verbessern.

Die Dokumentsammlung in Regensburg

Die UB Regensburg war für das Sammeln in den Fächern Biologie, Psychologie und Germanistik zuständig. Diese Augabe wurde im wesentlichen Studentischen Hilfskräften übertragen, die dazu das Werkzeug Pony einsetzten.

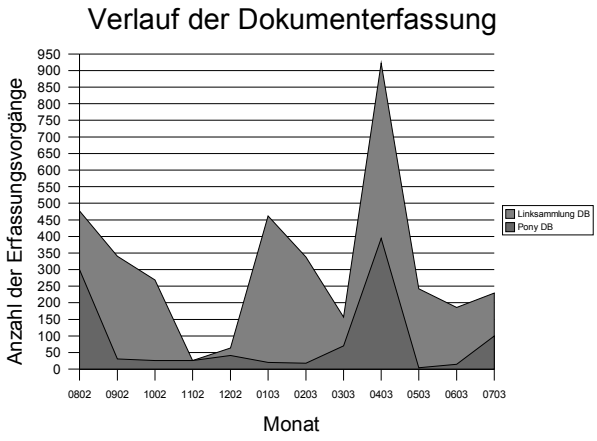

Abbildung 7: Erfassungsvorgänge des Sammelns und Erschließens in Regensburg

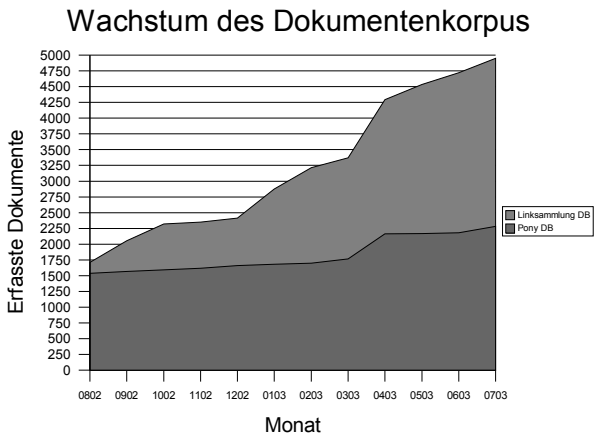

Abbildung 8: Wachstum der Sammlung in Regensburg. Dokumente, die in die Linksammlung DB eingestellt wurden, befinden sich nicht mehr in der Pony DB.

Dabei war das Vorgehen zweischrittig: Zunächst wird eine Sammlung von Dokumenten in der Datenbank von Pony angelegt. Hier kommt es vor allem auf den schnellen Aufbau einer möglichst großen Sammlung an, ohne dabei schon (mehr als triviale) Metadaten zu erheben. Im zweiten Schritt werden von einem anderen Bearbeiter die Dokumente in die Datenbank der vorläufigen Linksammlung Kaiserslautern transferiert. Dabei wurden mit Hilfe des Eingabeformulars der Linksammlung weitere Metadaten hinzugefügt.

Die Abbildungen 7 und 8 zeigen die Entwicklung der Sammlung in den letzten elf Monaten[17]. Insgesamt wurden bis Anfang Juli 2003 4950 Dokumente gesammelt. Davon befinden sich 2282 in der Datenbank von Pony (gesammelt) und 2667 in der Linksammlung Kaiserslautern (weiter erschlossen). Durchschnittlich wurden im dargestellten Zeitraum pro Monat 87 Dokumente mit Pony gesammelt und 222 Dokumente weiter erschlossen und in die Linksammlung eingetragen. Dazu wurden monatlich im ungefähren Durchschnitt 100 Arbeitstunden durch studentische Hilfskräfte eingesetzt.

17 Die Daten wurden nachträglich aus Log-Files rekonstruiert. Wahrscheinlich existieren gewisse Inkonsistenzen und nicht erfasste Überschneidungen der beiden Datenbanken. Die dargestellten Daten werden den tatsächlichen Status Quo daher geringfügig überschätzen.

Nicht zahlenmäßig dargestellt sind weitere Erfassungsvorgänge mit Pony, die aber zumindest genannt werden sollen:

1. Kennzeichnung eines Dokuments als *nicht geeignet.* Dieser Dokumenttyp wurde eingeführt, um den Rechercheuren das wiederholte „Auffinden“ nicht geeigneter Seiten zu ersparen und um negative Lernmengen für das automatische Sammelsystem zu generieren.
2. Ebenfalls erfasst werden *Sammlungen* und *Verteiler*, also Seiten, die vermutlich auf zahlreiche Lernressourcen verweisen und später noch „geerntet“ werden können.

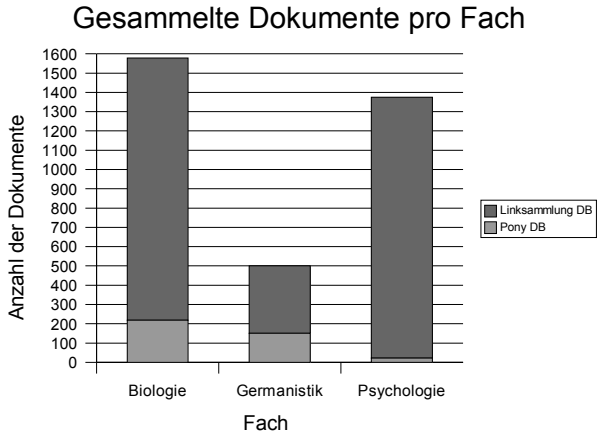

Abbildung 9: In Regensburg gesammelte Dokumente in den Fächern Biologie, Germanistik und Psychologie

Ziel dieses Arbeitspaketes war für Biologie, Germanistik und Psychologie eine Sammlung von 1000-2000 Dokumenten pro Fach aufzubauen. In Biologie und Psychologie wurde das mit Sammlungen von 1578 und 1375 voll erreicht (Abb. 9). Im Fach Germanistik wurde mit dem Sammeln erst im Januar 2003 begonnen. Dabei konnte ein schneller Fortschritt verzeichnet werden (derzeit 501 Dokumente), so dass die angestrebte Dokumentengröße voraussichtlich noch in diesem Jahr erreicht wird.

Insgesamt hat sich das Modell des kooperativen, arbeitsteiligen Sammeln mit Unterstützung durch das System Pony als sehr effizient und kostengünstig erwiesen.

II.3. Arbeitspaket 2: Definition von Metadaten

II.3.1. Definition des Metadatensets

Für META-AKAD wurde eine eigene Metadatenstruktur entwickelt, die auf die besonderen Anforderungen des Systems zugeschnitten ist. Die erarbeitete Definition der Metadaten umfasst Attribute zur Beschreibung von Dokumenten sowie von Gutachten und berücksichtigt insbesondere auch didaktische und rechtliche Aspekte.

Zur Definition der Metadaten wurden zunächst die bestehenden internationalen Standards ausgewertet, darunter vor allem das *Dublin Core Metadata Element Set* zusammen mit den *DCMES-Qualifikatoren*, die von der *Dublin Core Education Working Group* vorgeschlagenen Elemente für Lehrmaterialien und das Schema der *IMS Learning Resource Meta-data*. Anhand der vorliegenden Quellen wurde ein Kern von Elementen für das META-AKAD System herausgearbeitet.

Dabei war grundsätzlich eine Entscheidung darüber zu treffen, ob der Ansatz von Dublin Core oder der Ansatz von IMS/LOM als Basis für die eigene Metadaten-Definition gewählt werden sollte. Beide Ansätze gehen von unterschiedlichen Prinzipien aus. Dublin Core stellt einen allgemeinen Rahmen zur Beschreibung von Ressourcen bereit, der dann mit Hilfe eines Qualifikationsmechanismus den eigenen Anforderungen entsprechend verfeinert werden kann. Demgegenüber gibt IMS/LOM ein im wesentlichen festes Metadaten-Schema vor, das aber weitaus detaillierter ist und im Unterschied zu Dublin Core eine komplexe hierarchische Struktur aufweist. Insbesondere umfasst dieses Schema eigens eine Kategorie für lehrspezifische Angaben wie Zielgruppe, Lernbedingungen, Schwierigkeitsgrad, etc..

Die Vor- und Nachteile beider Ansätze wurden ausführlich diskutiert. Als Ergebnis wurde beschlossen, den Ansatz von Dublin Core als Grundlage zu wählen und darin Teile des IMS-Modells zu integrieren. Hierfür sprachen hauptsächlich folgende Gründe:

- Die Offenheit des Dublin Core Ansatzes gestattet eine spezifische Auswahl von Attributen.

- Die volle Komplexität des LOM Schemas kann in der Regel bei der Erschließung nicht ausgenutzt werden.
- Weite Teile des LOM Schemas lassen sich in Dublin Core abbilden.
- Die bibliographische Beschreibung nach Dublin Core ist im Bibliotheksbereich weit verbreitet.

Aufbauend auf dem Metadatensatz von Dublin Core wurden die für META-AKAD erforderlichen Erweiterungen vorgenommen. Dabei wurde insbesondere auch der Stand bei verwandten Projekten berücksichtigt, hierunter MathNet, Merlot, RENARDUS, ARIADNE, European School Net, Deutscher Bildungsserver und DLmeta. Darüber hinaus wurden aber auch gänzlich neue Elemente in das META-AKAD Schema aufgenommen. Diese betreffen vornehmlich den lehrspezifischen Teil der Metadaten und die Beschreibung der Gutachten.

Für den herausgearbeiteten Metadatensatz wurde ein Strukturmodell entwickelt, in dem die einzelnen Elemente ähnlich wie bei IMS/LOM in hierarchischer Form angeordnet sind. Die Darstellung basiert auf dem Qualifikationsmechanismus von Dublin Core; es werden bis zu drei Hierarchiestufen verwendet. Das Strukturmodell dient als Grundlage für das Datenmodell von META-AKAD wie auch für die Abbildung externer Schemata auf das eigene Modell. Letzteres ermöglicht die Erfassung und Ausgabe von Metadaten anderer Standards, die über eine Schnittstelle zur Datenbank verwaltet werden soll. Es wurde daher besonders auf Kompatibilität mit den bestehenden Standards geachtet. In diesem Zusammenhang wurde auch eine Abbildung der META-AKAD Attribute auf das Schema von Dublin Core definiert.

Desweiteren wurde für den Metadatensatz eine inhaltliche Beschreibung ausgearbeitet, die sich ebenfalls an gegebenen Standards orientiert. Darin sind die Bedeutung und intendierte Verwendung der Metadaten-Elemente dargestellt, in Frage kommende Werte durch Angabe von Wertemengen spezifiziert und für die Werte deren Multiplizität und Ordnung angegeben. Für die Festlegung der Werte-Schemata wurde nach vorhandenen Standards recherchiert, wie bei Datumsangaben, Formaten und Sprachangaben. Solche Standards wurden weitgehend übernommen. In einigen Fällen wurden für die Attribute aber auch eigene Vokabularien aufgestellt. Dies betrifft wiederum in erster Linie Attribute aus dem lehrspezifischen Bereich, wie Studienfächer, Lernkategorien, Zielgruppen, aber auch Dokument- und Medientypen.

Zur Identifizierung von Lehr-/Lernmaterialien unter den Netzpublikationen beim Sammeln wurde ein spezielles Vokabular ausgearbeitet. Dieses soll bei der Suche mit Stichwörtern aus den Dokumenten verglichen werden. Das Suchvokabular umfasst charakteristische Dokumententypen für Lehrmaterialien und weitere didaktische Kategorien.

Für die Sacherschließung in META-AKAD wurde eine spezielle Teilstruktur von Attributen definiert, die den Projektzielen in diesem Bereich angepasst sind. Über diese Attribute kann die Quelle von Schlagworten und Klassifikationen angegeben werden, und es kann zusätzlich auch vermerkt werden, ob deren Zuteilung innerhalb des Projekts erfolgte, und zwar durch intellektuelle bzw. automatische Erschließung, oder ob sie beim Sammeln externer Metadaten übernommen wurden. Die aufgeführte Liste der Quellen umfasst die Schlagwortnormdatei, die Regensburger Verbundklassifikation, die Mathematical Subject Classification MSC 2000, das Physics and Astronomy Classification Scheme und die Dewey Decimal Classification, die Liste kann jedoch nach Bedarf erweitert werden.

Als Ergebnis dieses Arbeitspakets liegt unter dem Titel „META-AKAD Metadata Element and Structure“ eine Metadaten-Definition in Form einer (natürlichsprachlichen) strukturierten Beschreibung vor (siehe Anhang; vgl. dort auch die „Kontrollierten Vokabularien der Metadatenelemente“). Auf der Basis dieser Definition wurde eine formale Spezifikation des Metadaten-Schemas als XML-Schema ausgearbeitet, das die Struktur einer XML-Repräsentation der Metadaten exakt beschreibt. Die Verarbeitung der Metadaten im META-AKAD-System erfolgt zu einem großen

Teil in Form von XML-Dokumenten, wobei das Schema als integraler Bestandteil der Software zum Einsatz kommt. Dies ist ausführlicher in Kap. II.5.1 auf Seite22 beschrieben. Das „XML-Schema der META-AKAD-Metadaten“ und eine „Dokumentation des XML-Schemas der META-AKAD-Metadaten“ finden sich im Anhang.

Das Metadaten-Schema von META-AKAD weist diejenigen Attribute aus, die bei der Dokument-Erschließung zur Beschreibung der Dokumente verwendet werden können. Diese dienen der Anzeige der Datensätze nach außen und können, unterlegt mit entsprechender Funktionalität, Sucheinstiege für den Endbenutzer bereit stellen. Neben der externen Repräsentation der Dokumente ist eine system-interne Spezifikation von Benutzer- und Verwaltungsdaten erforderlich. Hierzu wurde in Verbindung mit AP4 ein Konzept entwickelt, das u.a. Angaben zur Benutzerverwaltung, über den Stand der Bearbeitung von Dokumenten im Erschließungsprozess, zur Historie und Versionsverwaltung von Dokumenten und zu deren Verfügbarkeit umfasst. Hier sind auch Attribute für redaktionelle Bemerkungen und interne Beurteilungshinweise vorgesehen.

II.3.2. Evaluation des Prototyps, internationale Abstimmung

Die internationale Diskussion der Metadaten-Initiativen wird laufend verfolgt. Insbesondere wird derzeit der didaktische Teil der Metadaten weiterer externer Schemata ausgewertet, und es werden diesbezüglich Vorschläge zur Ergänzung der Metadaten von META-AKAD ausgearbeitet. Als zunehmend wichtig erscheinen zudem die in der Entwicklung befindlichen Metadaten-Schemata *Electronic Resource Citation* (*ERC*)[18] und *Archival Resource Key* (*ARK*)[19]. Metadaten anderer Quellen sollen bei der Datenaufnahme über eine Schnittstelle zur Datenbank verwaltet werden. Hierfür werden die erforderlichen Abbildungen der externen Schemata (vornehmlich DC, IEEE/LOM und OAI) auf unser Modell weiter spezifiziert.
Um die Nutzung der Nachweise von Lehr- und Lernmaterial in META-AKAD für einen möglichst großen Benutzerkreis zu erleichtern, soll das erstmalige Auffinden des META-AKAD-Systems und der erarbeiteten Metadaten von außerhalb gefördert werden. Dazu werden die Metadaten auch in Formen angeboten werden, die die Entdeckung durch externe Suchsysteme fördern. So werden die erarbeiteten Metadaten in Form von Meta-Tags, die den gebräuchlichen, internationalen Standards (insb. Dublin Core) entsprechen, in die Webseiten der Endbenutzerschnittstelle eingebettet werden. Diese Metadaten-Auszeichnung wird die gezielte Suche nach akademischem Lehr- und Lernmaterial mit Meta-Tag-auswertenden Suchmaschinen von Nutzern überall im Internet erleichtern. Außerdem wird das META-AKAD-System eine Schnittstelle nach dem *Open Archives Protocol for Metadata Harvesting* (OAI-PMH)[20] erhalten. Diese Schnittstelle wird an der zentralen Nachweisstelle der *Open Archives Initiative* (OAI)[21] als OAI Data Provider registriert werden. Mit der Registrierung stellt das META-AKAD-System zusammen mit zahlreichen namhaften digitalen Archiven aus der ganzen Welt seine Metadaten für die strukturierte Suche über die OAI-Schnittstelle durch sogenannte Service Provider zur Verfügung. Solche Service Provider können in den Metadaten einzelner oder auch simultan mehrerer Data Provider suchen. Ihren Benutzern können die Service Provider alle Vorteile eines Gateways mit einer einheitlichen Suchmaske für unterschiedlichste Sammlungen, Kataloge und Archive bieten. Wir erwarten, daß META-AKAD durch die OAI-Schnittstelle wesentlich mehr Benutzer aus einem auch geographisch sehr weiten Kreis haben wird. Mit der Implementierung der OAI-Schnittstelle kommen wir einer der wesentliche Forderungen der Deutschen Initiative für NetzwerkInformation (DINI)[22] nach. Die Struktur der Metadaten in META-AKAD erlaubt durch die Berücksichtigung des Dublin-Core-Standards die einfache Übernahme auch für die OAI-Schnittstelle. Das OAI-PMH erlaubt es, Metadaten nach alternativen Standards anzubieten. Voraussichtlich werden von META-AKAD zwei Metadatenformate über die OAI-Schnittstelle

18 <http://jodi.ecs.soton.ac.uk/Articles/v02/i02/Kunze/>

19 <http://www.ietf.org/internet-drafts/draft-kunze-ark-06.txt>
20 <http://www.openarchives.org/OAI/openarchivesprotocol.html>
21 <http://www.openarchives.org/Register/BrowseSites.pl >
22 <http://www.dini.de/dokumente.php>

angeboten werden: OAI-Dublin-Core mit einem Mindestmaß an Metadaten für größtmögliche Kompatibilität zwischen Systemen und zusätzlich das META-AKAD-eigene Format zur bestmöglichen Beschreibung des erfaßten Lehr- und Lernmaterials.

II.4. Arbeitspaket 3: Bewertung des Lehr- und Lernmaterials

Ein spezifisches Problem bei online verfügbaren Ressourcen ist das Fehlen einer Qualitätssicherung, wie sie bei gedruckten Medien im allgemeinen und in der wissenschaftlichen Literatur in Form des Review-Prozesses besteht. Aus diesem Grund sollte im Projekt Meta-Akad ein eigenes Gutachtersystem entwickelt werden. Dazu gehören die Aspekte der inhaltlichen Konzeption, des organisatorischen und technischen Aufbaus.

In technischer und organisatorischer Hinsicht wird dieses System als Teil der Redaktionsschnittstelle für den Erschließungsworkflow umgesetzt. Dieses wird in AP 4 näher beschrieben.

Ein Review bestehender Verfahren zur Qualitätsbeurteilung von Onlineressourcen ergab zwei wesentliche Aspekte der Qualität: die inhaltliche Qualität und die didaktische Qualität. Hinzu genommen wurde der Aspekt der Benutzerfreundlichkeit, der besonders bei interaktiven Lernanwendungen und Hypertexten eine Rolle spielt. Alle drei sind relativ komplexe Merkmale, die sich nur ungenügend mit einer einzigen Skala erfassen lassen. Um zu einer möglichst objektiven und reliablen Erfassung zu kommen, lag es nahe, die komplexen Merkmale in einfachere zu zerlegen. Die folgenden Abschnitte geben einen Überblick über das Gutachtersystem, eine detailliertere Darstellung findet sich im Anhang unter „Qualitätsbeurteilung von Web-Lernobjekten: Erweiterung der Metadatendefinition“, dort wird das System auch anhand eines skizzierten Fragebogen konkretisiert und veranschaulicht.

Der mehrstufige Auswahlprozeß zum Auffinden von Lehr- und Lernmaterial und anschließenden Aussondern ungeeigneten Materials bewährte sich. Es bestehen jedoch gewisse Schwierigkeiten, ausreichend Fachwissenschaftler als Gutachter zu gewinnen, da sich die strukturierte Begutachtung des Materials nach inhaltlichen, didaktischen und Usability-Kriterien als recht aufwendig erwies.

Das Gutachtersystem konnte bisher nicht implementiert werden, da die verzögerte Fertigstellung des Redaktionssystems Vorrang besaß. Die Implementierung wird unmittelbar nach dem Ende der Arbeiten am Redaktionsysytem erfolgen. Aufgrund der, verglichen mit dem Redaktionssystem, sehr viel einfacheren Struktur wird eine zügige Fertigstellung des Gutachtersystems erwartet.

Die bisher vorhandenen Bewertungen konnten noch nicht in die Datenbank eingespielt werden.

II.4.1. Inhaltliche Qualität

Grundvoraussetzung für die Eignung einer Ressource als Lernmaterial ist die sachliche Korrektheit des Inhalts. Dieses Merkmal setzt sich aus zumindest den folgenden Teilmerkmalen zusammen:

1. Die Argumentation ist nachvollziehbar und die Schlussfolgerungen logisch korrekt.
2. Die Sachverhalte sind sachlich einwandfrei dargestellt und gelten in der Disziplin als gut belegt bzw. etabliert.
3. Die verwendeten Begriffe entsprechen dem in der Disziplin üblichen Vokabular.
4. Historische oder nach heutiger Lehrmeinung überholte Positionen werden in der Ressource eindeutig gekennzeichnet bzw. kritisch diskutiert.

Eine „abgekürzte“ Beurteilung ist mit den folgenden beiden Kriterien möglich:

- Der Urheber der Ressource kann als anerkannter Vertreter seiner Disziplin gelten.
- Die Ressource hat anderweitig bereits einen genügend strengen Reviewprozess durchlaufen.

Die Beurteilung wird in Form eines Ratings sowie eines Beurteilungstextes vergeben.

Zusätzlich zur inhaltlichen Qualität werden noch zwei weitere Merkmale zum inhaltlichen Umfang der Ressource erfasst. Diese sind nicht im engeren Sinne eine Qualitätskontrolle, sondern beziehen sich auf den Lernzweck, für den die Ressource geeignet ist (schneller Überblick über ein Thema *vs*. Vertiefung). Diese beiden Merkmale sind:

- die inhaltliche Tiefe
- die inhaltliche Breite

Für die Benutzer ergibt sich mit diesen Merkmalen die Möglichkeit, gezielt nach Dokumenten zu suchen, die ihrem Lernbedürfnis entsprechen, z.B. einen Überblick über einen Themenbereich zu bekommen (große inhaltliche Breite) oder ein bestimmtes Thema zu vertiefen (inhaltliche Tiefe).

II.4.2. Didaktische Qualität

Die Besonderheit bei der Bestimmung der didaktischen Qualität liegt darin, dass sich eine didaktische Eignung nicht per se, sondern nur in Bezug auf einen bestimmten Lernkontext bestimmen lässt. Daher wurde die Beurteilung der didaktischen Qualität in zwei Schritte gefasst. Im ersten Schritt legt der Beurteiler an Hand mehrerer Kriterien den Lernkontext fest, für den eine Ressource ihm besonders geeignet erscheint. Diese Kriterien sind:

- die Eignung für bestimmte Zielgruppen in Hinblick auf den Schwierigkeitsgrad
- die Lernsituationen, in der die Ressource sinnvoll angewendet werden kann
- das Vorwissen, das beim Lernenden zum guten Verständnis vorhanden sein muss
- die Lernziele, die mit der Ressource erreicht oder unterstützt werden können

Zielgruppen und Lernsituationen wählt der Gutachter aus einer vorgegebenen Liste aus. Vorwissen und Lernziele werden dagegen als Text erfasst.

Im zweiten Schritt beurteilt der Gutachter die didaktische Qualität, wobei angehalten ist, sich explizit auf die vorher identifizierten Kontexte zu beziehen. Das Urteil wird auch hier als Rating und als Text vergeben.

II.4.3. Benutzerfreundlichkeit

Die Benutzerfreundlichkeit ist ebenfalls ein komplexes Merkmal. Hinzu kommt, dass nur wenige Gutachter in diesem Bereich eine ausreichende Erfahrung haben dürften. Aus diesem Grund wurden auch hier detaillierte Teilkriterien konstruiert, die somit leichter zu beurteilen sind. Diese Kriterien sind angelehnt an die zehn Usability Heuristiken von Jacob Nielsen [Nie93]. Die hier umgesetzten Heuristiken waren:

- Sichtbarkeit des Systemzustands
- Benutzerkontrolle und Freiheit
- Konsistenz und Standards
- Vermeidung von Fehlern
- Ästhetik und minimalistisches Design
- Hilfe und Dokumentation

Die Beurteilung der Benutzerfreundlichkeit ist im übrigen nur bei interaktiven Anwendungen und Hypertexten sinnvoll und wird demnach nur bei solchen Ressourcen vergeben. Die Beurteilung erfolgt ebenfalls als Rating und als Text.

II.5. Arbeitspaket 4: Entwicklung und prototypische Realisierung eines DB-gestützten, Web-basierten Informationssystems

II.5.1. Dokumentenverwaltung

Die Suche nach Informationen im Internet liefert immer wieder Suchergebnisse von extrem schlechter Qualität, weil die heterogenen Dokumente weder systematisch nach ihrem Inhalt erschlossen wurden und kaum durch geeignete Metadaten beschrieben sind, noch die Suchmaschinen effektive Verfahren anwenden. Sie benutzen meist nur Schlüsselwort-basierte Suche und setzen vor allem keine Domänen-spezifischen Ontologien oder Klassifikationsschemata ein, um beispielsweise semantische Ähnlichkeit in den Griff zu bekommen. Um wesentlich bessere Suchergebnisse zu erzielen, muss zunächst mehr Aufwand in die inhaltliche Erschließung der Dokumente gesteckt werden, damit später dann Verfahren der Ähnlichkeitssuche (nach inhaltlichen und strukturellen Kriterien) gewinnbringend eingesetzt werden können. Deshalb betrachten wir das Problem der Dokumentensuche in einem größeren Zusammenhang und wollen uns dabei wiederum auf die Aspekte der Datenverwaltung konzentrieren.

Datenmodell

Es sind folgende verschiedenen Arten von Daten auf jeweils spezifische Weise zu verwalten. Beschreibende Daten (auch Metadaten genannt) folgen einem speziell zu entwickelnden Metadatenmodell. Ein Metadatensatz wird genau einem Dokument zugeordnet und die enthaltenen Attributwerte sind das Ergebnis der Erschließung dieses Dokuments. Neben beschreibenden Daten müssen die zugehörigen Dokumente selbst repräsentiert werden. Dies kann einerseits in Form von Referenzen (URLs) geschehen; es kann aber auch zusätzlich eine direkte Erfassung von Dokumenten mit anschließender Verwaltung von Quelldokumenten durch das Datenverwaltungssystem vorgesehen werden. Als dritte Gruppe von Daten sind die Personen- bzw. Organisations-bezogenen Daten zu nennen. Dies sind im wesentlichen Daten zu den verschiedenen Benutzergruppen, ihren Rollen (Autoren, Gutachter, Lernende usw.), Rechten/Pflichten und ihrer Einbindung in gewisse Organisationsstrukturen (Arbeitsgruppen, Fachbereiche, Institute usw.). Ebenfalls wichtig sind die ablaufbezogenen Daten, die Instanzen und Zustände von mehrschrittigen Abläufen („Workflows“) enthalten, sofern die Schemata dieser Abläufe an der Systemschnittstelle sichtbar sind und den Benutzern ihre Einbindung in solche Vorgänge klar ist.

Der Umfang an Attributen zur Beschreibung der Metadaten wurde für den Einsatz von XML in eine entsprechende formale Beschreibung der Struktur überführt. Die Verwendung von XML legt die Spezifikation der Struktur durch eine Document Type Definition (DTD[23]) oder durch ein XML-Schema-Dokument[24] nahe. XML-Schema erlaubt eine sehr differenzierte Spezifikation der Struktur und der verwendeten Datentypen innerhalb eines XML-Dokuments. Während eine DTD nur sehr einfache Beschreibungskonzepte anbietet, erlaubt XML-Schema beispielsweise den Einsatz von Abstraktionskonzepten wie die Spezialisierung (Vererbung). Da ein XML-Schema-Dokument allerdings ungleich schwieriger zu lesen ist als eine DTD und hier zunächst allein die Struktur und das Aussehen eines dieser Struktur entsprechenden XML-Dokuments wichtig ist, beschränken wir uns zunächst auf die Betrachtung von DTDs im Rahmen der Realisierung des prototypischen Systems. Im endgültigen System allerdings finden XML-Schema-Dokumente Einsatz.

Repräsentation der Metadaten durch XML

Es wurde bisher von Metadatensätzen und ihren Attributen gesprochen. Ein Metadatensatz beschreibt dabei eine Web-Ressource mit Hilfe einer Menge von Attributen, die in unserem Ansatz

23 <http://www.w3.org/xml>

24 <http://www.w3.org/xml-schema>

an den Dublin-Core-Standard[25] angelehnt sind. Das Attribut „Identifier“ beispielweise beinhaltet die URL des entsprechenden Lehr-/Lernangebots. Weitere Attribute wie „Creator“ oder „Format“ geben Auskunft über den Autor oder das verwendete Format. Ein diese Strukturen repräsentierendes XML-Dokument könnte wie in Abb. 10 aussehen.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Written by Marcus Flehmig "flehmig@informatik.uni-kl.de" -->
<metabase>
  <learningresource guid="K480000038">
    <title>
      <main origin="external">Vektoranalysis</main>
    </title>
    <creator origin="external">
      <name>Trautmann, Günther</name>
      <organization>Universität Kaiserslautern, Kaiserslautern, DE</organization>
    </creator>
    <subject scheme="freetext" origin="external" xml:lang="de">Vektoranalysis</subject>
    <date> <unspecified format="yyyy">1999</unspecified></date>
    <type>
      <document><vorlesungsskript/></document>
      <media><text/> </media>
    </type>
    <format>
      <mimetype>application/postscript</mimetype>
      <encoding>application/x-gzip</encoding>
    </format>
    <identifier href="http://www.mathematik.uni-kl.de/~wwwagag/preprints/skripte/Vektoranalysis.ps.gz"/>
    <language scheme="RFC1766">de</language>
    <educational>
      <verified>true</verified>
      <fieldofstudy><mathematik/></fieldofstudy>
    </educational>
    <annotations xml:lang="de">Kurzskript</annotations>
  </learningresource>
</metabase>
```

Abbildung 10: Beispiel einer XML-Repräsentation.

Wie man in der Abbildung Abb. 10 auch leicht sehen kann, werden die Metadaten-Attribute auf XML-Elemente abgebildet. Angaben über den Ursprung („Origin“) des Inhalts eines XML-Elements bzw. Angaben zum Format oder Schema („Scheme“) werden unter Einsatz von XML-Attributen gemacht. Das mit XML verbundene semi-strukturierte Datenmodell erlaubt eine sehr intuitive Verwendung der Metadaten-Attribute. Bei der Erschließung eines Lehr-/Lernangebots können nicht in jedem Fall alle Metadaten-Attribute sinnvoll ausgefüllt werden, da es vorkommen kann, dass nicht alle Informationen verfügbar sind. Aber auch Mehrfachnennungen, bei denen nicht von Vorneherein die Kardinalitäten bekannt sind, kommen sehr häufig vor. So können beispielsweise beliebig viele Autoren („Creator“) an dem Verfassen eines Artikels beteiligt gewesen sein. Diese Freiheitsgrade können durch eine Strukturbeschreibung entsprechend sinnvoll eingeschränkt werden. Einen Auszug aus der verwendeten DTD findet man in Abbildung Abb. 11. Man erkennt, dass in der Datenbank („Metabase“) Lehr-/Lernangebote („Learning Resource“), Gutachten („Peer Review“) oder Benutzerkommentare („User Comment“) vorkommen können. Ein Lehr-/Lern-

25 <http://www.dublincore.org>

angebote lässt sich vielfältig mittels weiterer XML-Elemente beschreiben. Dabei ist es z. B., wie man anhand der DTD sehen kann, notwendig, einen Titel anzugeben, aber es können beliebig viele Autoren genannt werden, was durch den Asteriskus (*) hinter dem Elementnamen ausgedrückt wird. Mit Fragezeichen gekennzeichnete Elemente sind optional.

Jedes erfasste Lehr-/Lernangebot wird zunächst durch ein eigenes XML-Dokument repräsentiert. Allerdings können Lehr-/Lernangebote auch in Beziehung („Relation“) zueinander stehen. Die Definition der Metadaten-Attribute sieht verschiedene Arten von Beziehungen vor: Version, Teil („Part“), Format, sowie Verweise auf Gutachten („Review“) und Benutzerkommentare („User Comment“). Besteht beispielsweise ein Lehr-/Lernangebote aus verschiedenen Kapiteln, so können für jedes Kapitel eigene Einträge vorgesehen werden. Diese miteinander verknüpften Einträge werden durch ein einziges XML-Dokument repräsentiert, wobei die Verknüpfung durch XML-Referenzen realisiert wird. So können Strukturen (Beziehungen) sehr einfach gehandhabt und visualisiert werden, d. h. zusammengehörige Daten können gemeinsam durch nur eine Anfrage angezeigt werden. Auch das Erfassen von neuen Einträgen wird so optimal unterstützt. Es können zusammengehörige Daten gemeinsam ohne Kommunikationsaufwand mit der Datenverwaltungskomponente erfasst werden und als Ganzes zur weiteren Verarbeitung an die Datenbankkomponente geschickt werden. Um die Definition der XML-Dokumente bei der Erschließung weiter zu vereinfachen, können Referenzen auch an Ort und Stelle („inline“) definiert werden, d. h. ohne die Notwendigkeit, einen eigenen Eintrag für das Lehr-/Lernangebote zu erfassen. Allerdings geht damit die Möglichkeit verloren, zusätzlich einen Autor oder ähnliche Metadaten-Attribute für die referenzierten Einträge spezifizieren zu können. Aber so wird sehr einfach das Problem umgangen, lokal und Client-seitig bereits eindeutige Identifikatoren definieren zu müssen.

```
<!ENTITY % SubjectScheme "(SWD|DDC|MSC2000|PACS|RVK|freetext)">

<!ELEMENT  metabase (learningresource+, peerreview*, usercomment*)>
<!ELEMENT  learningresource (title, creator*, subject*, description?, publisher*,
           contributor*, date?, type?, format?, identifier*, source?, language*,
           relation?, coverage?, rights?, educational?, audience?, annotations?, image?)>
<!ATTLIST  learningresource guid ID #REQUIRED >
<!-- ============ Titel ================================== -->
<!ELEMENT  title (main+, alternative*, version?)>
<!ELEMENT  main (#PCDATA)>
<!ATTLIST  main origin %OriginEnum; #REQUIRED
           xml:lang CDATA "en">
<!ELEMENT  alternative (#PCDATA)>
<!ATTLIST  alternative origin %OriginEnum; #REQUIRED
           xml:lang CDATA "en">
<!ELEMENT  version (#PCDATA)>
<!ATTLIST  version origin %OriginEnum; #REQUIRED
           xml:lang CDATA "en">
<!-- ============ Autoren ================================== -->
<!ELEMENT  creator (name, email*, organization*)>
<!ATTLIST  creator origin %OriginEnum; #REQUIRED>
<!ELEMENT  name (#PCDATA)>
<!ELEMENT  email (#PCDATA)>
<!ELEMENT  organization (#PCDATA)>
<!-- ============ Thema ================================== -->
<!ELEMENT  subject (#PCDATA)>
<!ATTLIST  subject scheme %SubjectScheme; #REQUIRED
           origin %OriginEnum; #REQUIRED
           xml:lang CDATA "de">
```

Abbildung 11: Auszug aus einer DTD

```
<metabase>
    <learningresource guid="K480001276a">
    <title><main origin="external">Cosmology: Mankind s Grand Investigation</main></title>
    <creator origin="external">
      <name>Bothun, Greg</name>
      <organization>University of Oregon, Eugene, US</organization>
    </creator>
    <subject scheme="freetext" origin="external" xml:lang="de"> Kosmologie </subject>
    <subject scheme="freetext" origin="external" xml:lang="de"> Weltall </subject>
    <subject scheme="freetext" origin="external" xml:lang="de"> Außerirdisches Leben </subject>
    <identifier href="http://zebu.uoregon.edu/hb/c1w.html"/>
    <relation>
      <haspart><reference guid="K480001276b" /></haspart>
      <haspart><reference guid="K480001276c" /></haspart>
    </relation>
  </learningresource>
  <learningresource guid="K480001276b">
    <title><main origin="external">Cosmology: Mankind s Grand Investigation</main></title>
    <creator origin="external">
      <name>Bothun, Greg</name>
      <organization>University of Oregon, Eugene, US</organization>
    </creator>
    <subject scheme="freetext" origin="external" xml:lang="de"> Kosmologie </subject>
    <subject scheme="freetext" origin="external" xml:lang="de"> Weltall </subject>
    <subject scheme="freetext" origin="external" xml:lang="de"> Außerirdisches Leben </subject>
    <identifier href="http://zebu.uoregon.edu/hb/c2w.html"/>
  </learningresource>
  <learningresource guid="K480001276c">
    <title><main origin="external">Cosmology: Mankind s Grand Investigation</main></title>
    <creator origin="external">
      <name>Bothun, Greg</name>
      <organization>University of Oregon, Eugene, US</organization>
    </creator>
    <subject scheme="freetext" origin="external" xml:lang="de"> Kosmologie </subject>
    <subject scheme="freetext" origin="external" xml:lang="de"> Weltall </subject>
    <subject scheme="freetext" origin="external" xml:lang="de"> Außerirdisches Leben </subject>
    <identifier href="http://zebu.uoregon.edu/hb/c3w.html"/>
  </learningresource>
</metabase>
```

Abbildung 12: Beispiel einer Relation.HasPart-Beziehung

Ein Beispiel soll dies nun verdeutlichen: ein interessantes Skript aus dem Bereich Physik soll erschlossen werden. Es besteht aus mehreren Kapitel (hier drei) und das erste Kapitel enthält Verweise auf die Übrigen. Entsprechend obiger Ausführungen werden drei Einträge vom Type „Learning Resource“ definiert und miteinander verknüpft (siehe: Relation.HasPart.Reference-Element in Abb. 12). Müsste man bereits bei der initialen Erschließung Client-seitig die in Abb. 12 gezeigte Struktur erzeugen und an die Datenbankkomponente schicken, so ergäben sich eine Menge vermeidbarer Redundanzen und damit auch an eine Menge potenzieller Fehlerquellen. Daher ist für die initiale Erschließung eine Vereinfachung vorgesehen. In Abb. 13 findet man ein Beispiel für diese Vereinfachung, das die bereits genannte Inline-Definition verwendet. Bei dieser Variante werden in dem Relation.hasPart-Element des referenzierenden Eintrags nur die zusätzlichen Elemente definiert, die für die einzelnen Teile unterschiedlich sind.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<metabase>
  <learningresource guid="K480001276">
    <title><main origin="external">Cosmology: Mankind s Grand Investigation</main></title>
    <creator origin="external">
      <name>Bothun, Greg</name>
      <organization>University of Oregon, Eugene, US</organization>
    </creator>
    <subject scheme="freetext" origin="external" xml:lang="de">Kosmologie</subject>
    <subject scheme="freetext" origin="external" xml:lang="de">Weltall</subject>
    <subject scheme="freetext" origin="external" xml:lang="de">Außerirdisches Leben</subject>
    <identifier href="http://zebu.uoregon.edu/hb/c1w.html"/>
    <relation>
      <haspart>
        <inlineresource><identifier href="http://zebu.uoregon.edu/hb/c2w.html" /> </inlineresource>
      </haspart>
      <haspart>
        <inlineresource><identifier href="http://zebu.uoregon.edu/hb/c3w.html" /> </inlineresource>
      </haspart>
    </relation>
  </learningresource>
</metabase>
```

Abbildung 13: Beispiel einer Inline-Definition

Das Dokument aus Abb. 13 wird an die Datenbankkomponente geschickt und erst dort zu der Struktur aus Abb. 12 expandiert. Es befinden sich keine Inline-Definitionen in der Datenbank, so dass ein Pfadausdruck in einer Suchanfrage immer eindeutig ist. Die Werte des referenzierenden Eintrags werden übernommen. Bei Verwendung der Inline-Variante können ferner auch Angaben über das Format, den Dokument- bzw. Medien-Typ sowie auch zum Lehrmaterial gemacht werden. Die Möglichkeiten beschränken sich aber auf die Definition von Ein-Ebenen-Beziehungen, komplexe Strukturen lassen sich derart nicht definieren. Auch in dem Fall, dass jedem Eintrag beispielsweise ein eigener Autor zugewiesen werden soll, ist die Variante aus Abb. 12 zu wählen. Einen entsprechenden Auszug aus der DTD findet sich in Abb. 14.

```
<!-- ============= Verweise ========== ========== ========== ======= -->
<!ELEMENT relation (hasVersion|hasPart|hasFormat|hasReview|hasUserComments)+ >
<!ELEMENT hasVersion (reference|inlineresource) >
<!ELEMENT hasPart (reference|inlineresource) >
<!ELEMENT hasFormat (reference|inlineresource) >
<!ELEMENT hasReview (reference) >
<!ELEMENT hasUserComments (reference) >
<!ELEMENT reference EMPTY >
<!ATTLIST    reference guid IDREF #REQUIRED>
<!ELEMENT inlineresource (title?, type?, format, identifier, educational?, annotations?)>
```

Abbildung 14: Strukturbeschreibung (DTD) der Inline-Definition.

Verarbeitungsmodell

In der bisherigen Diskussion wurde immer wieder von XML-Elementen und XML-Dokumenten gesprochen. In Bezug darauf sind zwei verschiedenartige Verarbeitungsmodelle vorstellbar:

- ein Element-basiertes und
- ein Dokument-basiertes Verarbeitungsmodell.

Die Element-basierte Sichtweise betrachtet ein Element (also z. B. Autor oder Titel) als Granulat der Verarbeitung. Eine definierte Menge von Elementen bildet einen Metadatensatz und der Zustand der Erschließung ergibt sich allein aus der Menge der belegten Elemente. Leere Elemente können als Symbol für eine vorgenommene Eintragung dienen. Jedes Element kann für sich verwaltet, verändert und auch gesperrt werden. Auch die Autorisierung geschieht auf der Ebene der Elemente. Automatisch ausgefüllte bzw. gesammelte Elemente müssen alle einzeln verifiziert werden und für eine weitere externe Benutzung freigegeben werden. Diese feingranulare Verarbeitung ist aber in der Realisierung sehr aufwändig, teuer und in Betracht der zu realisierenden Anwendungen nicht sinnvoll. Aufgrund der eher einfachen Struktur der zugrundeliegenden Prozesse und damit vor allem aufgrund des grundlegenden MAS-Mehrschrittvorgangs erscheint ein gröberes Granulat der Verarbeitung sinnvoll. Wie es sich bereits im vorangegangen Abschnitt abgezeichnet hat, sind XML-Dokumente mit der bereits beschriebenen Struktur als Verarbeitungseinheit zu präferieren. Dokumente tragen dabei allerdings keine Verarbeitungsanweisungen oder -informationen, sondern repräsentieren allein die operationalen Daten. Ein weiteres Konzept sah vor, die jeweilige Operation auf diese Daten durch eine entsprechende WebDAV-Methode (Web-based Distributed Authoring and Versioning[26]) zu definieren, wobei, die im vorangegangen Abschnitt beschriebenen Dokumente, Parameter eben dieser Methoden sein könnten. Es können beispielsweise mittels WebDAV Sperren für ein Dokument gesetzt werden und so konkurrierende Zugriffe behandelt werden. Änderungen am Inhalt eines Dokuments können durchgeführt werden und nach der Bearbeitung kann das Dokument wieder an die Datenverwaltungskomponente zurückgeschickt werden. Dort würde das Dokument von einem WebDAV-Server verarbeitet und die enthaltenen Daten können in einem objekt-relationalen Datenbanksystem gespeichert werden.

Dokumenten-basierte Verarbeitung mit WebDAV

Die HTTP-Erweiterung WebDAV als Protokoll bzw. Schnittstelle zu der Datenverwaltungskomponente einzusetzen, besaß den Vorteil, schon sehr früh mit der Entwicklung weiterer Komponenten (z. B. Redaktionssystem) beginnen zu können, die auf der Funktionalität der Datenverwaltungskomponente aufbauen, ohne dass diese bereits vollständig realisiert sein muss. Neben der Schnittstelle für die Definition der möglichen Operationen, war mit der Beschreibung der Abbildung der Metadaten-Attribute auf XML ein konkretes Datenmodell gegeben, das dem Austausch der zu verwaltenden Daten dienen kann. Um nicht einen vollständigen WebDAV-Server realisieren zu müssen, wurde ein Framework ausgewählt, das an den notwendigen Stellen erweitert wurde. WebDAV kann auch vereinfacht als Web-basiertes Dateisystem bezeichnet werden, das neben HTTP-basierten Leseoperationen auch Schreiboperationen und den Einsatz von Sperren auf der Ebene von Dokumenten erlaubt.

II.5.2. WebDAV-basierte prototypische Realisierung mit Slide

Eine Realisierung des WebDAV- Standards stellt das Slide-Projekt[27] der Apache-Jakarta-Gruppe[28] dar. Slide besteht aus mehreren Server- und Client-Komponenten, zu deren Integration WebDAV verwendet und als genanntes Framework eingesetzt wird. Dieses Framework wird um eine spezialisierte Web-Anbindung zur Realisierung der Ablaufkoordination, um spezielle Helper-Komponenten für die Bereitstellung der Metadaten in XML sowie einer OR-basierten Data-Store-Implementierung ergänzt.

26 <http://www.webdav.org>

27 <http://jakarta.apache.org/slide>

28 <http://jakarta.apache.org>

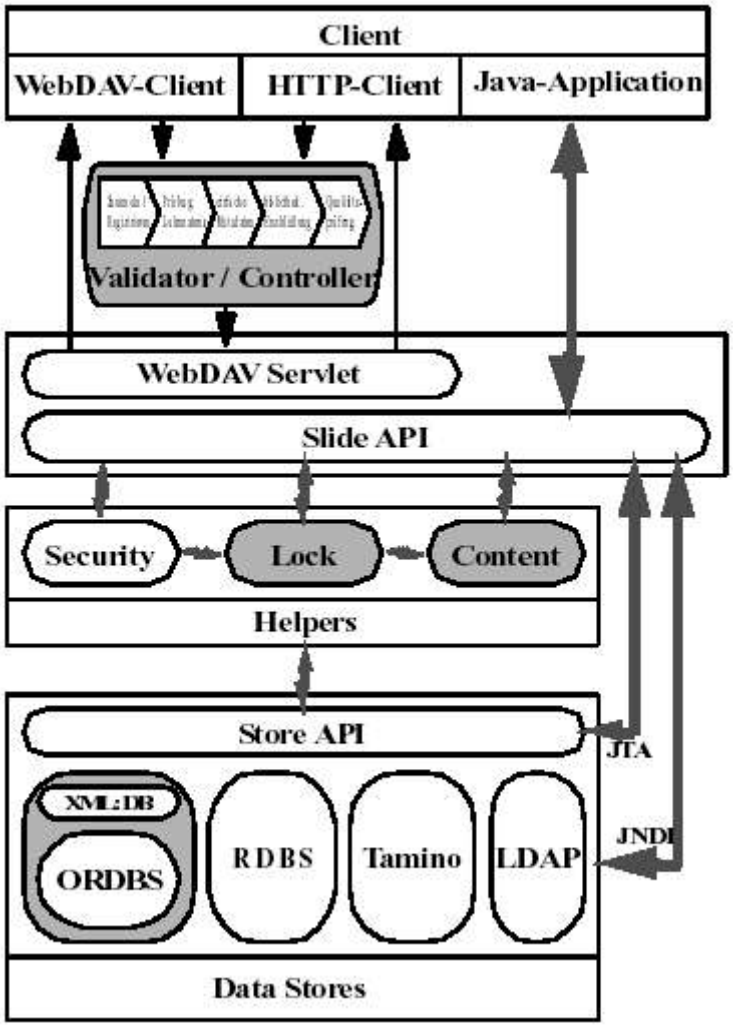

Abbildung 15: Slide/WebDAV-basierter Architekturentwurf

Eine Übersicht über die zugrundeliegende Systemarchitektur findet man in Abb. 15 . Die grau hinterlegten Komponenten markieren die Stelle, an denen eine Anpassung notwendig ist. Neben der Erweiterung durch die Realisierung zusätzlicher Komponenten kommt aber bei dem Einsatz eines Frameworks auch der Konfiguration (hier: das WebDAV-basierte Content-Management-System Slide) eine wichtige Bedeutung zu. Zu diesem Zweck wurden zunächst die verschiedenen Rollen identifiziert, deren möglichen Operationen in den verschiedenen Stufen des Erschließungsprozesses bestimmt und die Rechte entsprechend spezifiziert. Dies geschah unter Berücksichtigung der besonderen Eigenschaften des WebDAV-Protokolls bzw. unter Berücksichtigung der Eigenarten Web-basierter Anwendungen. Für die Abbildung der verschiedenen Verarbeitszustände eines Dokuments wurde naheliegenderweise für jeden Zustand ein eigenes Verzeichnisse erzeugt. Dokumente eines bestimmten Verarbeitungszustandes befinden sich somit in den jeweiligen Verzeichnissen und auf sie kann so sehr leicht mittels WebDAV zugegriffen werden. Zusätzlich können über so genannte Properties detailliertere Informationen über den Zustand der Erschließung mittels WebDAV angefragt werden. Auch darüber hinausgehende Informationen, beispielsweise über das betreffende Studienfach (Physik, Biologie etc.) können über Properties bezogen werden, so dass die Erschließung optimal unterstützt werden kann. Um die Spezifikation der Rechte bestimmte Operationen ausführen zu dürfen, müssen zunächst die möglichen Rollen identifiziert werden, die ein Benutzer im System einnehmen kann.

Rollen

Rollen ermöglichen es, die Interaktionen zwischen den Teilnehmern zu strukturieren und die Funktionalitäten abhängig von der Rollenverteilung zu definieren. Zwei Aspekte sind zu berücksichtigen [BS95]:

1. Die Rolle definiert die soziale Funktion eines Einzelnen in Beziehung zum Gruppenprozess, zur Organisation und zu anderen Gruppenteilnehmern.
2. Die Rolle definiert die Rechte und Pflichten im Rahmen des Gruppenprozesses. Die Kontrolle über die Informationseinheiten (z. B. Lese- und Schreibrechte) und die Aktivitäten, welche die Einzelnen ausführen dürfen oder müssen, werden festgelegt. Ebenso können Privilegien vergeben werden.

Die Informationseinheiten im o. g. Sinn bestehen aus Dokumenten, d. h., das Granulat der Verarbeitung ist das Dokument. Diese Dokumente sind XML-basiert und somit textbasiert. Es gibt Dokumente, die Metadaten, Gutachten, Benutzerkommentare oder Kombinationen beinhalten können. Die Schnittstelle für den Web-basierten bzw. entfernten Zugriff bildet WebDAV. Entsprechend der o. g. Definition lassen sich aus den Erschließungsstufen direkt die primären Rollen ableiten:

- Endbenutzer (auch externer Benutzer bzw. WebUser),
- Redakteur,
- Gutachter,
- Systemverwalter.

Durch die Trennung von formaler und inhaltlicher Erschließung wird allerdings eine Verfeinerung der Rolle Redakteur notwendig:

- Zuarbeiter (wissenschaftliche Hilfskräfte oder automatische Systeme),
- Bibliothekar,
- Verantwortlicher für den Inhalt (Administrator).

Ferner ist eine besondere Rolle für die Verwaltung (Management) des Systems sinnvoll. Dieser so genannte Maintainer ist nicht verantwortlich für den Inhalt im Sinne des Administrators, sondern für die Verwaltungsaufgaben innerhalb des Systems. Die möglichen Aufgaben betreffen beispielsweise die Benutzerverwaltung oder Datensicherung. Somit wurde die ursprüngliche Systemverwalterrolle in eine technische und eine nicht-technische Rolle unterteilt.

Zusammenfassend lassen sich die Rollen wie folgt definieren:

- WebUser: Endbenutzer mit Suchmöglichkeiten (lesender Zugriff) und der Möglichkeit, Link-Vorschläge und Kommentare in die Sammlung einzubringen.
- Bibliothekar: Die Rolle der Bibliothekare umfasst alle Kompetenzstufen bis auf den Gutachter, d. h., ein Bibliothekar kann auf allen Stufen des Erschließungsprozesses wirken und auch nur ein Bibliothekar kann entscheiden, ob ein XML-Dokument mit den zugehörigen Metadaten öffentlich zugänglich gemacht werden soll.
- Robot: Ein Robot beschreibt die Rolle eines Zuarbeiters, die einem Werkzeug zum automatischen Sammeln und Erschließen zugewiesen wird. Solche Helfer können allerdings nur intern wirken und zuarbeiten. Ein Bibliothekar muss von ihnen erfasste Daten validieren und explizit einer Verwendung zustimmen.
- HiWi: Diese Rolle beschreibt wissenschaftliche Hilfskräfte. Auch solche Helfer können nur intern wirken und zuarbeiten. Auch die von ihnen erfassten Daten müssen validiert werden und es muss explizit einer Verwendung zugestimmt werden.

• Gutachter: Die genaue Funktion der Rolle des Gutachters muss im Detail noch definiert werden. Es ist aber schon hier wichtig zu erwähnen, dass Benutzer in dieser Rollen nur auf ihre eigenen Gutachten uneingeschränkt zugreifen können dürfen. Es sollte beispielsweise nicht möglich sein, dass ein Gutachter Gutachten Anderer ändern kann.

• Admin: Die Administratorrolle ist für die Inhalte des Autorensystems verantwortlich und kann Änderungen unabhängig von den verschiedenen Verarbeitungsstati durchführen. Sie repräsentiert letztlich einen Bibliothekar mit umfassenden Rechten in Bezug auf die jeweiligen Inhalte bzw. Dokumente.

• Maintain: Die Funktion der Systemadministration und Wartung kommt der Maintainer- Rolle zu. Benutzern ist es erlaubt, in dieser Rolle auch so genannte Managementfunktionen durchzuführen (bspw. Benutzerverwaltung, Sicherungen etc.).

II.5.3. Produktionssystem

Die HTTP-Erweiterung WebDAV als Protokoll bzw. Schnittstelle zu der Datenverwaltungskomponente einzusetzen, hatte den Vorteil schon sehr früh mit der Entwicklung weiterer Komponenten, beispielsweise dem Redaktionssystem, beginnen zu können, ohne dass eine vollständige Implementierung des Gesamtsystems hätte existiert müssen. Neben der Schnittstelle für die Definition der möglichen Operationen ist mit der Beschreibung der Abbildung der Metadaten-Attribute auf XML ein konkretes Datenmodell gegeben, das dem Austausch der zu verwaltenden Daten dienen kann. Die prototypische Realisierung der Datenverwaltungskomponente mittels des WebDAV- Standards unter Benutzung des Slide-Projekts der Apache-Jakarta-Gruppe erlaubte es innerhalb kurzer Zeit, ein evaluierbares System zur Verfügung zu stellen. Für die Realisierung der verfeinerten Konzepte allerdings reichte die Mächtigkeit des Frameworks nicht aus. Daher wurde die Slide- basierte Architektur im Kern, d. h. an den Stellen der Dokumenten- bzw. Datenverwaltung, durch eine eigene Realisierung ersetzt. Eine Anpassung der Slide-spezifischen Komponenten hätte teilweise einen größeren Aufwand dargestellt als eine eigene Implementierung. Dies lag insbesondere daran, dass sich die Anforderungen in Bezug auf die Web-basierte Suche, dem Redaktionssystem und dem zugrundeliegenden Erschließungsprozess und damit auch an die Datenverwaltungskomponente zu einer sehr späten Phase des Projekts nicht unwesentlich geändert hatten. Die erweiterte Implementierung basiert auf dem Java™ 2-Enterprise-Edition-Framework (J2EE)[29] und nutzt dessen Vorteile durch Einsatz von Enterprise JavaBeans™ und anderen wohl etablierten Techniken. Dabei sollte auch weiterhin WebDAV als eine der möglichen Schnittstelle unterstützt werden. Die Implementierung des Systems zur Realisierung der verfeinerten Konzepte fand im Rahmen zweier Diplomarbeiten, einiger Projektarbeiten und eines Praktikums sowie unter Einsatz von hilfswissenschaftlichen Mitarbeitern an der Universität Kaiserslautern statt. Sie umfasst ca. 80.000 LOC verteilt auf über 460 Java-Klassen und ca. 80 J2EE-Komponenten.

Java2-Enterprise-Edition: Plattform für unternehmensweite Anwendungen

Die „Java™ 2 Platform, Enterprise Edition“ (J2EE) bildet den Standard für unternehmensweite Java-basierte Anwendungen. Durch die Definition von allgemeinen Diensten (beispielsweise Transaktionsverwaltung, Nachrichten-, Namens- oder Verzeichnisdiensten) und die systemseitige Unterstützung fundamentaler Anwendungssemantik (beispielsweise Persistenz von Geschäftsobjekten) lassen sich nun mehrschichtige Architekturen definieren, die allein auf standardisierten, modularen Komponenten basieren. Dabei baut J2EE auf etablierten Techniken und Konzepten der Java2-Standard-Edition (J2SE) auf, geht mit der Unterstützung für JDBC™ (Datenbankzugriff), CORBA (Kommunikation in verteilten, heterogenen Systemumgebungen), Enterprise-JavaBeans™-Komponenten (EJB), Java-Servlet-Schnittstelle, Java-Server-Pages™ (JSP) und XML weit darüber hinaus und überträgt das mit Java im allgemeinen verbundene Konzept „Write Once, Run Anywhe-

29 <http://java.sun.org/j2ee>

re™“ auf unternehmensweite Anwendungen: „With simplicity, portability, scalability and legacy integration, J2EE is the platform for enterprise solutions.“ (SUN Microsystems, 2000).

J2EE-Programmier- und -Anwendungsmodell

Dem durch J2EE definierten Modell für die Entwicklung unternehmensweiter Anwendungen liegt die Trennung von Präsentations-, Anwendungs- und Datenhaltungsaspekten zugrunde (Abb. 16). Die Trennung erfolgt durch die Definition unterschiedlicher Schichten für die Realisierung der Präsentationslogik, Anwendungslogik („Server-Side Business Logic“) und Datenhaltungskomponente („Enterprise Information System“). Die Präsentationsschicht ist ihrerseits wieder unterteilt in eine Client-seitige und eine Server-seitige Schicht. In der Client-seitigen Präsentationsschicht sind verschiedene Endgeräte berücksichtigt. So ist vorgesehen, über Web-Browser („HTTP-User-Agents“), Applets oder auch eigenständigen Java-Clients auf die entsprechenden Server-seitigen Komponenten zugreifen zu können. Die Server-seitige Präsentationsschicht wird ausgefüllt durch den Web- Server. Dieser beinhaltet einen so genannten Web-Container zur Ausführung von Servlets und JSP- Seiten. Den EJB-Container zur Ausführung von EJB-Komponenten findet man in der Schicht der Server-Side-Business-Logic. Web-Server bzw. Web-Container und EJB-Container bilden zusammen mit den entsprechenden Diensten den so genannten J2EE-Application-Server. Datenbanksysteme, bzw. so genannte Enterprise-Information- Systems (EIS), findet man auf der Ebene der Datenhaltung wieder.

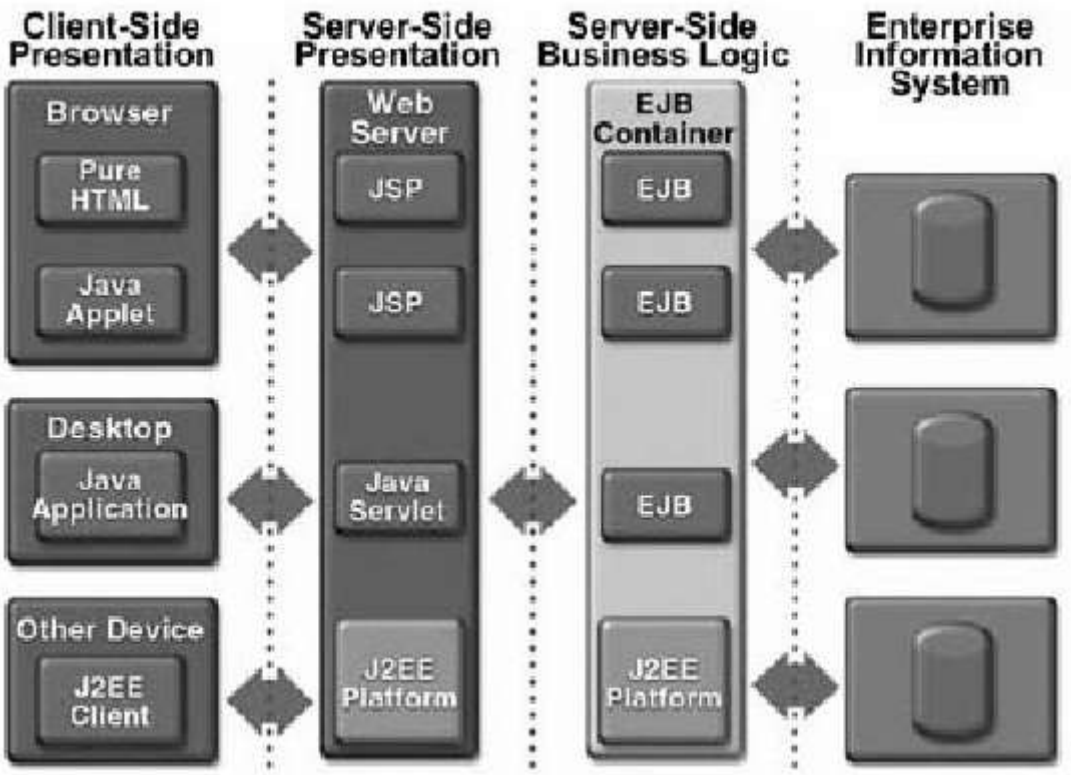

Abbildung 16: J2EE-Mehrschichtenarchitektur

Enterprise-JavaBeans-Komponenten

Enterprise-JavaBeans (EJB) sind Komponenten eines flexiblen Komponentenmodells für Geschäftsobjekte in verteilten Systemumgebungen. Ein Komponentenmodell ermöglicht die Erstellung von wieder verwendbaren Software-Teilen (Komponenten). Es beschreibt eine Infrastruktur für den

Einsatz und die Kommunikation von diesen so genannten Komponenten. Eine Komponente ist ein Stück Software, das klein genug ist, um es in einem Stück erzeugen und pflegen zu können, groß genug ist, um eine sinnvoll einsetzbare Funktionalität zu bieten und eine individuelle Unterstützung zu rechtfertigen sowie mit standardisierten Schnittstellen ausgestattet ist, um mit anderen Komponenten zusammenzuarbeiten [Gri98]. Komponenten und damit auch EJBs können zur Installationszeit konfiguriert und angepasst werden. Ihr Verhalten kann bedingt deklarativ spezifiziert werden. J2EE kennt zwei unterschiedliche Arten von EJB-Komponenten: Session-Beans und Entity-Beans. Entity-Beans repräsentieren persistente Geschäftsobjekte (beispielsweise im einfachsten Fall ein einzelnes Tupel in einer Tabelle einer relationalen DB), während Session-Beans das Verhalten bzw. die Geschäftslogik realisieren und somit auch das Zusammenspiel von Entity-Beans organisieren. Die Erwartungen, die in J2EE gesetzt werden, sind durch vollmundige Ankündigungen entsprechend hoch, es bleibt allerdings abzuwarten, ob diese Erwartungen auch erfüllt werden können: “Based on these flexible component configurations, the J2EE application model means quicker development, easier customization and greater ability to develop powerful enterprise applications. And, because it is based on the Java programming language, this model enables all J2EE applications to achieve all the benefits of Java technology: scalability, portability, and programming ease.“ (Sun Microsystems, 2002).

Übersicht über den J2EE-basierten Implementierungsansatz

Um den gewachsenen Ansprüchen an die Datenverwaltungskomponente gerecht zu werden, wurde bei der Erweiterung des bisherigen Ansatzes auf sinnvolle Modularität geachtet. Die Re-Implementierung mittels Java 2 Enterprise-Edition (J2EE) umfasst daher folgende Teilbereiche (siehe Abb. 17):

- Anfrageverarbeitung (Query-Engine),
- Ergebnisverarbeitung (Result-Set-Processor),
- XML-Dokumenten-Management (XML-Processor),
- Web-basierte Suche (Web-Search),
- Redaktionssystem (Indexing-Tool) und
- Support-Komponenten (bspw. zur Unterstützung der Navigation über die RVK-Klassifikation oder Benutzerverwaltung).

Dabei konnten die Ansätze aus den bisherigen Arbeiten aufgrund entsprechender Kapselung und Festhalten an Java ohne große Änderungen weiterverfolgt werden. Alle genannten Teilbereiche sind durch entsprechende Diplom- oder Projektarbeiten bzw. Studenten (d. h. Hiwis) besetzt. Die Web-basierte Suche und die entsprechende Infrastruktur zum Testen der Such-Schnittstelle sind bereits nahezu vollständig implementiert [FFW02]. Diese Testumgebung wird nun sukzessive durch die verschiedenen Projekt- und Diplomarbeiten mit Funktionalität ausgefüllt. Die Entwicklung des Redaktionssystems basierend auf der neuen Technologie ist bereits weitgehend abgeschlossen.

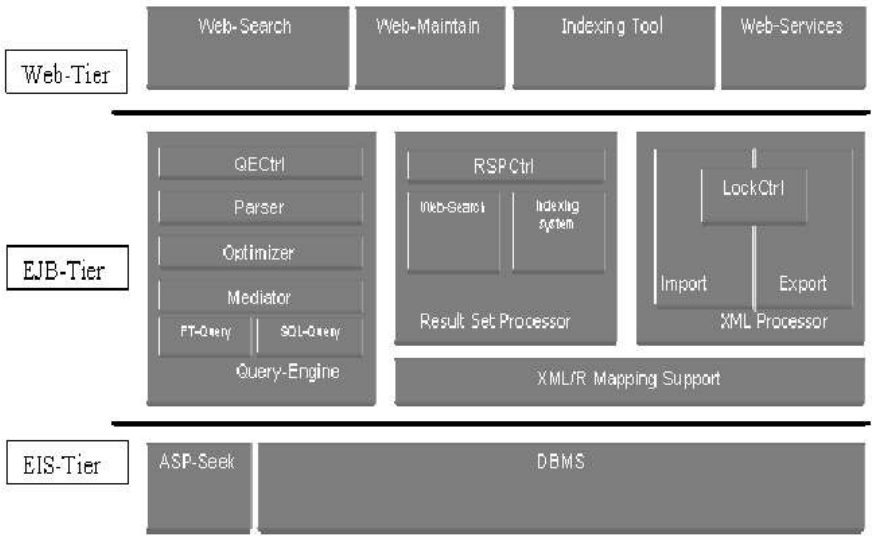

Abbildung 17: Der J2EE-basierte Implementierungsansatz

Grundlegende Konzepte der Re-Implementierung

Der Abbildung deutet ferner an, wie die genannten Komponenten in Beziehung stehen. Die Datenhaltungsschicht („EIS-Tier"), bestehend aus einem relationalen Datenbanksystem und einem Volltextindexierungssystem, bildet die Basis. Auf dieser baut die mittlere Schicht der EJB-Komponenten auf („EJB-Tier"). EJBs realisieren die eigentliche Anwendungslogik. Die Schicht der Server- seitigen Präsentation („Web-Tier") wiederum nutzt die EJB-Komponenten und realisiert die verschiedenen Module beispielsweise zur Unterstützung der Web-basierten Suche, der Redaktionsschnittstelle oder so genannter Web-Services.

Nochmals soll besonders betont werden, dass die Aufgaben der eigentlichen Anfrageverarbeitung („Query-Engine"), der Ergebnisverarbeitung (Result-Set-Processor) und des XML-Dokumenten-Managements („XML-Processor") getrennt betrachtet werden und auch separat realisiert werden („Separation of Concerns"). Es soll nun auf die Komponenten im einzelnen eingegangen und ihre Besonderheiten näher erläutert werden.

Basisfunktionalität (Foundation)

Zunächst seien die fundamentalen Dienste erwähnt, die eine besondere Unterstützung der Regensburger Verbundklassifikation (RVK) realisieren, sowie Unterstützung für Schema-Informationen (bspw. Wertebereiche, vordefinierte Vokabularien oder interne Abbildungsinformationen), Benutzerverwaltung und Konfiguration anbieten. Die Aufgabe der Verwaltung der strukturierten Daten übernimmt ein objekt-relationales Datenbankverwaltungssystem (ORDBVS). Die Verwaltung von unstrukturierten Daten wird durch ein entsprechendes Volltext-System (hier: ASP-Seek) unterstützt und bei der Anfrageverarbeitung integriert.

XML-Dokumenten-Management (XML-Processing)

Die jeweiligen Metadatensätze und die miteinander verknüpften Einträge werden, wie eingangs bereits erwähnt, durch ein einziges XML-Dokument repräsentiert, wobei die Verknüpfung durch XML-Referenzen realisiert wird. So können Strukturen (Beziehungen) sehr einfach gehandhabt und visualisiert werden, d.h., zusammengehörige Daten können gemeinsam durch nur eine Anfrage angezeigt werden. Auch das Erfassen von neuen Einträgen wird so unterstützt. Es können zusammengehörige Daten gemeinsam erfasst und als Ganzes zwecks weiterer Verarbeitung verschickt werden. Da die Verarbeitung also Dokumenten-orientiert erfolgt, muss die XML-Processor-

Komponente die Aufgabe übernehmen können, XML-Dokumente auszutauschen. Diese Import- und Exportfunktionalität von Meta-Akad-Dokumenten wird durch geeignete Sperrmechanismen weiter unterstützt.

Anfrageverarbeitung (Query-Engine)

Anfragen an die Menge der Meta-Akad-Dokumente sollten beliebig formuliert werden können, d. h., es sollten beliebige Meta-Akad-Attribute bzw. XML-Elemente benutzt werden können, um einfache Prädikate zu bilden und aus diesen dann komplexe Ausdrücke zusammenzusetzen. Ferner sollte eine Volltextsuche unterstützt werden und es sollte möglich sein, statistische Informationen („Meta-Queries“) über das Anfrageergebnis zu erfahren, beispielsweise über die Anzahl der Treffer, die häufigsten Schlagworte oder Klassifikationen. Es sollte aber auch möglich sein, das Anfrageergebnis nach möglichst beliebigen Kriterien zu sortieren. Möchte man nun aber die Anfrage nicht mehrfach ausführen müssen, da die notwendigen Schritte gegebenenfalls sehr kostenintensiv sein können, so liegt es nahe, die Anfrageergebnisse vollständig zu materialisieren. Die Materialisierung erlaubt auch eine sinnvolle Trennung von Anfrageverarbeitung und Ergebnisverarbeitung.

Die Anfrageverarbeitung umfasst im allgemeinen die syntaktische Analyse der Anfrage, ihre Optimierung, Übersetzung und Ausführung. Da aber im Falle der Anfrageverarbeitung im Meta-Akad- Projekt zwei Datenquellen mit unterschiedlichem Datenmodell zu kombinieren sind, wurde ein zusätzlicher Verarbeitungsschritt notwendig. In diesem Schritt übernimmt ein so genannter Mediator die Zerlegung der Anfrage und die Kombination der Ergebnismengen.

Die Teilanfragen werden der jeweiligen Ausführungskomponente übergeben. Die Ausführungskomponente zur Unterstützung des Volltextindexierungssystems besitzt die Fähigkeit, Anfragen in einem Cache zwischenzuspeichern und wieder zu verwenden. Das integrierte Gesamtergebnis wird in der relationalen Datenbank materialisiert und kann von der Ergebnisverarbeitung weiterverwendet werden.

Die Anfrageverarbeitung führt somit die Anfrage nur in dem Sinne aus, dass sie die interne Repräsentation einer Anfrage („Query-Tree“, Abb. 18) in entsprechende SQL-Anfragen bzw. Volltextsuchanfragen transformiert und ausführt. Ergebnisse werden in temporären Tabellen gespeichert, was wiederum zu einem günstigeren Zugriffsverhalten auf der Datenbankebene führt, da die Sperrproblematik entschieden entschärft wird. Des Weiteren ermöglicht die Trennung überhaupt erst das Konzept des Stateless-Cursor, welches im nachfolgenden Abschnitt im Rahmen der Ergebnisverarbeitung beschrieben werden soll.

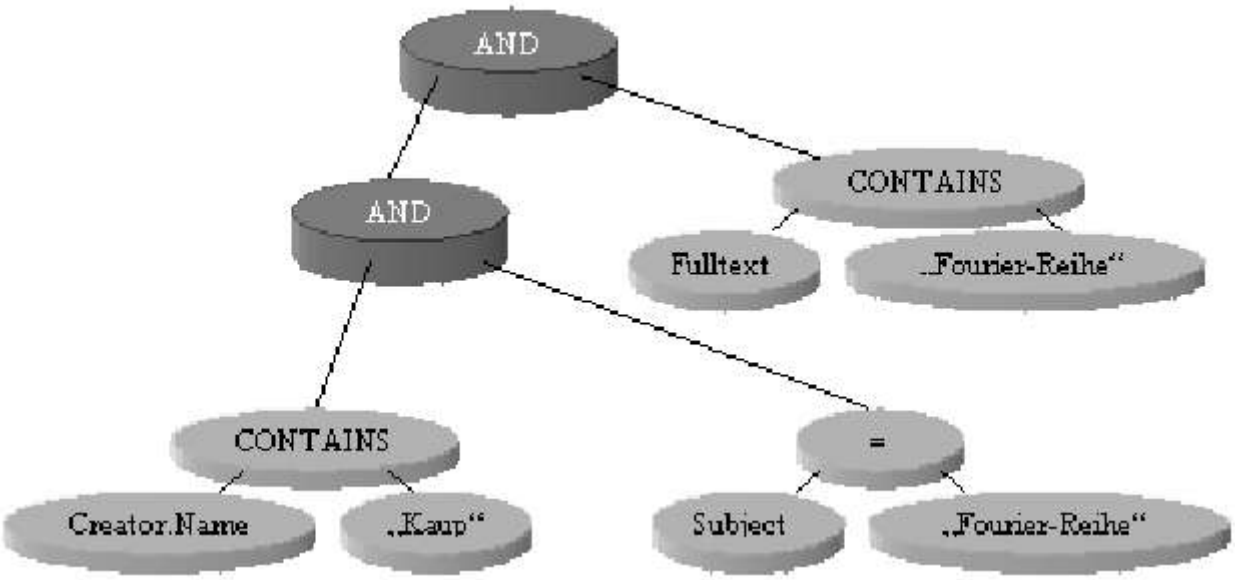

Abbildung 18: Anfrageverarbeitung mittels eines Query-Tree

Ergebnisverarbeitung (Result-Set-Processing)

Das materialisierte Anfrageergebnis dient der Ergebnisverarbeitung als Grundlage für die Beantwortung von weiteren Anfragen nach der Anzahl von Treffern oder den am häufigsten verwendeten Schlagworten. Diese Anfragen und auch die Behandlung der Sortierung der Anfrageergebnisse nach den verschiedenen Kriterien erfolgt ohne die wiederholte Ausführung der gesamten Anfrage. Ebenso ist der Wechsel zwischen verschiedenen bereits schon einmal ausgewählten Sortierungen ohne erneute Ausführung möglich.

Dabei dient als Schnittstelle zum Web-Tier der Result-Set-Processor-Controller (RSPCtrl). Die RSPCtrl-Komponente unterstützt zwei unterschiedliche Implementierungen: eine zustandslose („stateless“) und eine zustandsbehaftete („stateful“) Variante. Zwischen diesen beiden kann frei gewählt werden. Allerdings ist die zustandslose für den Zugriff über das Web optimiert und benutzt das bereits genannte Stateless-Cursor-Konzept, während die andere Variante in Bezug auf Zugriffe durch das Redaktionssystem optimiert wurde. Letztere bietet weitere Sortiermöglichkeiten an und benutzt Datenbank-spezifische Cursor zum Durchlaufen der Ergebnismenge. Dies schont zwar die Datenbank-seitig aufzubringenden Ressourcen für die Speicherung der Ergebnisse, allerdings erhöht sich die Anzahl der offenen Datenbank-Verbindungen, die nicht gemeinsam benutzt werden können.

Stateless-Cursor-Konzept

Bei Zugriffen über die Web-basierte Suchschnittstelle müssen viele parallele Anfragen gleichzeitig verarbeitet werden können. Im Allgemeinen sollte daher ein Ziel immer die gemeinsame Nutzung von DB-Ressourcen sein. Eine besonders kritische Ressource stellt in der Regel die Datenbankverbindung dar. Daher wird bei der Web-basierten Suche versucht, die Dauer der exklusiven Nutzung einer Datenbankverbindung zu minimieren. Aus diesem Grund erhält jedes Objekt bzw. Tupel, das sich bei einer Anfrage qualifiziert hat, einen eindeutigen und fortlaufenden Schlüssel, über den Bereichsanfragen gestellt werden können. So ist es möglich, auf Teilergebnisse zuzugreifen, ohne während der Verarbeitung und insbesondere ihrer Benutzer-seitigen Präsentation eine Datenbankverbindung oder einen Datenbank-Cursor zu halten. Soll über die Gesamtmenge der Ergebnisse iteriert werden, können sukzessive Pakete beliebiger Größe angefordert werden. Die dabei zu verwal-

tende Cursor- bzw. Iterator-Position wird in der Datenbank abgelegt. Es muss nur der entsprechende Schlüsselwert des zuletzt gelesenen Objekts bzw. Tupels gespeichert werden. Um auch die Kosten für Auf- bzw. Abbau einer Verbindung zu minimieren, werden die Verbindungen in einem so genannten Pool verwaltet. Auch die Komponenten (d. h. Session-EJBs), welche die oben genannten Bereichsanfragen verarbeiten, werden gemeinsam verwendet und in einem Pool verwaltet, um die Kosten bei Instanziierung und Löschung zu minimieren. Diese insgesamt zustandlose Verarbeitung ist essentiell für das Stateless-Cursor-Konzept.

Beispielhafte Anfrageverarbeitung

Auf die Anfrageverarbeitung soll nun anhand eines Beispiels näher eingegangen werden. Zunächst muss eine Query-Tree-Objekt-Struktur (siehe Abb. 18) definiert werden, welche die eigentliche Anfrage repräsentiert. Dieser so genannte Query-Tree wird beispielsweise von einer Komponente der Web-basierten Suche nach Ausfüllen des Formulars der erweiterten Suche generiert und dem Query-Engine-Controller (QECtrl) übergeben, der als zentraler Einstiegspunkt der Query-Engine-Komponente fungiert (siehe Abb. 18).

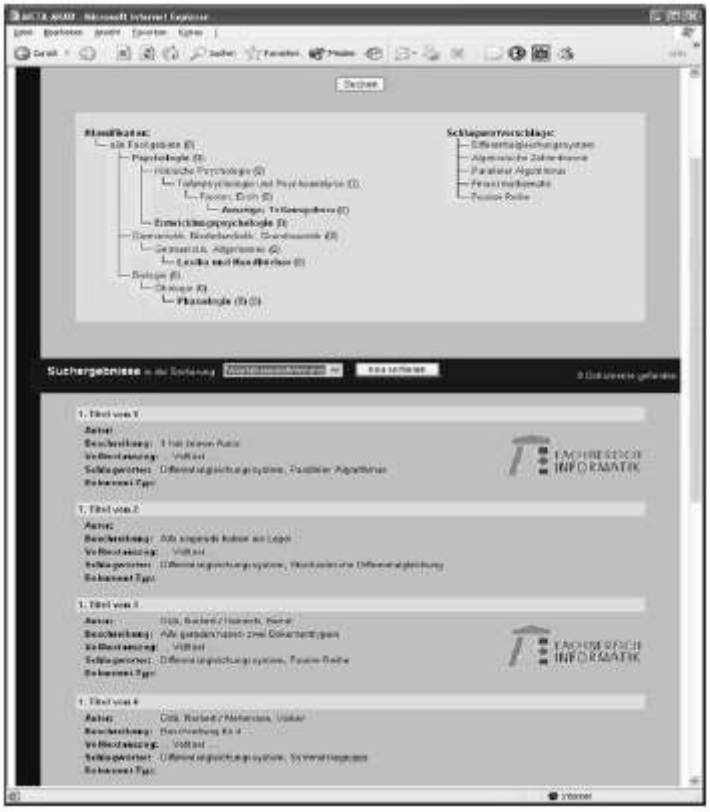

Abbildung 19: Präsentation der Ergebnisliste

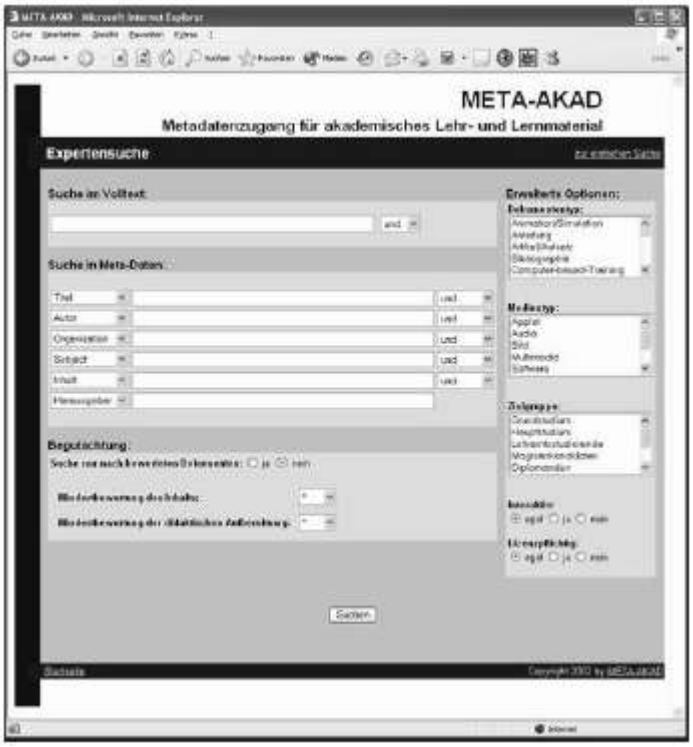

Abbildung 20: Expertensuchschnittstelle

Die Query-Engine verarbeitet die Anfrage und erstellt, wie bereits erwähnt, eine temporäre Tabelle und lässt, wiederum durch die QECtrl, ein Stellvertreter-Objekt („Proxy“oder „Handle“) für das Anfrageergebnis zurückliefern.

Dieses Handle wird nun bei jedem der weiteren Schritten als Kontextinformation benötigt und muss den weiteren Methodenaufrufen mitgegeben werden. Es muss daher zunächst auch dem Result-Set-Processor-Controller (RSPCtrl) übergeben werden, über den es möglich ist, eine Übersicht der sich qualifizierenden Tupel zu beziehen und über die gesamte Ergebnismenge zu iterieren (s.Abb. 1.5). Anhand eines eindeutigen Identifikators kann von dem XML-Processor das vollständige Metadaten-Dokument bezogen werden und nach Anforderung einer Schreibsperre auch verändert werden.

II.5.4. Realisierung der Anfrageverarbeitung

Dieser Abschnitt beschreibt die Anforderungen, den Ablauf, die Implementierung der Anfrageverarbeitung und die Funktionsweise einzelner Komponenten. Die Anfrageverarbeitung (AV) ist eine der drei Hauptkomponenten der Anwendungslogik der Dokumentenverwaltung. Sie hat zur Aufgabe, möglichst beliebige Suchanfragen entgegenzunehmen und ein entsprechendes Ergebnis zurückzuliefern. Es soll einfach möglich sein, aus den Metadaten-Attributen bzw. aus den zugehörigen XML-Elementen Prädikate (beispielsweise **Autor = 'Göb, Norbert'**) zu bilden und diese dann zu komplexen Ausdrücken zusammenzusetzen (z. B. **[Autor = 'Göb, Norbert'] AND [Titel enthält 'Algebra']**). Weiterhin sollte die Möglichkeit bestehen, die Suche in den strukturierten Meta-Daten mit einer Volltextsuche zu kombinieren. Auch muss die Möglichkeit berücksichtigt werden, Anfragen zu materialisieren und in einem Cache abzulegen, um kostenintensive Wiederholungen von Anfragen (beispielsweise bei Umsortierung, Anfrageverfeinerung und statistischer Auswertung über das gesamte Anfrageergebnis) effizienter bearbeiten zu können. Das von der AV gelieferte Ergebnis muss in einer Form vorliegen, die es erlaubt, die AV sinnvoll von der Ergebnisverarbeitung zu trennen. Es sollte möglich sein, das Ergebnis nach verschiedenen Kriterien zu sortieren und eine aktuelle Position (Offset des Cursor) zu setzen. Falls eine Volltextsuche durchge-

führt wird, soll ein zu dem jeweiligen Dokument passender Volltextauszug mitgeliefert werden. Die Ergebnisse einer Anfrage werden daher in temporären Tabellen Datenbank-seitig gespeichert, was zu einem einfachen Zugriff seitens der Ergebnisverarbeitung führt.

Im Wesentlichen besteht die Aufgabe der AV in der syntaktischen Analyse der Anfrage, ihrer Optimierung, Übersetzung und Ausführung. Insbesondere soll jedoch eine Anfrage an zwei verschiedene Datenhaltungssysteme gestellt werden können, die sich zudem in ihrem Datenmodell stark unterscheiden: Zum einen ein relationale DBS mit Metadaten und zum anderen ein Volltextindexierungssystem. Die AV muss Anfragen an beide Systeme stellen und die Ergebnisse entsprechend kombinieren können.

Phasen der Anfragenverarbeitung

Die Verarbeitung einer Anfrage kann mit einem einfachen „Top-Down"-Ansatz beschrieben werden. An [Mit95] angelehnt kann die AV in fünf Phasen (siehe Abb. 21) eingeteilt werden:

Schritt 1: *Interndarstellung der Anfrage*. Um die Anfrage effizient verarbeiten zu können, wird sie vom *QueryParser* in ein geeignetes internes Format umgewandelt. Diese interne Darstellung soll einfach zu übersetzen sein und die anschließend folgende Transformation der Anfrage unterstützen. Weiterhin wird eine Syntaxanalyse der Anfrage durchgeführt.

Schritt 2: *Transformation der Anfrage.* Logische Umformungen sollen die Anfrage standardisieren und, falls erforderlich, auch zusammenfassen. Geeignete Transformationen führen zu einem effizienteren Anfrageplan.

Schritt 3: *Erzeugung eines Anfrageplans*. Der *Plangenerator* erzeugt aus der internen Darstellung einen Anfrageplan (QueryPlan), der die Grundlage für die Generierung der späteren SQL-Anfrage an die relationale Datenbank bildet. Ferner führt er Vorbereitungen für die Volltextsuche durch, indem er entsprechende temporäre Tabellen in der Datenbank erstellt.

Schritt 4: *Optimieren des Anfrageplans.* Der im letzten Schritt erzeugte Anfrageplan führt nicht automatisch zu einer optimalen SQL-Anfrage. Der *PlanOptimizer* versucht nun den Anfrageplan entsprechend umzuformen, um eine möglichst optimale SQL-Anfrage zu erhalten.

Schritt 5: *Ausführung des Anfrageplans*. Im fünften und letzten Schritt wird aus dem Anfrageplan eine SQL-Anfrage generiert und ausgeführt. Das Resultat der Anfrage wird entsprechend aufbereitet und in eine temporäre Tabelle geschrieben. Ein Proxy-Objekt dient im weiteren als Stellvertreter für das Anfrageergebnis und als Schnittstelle für den Zugriff darauf.

Diese fünf Schritte beschreiben im Grunde die aufeinander folgenden Verarbeitungsphasen einer Anfrage.

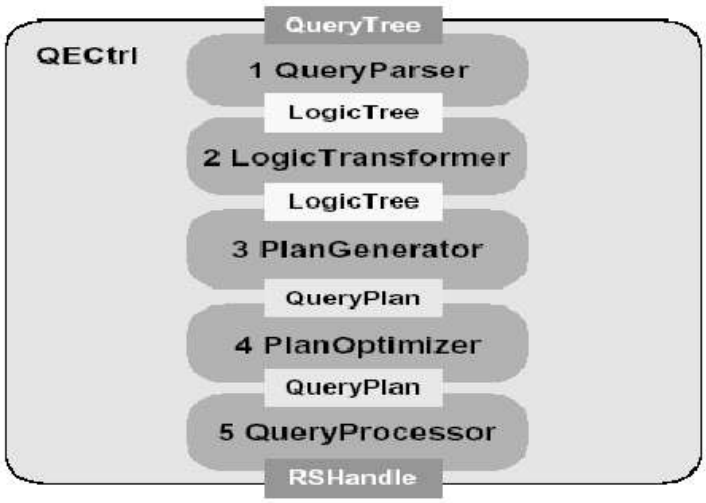

Abbildung 21: Phasen der Anfrageverarbeitung.

Die Kontrolle des Ablaufs obliegt dem Kontrollmodul (*QECtrl*). Diese Enterprise-JavaBean wird von den Komponenten des WEB-Tier benutzt und liefert ein Objekt für die Repräsenation des Ergebnisses zurück.

Interne Repräsentation und Verarbeitung einer Anfrage

Die AV erhält die Anfrage in dem schon zuvor vorgestellten speziellen Format. Dieser Suchbaum („QueryTree“) bietet dem Anwender eine relativ einfache, Schema-unabhängige Möglichkeit, eine Anfrage an die Datenbank zu stellen. Ferner bietet sie die Möglichkeit, die Volltextsuche unkompliziert zu integrieren. Die Einschränkung dieser besonderen Art der „Anfragesprache“ liegt darin, dass nicht mehr Möglichkeiten als die Formulierung einer Selektion gegeben sind. Die Struktur des QueryTree ermöglicht es aber, einfache Prädikate mit Hilfe der XML-Elemente zu bilden. Um auch komplexere Anfragen formulieren zu können, besteht die Möglichkeit mehrere solcher Prädikate mit den logischen Operatoren AND, OR und AND NOT zu Ausdrücken zu kombinieren.

Um die Anfrage effizient und einfach verarbeiten zu können, wird sie in eine entsprechende interne Repräsention umgewandelt. Die interne Darstellung („LogicTree“) einer Anfrage ist, wie die Anfrage selbst, als binärer Baum realisiert: mit Prädikaten als Blätter und Operatoren als Knoten. Die zur Verfügung stehenden logischen Operatoren AND, OR und NOT bilden eine Basis, d.h., jeder logische Term ist somit als ein LogicTree darstellbar. Diese Eigenschaft ist vor allem wichtig für spätere Transformationen der internen Darstellung.

Der QueryParser hat nun die Aufgabe, für einen gegebenen QueryTree einen äquivalenten LogicTree zu erzeugen. Die binäre Baumstruktur erlaubt es einen recht einfachen rekursiven Algorithmus zu formulieren, der diese Arbeit durchführt. Der Algorithmus erstellt für einen übergebenen Konten des QueryTree einen entsprechenden Knoten des LogicTree.

Die logische Transformation stellt verschiedene Methoden zur Verfügung, um die Interndarstellung in eine geeignete Normalform zu bringen. Im Wesentlichen sind dies die konjunktive (KNF) und die disjunktive (DNF) Normalform. Je nach Struktur der Anfrage kann eine solche Transformation die spätere Optimierung begünstigen. Die im LogicTransformer verwendeten Algorithmen sind alle in [Mit95] zu finden. Die Normalisierung basiert auf drei Regeln:

- Ersetze jedes Vorkommen von **`Not(Not(a))`**
  durch **`a`**.
- Ersetze jedes Vorkommen von **`Not(And(a, b))`**
  durch **`Or(Not(a), Not(b))`**.

• Ersetze jedes Vorkommen von **`Not(Or(a, b))`**
durch **`And(Not(a), Not(b))`**.

Die Transformation in die konjunktive Normalform lässt sich also rekursiv formulieren:

• Ersetze jedes Vorkommen von **`Or(a, And(b, c))`**
durch **`And(Or(a, b),Or(a, c))`**.

• Ersetze jedes Vorkommen von **`Or(And(a, b), c)`**
durch **`And(Or(a, c), Or(b, c))`**.

Für die disjunktive Normalform gilt analog:

• Ersetze jedes Vorkommen von **`And(a, Or(b, c))`**
durch **`Or(And(a ,b), And(a, c))`**.

• Ersetze jedes Vorkommen von **`And(Or(a, b), c)`**
durch **`Or(And(a, c), And(b, c))`**.

Der Anfrageplan ist eine Repräsentation der späteren SQL-Anfrage, der allerdings eine einfache Manipulation und Analyse zulässt. Das Erzeugen des Anfrageplans ist einer der komplexesten Schritte der Anfrageverarbeitung. In dieser Phase findet die Zuordnung von XML-Elementen und Datenbank-Tabellen bzw. DB-Attributen sowie die Vorbereitung, Ausführung und Aufbereitung der Volltextsuche statt.

Die Metadaten-Attribute der erfassten Lehr-/Lern-Materialien (oder auch Ressourcen) werden in einem relationalen Schema verwaltet, wobei die einzelnen Merkmale in verschiedenen Tabellen gespeichert sein können und über Beziehungen den jeweiligen Ressourcen zugeordnet werden können. Die zu erzeugende SQL-Anfrage soll nun die Schlüssel der Datensätze liefern, deren Merkmale die vom *LogicTree* beschriebenen Eigenschaften erfüllen. Um nun eine entsprechende Anfrage generieren zu können, muss das Schema der Datenbank und die Abbildung der XML-Elemente auf die SQL-Tabellen bzw. Attribute in einer geeigneten Form verfügbar gemacht werden. Diese Informationen werden durch ein so genanntes Metamodell beschrieben.

Es handelt es sich dabei auch um ein relationales Schema, welche die Zuordnung von XML-Attributen und Tabellenspalten sowie die Beziehung zwischen den einzelnen Attributen beschreibt. Die Struktur dieses Schemas ist aus dem Diagramm in Abb. 22 ersichtlich.

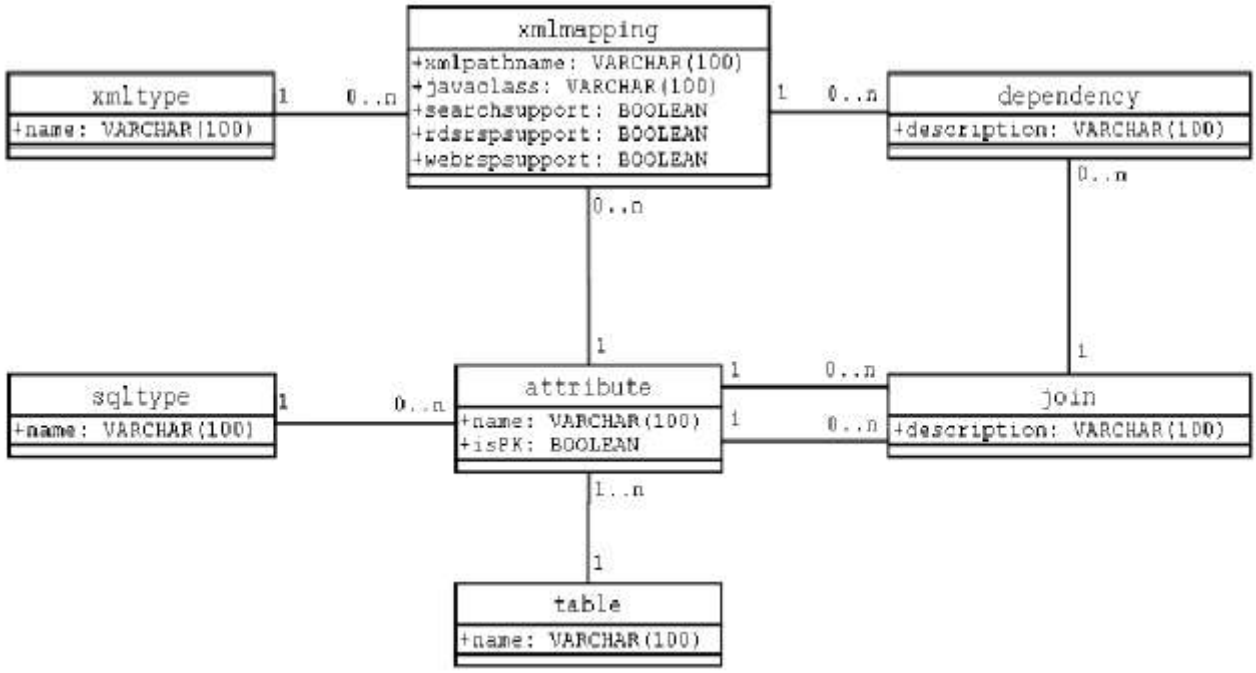

Abbildung 22: Metamodell-Schema.

Um die Volltextsuche möglichst generisch in das System zu integrieren, wurde beschlossen, das Ergebnis der Suche als temporäre Tabelle in der relationalen Datenbank zu materialisieren. Diese Vorgehensweise ermöglicht es, das Ergebnis der Volltextsuche direkt in die SQL-Anfrage einzubeziehen, ohne dass ein externes Zusammenführen der Ergebnisse erforderlich wäre. Ferner können Suchergebnisse gespeichert und für andere Anfragen wiederverwendet werden (Caching“).

Da die Merkmale der Ressourcen in einer relationalen Datenbank mit mehreren Tabellen gespeichert werden, ist es in vielen Fällen nötig, ein Kreuzprodukt („Join“) der beteiligten Tabellen zu erstellen und die interessanten Zeilen mit einer entsprechenden Verknüpfungsvorschrift („Join predicate“) auszuwählen. Die Information, welche Tabellen unter welchen Bedingungen an einem Join teilnehmen, liefert auch das Metamodell.

Bei der Formulierung der Anfrage ist jedoch zu beachten, dass eine einfache Verknüpfung aller beteiligten Tabellen in einem einzigen *Join* nur in Ausnahmefällen zum gewünschten Ergebnis führt. Eine solche Anfrage würde nur Datensätze liefern, für die in jeder beteiligten Tabelle mindestens ein Datensatz vorhanden ist, der die Join-Bedingung erfüllt. Da jedoch nicht für alle Dokumente jedes mögliche Merkmal erfasst wird, können Datensätze aus dem Ergebnis herausfallen, obwohl sie den vom Benutzer festgelegten Kriterien genügen. Auch die Benutzung von äußeren Verbundoperatoren ist nicht einfach möglich. Einen weiteren Stolperstein bilden die 1:N- oder M:N-Beziehungen. Dies wird allerdings bei der Generierung der SQL-Anfrage berücksichtigt. Zuvor ist der Anfrageplan jedoch noch zu optimieren.

Das Ziel der Anfrageoptimierung durch eine geeignete Komponente („PlanOptimizer“) besteht darin, den Anfrageplan in eine für die Ausführung möglichst günstige Form zu bringen. In diesem speziellen Fall bedeutet dies das Zusammenfassen von mehreren Unteranfragen zu möglichst nur einer Anfrage. Dabei müssen jedoch wiederum die Besonderheiten des Schemas beachtet werden, die bereits kurz erläutert wurden. Diese Eigenschaften führen zu zwei wichtigen Regeln für die Optimierung:

- Fasse alle Disjunktionen (OR) zusammen, welche die gleichen Tabellen betreffen.
- Fasse alle Konjunktionen (AND) zusammen, welche nicht die gleichen Attribute betreffen

Da der PlanOptimizer keine logischen Umformungen vornimmt, kann es von Vorteil sein, die Anfrage schon im Hinblick auf die Optimierung zu formulieren oder vorher entsprechende logische

Transformationen durchzuführen. Negationen können allerdings nicht so einfach zusammengefasst werden, da dies in der Regel zu komplexe Umformulierungen der Anfrage erfordern würde.

Die letzte Phase der Anfrageverarbeitung ist die Ausführung der Anfrage. Dabei wird aus dem Anfrageplan eine SQL-Anfrage generiert, diese an die Datenbank gestellt und das Ergebnis in einer temporären Tabelle gespeichert. Diese Tabelle wird registriert und ein eindeutiger Schlüssel zur Verfügung gestellt, der externen Moduln des Systems (beispielsweise der Ergebnisverwaltung) den Zugriff auf das Resultat der Suche ermöglicht. Überdies werden den Datensätzen, sofern vorhanden, Volltextauszüge und eine fortlaufende Nummer, abhängig von der Volltextsuche, zugeordnet, um die Ergebnisse entsprechend gewichten zu können und gemäß des Stateless-Cursor-Konzeptes darauf zugreifen zu können, was eine fortlaufende Nummerierung der Ergebnisse voraussetzt.

II.5.5. Realisierung der Ergebnisverarbeitung

In der Diplomarbeit, die sich mit Ergebnisverarbeitung befasste [Gau03], werden die beiden o. g. Implementierungs- bzw. Entwurfskonzepte („stateless“ und „stateful“) hinsichtlich ihrer Fähigkeit zu skalieren untersucht. Beide Konzepte bieten diesbezüglich Vor- und Nachteile. Die zustandslosen Komponenten lassen ein gutes Skalierungsverhalten erwarten: Es müssen immer nur so viele Instanzen einer Komponente erzeugt werden, wie parallel zum Einsatz kommen. Der Nachteil dieses Entwurfskonzeptes ist der hohe Kommunikationsaufwand zum Austausch der zusätzlichen Informationen zwischen aufrufender und aufgerufener Instanz. Diese Problematik existiert bei einem Komponentenmodell mit zustandsbehafteten Komponenten nicht. In jeder Instanz können die benötigten Zustandsinformationen direkt gespeichert werden. Durch die exklusive Bindung einer Instanz an ihren Erzeuger, muss allerdings eine viel größere Anzahl von Instanzen erzeugt und verwaltet werden: Für jeden Client wird eine eigene Instanz erzeugt, die Verwaltung der Instanzen in einem Pool ist nicht möglich. Um das Skalierungsverhalten zustandsloser und zustandsbehafteter Komponenten evaluieren zu können, wurden mehrere unterschiedliche Implementierungen der Resultset- Prozessor Komponente im Meta-Akad Projekt vorgenommen. Mittels eines Lastgenerators sollen die maximale Anzahl der Interaktionen pro Sekunde und die durchschnittliche Antwortzeit einer Interaktion in Abhängigkeit von der Last für jede dieser Implementierungen bestimmt werden. Da die übrigen Komponenten des Systems hierbei unangetastet bleiben, erlauben die Messergebnisse Rückschlüsse auf das Skalierungsverhalten der unterschiedlichen Implementierungen.

In der Literatur sind etliche Definitionen zum Thema Komponenten und Komponentenmodell zu finden:

- „Software components enable practical reuse of software parts and amortization of investments over multiple applications. There are other units of reuse, such as source code libraries, design, or architectures. Therefore, to be specific, software components are binary units of independent production, acquisition, and deployment that interact to form a functioning system (Clemens Szyperski in [Szy98]).“
- „A business component represents the software implementation of an autonomous business concept or business process. It consists of the software artifacts necessary to express, implement, and deploy the concept as a reusable element of a larger business system (Wojtek Kozaczynski in [Koz99]).“
- „Components are self-contained instances of abstract data types (ADTs) that can be plugged together to form complete applications (Dough Schmidt in [Sch99]).“

Die Unterschiede in den genannten Definitionen liegen lediglich in der Gewichtung der einzelnen Forderungen, so dass sich die Aussagen wie folgt zusammenfassen lassen:

1. Komponenten sind binäre Software-Artefakte. Zum Zeitpunkt ihrer Komposition liegen sie als übersetzte Software-Module vor.
2. Komponenten sind grob granuliert. Sie umfassen immer einen größeren Funktionskomplex.
3. Eine Komponente enthält immer nur die Elemente höchstens eines Vorgangs bzw. Geschäfts-Konzeptes. Die Abläufe verschiedener Konzepte sollten nicht vermischt werden.
4. Komponenten sind in sich abgeschlossene Einheiten. Sie haben möglichst wenige Abhängigkeitsbeziehungen zu anderen Komponenten. Das heißt, sie besitzen eine geringe Kopplung.
5. Komponenten werden speziell im Hinblick auf ihre Wiederverwendbarkeit entworfen.
6. Das Verhalten von Komponenten kann zum Zeitpunkt der Verwendung konfiguriert werden. Änderungen am Quellcode (incl. Neu-Übersetzen) sind nicht nötig.
7. Komponenten kommunizieren ausschließlich ¨über explizit definierte Schnittstellen miteinander.

Zu einer Komponente gehört immer eine passende Laufzeitumgebung, die die notwendigen Infrastruktur-Dienste zur Verfügung stellt. Im einfachsten Fall handelt es sich hierbei um entsprechend der Bedürfnisse zur Komposition der einzelnen Komponenten entwickelten Code, auch „Glue-Code“ genannt. Meist steht jedoch eine fertige Infrastruktur, wie etwa ein J2EE Applikations-Server, zur Verfügung. Die Laufzeitumgebung der Komponenten ist dann Teil einer solchen Infrastruktur (Framework). Das Framework ist für die Komposition der einzelnen Komponenten verantwortlich, es stellt die zur Kommunikation zwischen den Komponenten nötige Funktionalität zur Verfügung und es ist verantwortlich für die Erzeugung und Verwaltung von Instanzen der Komponenten. Was das Framework im Einzelnen leistet, hängt jedoch stark von der jeweils zugrunde liegenden Technologie ab. So stellt die Java 2 Enterprise Edition viele weitergehende Dienste zur Verfügung. Etliche heute verwendeten Frameworks bieten die Möglichkeit, sowohl zustandsbehaftete als auch zustandslose Komponenten einzusetzen.

Die an diese Komponenten zu stellenden funktionalen Anforderungen sind:

8. Ausführen von einfachen und komplexen Suchanfragen,
9. Ausführen von Meta-Queries auf der Menge der Suchergebnisse (Abfrage der Trefferanzahl, Bestimmen der häufigsten Kategorien und Schlagworte in der Treffermenge),
10. Erzeugen und Abrufen von Übersichtsseiten der Treffermenge,
11. Durchführen unterschiedlicher Sortierungen der Treffermenge,
12. Erzeugen einer Kategorie- und Schlagwortvorschlagliste entsprechend der Suchanfrage.

Eine Anforderung an die zustandslose Variante des Resultset-Prozessors, war das effiziente Durchführen von Sortierungen der Ergebnismenge nach vorgegebenen Attributen. Um eine einmal durchgeführte Sortierung nicht immer wiederholen zu müssen (zustandslose Verarbeitung), wurde ein Konzept erstellt, das es erlaubt Sortierreihenfolgen zu speichern. Hierzu wurde das Stateless-Cursor-Konzept erweitert: Es wird nicht nur ein zusätzliches Attribut für die Cursor-Position eingeführt, sondern eine Reihe von Attributen. Jedes dieser Attribute enthält eine fortlaufende Nummerierung für die Tupel in der Ergebnistabelle. Diese fortlaufenden Nummernfolgen entsprechen jeweils einer bestimmten Sortierreihenfolge. In der Verwaltungstabelle wird, zusätzlich zur Cursor-Position, die aktuelle Sortiermethode gespeichert. So kann jederzeit ermittelt werden, nach welchen Attribut die Ergebnistabelle zu sortieren ist. Um die Tupel in einer bestimmten Reihenfolge zu lesen, genügt es die Ergebnistabelle nach dem entsprechenden Attribut zu sortieren und mit einem DB-Cursor darüber zu iterieren. Muss die Cursorposition gesichert werden, so wird an Cursorposition die Nummer des Attributs nach dem sortiert wurde in die Verwaltungstabelle geschrieben.

Zusätzlich wird noch der Name des Attributs nach dem sortiert wurde in der Verwaltungstabelle vermerkt. Soll die Position des DB-Cursors (unter Beibehaltung der Sortierreihenfolge) rekonstruiert werden, so wird die Positionsnummer und der Name des entsprechenden Attributs aus der Verwaltungstabelle gelesen. Die Ergebnistabelle kann nun sortiert nach diesem Attribut gelesen werden. Ein neuer DB-Cursor kann an die richtige Position gesetzt und die ursprüngliche Iteration fortgesetzt werden. Jedem Attribut in der Ergebnistabelle wird hierbei ein festes Sortiermuster zugeordnet. Derzeit werden vier vorgenerierte Sortiermuster in jeweils auf- und absteigender Reihenfolge unterstütz. Soll das Sortierkriterium geändert werden, so sieht die Sortierkomponente zunächst in der Ergebnistabelle nach. Findet sie dort in der entsprechenden Spalte bereits eine Sortierung vor, genügt ein simples „ORDER BY“, um die Ergebnismenge wie gewünscht zu sortieren. Ist in der entsprechenden Spalte noch keine Sortierreihenfolge eingetragen, so wurde dieses Sortiermuster bisher noch nicht verwendet. Die Sortier-Komponente muss somit eine entsprechende Sortierung erst noch vornehmen. Hierzu werden mittels des Metamodells alle benötigten Tabellen ermittelt, ein Verbund gebildet und entsprechend der gewünschten Attribute (Sortiermuster) sortiert. In der Ergebnistabelle wird daraufhin in der dem Sortiermuster entsprechenden Spalte die Sortierreihenfolge eingetragen. Auf diese Weise kann eine wiederholte Ausführung ein und derselben Anfrage vermieden werden, da die Sortierfolge für jedes Sortiermuster erst bei ihrer erstmaligen Verwendung erzeugt wird.

Um die Evaluierung der verschiedenen Entwurfskonzepte vornehmen zu können, musste nun ein Messverfahren erarbeitet werden. Hierzu wurden in Anlehnung an den TPC-W Benchmark eine Reihe von Konzepten entworfen und Metriken identifiziert. Nun konnte die Evaluierung des Meta-Akad Systems und der verschiedenen Implementierungen des Resultset-Prozessors vorgenommen werden. Dabei stellte sich heraus, dass die zustandslose Variante des Resultset-Prozessors im Falle unbeschränkter Systemressourcen deutlich schlechter skaliert als die zustandsbehafteten Versionen. Dies lag besonders an der hohen Belastung des DBVS durch die vielen Operationen zum Sichern und Wiederherstellen von Zustandsinformationen durch den (zustandslosen) Resultset-Prozessor. Dies führte sogar dazu, dass andere Komponenten (etwa die Query-Engine) die häufig auf das DBVS zugreifen hierbei ebenfalls ein schlechteres Skalierungsverhalten zeigten als im Zusammenspiel mit den zustandsbehafteten Varianten des Resultset-Prozessors. Die Gründe hierfür liegen im Optimierungsverhalten des eingesetzten Anwendungs-Servers. Dieser kann lesende Zugriffe bei kurzen Lesesperren über eine gemeinsame DB-Verbindung abwickeln. Zusammenfassend lässt sich allerdings sagen, dass die verwendeten Konzepte sich auch im realen Einsatz bewährten und insbesondere bei mehrfachen Umsortierungen große Vorteile boten.

II.5.6. Import und Speicherung von Metadaten

Bei der Einfügung von Metadaten in das System müssen zwei Hauptbestandteile berücksichtigt werden. Zum einen muss das XML-Fragment in der Datenbank gespeichert werden um eine ausreichend schnelle sowie verlustfreie Rekonstruktion zu ermöglichen, zum anderen müssen die Metadaten in einer Datenbankstruktur gespeichert werden, die ein schnelles Suchen über dem Gesamtbestand ermöglicht. Eine Suche ausschließlich auf dem XML-Fragment wäre bei großen Datenmengen viel zu langsam.

Beim Hinzufügen eines Dokumentes wird im ersten Schritt das erhaltene XML-Dokument in die Bestandteile *„learningresource“, „peerReview“* und *„userComment“* zerlegt. Jedes Fragment dieser drei Bereiche wird in einer eigenen Struktur zwischengespeichert und an eine typspezifische Komponente weitergeleitet, wobei Beziehungen zwischen den einzelnen Fragmenten entkoppelt davon weitergeleitet werden. In diesem ersten Schritt werden bereits neue systemweit eindeutige Identifier vergeben.

In der zweiten und deutlich komplexeren Stufe erfolgt die typspezifische Bearbeitung der XML-Fragmente. Ein Parser sorgt dafür, dass alle relevanten Informationen des XML-Dokument in das Datenbanksystem eingetragen werden.

Der Parser durchläuft elementweise das XML-Dokument und tritt für jedes abgeschlossene Element und jedes Attribut in Aktion.

Für diesen Schritt wird das bereits angesprochene Metamodell verwendet. Durch eine Anfrage an das XML-Mappingschema mit dem XML-Pfad des aktuell zu behandelnden Elementes (z.B. *learningresource.creator.organization.origin*) erhält man den genauen Ort (Tabelle, Spalte) in der Datenbank, an dem dieser Wert festgehalten werden soll. Ebenso enthält des Metamodell Informationen über den zu verwendenden Datentyp, vom aktuellen Pfad abhängige Spalten/Tabellen und zur Multiplizität des Elementes. Für nicht-systemkontrollierte Vokabularien erfolgt nun sofort (geschützt durch einen dokumentbasierten Transaktionskontext) der Eintrag in die Datenbank. Im Anschluss daran werden rekursiv alle Abhängigkeiten (Fremdschlüsselbeziehungen) geprüft und zum Schreiben vorgemerkt. Bei systemkontrollierten Vokabularien (bspw. Medientyp) existiert der Eintrag entweder, dann wird ebenso die Erfüllung der Abhängigkeiten ausgeführt, andernfalls wird das Einfügen des Dokumentes abgelehnt, da es gegen systemseitige Einschränkungen verstößt.

Wenn das Fragment vollständig durch den Parser gelaufen ist, werden abschließend alle vorgemerkten Werte tabellenweise gebündelt auf einmal in die Datenbank eingetragen.

Für nahezu alle möglichen Attribute erfolgt die Verarbeitung rein generisch, d.h. alleine aus dem Metamodell können alle Informationen zur Speicherung abgeleitet werden. Einen Sonderfall bieten Schlagworte und Klassifikationen, insbesondere die Klassifikation nach RVK. Hierfür wurden einige besondere, nur teil-generische Lösungen realisiert.

Abschließend erfolgt die Speicherung des kompletten XML-Fragmentes sowie einiger Verwaltungsattribute, die nicht direkt zum Inhalt des XML-Dokumentes gehören (Datum der letzten Änderung, letzter Bearbeiter, etc.).

Die Speicherung des Fragmentes zusätzlich zu den Datenbankstrukturen erfolgt aus zwei Gründen: Zum einen stellt die Speicherung des Fragmentes die verlustfreie Wiederherstellung sicher, vor allem, was die Reihenfolge von Attributen angeht, welche schwer in einem rein relationalen Datenbankschema festzuhalten ist. Zum anderen können Fragmente deutlich schneller an die Benutzer- oder Redaktionsschnittstelle „geliefert“ werden, wenn sie bereits vollständig materialisiert sind, als wenn sie erst zusammengesetzt werden müssten.

Bei Gutachten und Benutzerkommentaren kommt man auf Grund der deutlich einfacheren Struktur mit rein generischen Mitteln aus. Auch hier liegt der Vorteil des generischen Modells klar in der einfachen Erweiterbarkeit des Datenmodells.

Verknüpfungen zwischen einzelnen Dokumenten (*hasPart, hasUserComment*) werden ebenfalls an zwei Stellen realisiert. Zum einen muss die Beziehung zwischen zwei Ressourcen im Datenbankschema festgehalten werden, so dass eine Suche einfach und schnell erfolgen kann. Zum anderen müssen die XML-Fragmente modifiziert werden, da die Beziehung zwischen zwei Ressourcen immer auch dort festgehalten wird.

Die Funktionalität wird durch einen simplen Funktionsaufruf beim XMLPController im EJB-Tier zur Verfügung gestellt und auf unterster Ebene von den typspezifischen Controllern realisiert.

Das Redaktionssystem war bis zum Zeitpunkt dieses Berichts noch nicht mit der kompletten Funktionalität fertiggestellt. Es wird jedoch intensiv daran gearbeitet und die Beendigung dieser Arbeiten steht kurz bevor.

II.6. Arbeitspaket 5: Verbesserung der Qualität der Erschließung bei schlecht erschlossenem Material

II.6.1. Semi-automatische Formalerschließung

Ein großes Problem ist nach wie vor die Erschließung großer Sammlungen von Dokumenten. Hier können geeignete Werkzeuge [Cha02] viele der anfallenden Arbeiten erleichtern oder sogar vollständig übernehmen. Solche Werkzeuge sollten sich dadurch auszeichnen, dass sie nicht nur komfortable Formulare und graphische Benutzeroberflächen anbieten, sondern insbesondere dadurch, dass sinnvolle Vorschläge für das Ausfüllen solcher Formulare gemacht werden. Es ist denkbar, dass Namen erkannt, aus den zu erschließenden Dokumenten extrahiert und dem Benutzer vorgeschlagen werden. Dabei möchte man durch die Verwendung von Text-Mining-Techniken zu solchen Vorschlägen gelangen.

Informationsextraktion

Bei der Informationsextraktion bzw. beim Text-Mining geht es um das Aufspüren und Strukturieren relevanter Informationseinheiten aus einer Menge von unstrukturierten oder semi-strukturierten Texten. Ein wichtiger Teilbereich hierbei beschäftigt sich mit der Erkennung so genannter „Named-Entities“. In den letzten Jahren werden in diesem Teilbereich verstärkt maschinelle Lernverfahren eingesetzt. Viele Text-Mining-Algorithmen wurden entwickelt, um Namen in unstrukturierten Dokumenten zu finden und zu klassifizieren. Unter „Named-Entities“ [Ste01] werden auch Personennamen verstanden. Personennamen in einem Dokument zu erkennen ist eine schwierige Aufgabe. Normalerweise kann man nur, wenn man die interne Struktur des Satzes oder den Kontext genau betrachtet, entscheiden, ob ein Satzteil ein Personenname ist. Allein durch Betrachtung einzelner Wörter kann dies nicht entschieden werden.

Sehr hilfreich ist hierbei der Einsatz von Wörterbüchern bzw. Datenbanken. Hierbei stößt man allerdings auf zwei Probleme: Zum einen dass die Wörterbücher nie vollständig sein können, da die Anzahl der Personennamen nicht beschränkt ist und diese Namen in verschiedenen Formen vorkommen; zum anderen dass die Regeln und Heuristiken, die verwendet werden müssen, nicht alle Fälle abdecken können. Ferner führt der Umstand, dass im Deutschen keine Unterscheidung zwischen Substantiven und Namen in Bezug auf ihre Schreibweise gemacht wird, zu weiteren Problemen.

Text-Mining-Konzepte

Ähnlich wie Data Mining die Analyse strukturierter numerischer Daten kennzeichnet, beschreibt der Begriff des Text Mining eine Menge von Methoden zur (halb-)automatischen Auswertung großer Mengen natürlichsprachlicher Texte. Das Gebiet des Text Mining umfasst vielfältige Methoden zur Extraktion von Informationen aus natürlichsprachlichen Texten.

In diesem Gebiet bzw. in dem Forschungsgebiet der Namenserkennung und Namensklassifikation werden viele Algorithmen vorgestellt und dokumentiert. Bei diesen Namenerkennungssystemen, die Teile der so genannten „Information Extraction Systems“ sind, werden hauptsächlich zwei Ansätze verwendet und zwar der sogenannte „Knowledge Engineering Approach“ und „Automatic Training Approach“ [AI99]. Der hier verfolgte Ansatz folgt dem „Knowledge Engineering Approach“, indem Wörterbücher und Heuristiken benutzt werden, um Namen in Dokumenten zu erkennen.

Beispiel

Beispiel aus einem HTML-Dokument mit Personennamen, die teilweise mit akademischen Titeln eingeführt werden (Abb. 23). In diesem Fall werden mit Hilfe von Vor- und Nachnamenwörterbü-

cher verschiedene Heuristiken angewendet. Das Ergebnis als Vorschlag in den Feldern des semi-automatischen Erschließungswerkzeuges sieht man in Abbildung 24.

Abbildung 23: Beispiel einer HTML-Seite, die Personennamen enthält.

Abbildung 24: Ergebnis als Vorschlag in den Feldern des semi-automatischen Erschließungswerkzeuges.

II.6.2. Semi-automatische Sacherschließung

Nachdem erste Versuche bereits gezeigt hatten, dass ein statistischer Ansatz mit Zuhilfenahme von Wortlisten zur automatischen Klassifikation nach der Regensburger Verbundklassifikation (RVK)[30] nicht die gewünschten Ergebnisse erbringt, wurde über ein alternatives Verfahren nachgedacht [Ber02]. Dabei wurde auch nach einer Möglichkeit gesucht, die automatische Beschlagwortung mit Vokabular der SWD (Schlagwortnormdatei) zu integrieren. Bei den Überlegungen, welcher neue Ansatz zur Klassifizierung und Beschlagwortung gewählt wird, wurde darauf geachtet, dass man nach Möglichkeit ohne Wörterbücher auskommt, da es sehr aufwändig ist, diese zu erstellen. Außerdem sollte ein neuer Ansatz ohne große Anpassung auf verschiedene Sprachen anwendbar sein. Aus diesen Gründen wurde ein Lernverfahren ausgewählt, um aus bereits klassifizierten und beschlagworteten Dokumenten die Informationen, die zur Klassifizierung und Beschlagwortung neuer Dokumente benötigt werden, zu extrahieren. Als eines der besten Lernverfahren hat sich die sogenannte „Support Vector Machine“ (SVM) herausgestellt. Das Lernverfahren „Support Vector Machine“ ist ein noch recht junges Verfahren, das bereits in vielen Anwendungsgebieten die meisten anderen Systeme übertroffen hat. SVM-Verfahren werden nicht nur zur Klassifizierung von Texten genutzt, sondern finden auch in Bereichen der Klassifizierung von Bildern, Schrifterkennung, Objekterkennung und vielen anderen Bereichen Verwendung [Web02].

Beschreibung des Klassifizierungs- und Beschlagwortungssystems

Abbildung 25 zeichnet den Arbeitsablauf des neuen Klassifizierungs- und Beschlagwortungssystems nach:

30 <http://www.bibliothek.uni-regensburg.de/Systematik/systemat.html>

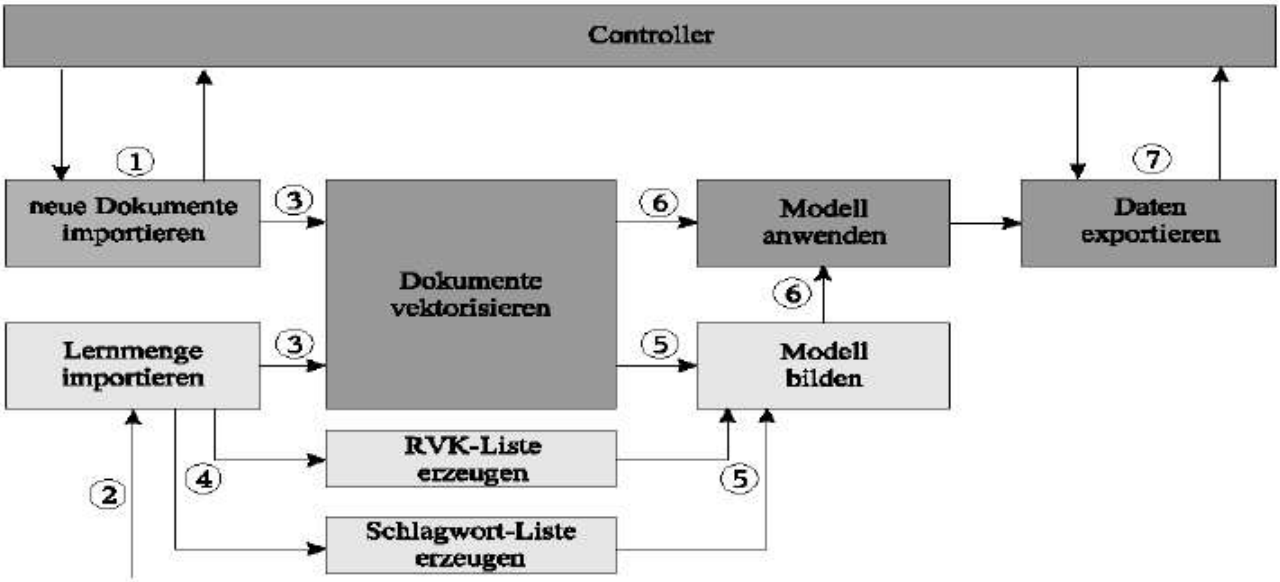

Abbildung 25: Arbeitsablauf der semi-automatischen Sacherschließung.

13. Import neuer Dokumente

    Im ersten Arbeitsschritt wird eine Anfrage an den Controller der zentralen Datenverwaltungskomponente (DVK) gestellt, um alle neuen, noch nicht klassifizierten oder beschlagworteten Dokumente zu suchen. Dieser liefert dann alle relevanten Daten, die für die Klassifizierung und Beschlagwortung dieser Dokumente notwendig sind.

14. Import der Lernmenge

    Hier werden die speziell ausgewählten Dokumente, die typisch für bestimme Klassen und Schlagwörter sind, importiert.

15. Dokumente vektorisieren

    Um mit den bereits importierten Dokumenten arbeiten zu können, müssen diese erstmal in die Vektordarstellung überführt werden. In Abbildung 26 ist die Konvertierung eines Dokumentes in eine Vektorrepräsentation dargestellt. Danach müssen die Wörter nach der Häufigkeit des Vorkommens in den einzelnen Dokumenten gewichtet werden.

16. Listen erzeugen

    In diesem Arbeitsschritt wird aus der Lernmenge, die im zweiten Schritt importiert wurde, eine Liste aller in der Lernmenge vorhandenen Klassifikationen und Schlagwörter bestimmt. Dieses Verfahren ist leider nur in der Lage, Klassifikationen und Schlagwörter zu vergeben, die in der Lernmenge enthalten sind.

17. Modell bilden

    Für jede Klassifikation und jedes Schlagwort muss ein Modell gebildet werden, mit dem man für neue Dokumente entscheiden kann, ob sie zu einer bestimmten Klasse gehören oder nicht. Dabei teilt man für jede Klasse die Lernmenge in eine Positiv- und eine Negativmenge auf. Ziel ist es nun eine Funktion zu finden, die diese beiden Mengen linear trennt. SVM bildet ein System zum Lernen von linearen Funktionen in einem kerninduzierten Merkmalsraum unter Berücksichtigung der Ergebnisse der Generalisierungstheorie und der Ausnutzung der Optimierungstheorie. Man versucht lineare Funktionen zu nutzen, da sie recht gut erforscht und einfach realisierbar sind. Jedoch sind lineare Funktionen nicht geeignet, um reale Probleme zu lösen. Daher geht man in einen hochdimensionalen kerninduzierten Merkmalsraum über, in dem viele Probleme linear trennbar werden.

    In Abbildung 27 ist die Überführung in einen höherdimensionalen Merkmalsraum und die Trennung durch eine lineare Funktion dargestellt. In der Anwendung entstehen oft Vektoren mit mehr als 10000 Einträgen.

18. Modell anwenden
    Für alle neuen Dokumente muss nun in jedem Modell getestet werden, ob sie zur Positiv- oder zur Negativmenge gehören. Falls ein Dokument zur Positivmenge gehört, wird die entsprechende Eigenschaft des Models gespeichert.

19. Daten exportieren
    Nun wird jedes Dokument, für das eine Klassifikation oder ein Schlagwort gefunden wurde, von der DVK im XML-Format angefordert. Dann wird das XML-Dokument um die neuen Eigenschaften ergänzt und wieder an die DVK zurückgeschickt.

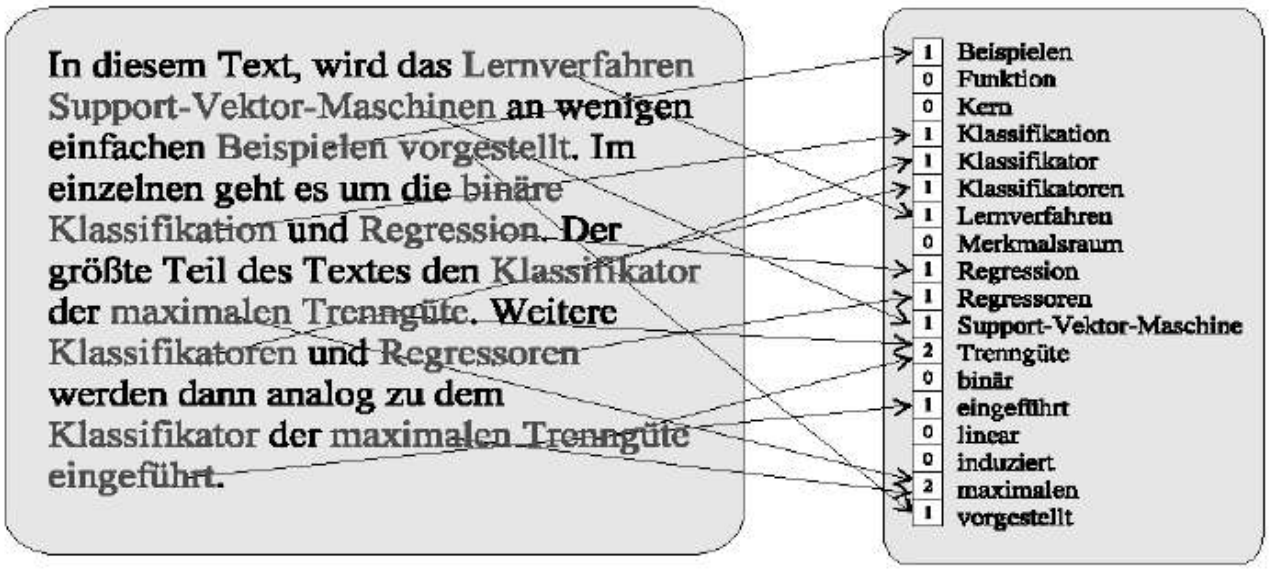

Abbildung 26: Attribute-Wert-Darstellung eines Dokumentes.

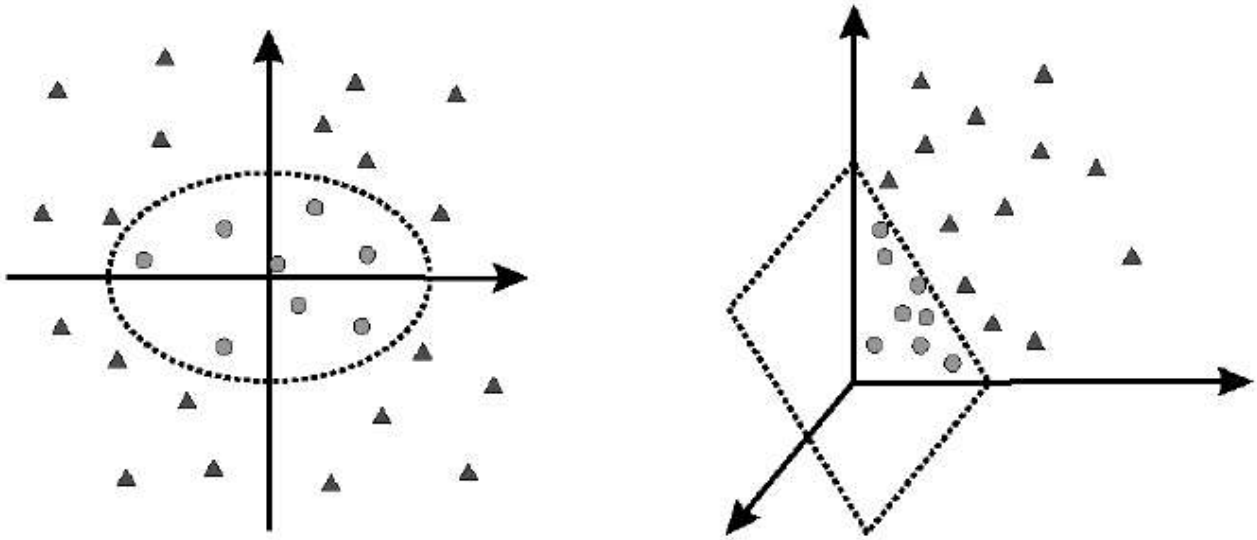

Abbildung 27: Transformation in einen höherdimensionalen Merkmalsraum

Evaluierung

Die bisherigen Testläufe wurden mit deutschsprachigen Dokumenten aus dem Fachgebiet Mathematik durchgeführt. Dabei wurde aus der vorläufigen Linksammlung eine Lernmenge von ca. 700 Dokumenten entnommen. Als Testmenge wurden ca. 170 neue Dokumente gesucht und intellektu-

ell erschlossen. Die im Arbeitsschritt 5 beschriebene Modellbildung kann auf unterschiedliche Weise erfolgen. Bisher wurden zwei unterschiedliche Ansätze untersucht:

20. Virtuelle Klasse
    Bei diesem Ansatz wurden jeweils Kombinationen von Klassifikationen zu einer neuen virtuellen Klasse vereint. Die SVM wurde dann dazu benutzt, um eine Klasse aus der Menge der virtuellen Klassen auszuwählen. Die ausgewählte virtuelle Klasse wurde dann wieder in ihre einzelnen Klassifikationen zerlegt. Die Beschlagwortung erfolgte analog.
21. Binäre Entscheidung
    In dem Fall "Binäre Entscheidung" wird für jede Klassifikation und für jedes Schlagwort einzeln an Hand einer Positiv- und Negativmenge untersucht, ob eine Klassifikation oder ein Schlagwort vergeben werden soll.

Tabelle 1: Automatische Sacherschließung. Vergleich von Ansatz 1 und 2

	Klassifikation: precision	Klassifikation: recall	Beschlagwortung: precision	Beschlagwortung: recall
Virtuelle Klasse	80.3%	59.0%	80.1%	62.4%
Binäre Entscheidung	89.2%	52.7%	90.4%	58.4%

Der Versuch, die Lernmenge mit Daten des Bibliotheksverbundes Bayern (BVB) auf ca. 4000 Dokumente zu vergrößern, brachte eine Verschlechterung des Ergebnisses bei der Klassifikation. Die Precision sank auf 65.2% und der Recall auf 35.9%. Auf Grund dieser Ergebnisse vermuten wir, dass die Struktur der Titeldaten bei Online-verfügbarem Lern- und Lehrmaterial nicht mit dem Datenbestand des BVB vergleichbar ist, denn die Anwendung der Lernmenge auf BVB-Daten brachte wieder bessere Ergebnisse [Web02].

II.7. Arbeitspaket 6: Gestaltung der Benutzerschnittstelle

II.7.1. Konzept und Design

Die Suchschnittstelle für den Endbenutzer soll den Kriterien der Benutzerfreundlichkeit genügen. Ihre Gestaltung orientiert sich dabei eher an den Vorgaben von Web-Suchmaschinen als an klassischen Bibliothekssystemen. Die Benutzerschnittststelle soll so einfach, selbsterklärend und leicht lernbar wie möglich sein, aber dennoch mächtige Funktionen zur Suche bereitstellen.

Die Suchfunktion mit einer formalen Abfragesprache oder einem umfangreichen Formular umzusetzen wäre zwar ein sehr mächtiger Ansatz, entspricht jedoch nicht den Kriterien der Benutzerfreundlichkeit. Die Alternativen sind eine allgemeine Stichwortsuche und eine verzeichnisbasierte Suche. Beide Konzepte sind für sich genommen benutzerfreundlich, jedoch wenig mächtig. Die reine Stichwortsuche würde die besonders hochwertigen Metadaten der bibliothekarischen Sacherschließung (RVK und SWD) nicht genügend für das Retrieval ausnutzen. Das wäre zwar bei einer verzeichnisbasierten Suche der Fall, diese erlaubt aber wiederum keine fein abgestimmten Suchausdrücke. Vielversprechend ist hingegen eine Kombination beider Techniken. Die resultierende Schnittstelle ist zwar etwas komplexer, bietet dem Benutzer aber eine hohe Flexibilität und macht dabei ausgiebigen Gebrauch von den verfügbaren Metadaten.

Die Startseite des Dienstes bietet dem Benutzer zwei unterschiedliche Wege, seine Suche zu beginnen:

22. über die Schnellsuche, die eine Stichwortsuche über den Volltext der Dokumente sowie ausgewählte Metadatenfelder ausführt.
23. über das Browsing in der RVK-basierten Kategorienansicht.

Die Abbildung 28 zeigt die Ergebnisanzeige einer Stichwortsuche. Diese bietet dem Benutzer zwei Arten von Treffern: In „Themengebiete“ und „Schlagwörter“ werden metadatenbasierte Kategorien angeboten, die in den gefundenen Dokumenten besonders häufig aufgetreten sind. Der Benutzer hat über die angebotenen Kategorien die Möglichkeit, zu einer verzeichnisbasierten Suche überzugehen. Die Treffermenge der Dokumente wird mit einigen wichtigen Metadaten angezeigt. Weiterhin erhält der Benutzer die Möglichkeit, die Treffer nach einem oder mehreren Kriterien zu sortieren. Es wird erwartet, dass dadurch ein sehr zielorientiertes Suchen entsprechend den Lernbedürfnissen des Benutzers möglich wird.

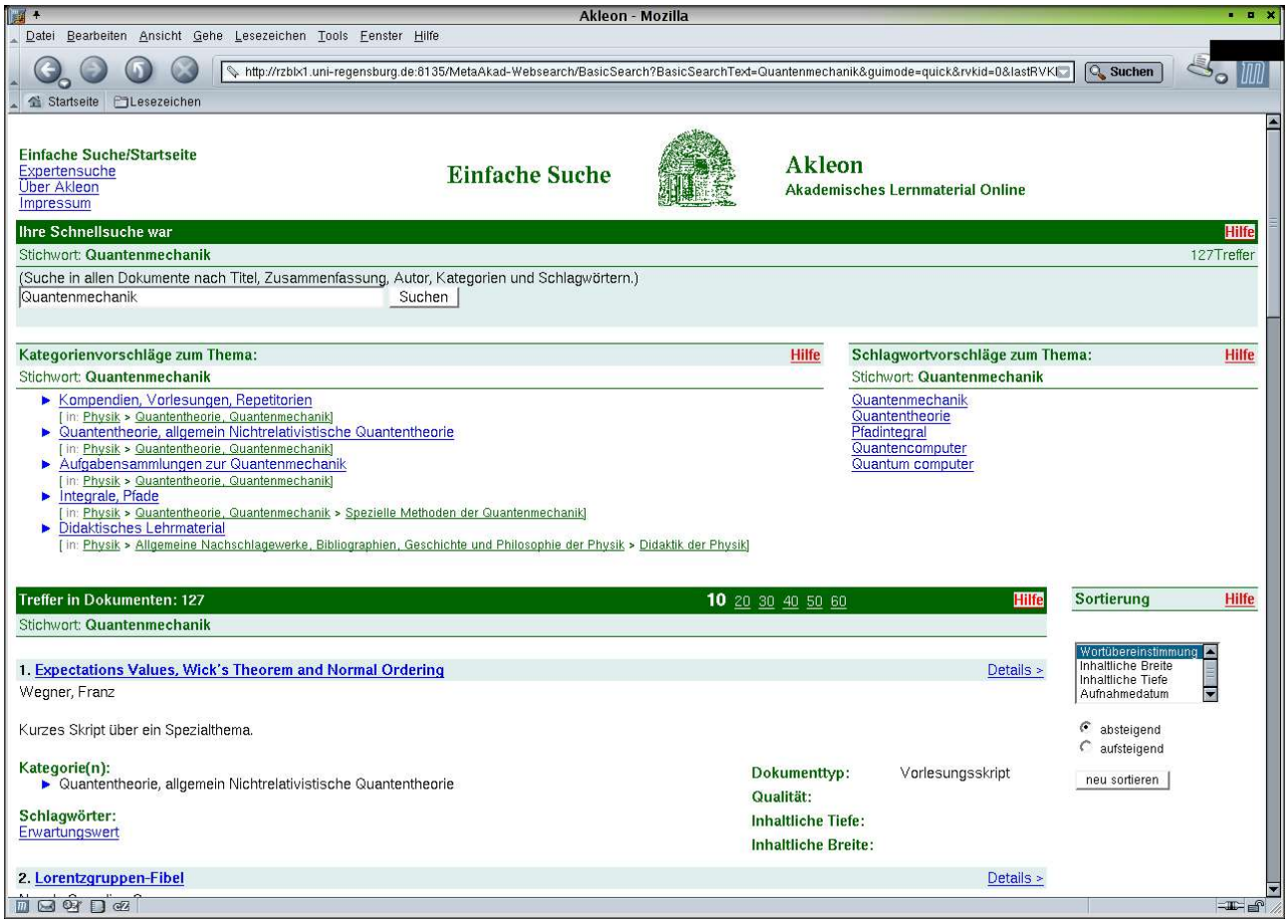

Abbildung 28: Ergebnis einer Stichwortsuche.

In der Detailansicht (Abb. 29) werden dem Benutzer alle verfügbaren Metadaten strukturiert angezeigt. Er hat hier auch die Möglichkeit die Kommentare anderer Benutzer zu lesen und selbst eine Bewertung abzugeben.

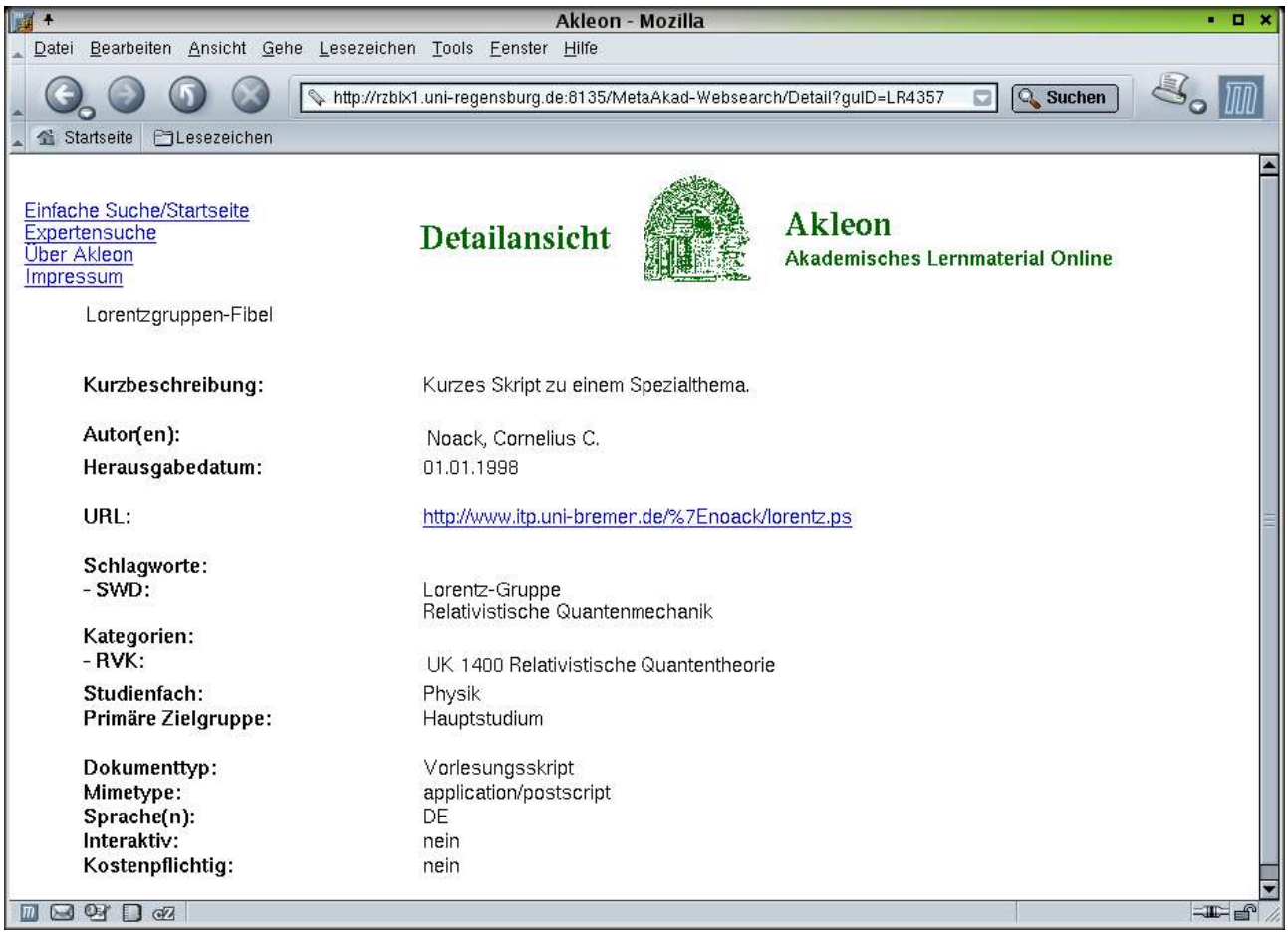

Abbildung 29: Detailansicht einer Ressource mit den verfügbaren Metadaten.

Über die Startseite oder einen kategorialen Treffer kann der Benutzer sich alle Dokumente zu einem bestimmten Thema anzeigen lassen (Abb. 30). Das geschieht entweder über ein bestimmtes Schlagwort oder ein Gebiet der entsprechenden Fachsystematik (hier: RVK). Bei der Auswahl eines Gebiets wird analog zur Stichwortsuche die Schlagwortmenge analysiert und die häufigsten angeboten. Bei der Schlagwortsuche werden sowohl die Themengebiete als auch die Schlagwörter berücksichtigt und als kategoriale Treffer angezeigt.

Zuletzt ist es auch möglich aus einer Kategorie (Schlagwort oder Themengebiet) heraus eine Stichwortsuche zu starten und dabei auf Wunsch die Suche auf die aktuelle Kategorie zu beschränken (Abb. 31). Damit ist eine beidseitige Interoperabilität zwischen stichwortbasierter und kategorialer Suche gegeben.

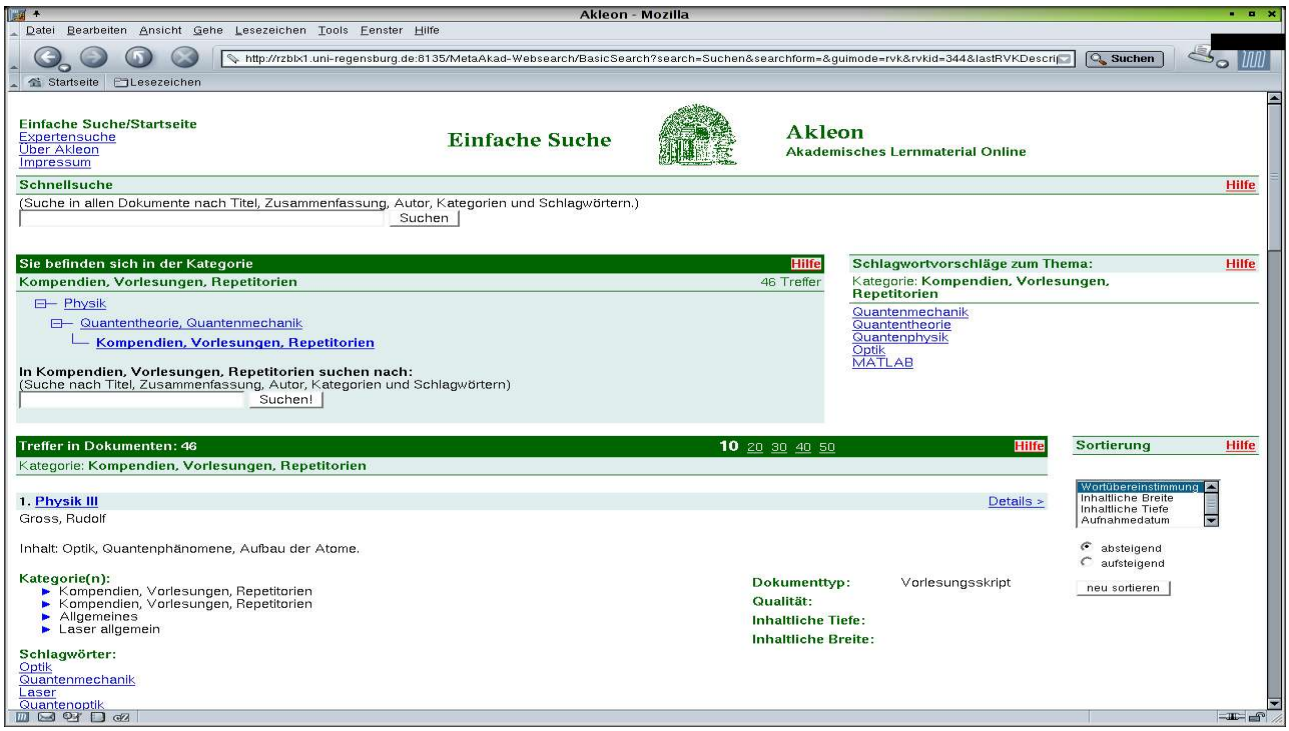

Abbildung 30: Anzeige aller Lerndokumente einer Kategorie (Physik - Quantentheorie, Quantenmechanik - Kompendien, Vorlesungen, Repetitorien).

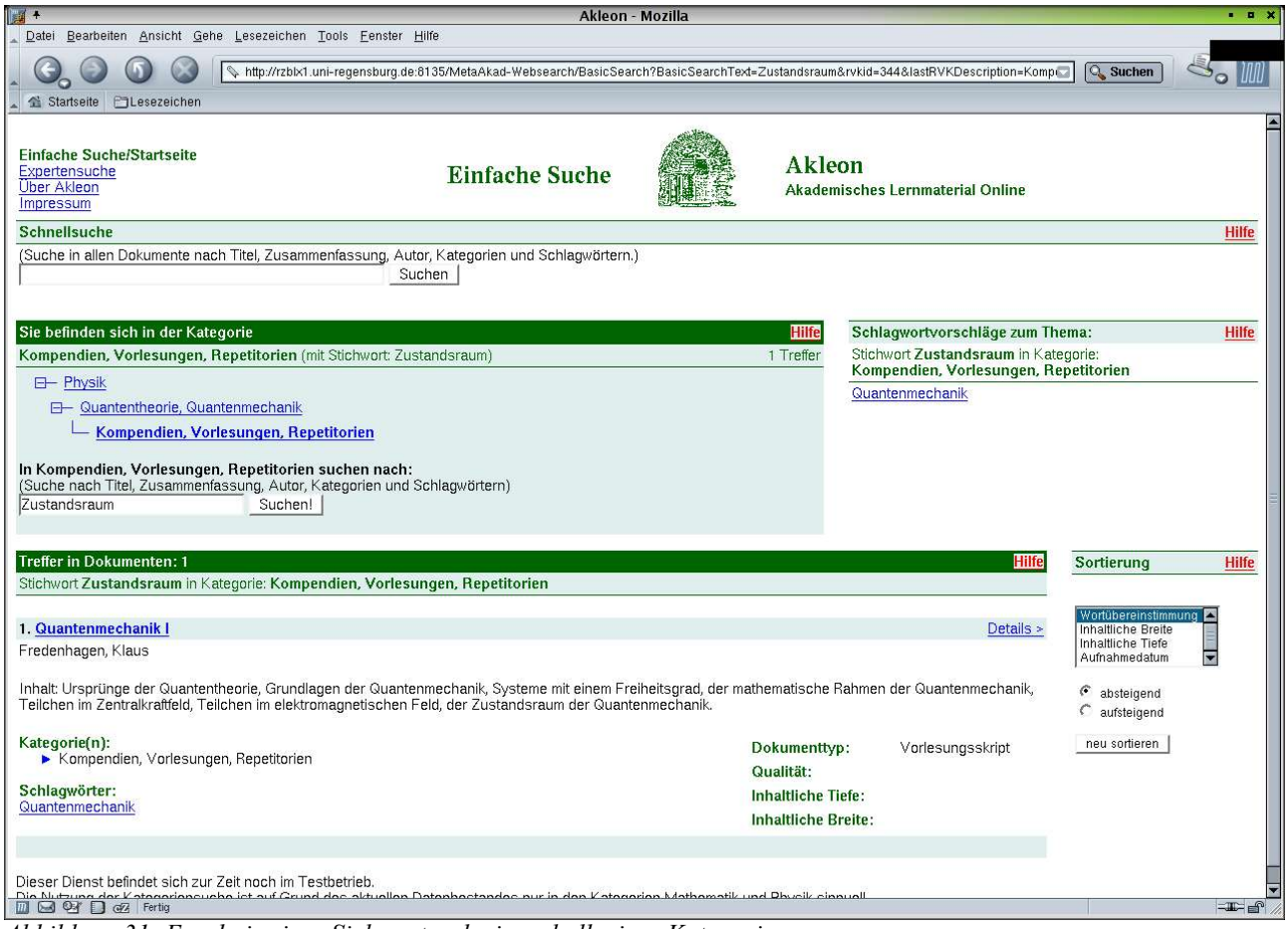

Abbildung 31: Ergebnis einer Sichwortsuche innerhalb einer Kategorie.

II.7.2. Prototyping

Die Benutzerschnittstelle wurde zunächst als statischer Mockup umgesetzt. Dieser diente als erste Diskussionsgrundlage innerhalb des Projektteams. Für eine empirische Evaluation durch Benutzertests war er jedoch nicht geeignet, da die Funktionalität nicht hinreichend abgebildet werden konnte.

Da die Entwicklung der Benutzerschnittstelle auf dem Zielsystem deutlich verzögert war, wurde im Winter 2002/03 ein funktionstüchtiger Prototyp entwickelt. In technischer Hinsicht handelte es sich dabei um ein einfaches Drei-Schichten-System:

24. Datenhaltung: Hier wurden die Datenbank der Linksammlung in Kaiserslautern und eine Kopie der RVK-Online Datenbank verwendet. Beide basieren auf dem relationalen MySQL.
25. Anwendungslogik: Diese Schicht wurde objektorientiert in der Skriptsprache Python implementiert. Die Ausgabe dieser Schicht findet in einem XML-Dialekt statt, der keinerlei Formatierungsanweisungen enthält, sondern nur die Informationen, die auf der Benutzeroberfläche dargestellt werden.
26. Benutzeroberfläche: Der XML-Output der Anwendungsschicht wird durch XSLT-Stylesheets in browserdarstellungsfähiges XHTML umgewandelt.

Diese Architektur wurde gewählt, um während der Benutzertests ad hoc Verbesserungen an der Benutzerschnittstelle durchführen zu können. In den meisten Fällen ist für solche Änderungen nur noch die Anpassung des XSLT-Stylesheets notwendig.

II.7.3. Benutzertests

Mit dem Prototypen wurde eine Benutzerstudie durchgeführt. Das primäre Ziel dabei war das frühzeitige Erkennen von Usability-Fehlern.

Stichprobe und Methoden

Die Probanden in dieser Studie bestand aus 15 zufällig ausgewählten Studenten, die durch die Teilnahme an einer Verlosung motiviert wurden. Der Versuchsablauf umfasste ein kurzes Briefing, die Durchführung des Usability Tests und einen abschließenden Fragebogen zur Erfassung der Vorerfahrung des Probanden in Bezug auf (webbasierte und bibliothekarische) Suchsysteme.

Der Usability Test bestand aus sechs Aufgaben, die selbständig zu lösen der Proband gebeten wurde. Die Aufgaben waren unterschiedlich komplex und berührten unterschiedliche Funktionen der Benutzeroberfläche sowie unterschiedliche Sachthemen. Viele der Probanden mussten deswegen fachfremde Aufgaben bearbeiten. Von den Aufgaben waren vier reine Retrievalaufgaben, von denen zumindest zwei einen höheren Komplexitätsgrad hatten als die üblicherweise beobachtbaren Benutzeranfragen in Suchsystemen [Kra00]. Die Versuchsanleitung befindet sich im Anhang.

Während des Tests wurden halbstrukturierte Beobachtungsprotokolle erstellt. In diesen wurden alle Benutzeraktionen, Bedienprobleme und wichtige Kommentare protokolliert. Zu den Bedienproblemen wurde zusätzlich der betroffene Funktionsbereich sowie eine Mutmaßung über die Ursache des Problems (den Designfehler) festgehalten. Aus diesen Rohdaten wurden durch Kategorisierung der Bedienprobleme Variablen gebildet. Ein Bedienproblem wurde im allgemeinen einem bestimmten Funktionsbereich der Benutzerschnittstelle zugeordnet.

Ergebnisse

Die Studie wurde in drei Abschnitten mit je fünf Probanden durchgeführt. Zwischen den Abschnitten wurden die aufgetretenen Bedienprobleme durch Designänderungen bearbeitet. Im

folgenden Durchlauf wurden besonders auf den Erfolg bzw. unerwünschte Nebeneffekte der Änderungen geachtet. In einem Fall schien sich ein solcher Nebeneffekt zu zeigen, so dass eine erneute Modifikation notwendig wurde. Als besonders kritisch stellte sich die Darstellung der RVK-basierten Vorschläge und Kategorien heraus. Dieser Bereich ist zwar funktional bekannten und etablierten Webdiensten (Amazon, Yahoo) nachempfunden, die Benennungen der RVK-Kategorien ist jedoch für diese Art der Darstellung nur bedingt geeignet. In diesem Funktionsbereich wurden daher recht umfangreiche Änderungen durchgeführt. Der Erfolg der Maßnahmen konnte inferenzstatistisch mit einer sehr hohen Effektstärke nachgewiesen werden.

Fast alle Probanden waren in der Lage, mindestens fünf der Aufgaben mindestens annähernd zu lösen. So konnten auch Studenten im ersten Semester mit eher geringen Vorerfahrungswerten gut eine komplexe Retrievalaufgabe im Bereich der Teilchenphysik zu bearbeiten. Insgesamt deuten diese Ergebnisse auf eine allgemein recht gute Benutzbarkeit der Oberfläche hin.

II.7.4. Implementierung und Test

Nach Abschluss des Usability Tests wurde der Prototyp dazu verwendet, den HTML-Code der Benutzeroberfläche browsertauglich zu machen. Das war notwendig, da an der Universität Regensburg noch Netscape 4.xx als Standardbrowser eingesetzt wird. Dieser ist technisch veraltet, stellt moderne HTML/CSS-Konstrukte fehlerhaft dar und zeigt unter bestimmten Umständen weitere Darstellungsprobleme (z.B. mehrfach bei verschachtelte Tabellen). Damit wird ein speziell optimierter HTML/CSS-Code nötig, um diesen Browser zu unterstützen.

Daraufhin wurde die Benutzerschnittstelle auf das Zielsystem portiert. Dabei kamen im wesentlichen Java Server Pages (JSP) zum Einsatz. Während dieser Arbeit trat ein zusätzliches Problem zu Tage: Das Navigieren in der Kategorienansicht war bisher nur als synchroner Modus umgesetzt worden, indem die Auswahl einer Kategorie immer die Generierung und Darstellung einer neuen Treffermenge auslöste. Dieser Vorgang war aber aus Gründen der Performanz nicht wünschenswert. Um diesem Problem zu begegnen, wurde die Kategorienansicht neu gestaltet, so dass sie sowohl die asynchrone Navigation (ohne neue Suche) als auch einen synchronen Modus (Aufruf der Kategorie mit Suche) unterstützt. Die Neugestaltung der Baumansicht orientierte sich stark an der üblichen Darstellung in Dateimanagern (z.B. MS Explorer). Damit kann angenommen werden, dass diese Darstellung für die Benutzer erwartungskonform ist und keine neuen Bedienprobleme verursacht.

Nach Abschluss der Implementierung wurde wiederum die Browsertauglichkeit der Oberfläche bewertet und weitere Verbesserungen daran vorgenommen, so dass Kompatibilität zu den meisten im Umlauf befindlichen Browser besteht.

Die Auswirkungen einer verstärkten Nutzung von Multimedia-Dokumenten mit sehr hohen Datentransferraten auf das G-Win konnten bisher nicht untersucht werden. Solche Belastungstest zur Netzlast sollen aber noch stattfinden. Die Ergebnisse werden nachgereicht werden. Die Datentransferraten beim Sammeln von Dokumenten bei der semiautomatischen Suche nach Lernmaterial mit Harvest bzw. ASPSeek waren teilweise erheblich, wurden zumeist aber ausdrücklich beschränkt, um eine Überlastung fremder Server unbekannter Leistungsfähigkeit zu verhindern.

II.8. Arbeitspaket 7: Evaluation von META-AKAD durch die Benutzer

Ziel dieses Arbeitspakets war es, die Qualität des Dienstes in Hinblick auf Funktionalität und Informationsangebot empirisch zu erfassen, zu evaluieren und die Ergebnisse für eine Verbesserung des Dienstes zu nutzen.

Durch die starke Verzögerung in den anderen Arbeitspaketen konnte die abschließende Evaluation noch nicht durchgeführt werden. Allerdings wurde schon in der frühen Planungsphase des Projekts

erkannt, dass eine vorausschauende Planung in Hinblick auf die benutzerorientierte Gestaltung sinnvoller ist als eine post hoc Evaluation.

Daher wurden im gesamten Projektverlauf erhebliche Anstrengungen unternommen, um diesem wichtigen Aspekt gerecht zu werden:

Zunächste wurde die verfügbare Literatur zur *Usability von Information Retrieval Systemen* zu Rate gezogen. Die wesentlichen Ergebnisse dabei waren

- Die wenigsten Benutzer bevorzugen Suchschnittstellen mit komplexen Formularen oder formalen Sprachen
- Wenn formale Abfragesprachen benutzt werden, sind häufige Bedienfehler zu beobachten (z.B. Verwechslung des logischen UND und ODER)
- Meistens werden verhältnismäßig einfache Suchanfragen verwendet, die nur wenige Metadatenfelder (wenn vorhanden) nutzen (z.B. Titelwort und Autor)
- Die meisten Benutzer bevorzugen allgemein eine Volltextsuche vor einer kategorialen Suche.
- Ob eine kategoriale Suche oder eine Volltextsuche geeigneter ist, hängt stark vom aktuellen Informationsbedürfnis ab (gezielte Suche vs. Stöbern)

Daraufhin wurden einige existierende Digital Library Systeme einem Vergleich unterzogen, um festzustellen, wie gut sie den genannten Anforderungen entsprechen. Dabei zeigte sich, dass diese Systeme sich noch stark an die Gestaltung klassischer bibliothekarischer Recherchesysteme (z.B. OPAC) anlehnen und nicht als besonders benutzerfreundlich gelten können. Eine detaillierte Darstellung dazu findet sich im Anhang unter „Verwaltungssysteme für Online-Lernmaterial. Ein benutzerorientierter Vergleich“.

Vielversprechender erschien der Ansatz sich bei Funktionalität und Gestaltung an modernen Web-Suchdiensten zu orientieren. Eine detaillierte Begründung findet sich unter „Gestaltung der Schnittstelle für den Endbenutzer“ im Anhang.

Auch bei der Definition der Metadaten blieb der Aspekt der Benutzerorientierung nicht unberücksichtigt. Gerade bei der Qualitätsbeurteilung wurden Metadaten mit aufgenommen, die dem Benutzer eine sehr zielorientierte Suche in der Sammlung ermöglichen, z.B. die inhaltliche Tiefe und Breite.

Wie bereits in Kap. II.7 beschrieben, wurde die Benutzerschnittstelle sorgfältig entworfen und getestet. Hier fehlt bislang noch eine Studie, die die Benutzerzufriedenheit explizit erfasst. Diese Studie befindet sich jedoch in Vorbereitung. Dazu wurde ein standardisierter Fragebogen zur Erfassung der Benutzerfreundlichkeit, der IBM Post-Study System Usability Questionaire (PSSUQ) [Lew95], bereits ins Deutsche übersetzt.

Insgesamt konnte dieses Arbeitspaket aufgrund der Verzögerungen bei der Systementwicklung nicht wie im Antrag vorgesehen durchgeführt werden. Wir sind jedoch der Überzeugung, das dahinter stehende Ziel durch eine Vielzahl anderweitiger Maßnahmen voll erfüllt zu haben.

III Zukünftige Entwicklung

III.1. Weiterbetrieb

AKLEON wird nach Projektende als regulärer Dienst kooperativ von den Universitätsbibliotheken Regensburg und Kaiserslautern aus laufenden Mitteln weitergeführt werden. Die Datenbank, der J2EE-Server und der Webserver für den öffentlichen Dienst AKLEON werden wie bisher in Kaiserslautern verbleiben und von der Universitätsbibliothek in Zusammenarbeit mit der AG Datenbanken und Informationssysteme betrieben werden. Das Such- und Sammelunterstützungssystem Pony wird an der Universitätsbibliothek Regensburg verbleiben und dort weitergepflegt werden.

III.2. Weiterentwicklung

Über den bloßen Weiterbetrieb hinaus soll der Dienst AKLEON auch weiterentwickelt und ausgebaut werden. Dies soll aus Eigenmittel und aus Sondermitteln der beteiligten Universitätsbibliotheken finanziert werden. Zunächst werden die noch ausstehenden Aufgaben des Projekts Meta-Akad erfüllt. Hierfür stehen Eigenmittel zur Verfügung:

- Redaktionssystem (wird in Kürze erledigt)
- Gutachtersystem
- Test der Belastung des G-Win durch die verstärkte Nutzung von Multimedia-Dokumenten

Für die Weiterentwicklung werden die beteiligten Einrichtungen zusätzliche Gelder einwerben. Hierfür bestehen relativ gute Aussichten. Die wichtigsten Ziele sind dabei:

- Ausbau der Sammlung von Lehr- und Lernmaterialien in AKLEON: Dies ist für die gesamte Bandbreite der wichtigsten Wissenschaftsfächer notwendig. Hierfür soll die in Meta-Akad erarbeitete Software eingesetzt werden.
- Mit der Virtuellen Hochschule Bayern soll ein maschineller Datenaustausch eingerichtet werden.
- Nutzung von AKLEON für die Speicherung eigener Lehr- und Lerndokumente auf dem Server.
- Einbindung in das Projekt Metadaten Registry der AG der Verbundsysteme. Dieses Projekt wird in das Projekt Vascoda (BMBF und DFG) integriert. Die Daten werden damit über das Portal vascoda[31] zugänglich.
- Ausbau des Gutachtersystems, das in AKLEON entwickelt worden ist.
- Schulung und Öffentlichkeitsarbeit an den Universitäten, um sowohl die Nutzung von AKLEON zu fördern als auch zur Mitarbeit anzuregen.
- Entwicklung einer OAI-Schnittstelle.
- Erarbeitung von Schnittstellen zwischen Lernplattformen und AKLEON.
- Erarbeitung eines Workflows für die Verwaltung von selbsterstellten und gekauften Lehr- und Lernmaterialien in Zusammenarbeit mit den Rechenzentren und Multimediazentren.
- Erarbeitung eines Konzepts für Rechte und Zugriffsverwaltung in Zusammenarbeit mit der vhb und den Hochschulen.

31 <http://www.vascoda.de/>

- Erarbeitung eines Konzepts zur Integration von Servern von Dritten, z. B. www.knowledge-bay.de (eine studentische Initiative an der Universität Regensburg).
- Integration von Lehr- und Lernmaterialien in die bibliothekarischen Nachweisinstrumente.
- Zusammenarbeit mit Hochschuldidaktischen Zentren.

IV Literatur

IV.1. Eigene Literatur

[Aml03] Amlinger, C., Komponente zur Web-Service-unterstützten Integration von XML-Dokumenten in einrelationales DBS im Rahmen des Projekts Meta-Akad, Projektarbeit, FB Informatik, Universität Kaiserslautern, 2003.

[Ber02] Berscheid, F., Ablaufkoordination, automatisiertes Sammeln und Erkennen von Lehr-/ Lernmaterialien im Projekt META-AKAD, Diplomarbeit, FB Informatik, Universität Kaiserslautern, Mai 2002.

[Cha02] Chatti, A., Einsatz von Text-Mining-Algorithmen bei der Realisierung eines semi-automatischen Erschließungswerkzeugs im Projekt Meta-Akad, Projektarbeit, FB Informatik, Universität Kaiserslautern, Dezember 2002.

[FFW02] Flehmig, M., Fudeus, S., Weber, C., Realisierung einer Web-basierten Suchschnittstelle für die Verwendung in Meta-Akad, interner Praktikumsbericht, FB Informatik, Universität Kaiserslautern, August 2002.

[Fle03] Flehmig, M., A Scalable Component-based Architecture for Online Services of Library Catalogs (Supporting Digital Libraries Utilizing J2EE), to be submitted, 2003

[Fud03] Fudeus, S., Generische Verwaltung von XML-basierten Daten in relationalen DBS (Projekt MetaAkad), Projektarbeit, FB Informatik, Universität Kaiserslautern, 2003.

[Gau03] Gauß, B., Skalierungskonzepte in datenintensiven, mehrschichtigen und verteilten Anwendungensarchitekturen (J2EE) und ihre quantitative Bewertung (Projekt Meta-Akad), Diplomarbeit, FB Informatik, Universität Kaiserslautern, 2003.

[GW1] M.E. Berbenni, G. Weber, H. J. Jodl, Neues Portal für elektronische Lehr- und Lernmaterialien: <http://www.akleon.de/>, eingereicht bei Physik und Didaktik in Schule und Hochschule: <http://www.phydid.de/.>

[GW2] Maria Elisabetta Berbenni, Gisela Weber, Hansjörg Jodl Neuer Dienst für elektronische Lehr- und Lernmaterialien: <http://www.akleon.de/>, eingereicht bei Plus Lucis: <http://pluslucis.univie.ac.at/>

[HJJ] H. J. Jodl, Report on Available Multimedia Material for a Lecture in Quantum Mechanics, 7th Workshop on Multimedia Physics Teaching and Learning of the European Physical Society, Parma (Italy), 22-24 September 2002.

[Kre03] Krennrich, K., Möglichkeiten der Identifikation und Notifikation von Änderungen bei Web-basierten Dokumenten durch Hashcodes, Projektarbeit, FB Informatik, Universität Kaiserslautern, 2003.

[MB] M. Benedict, E. Debowska, H. J. Jodl, L. Mathelitsch, R. Sporken, Recommendations for material on quantum mechanics and for evaluation criteria, 7th Workshop on Multimedia Physics Teaching and Learning of the European Physical Society, Parma (Italy), 22-24 September 2002.

[Tuc03] Tuchbreiter, J., Integration und Caching von Volltextanfragen in MetaAkad, Projektarbeit, FB Informatik, Universität Kaiserslautern, 2003.

[Wag02] Wagner, B., XML-basierte Anfrageverarbeitung der Datenverwaltungskomponente im Rahmen des Projekts Meta-Akad, Projektarbeit, FB Informatik, Universität Kaiserslautern, 2003.

[Web02] Weber, C., Realisierung einer automatischen Klassifizierungs- und Beschlagwortungskomponete im Rahmen des Projekts META-AKAD, Projektarbeit, FB Informatik, Universität Kaiserslautern, 2003. <http://kluedo.ub.uni-kl.de/volltexte/2003/1567/>

IV.2. Weitere Literatur

[AI99] Appelt, D. E., Israel, D. J., Introduction to Information Extraction Technology, Tutorial for IJCAI-99, Stockholm, Sweden, 1999.

[BS95] Borghoff, U. M., Schlichter, J. H., Rechnergestützte Gruppenarbeit — Eine Einführung in Verteilte Anwendungen, Springer- Lehrbuch, Berlin Heidelberg, Deutschland, 1995.

[Gri98] Griffel, F., Componentware- Konzepte und Techniken eines Softwareparadigmas, dpunkt-Verlag, Heidelberg, Deutschland, 1998.

[Koz99] Kozaczynski, W., Composite Nature of Components, International Workshop on Component-based Software Engineering, Pittsburgh, USA, 1999.

[Kra00] Kralj, Andre, Softwareergonomische Aspekte eines Bibliotheksrecherchesystems, Magisterarbeit, Universität Regensburg, 2000.

[Lew95] Lewis, James R., IBM Computer Usability Satisfaction Questionaires: Psychometric Evaluation and Instructions for Use, International Journal of Human-Computer Interaction 7(1),1995, S. 57-78.

[Mit95] Mitschang, B., Anfrageverarbeitung in Datenbanksystemen — Entwurfs- und Implementierungskonzepte, Reihe Datenbanksysteme, Vieweg, 1995.

[Nie93] Nielsen, Jakob, Usability Engineering, Morgan Kaufmann: San Diego, USA, 1993 <http://www.useit.com/papers/heuristic/heuristic_list.html>

[Sch99] Schmidt, D., How to make Software Reuse Work for You, Technical Report, C++.Report, 1999.

[Ste01] Steiner, I., Warum “Named Entities“ für die Chunk-Analyse wichtig sind, in: Proc. der GLDV-Frühjahrstagung 2001, Henning Lobin (Hrsg.), Universität Gießen, März 2001, pp. 245-252.

[Szy98] Szperski, C., Component Software — Beyond Object-Oriented Programming, Addison-Wesley, 1998.

V Anhang

1. META- AKAD Metadata Element Set and Structure
2. Kontrollierte Vokabularien der Metadatenelemente
3. XML-Schema der META-AKAD-Metadaten
4. Dokumentation des XML-Schemas der META-AKAD-Metadaten
5. CCG Benutzeranleitung
6. Sammeln von Lerndokumenten: Strategie und Softwareunterstützung
7. Pony Bedienungsanleitung
8. Qualitätsbeurteilung von Web-Lernobjekten: Erweiterung der Metadatendefinition
9. Benutzerschnittstelle für das kollaborative Erschließen von Dokumenten: Ein Rahmenkonzept
10. Versuchsanleitung der Usability Studie
11. Verwaltungssysteme für Online-Lernmaterial. Ein benutzerorientierter Vergleich
12. Gestaltung der Schnittstelle für den Endbenutzer