Otto-von-Guericke-Universität Magdeburg
Institut für Technische und Betriebliche Informationssysteme

Preprint

Multimediaunterstützung von Vorlesungen – Ein Erfahrungsbericht –

Ingo Schmitt Dirk Jesko Gunter Saake

6. November 2003

Ingo Schmitt, *Dirk Jesko*, *Gunter Saake*
Otto-von-Guericke-Universität Magdeburg
Institut für Technische und Betriebliche Informationssysteme
Postfach 4120

39016 Magdeburg

{schmitt|jesko|saake}@iti.cs.uni-magdeburg.de

Zusammenfassung

Am Datenbanklehrstuhl der Otto-von-Guericke-Universität Magdeburg wurden in den letzten Jahren zwei Datenbankvorlesungen per Video aufgezeichnet, multimedial aufbereitet und den Studierenden über eine Webschnittstelle (`http://mmdb.cs.uni-magdeburg.de/`) verfügbar gemacht. Allgemeines Ziel war eine vorlesungsbegleitende Unterstützung von Studierenden bei der Wiederholung von Lehrinhalten und bei der Prüfungsvorbereitung. Im ersten Teil dieses Berichts werden Erfahrungen und Ergebnisse mit dem Multimediaprojekt vorgestellt. Der zweite Teil beinhaltet die ausführliche Dokumentation der Vorgehensweisen, Werkzeuge, etc. Die Bereitstellung der Erfahrungen, Dokumentationen und Werkzeuge soll eine Nachnutzung und eine eventuelle Weiterentwicklung durch andere Bildungseinrichtungen ermöglichen.

Inhaltsverzeichnis

1 Erfahrungsbericht

1.1 Einführung

Der heutige Stand der Multimediatechnologie und deren weite Verfügbarkeit ermöglichen deren Einsatz in der Universitätsausbildung. Insbesondere lässt sich die Qualität der Lehre erhöhen, wenn Studierende neben dem Besuch von Lehrveranstaltungen zusätzliche Möglichkeiten haben, den Lehrstoff selbständig zu vertiefen oder zur Prüfungsvorbereitung geeignet zu wiederholen.

Durch eine multimediale Unterstützung von Vorlesungen werden Lehrinhalte orts- und zeitunabhängig verfügbar. Derzeit existieren mehrere Projekte, die sich damit beschäftigen, einen zentralen Zugang zu Vorlesungsmaterialien aus verschiedensten Fachbereichen zu schaffen. Beispiele hierfür sind die Open-Courseware-Initiative des MIT [Mas03], das Projekt 100-online der Universität Stuttgart [Stu03] oder die World Lecture Hall der Universität Texas [The03]. Die jeweiligen Portale bieten einen zentralen Zugriff auf die Materialien der Veranstaltungen. Diese reichen von Skripten (in unterschiedlichen Formaten), Foliensammlungen (die teilweise an die Präsentation im WWW angepasst wurden) bis hin zu Animationen und Werkzeugen (z.B. als Java-Applets) für Experimente.

Ein erwünschter Effekt ist ein universitätsübergreifender Wettbewerb sowie eine Kooperation bei der Erstellung von Lehrinhalten und Methoden der Wissensvermittlung mit dem Ziel der Erhöhung der Lehrqualität. Nicht zuletzt werden dadurch Lehrinhalte auch für Unternehmen, die Absolventen einstellen wollen, transparenter. Damit kann das Projekt als ein Beitrag gesehen werden, die Kluft zwischen Universitäten und der freien Wirtschaft zu verringern.

In den letzten Jahren wurden in der Arbeitsgruppe Datenbanken der Universität Magdeburg, teilweise im Rahmen des Projekts „MuSofT – Multimedia in der SoftwareTechnik“[1] [ADE+03, MuS03], der Einsatz neuer Medien und Kommunikationstechniken für die Lehre untersucht. Ein Ziel war die Bereitstellung von Materialien zweier Datenbankvorlesungen. Zusätzlich erfolgte die Aufzeichnung der Vorlesungen per Video. Diese wurden anschließend aufbereitet und zusammen mit weiteren Materialien und Informationen (Folien und Literaturhinweise) über eine Webschnittstelle (siehe Abbildung 1.1) verfügbar gemacht.

[1] Speziell im Teilprojekt 1.2 „Entwicklung von Informationssystemen“, welches im Rahmen des Zukunftsinvestitionsprogramms der Bundesregierung unter der Fördernummer 01NM089B gefördert wird.

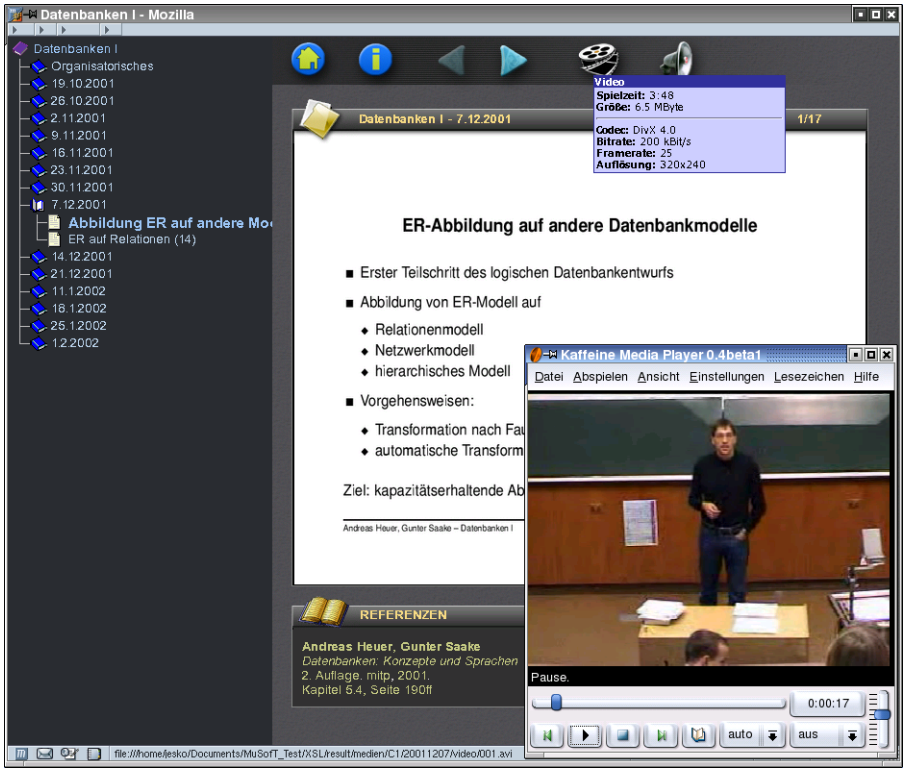


Abbildung 1.1: Screenshot der Vorlesungswebseite

Ziel dieses Berichts ist die Weitergabe unserer Ergebnisse und Erfahrungen an interessierte Dozenten und Dozentinnen, die eine ähnliche Unterstützung ihrer Lehrveranstaltungen anstreben.

1.2 Hintergrund

Ein wichtiges Ziel unseres Projektes lag in der Unterstützung, aber nicht in der Substitution von Vorlesungen. Nach allgemeinen Erfahrungen ist es fraglich, inwieweit eine rein multimedial aufbereitete Ausbildung eine klassische Präsenzveranstaltung mit einem Dozenten ersetzen kann.

Durch unser Projekt sollten Studierende bei der Wiederholung und Vertiefung von Lehrinhalten unabhängig von Zeit und Ort unterstützt werden. Dazu wurde eine Webseite entwickelt, so dass jeder Studierende zu deren Nutzung nur einen gängigen PC, einen Internetanschluss und einen geeigneten Video- und Audiodekoder benötigt. Eine Einschränkung auf ein bestimmtes Betriebssystem (Linux, Windows, ...) sollte vermieden werden. Dies wurde durch den konse-

quenten Einsatz offener, allgemein akzeptierter und unterstützter Technologie wie XML, XSL, HTML, DivX, MP3 und JavaScript erreicht.

Eine weitere Forderung war die Unabhängigkeit von den in der Vorlesung verwendeten Medien. Es soll keine Rolle spielen, ob der Dozent Folien oder eine Powerpoint-Präsentation verwendet. Natürlich müssen auch Tafelanschriften unterstützt werden.

An notwendiger Technik sollte ein schnurloses Mikrofon sowie ein handelsüblicher, digitaler Camcorder mit FireWire-Anschluss ausreichen. An spezieller Software werden ein Webserver (Apache), XML-Software, Schnittsoftware sowie geeignete Codecs benötigt. Als Hardware für den Webserver haben wir einen handelsüblichen PC unter Linux eingesetzt.

Der erforderliche personelle und finanzielle Aufwand für die Erstellung von multimedialen Lehrmaterialien ist häufig für einen breiten, universitären Einsatz zu hoch. In diesem Projekt wurde daher versucht, einen Kompromiss zwischen Aufwand und Nutzen zu finden, in dem Videos direkt von Lehrveranstaltungen aufgenommen werden. Wir benötigten für die Aufnahme und Aufbereitung einer Vorlesung eine studentische Hilfskraft (40h/Monat) für die gesamte Vorlesungszeit.

1.3 Umfang der Lehrunterstützung

In unserem Projekt wurden die Vorlesungen per Video und schnurlosem Mikrofon vor Ort aufgezeichnet. Die Webschnittstelle zeigt als kleinste Struktureinheit Videoclips, die jeweils zu genau einem Folienbild korrespondieren. Damit können Studierende jeweils spezielle Themen auswählen und brauchen nur die für sie interessanten Clips zu laden. Die Zerlegung eines Videos in einzelne, folienorientierte Clips ermöglicht ein genaues Navigieren und spart unnötigen Übertragungsaufwand. Zur Navigation unterstützen wir Sturkturierungsvarianten mit folgenden Strukturebenen:

nach Terminen: Der Zugriff erfolgt an Hand der Vorlesungen und Termine (vgl. Abbildung 2.9):

1. Auswahl der Vorlesung
2. Auswahl des Vorlesungstermins
3. Auswahl des Themas (ein Thema umfasst durchschnittlich ca. 6 Folien)
4. Auswahl der Folie.

nach Themen: Der Zugriff erfolgt an Hand von Themen mit zunehmender Spezialisierung (vgl. Abbildung 2.10):

1. Auswahl eines Themas

2. Auswahl eines Unterthemas (entspricht 3. bei Struktur nach Terminen)
3. Auswahl einer Folie
4. Auswahl des Video/Audio vom gewünschten Datum (es werden alle Termine aufgelistet, die diese Folie verwenden)

Das Video zeigt den Dozenten mit Gestik. Gegebenenfalls ist auch die Erstellung eines Tafelbildes enthalten. Die wichtigste Information ist allerdings das Gesprochene, also das Audio-Signal. Der Nutzer kann weiterhin auswählen, ob nur die Audio-Spur oder auch die Video-Spur präsentiert werden soll. Dadurch lässt sich das Übertragungsvolumen der verfügbaren Bandbreite entsprechend anpassen.

Zusätzlich zu Video- und Audio-Daten wird auf der Webseite die jeweils aktuelle Folie dargestellt. Ein Klick auf die Folie ergibt die entsprechende PDF-Datei. Hinweise bezüglich vertiefender Literatur werden in einem separaten Bereich angezeigt (siehe Abbildung 1.1).

Um einen Überblick über die zugrundeliegenden Daten zu bekommen, zeigt Abbildung 2.1 diese in einem UML-Klassendiagramm. Die Daten selbst sind in einer XML-Datei abgelegt, deren Aufbau in Abbildung 2.2 verdeutlicht und im Abschnitt 2.1 detailliert beschrieben ist. Erfasst werden Informationen zu den Folien und deren inhaltlicher Strukturierung, als auch zu den eigentlichen Veranstaltungen und den dabei erstellten Medien (Video- und Audiodateien).

1.4 Aufnahme und Aufbereitung der Multimediadaten

Das Video wurde von einer studentischen Hilfskraft während einer Vorlesung aufgezeichnet. Da das Video später folienorientiert geschnitten werden musste, wurden die Zeiten der Folienwechsel während der Aufnahme notiert.

Zur Aufbereitung wurde das Video (ca. 20 GB) über die FireWire-Schnittstelle (IEEE 1394) auf eine Festplatte übertragen. Eine Schnittsoftware wurde danach eingesetzt, um die genauen Frame-Nummern für die Folienwechsel anhand der aufgeschriebenen Zeiten zu ermitteln. Das Zerlegen des Videos in einzelne Clip-Dateien und die anschließende Komprimierung und Kodierung geschah automatisch. Dies betrifft auch die Isolierung des Audio-Signals. Diese Zeit wurde genutzt, um notwendige Informationen, etwa Folienüberschriften, in einer XML-Datei abzulegen. Zusätzlich mussten die Vorlesungsfolien als einzelne Dateien in ein dafür bestimmten Verzeichnis abgelegt werden.

Nachdem alle Daten (XML, Folien, Video und Audio) vorbereitet wurden, können die HTML-Dateien für die Webseite mit einem XSLT-Skript automatisch generiert werden.

Zusätzlich zur Bereitstellung im WWW wurden die Dateien auf CD gebrannt und den Studierenden zur Verfügung gestellt. Dabei konnten auf einer CD zirka vier multimedial aufbereitete Vorlesungen untergebracht werden.

Tabelle 1.1: Technische Daten

Merkmal	Quantität
DV-Rohdaten je Vorlesungstermin (90 min)	20 GB
Gesamtes Material für WWW-Seite je Vorlesungstermin	160 MB
davon Videodaten	150 MB
Anzahl Folien je Vorlesungstermin (min/max/avg)	17/46/30
Größe der Videoclips (min/max/avg)	0,3/36,5/5 MB
Dauer eines Videoclips in Minuten (min/max/avg)	0:08/20:13/3:00
Umfang der WWW-Seite der beiden Veranstaltungen	4,5 GB
Zeit für Aufnahme und Bearbeitung eines Vorlesungstermins	6 bis 10 Stunden

Die skizzierte Vorgehensweise ist in Kapitel 2 und auf der WWW-Seite `http://mmdb.cs.uni-magdeburg.de/` detailliert beschrieben. Wir stellen dort weiterhin alle notwendigen Skripte zur Generierung der Webseiten und zur Zerlegung von PostScript-Folien in einzelne Dateien, sowie ein Video über den Aufbereitungsprozess zur Verfügung.

1.5 Erfahrungen und Bewertung

Eine wichtige Erfahrung war der benötigte Aufwand zur Aufnahme und Aufbereitung. Die Arbeiten zur Aufbereitung einer Vorlesung mit 2 SWS wurden durch eine studentische Hilfskraft (40h/Monat) durchgeführt. Zeitintensive, manuelle Arbeiten waren dabei die Aufnahme selbst, das Finden der Frames, an denen ein Schnitt erfolgen sollte und die Erstellung der XML-Datei. Zur Erstellung der Literaturhinweise für die XML-Datei waren Interviews mit dem Dozenten nötig.

An Veranstaltungen wurden im Wintersemester 2001/02 die Vorlesung „Datenbanken 1“ und im Wintersemester 2002/03 die Vorlesung „Multimedia-Datenbanken“ aufgenommen. Der Umfang betrug jeweils 14 Vorlesungstermine mit je 90 Minuten. In Tabelle 1.1 sind einige technische Daten zusammengefasst.

Das Projekt wurde von Studierenden der Veranstaltungen insgesamt positiv aufgenommen. Anfangs stand die Befürchtung, dass diese der Vorlesung fernbleiben werden. Obwohl einige Studierende tatsächlich auf den Besuch der Präsenzveranstaltung verzichteten, besuchte ein Großteil weiterhin die Vorlesung. Dies scheint zu bestätigen, dass die Webseiten von den Studierenden mehrheitlich nicht als Ersatz, sondern als hilfreiche Unterstützung der Lehrveranstaltung angesehen wurden.

Einzelne Studierende konnten aus terminlichen Gründen (Überlappung mit anderen Vorlesungen oder Auslandsaufenthalt) einen Großteil der Präsenzveranstaltungen nicht besuchen, konnten aber durch die Videos den Stoff gut verfol-

Tabelle 1.2: Nutzung des Videos

Kategorie	Vorl.-besucher	Vorl.-abwesende
nie	15%	0%
gelegentlich	61%	47%
häufig	15%	46%
keine Angabe	9%	7%

gen und erzielten in der Prüfung teilweise sehr gute Noten. Insbesondere wurden die Videos (vom Webserver oder lokal von CD) zur Prüfungsvorbereitung eingesetzt. Zwei Wochen vor dem Prüfungstermin (11. März 2002) war eine verstärkte Nutzung des Angebots zu verzeichnen. In Abbildung 1.2 ist beispielhaft ein Auszug der Zugriffsstatistik des Web-Servers für das Jahr 2002 und speziell für den Prüfungsmonat März dargestellt. Der Zeitraum bis einschließlich März gehört dabei zur Vorlesung Datenbanken 1, der Zeitraum ab September zur Veranstaltung Multimediadatenbanken. Die geringere Belastung bei letzterer Vorlesung ist dabei wohl vorrangig auf die geringere Teilnehmerzahl zurückzuführen. Auch in diesem Fall war mit näher rückendem Prüfungszeitpunkt (Februar 2003) ein nennenswerter Anstieg zu verzeichnen.

Am Ende des Semesters führten wir mittels eines Fragebogens eine Evaluation durch. Zunächst war von Interesse, ob seitens der Studierenden die technischen Möglichkeiten für den Zugriff auf die Materialien zur Verfügung standen. Hier hat die Auswertung ergeben, dass etwa 90% der befragten Studierenden entweder das Uni-Netzwerk nutzten oder eine DSL-Verbindung besaßen, so dass auch die Übertragung größerer Datenmengen kein Problem darstellte. Hinzu kam, dass die Studierenden selbst CDs anfertigten und verbreiteten.

Für die Nutzung der Videos und die Besuche der Vorlesung wurden bei der Befragung vier Möglichkeiten von „nie“ bis „häufig“ angeboten. Dabei erfolgte eine Unterscheidung von Studierende, welche die Vorlesungen häufig bzw. immer, und welche die Vorlesungen selten bzw. nie besuchten. Die Ergebnisse sind in Tabelle 1.2 zusammengefasst.

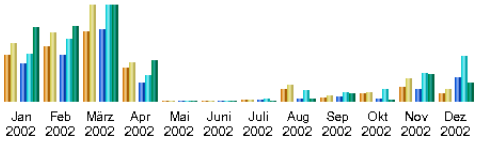
Bezüglich der Fragen, ob das Angebot hilfreich und die Gliederung in kleine Clips vorteilhaft war, zeigte sich, dass etwa 60% der Befragten die thematische Gliederung positiv aufnahmen. Die überwiegende Zahl der Befragten (etwa 70%) sah das Angebot insgesamt als hilfreich an.

Interessanterweise wurde das Video häufig dem Audio vorgezogen. Anscheinend kommen hier psychologische Effekte zum Tragen, die den visuellen Bezug zum Dozenten neben der reinen Stoffvermittlung wertvoll machen.

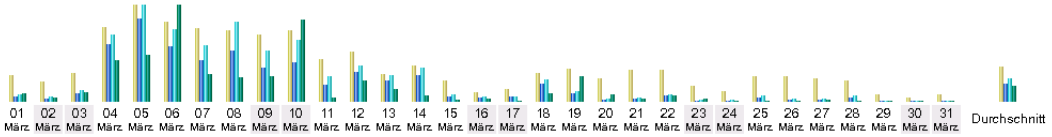
Einige Studierende wünschten neben den geschnittenen Videoclips die kompletten, ungeschnittenen Videos auf CD, da sie die Veranstaltungen komplett wiederholen und nicht von Folie zu Folie navigieren wollten.

Monat/Tag

Erster Zugriff	Zusammenfassung			Letzter Zugriff
01.03.2002 - 02:49	**Monat März 2002**			31.03.2002 - 23:31
Unterschiedliche Besucher	Anzahl der Besuche	Seiten	Zugriffe	Bytes
935	**1276** (1.36 Besuche/Besucher)	**90395** (70.84 seiten/besuch)	**120530** (94.45 zugriffe/besuch)	**32.12 GB** (26392.85 KB/besuch)



Monat	Unterschiedliche Besucher	Anzahl der Besuche	Seiten	Zugriffe	Bytes
Jan 2002	614	773	47147	60090	24.69 GB
Feb 2002	726	912	57832	78270	25.10 GB
März 2002	935	1276	90395	120530	32.12 GB
Apr 2002	444	521	23160	33477	13.80 GB
Mai 2002	0	0	0	0	0
Juni 2002	0	0	0	0	0
Juli 2002	20	23	1417	2780	161.34 MB
Aug 2002	159	217	3697	13659	1.06 GB
Sep 2002	56	75	5539	10921	2.66 GB
Okt 2002	101	127	3184	15545	646.41 MB
Nov 2002	195	305	15940	36052	9.03 GB
Dez 2002	110	157	29981	56800	6.08 GB
Total	3360	4386	278292	428124	115.33 GB



Tag	Anzahl der Besuche	Seiten	Zugriffe	Bytes
01.03.2002	31	877	1196	548.17 MB
02.03.2002	23	514	747	247.09 MB
03.03.2002	33	1356	1856	649.02 MB
04.03.2002	87	9692	11297	2.84 GB
05.03.2002	113	13972	16302	3.21 GB
06.03.2002	94	9315	12253	6.64 GB
07.03.2002	86	7043	9560	1.91 GB
08.03.2002	83	8640	13421	1.65 GB
09.03.2002	78	5784	8521	1.75 GB
10.03.2002	84	6679	10325	5.65 GB
11.03.2002	49	2808	4302	240.70 MB
12.03.2002	58	4996	6054	1.41 GB
13.03.2002	32	3581	4511	899.50 MB
14.03.2002	42	4356	5624	412.87 MB
15.03.2002	25	758	1174	136.73 MB
16.03.2002	11	633	904	167.88 MB
17.03.2002	15	757	831	65.83 MB
18.03.2002	33	3059	3710	570.28 MB
19.03.2002	38	1403	1710	1.73 GB
20.03.2002	27	324	451	461.65 MB
21.03.2002	37	483	698	183.35 MB
22.03.2002	37	976	1258	395.50 MB
23.03.2002	18	180	293	207.04 MB
24.03.2002	12	140	211	18.13 MB
25.03.2002	30	622	988	51.74 MB
26.03.2002	29	318	523	69.65 MB
27.03.2002	27	284	437	82.81 MB
28.03.2002	24	664	1066	30.83 MB
29.03.2002	8	91	147	8.52 MB
30.03.2002	4	22	44	1.11 MB
31.03.2002	8	68	116	20.07 MB
Durchschnitt	41.16	2915.97	3888.06	1.04 GB
Total	1276	90395	120530	32.12 GB

Abbildung 1.2: Netzstatistik

Bei der Auswertung zeigte sich, dass neben den Webzugriffen von unseren Studierenden die Webseite auch von anderen Bildungseinrichtungen kontaktiert wurde. Dies belegt ein allgemeines Interesse an unserem Projekt auch außerhalb der Otto-von-Guericke-Universität Magdeburg.

1.6 Probleme und Verbesserungsansätze

Neben den überwiegend positiven Erfahrungen zeigten sich auch einige Probleme und Ansätze für Verbesserungen. Diese sollen im Folgenden kurz diskutiert werden.

Handarbeit

Aufwändige Handarbeit zur Erfassung der XML-Daten war notwendig. Eine formularbasierte Eingabeschnittstelle könnte den Aufwand und die Fehleranfälligkeit beim Erfassen verringern. Ein Teil der Daten, etwa Folientitel und -nummer, lässt sich bereits während der Aufnahme erfassen.

Ein größeres Problem ist die erforderliche Handarbeit beim folienbasierten Zerlegen der Videos in einzelne folienbasierte Clips (Szenen). Aufwändig dabei ist die manuelle Bestimmung der Frames, bei denen die Schnitte zu erfolgen haben. Ideen zur Verringerung dieses Aufwandes gehen in zwei Richtungen:

1. Videoschnittsoftware unterstützen häufig eine automatische Szenenerkennung, die abrupte Änderungen des Bildinhalts als Szenenwechsel auffassen. Eine solche Änderung kann künstlich während der Aufnahme erzeugt werden, indem etwa ein schwarzes Blatt bei jedem Folienwechsel kurzzeitig vor das Objektiv gehalten wird.

2. Das Video wird, parallel zur Aufnahme auf Videokassette, direkt auf einen Rechner (z.B. einen Notebook) übertragen. Durch die Verfügbarkeit von leistungsfähigen Notebooks mit Festplatten hoher Kapazität stellt dies kein Problem dar. Eine Automatisierung des Videoschnitts ließe sich erreichen, wenn parallel zur Aufnahme ein Programm läuft, welches bei Tastendruck (beim Folienwechsel) den jeweils aktuellen Zeitwert speichert. Wird dieses Programm synchron mit der Videoaufnahme gestartet, könnte eine Job-Liste mit den Schnittzeiten für eine Videoschnittsoftware, etwa VirtualDub, automatisch erstellt werden. Weiterhin lassen sich auch erforderliche XML-Einträge, wie Folientitel und Foliennummer, während der Vorlesung erfassen. Allerdings muss beachtet werden, dass der Aufwand für eine Person bei der Aufnahme nicht zu hoch wird, da nebenbei auch die Kamera bedient werden muss.

Eine direkte Kopplung von Aufnahme und Powerpoint-Präsentation mit dem Ziel, einen Folienwechsel automatisch zu erkennen, ist aus zwei Gründen pro-

blematisch. Erstens sollte der Schnitt bei einer Sprechpause erfolgen, die aber bei einem Folienwechsel nicht vorliegen muss. Zweitens ist eine erforderliche Synchronisation zwischen der Präsentationszeit und der Videozeit nicht trivial.

Aktualisierung einer Vorlesung

Wir sind davon ausgegangen, dass eine neue Vorlesung komplett multimedial aufbereitet wird. In der Praxis werden jedoch in der Regel Vorlesungsunterlagen laufend aktualisiert. Dieser Aspekt der partiellen Aktualisierung von Vorlesungen wurde im Projekt nicht berücksichtigt, ist aber für die Zukunft geplant. Durch die Wahl von XML als präsentationsunabhängiges Format dürften Änderungen in der XML-Datei nicht schwierig sein.

Keine folienbasierte Vorlesung

Im Projekt sind wir davon ausgegangen, dass eine Vorlesung thematisch in einzelne Teile (Folien) von zirka 3 Minuten Länge gegliedert werden kann. Das mag zwar für ein Großteil der Vorlesungen zutreffen, aber nicht unbedingt für alle. Zum Beispiel kann während einer Vorlesung das Skript als Fließtext präsentiert werden. Ein weiteres Problem existiert, wenn in der Vorlesung ein Video oder eine Powerpoint-Animation abgespielt wird, die gesamte Vorlesung an der Tafel erfolgt, oder eine Demonstration an einem Computer stattfindet. Für diese Fälle müssen andere Verfahren zur Aufnahme und Strukturierung solcher Vorlesungsteile gefunden werden.

Anzeigen auf Folie

Während der Vorlesung verweisen Dozenten oft mit einem Zeigestock oder Laserpointer auf bestimmte Stellen einer Folie. Dies kann von der Kamera in der Regel nicht erfasst werden, da dazu die Kamera auf die angestrahlte Projektionsfläche gerichtet werden muss und die automatische Helligkeitseinstellung der Kamera den Folientext nicht gut genug erfasst. Daher fehlt im aufbereiteten Video der Bezug des Gesprochenen zu einzelnen Stellen der Folie. Eine nachträgliche Markierung des Textes und Synchronisation mit dem Gesprochenen ist jedoch nur sehr schwer zu realisieren.

Bewertung des Lerneffekts

Bei der Bewertung der Lerneffekts durch den zusätzlichen Einsatz multimedial aufbereiteter Vorlesungen gibt es unterschiedliche und teilweise sogar gegensätzliche Meinungen. Auf der einen Seite wird die Verbesserung der Möglichkeiten zum Erlernen eines Stoffes positiv bewertet. Auf der anderen Seite steht die Befürchtung, dass der Lerneffekt sinkt. Wenn Studierende vom regelmäßigen Besuch einer Vorlesung absehen, da eine Woche vor der Prüfung der gesamte Stoff anhand der Videos leicht nachgeholt werden kann, dann wird erfahrungsgemäß kein nachhaltiger Lerneffekt erzielt.

Dies ist ein generelles Problem, das immer auftritt, wenn umfangreiche Lernmaterialien zur Verfügung gestellt werden. Nach unserer Auffassung sind Studierende jedoch selbst verantwortlich, mit den erweiterten Möglichkeiten sinnvoll umzugehen. Zukünftige Evaluierungen müssen zeigen, welcher Lerneffekt durch zusätzliches Bereitstellen von Vorlesungsvideos konkret erreicht wird.

1.7 Ausblick

Weitere Arbeiten beschäftigen sich im Wesentlichen mit der Optimierung der Erstellungsprozesse. So sollen insbesondere die verschiedenen Shell-Skripte durch ein einzelnes Programm, z.B. auf Basis von Java oder PHP, ersetzt werden. Ziel ist es dabei die Bedienung intuitiver zu gestalten, indem z.B. die Auswahl der Parameter für den Aufruf des XSLT-Prozessors über eine GUI ermöglicht wird. Eine weitere Verbesserung soll durch die formularbasierte Eingabe der Metadaten über ein GUI erreicht werden. Durch die Nutzung von Java oder PHP wird auch eine plattformübergreifende Nutzung der Werkzeuge möglich. Weiterhin ist geplant, ein Werkzeug zur Zusammenstellung der Folien zu integrieren. Dazu werden in den LaTeX-Quellen die `ids` der Medienobjekte eingefügt. Nach der Auswahl der für eine Veranstaltung benötigten Folien, wird dann ein entsprechender Foliensatz automatisch zusammengestellt.

Weiterhin sollen auch die Metadaten um weitere Elemente aus LOM (Learning Object Metadata) [IEE02] und SCORM (Sharable Content Object Reference Model) [ADL02] erweitert werden. Insbesondere Beschreibungen und didaktische Informationen sollen erfasst werden, um so eine Anbindung an andere LOM-basierte Software zu ermöglichen.

1.8 Zusammenfassung und Danksagung

In diesem Beitrag haben wir über unsere Erfahrungen mit der multimedialen Unterstützung von Vorlesungen berichtet. Insgesamt wurde das Projekt von den Studierenden gut angenommen. Leider ist die Aufbereitung des Lehrstoffs derzeit noch mit einem erheblichen manuellen Aufwand verbunden. Daher sollten Wege gefunden werden, wie der Aufwand gesenkt werden kann. Dies betrifft in erster Linie die Zerlegung eines Videos in einzelne Clips.

Wir haben die entwickelten Werkzeuge zusammen mit einer detaillierten Beschreibung auf einer Webseite abgelegt und damit frei verfügbar gemacht. Damit ist die Hoffnung verbunden, dass unser Ansatz Nachahmung findet.

Unser Dank gilt den studentischen Hilfskräften Niko Zenker, Nico Suchold, Cornelius Huber, Reyk Hillert und im Besonderen Jan Henning und Timo Moeller, die am Projekt engagiert mitwirkten.

2 Technische Details

Die folgenden Abschnitte beinhalten die auch als Webseite verfügbare Anleitung zur Erstellung der Videos. Zunächst wird die Struktur der XML-Datei für die Verwaltung der Metadaten ausführlich beschrieben. Daran anschließend werden der Aufbau der WWW-Seite sowie die Vorgehensweisen und Werkzeuge ausführlich erläutert.

2.1 Verwaltung der Metadaten

Dieser Abschnitt gibt einen Überblick über Informationen, die erforderlich sind, um die WWW-Seite zu erstellen. Insbesondere gehen wir auf die Strukturierung in XML ein.

2.1.1 Übersicht

Die Erstellung der WWW-Seite für die Videomitschnitte der Vorlesungen erfordert eine Reihe von Informationen, z.B. Termine und Titel für die Strukturierung oder technische Informationen (Dateinamen, etc.) für die Generierung der HTML-Dateien. Die Informationen werden in einer XML-Datei gespeichert, so dass sie leicht erweitert und bearbeitet werden können.

2.1.2 Gliederung

Eine Übersicht der verwalteten Metadaten als UML-Klassendiagramm gibt Abbildung 2.1. Die daraus abgeleitete Struktur der XML-Datei ist in Abbildung 2.2 skizziert. Der folgende Abschnitt beschreibt die einzelnen Elemente im Detail. Zunächst soll jedoch auf die Aufteilung der enthaltenen Informationen eingegangen werden. Die Metadaten sind in drei Gruppen gegliedert (vgl. Packages in Abbildung 2.1):

`General`: Diese Gruppe beinhaltet allgemeine Elemente, z.B. `Book`, `VFormat` und `AFormat`, für die Spezifikation der Informationen zu Büchern und den verwendeten Codierungen.

`Content`: Die zweite Gruppe beinhaltet Elemente zur Beschreibungen einzelner Folien und deren Einordnung in eine, an den Standard LOM [IEE02]

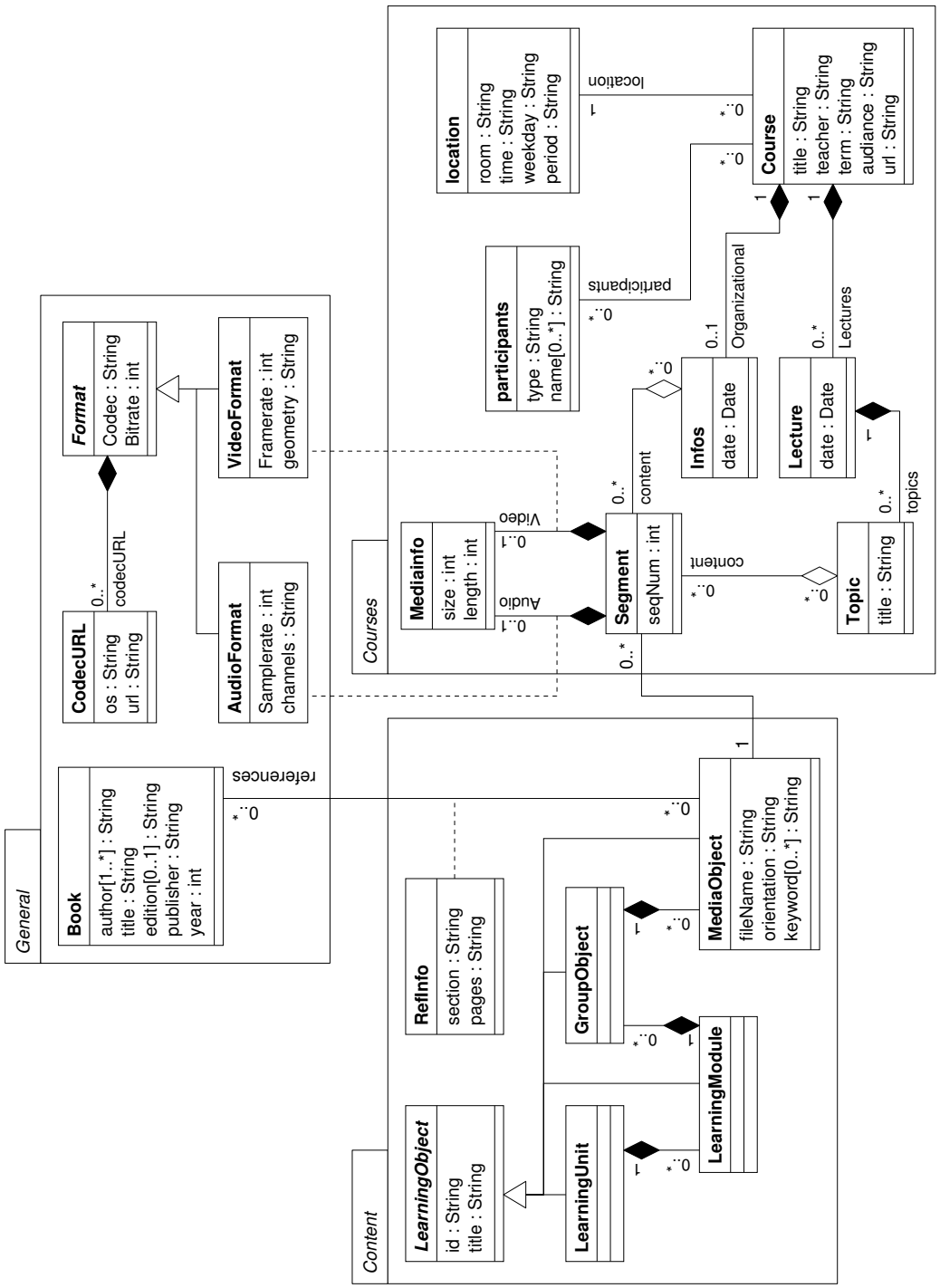


Abbildung 2.1: Daten zur Vorlesung als UML-Klassendiagramm

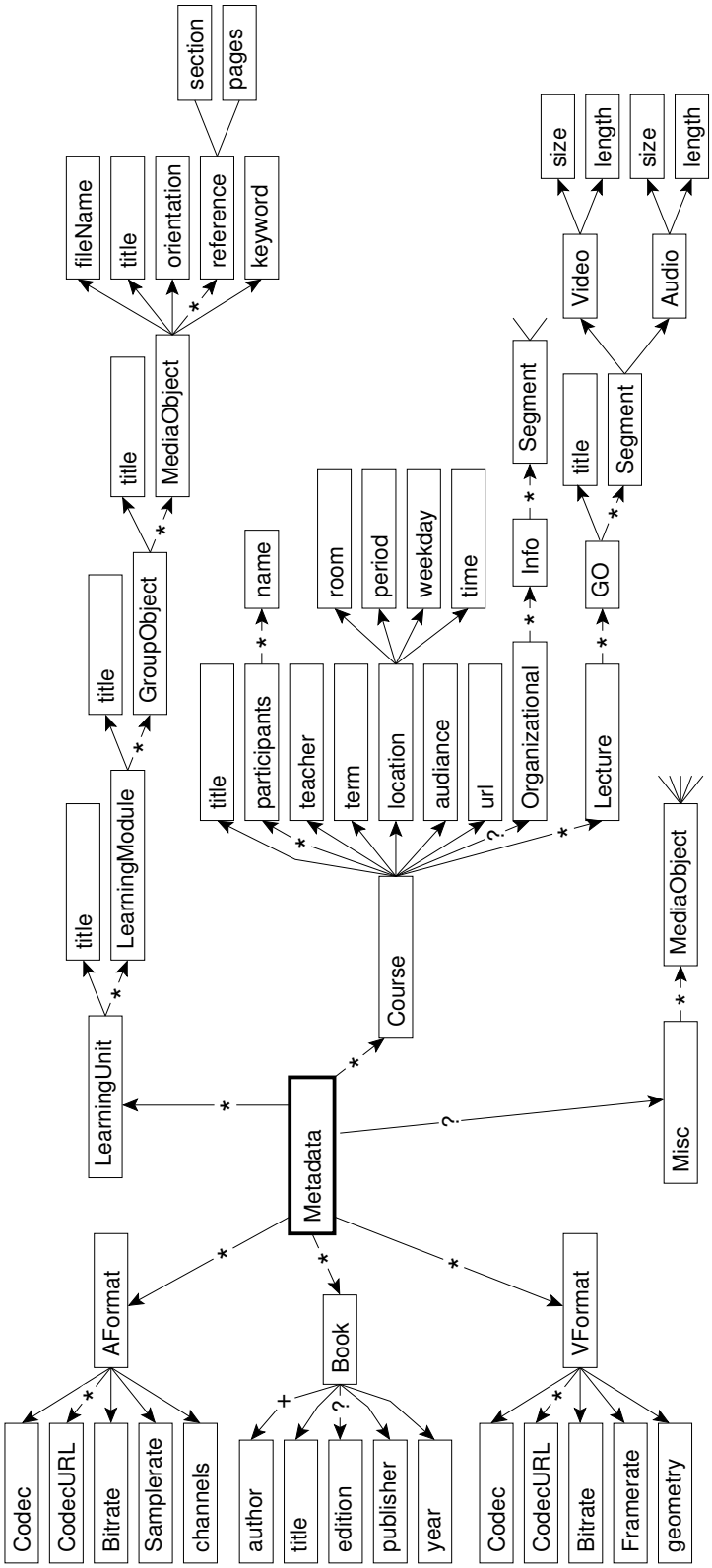


Abbildung 2.2: Elemente der DTD für die Beschreibung der Inhalte und Medien. `Metadata` ist das root-Element. Die Unterteilung von `MediaObject` und `Segment` wurde nur einmal vollständig dargestellt.

angelehnte, vierstufige Hierarchie. Die höchste Abstraktionsebene bildet dabei die `LearningUnit`, die niedrigste das `MediaObject`. Aus praktischen Gründen (Erstellung der WWW-Seite) verwenden wir eine strikte Gliederung, d.h. beispielsweise, dass ein `LearningModule` ausschließlich unterhalb einer `LearningUnit` auftreten kann und selbst wiederum nur `GroupObject`-Elemente enthalten darf. Ein weiterer Unterschied zu LOM besteht darin, dass die Hierarchie implizit durch die Struktur der Elemente in der XML-Datei gegeben ist, während in LOM die Referenzen explizit über entsprechende Elemente angegeben werden.

`Courses`: In dieser Gruppe werden schließlich die Informationen zu konkreten Kursen erfasst. Dies beinhaltet Referenzen zu den verwendeten Folien (`MediaObject`s), Informationen zu den Medien (`Video`, `Audio`) und die Strukturierung in Veranstaltungen (`Lecture`) und Themen `Topic`. Letztere beinhalten lediglich einen Titel für eine Reihe von `MediaObject`-Elemente. Dieser könnte auch aus dem zugehörigen `GroupObject` ermittelt werden. Wir verwenden jedoch aus zwei Gründen eine Kopie des Titels. Zum Einen vereinfacht sich das XSLT-Skript zur Erstellung der HTML-Dateien. Zum Anderen besteht die Möglichkeit, bei Bedarf einen kürzeren Titel für die WWW-Seite anzugeben. Aus dem Diagramm ist ersichtlich, dass die Segmente und somit die `MediaObject`s auf zwei Arten einem `Course` zugeordnet werden können. Zum Einen über `Lecture` und `Topic` und zum Anderen über `Info`. Letztere Zuordnung wurde speziell für Segmente mit allgemeinen Inhalten eingeführt, z.B. solchen mit organisatorischen Informationen. Derartige `MediaObject`s werden auch nicht in die Hierarchie eingeordnet. Während die Angaben in der zweiten Gruppe (`Content`) wiederverwendbar sind, ist dies bei den Informationen zu den Kursen in der Regel nicht der Fall, da sich die technischen Daten der Medien (z.B. Länge und Größe der Clips) mit jeder Veranstaltung ändern, auch wenn die selben Folien verwendet werden.

2.1.3 Erweiterungsbedarf

Derzeit beinhaltet die XML-Datei vorrangig Informationen, die zur Erstellung der WWW-Seite erforderlich sind. Erweiterungsbedarf besteht insbesondere bei didaktischen Informationen, etwa zur Bewertung des Schwierigkeitsgrades der Vorkenntnisse, etc. Auch eine kurze Beschreibung der Inhalte auf Ebene der Learningunits und der Learningmodules ist sinnvoll. Weiterhin sollten Informationen zu den Autoren der Materialien hinzugefügt werden. Ein Nachteil der beschriebenen Vorgehensweise ist weiterhin die Beschränkung auf Folien (PDF- und PNG-Dateien). Um auch andere Medien in die WWW-Seite einzubinden, sind entsprechende Erweiterungen an der XML-Struktur erforderlich. Dabei gilt

es jedoch abzuwägen, ob dann der Einsatz eines Content-Management-Systems oder einer speziellen Lernplattform nicht vorteilhafter wäre. Ein Kompromiss wäre beispielsweise die Möglichkeit, zu jedem `MediaObject` eine Reihe von Hyperlinks anzugeben, die zu weiteren Medien verweisen können. Dies wäre mit relativ wenig zusätzlichem Aufwand realisierbar.

2.1.4 Beschreibung der Elemente

Im folgenden werden die einzelnen Elemente der DTD beschrieben. Weiterhin ist jeweils ein Beispiel mit konkreten Daten aus der XML-Datei der Vorlesung „Datenbanken I“ angegeben. In Abbildung 2.2 ist die Hierarchie der Elemente grafisch dargestellt. Die Bezeichnung der Elemente ergibt sich mit geringfügigen Änderungen aus dem Klassendiagramm in Abbildung 2.1. Es sei darauf hingewiesen, dass ID–IDREF-Beziehungen zwischen den Elementen in der Abbildung der Übersichtlichkeit wegen nicht dargestellt sind. Weiterhin wurde auf die mehrfache detaillierte Darstellung der Elemente `Segment` und `MediaObject` verzichtet. Die folgende Beschreibung der Elemente ist als Referenz zu verstehen. Daher sind diese alphabetisch und nicht entsprechend der Struktur sortiert.

2.1.4.1 AFormat

Elternelement: Metadata

Kindelemente: Codec, CodecURL*, Bitrate, Samplerate, channels

Attribute: `id ID #REQUIRED`

Beschreibung: In diesem Element werden Informationen zu einem bei der Aufzeichnung verwendeten Audioformat angegeben. Jedes Format erhält dabei über das id-Attribut eine eindeutige Bezeichnung. Die `Segment`-Elemente referenzieren das Format über diese id. Wir verwenden als id ein `A` gefolgt von einer Nummer. Es gilt zu beachten, dass die id über alle Elemente eindeutig sein muss.

Beispiel:

```
<AFormat id="A1">
  <Codec>MP3</Codec>
  <CodecURL os="Windows 2000">http://www.divx.com/</CodecURL>
  <CodecURL os="Linux">http://www.divx.com/</CodecURL>
  <Bitrate>24 KBit/s</Bitrate>
  <Samplerate>22,5 kHz</Samplerate>
  <channels>Mono</channels>
</AFormat>
```

2.1.4.2 audiance

Elternelement: Course

Beschreibung: In diesem Element wird die Zielgruppe der Veranstaltung angegeben. Die Angabe wird derzeit jedoch noch nicht ausgewertet.

Beispiel:

```
<audience>CV,Wirtschaftslogistik,Informations- und
 Mikrosystemtechnik, WO:IF,WIF,SPTE</audience>
```

2.1.4.3 Audio

Elternelement: Segment

Kindelemente: size, length

Attribute: `codecId IDREF #REQUIRED`

Beschreibung: Dieses Element stellt Informationen über die Audio-Datei des Segments bereit. Diese Informationen werden auf der WWW-Seite dargestellt. Angegeben werden müssen eine Referenz auf das Format des Audio (siehe id-Attribut des AFormat) sowie Laufzeit und Größe.

Beispiel:

```
<Audio codecId="A1">
  <size>2.2 MByte</size>
  <length>12:55</length>
</Audio>
```

2.1.4.4 author

Elternelement: Book

Beschreibung: Gibt den Namen eines Autors eines Buches an. Sollen mehrere Autoren angegeben werden, können dem Book-Element mehrere `author`-Elemente zugeordnet werden. Jedem Buch muss wenigstens ein Autor zugeordnet werden.

Beispiel:

```
<author>Andreas Heuer, Gunter Saake</author>
```

2.1.4.5 Bitrate

Elternelemente: VFormat, AFormat

Beschreibung: Gibt die Bitrate des Video- bzw Audiostreams an. Der Inhalt des Elements wird ohne Änderung oder Ergänzung auf der WWW-Seite, angezeigt. Der Text ist daher frei formatierbar. Soll eine Einheit angezeigt werden, z.B. `kBit/sec`, so muss diese im `Bitrate`-Element mit angegeben werden.

Beispiel:

```
<Bitrate>200 KBit/s</Bitrate>
```

2.1.4.6 Book

Elternelement: Metadata

Kindelemente: author+, title, edition?, publisher, year

Attribute: `id ID #REQUIRED`

Beschreibung: Im `Book`-Element können bibliographische Angaben zu einem Buch angegeben werden. Verwendet wird ein so definiertes Buch anschließend in den Referenzen eines MediaObject-Element und kann somit einer Folie zugeordnet werden. Wir gehen davon aus, dass vorrangig Referenzen zu Büchern, z.B. Lehrbüchern oder Vorlesungsskripten, angegeben werden. Es ist aber auch möglich Zeitschriften- oder Konferenzbeiträge anzugeben. In diesem Fall bietet es sich an, im `Book`-Element die Zeitschrift bzw. den Konferenzband und in den `pages`- und `section`-Elementen die Angaben zum Beitrag anzugeben.

Das id-Attribut weißt jedem Buch einen eindeutigen Namen zu. Wir verwenden als id die Buchstaben `lb` gefolgt von einer Zahl, also z.B. `lb1`. Es ist zu beachten, dass die id über alle Elemente eindeutig sein muss.

Beispiel:

```
<Book id="B1">
  <author>Andreas Heuer, Gunter Saake</author>
  <title>Datenbanken: Konzepte und Sprachen</title>
  <edition>2</edition>
  <publisher>mitp</publisher>
  <year>2001</year>
</Book>
```

2.1.4.7 channels

Elternelement: AFormat

Beschreibung: Gibt die Anzahl der Kanäle des Audiostreams an. Der Inhalt des Elements wird ohne Änderung oder Ergänzung auf der WWW-Seite, angezeigt. Der Text ist daher frei formatierbar. Es sind daher numerische, aber auch textuelle Angaben, z.B. `Mono`, `Stereo` oder `5.1`, möglich.

Beispiel:

```
<channels>Mono</channels>
<channels>2</channels>
```

2.1.4.8 Codec

Elternelemente: VFormat, AFormat

Beschreibung: Gibt den Namen des verwendeten Codecs an, z.B. `DivX 4.0` oder `mp3`. Diese Information wird unverändert auf der WWW-Seite angezeigt.

Beispiel:

```
<Codec>DivX 4.0</Codec>
```

2.1.4.9 CodecURL

Elternelemente: VFormat, AFormat

Attribute: `os CDATA #IMPLIED`

Beschreibung: Jedem Format können eine oder mehrere URLs zugeordnet werden, die auf Downloadseiten verweisen, z.B. auf verwendete Player. Im os-Attribut kann dabei ein String mit dem Namen des zugehörigen Betriebssystems angegeben werden, falls beispielsweise für unterschiedliche Betriebssysteme auch unterschiedliche Software erforderlich ist. Die Angabe wird derzeit noch nicht ausgewertet.

Beispiel:

```
<CodecURL os="Windows 2000">http://www.divx.com/</CodecURL>
```

2.1.4.10 Course

Elternelement: Metadata

Kindelemente: title, teacher, participants*, term, location, audience, url, Organizational?, Lecture*

Attribute: `id ID #REQUIRED`

Beschreibung: Dieses Element fasst alle Informationen zu einer konkreten Lehrveranstaltung zusammen. Neben den eigentlichen Informationen muss eine id angegeben werden, die für den Aufruf der XSL-Skripte erforderlich ist.

Beispiel:

```
<Course id="C1">
  <title>Datenbanken I</title>
  <teacher>Dr. Ingo Schmitt</teacher>
  <participants type="Webdesign, XML & XSL">
    <name>Cornelius Huber</name>
    <name>Dirk Jesko</name>
  </participants>
  <participants type="Kamera & Schnitt">
    <name>Nico Zenker</name>
  </participants>
  <term>WS 2001/2002</term>
  <location>
    <room>Hörsaal 2</room>
    <period>wöchentlich</period>
    <weekday>Freitag</weekday>
    <time>7:30</time>
  </location>
  <audience>CV,Wirtschaftslogistik,Informations- und
 Mikrosystemtechnik, WO:IF,WIF,SPTE</audience>
  <url>http://wwwiti.cs.uni-magdeburg.de/iti_db/le...</url>
  ...
</Course>
```

2.1.4.11 edition

Elternelement: Book

Beschreibung: Jedem Buch kann über dieses Element optional die Nummer der Ausgabe zugeordnet werden.

Beispiel:

```
<edition>2</edition>
```

2.1.4.12 fileName

Elternelement: MediaObject

Beschreibung: In diesem Element wird der Dateiname der Folie angegeben. Dies erfolgt ohne Endung, da verschiedene Dateitypen verwendet werden und das XSL-Script jeweils die notwendige ergänzt. Die Angabe des Dateinamen kann ein Verzeichnis beinhalten, welches relativ zum Verzeichnis mit den Dateien des entsprechenden Typs betrachtet wird. Die Angabe `<fileName>01/3</fileName>` entspricht somit beispielsweise die PDF-Datei `folien/pdf/01/3.pdf` und die PNG-Datei `folien/pdf/01/3.png`.

Beispiel:

```
<fileName>DB1_01/1</fileName>
```

2.1.4.13 Framerate

Elternelement: VFormat

Beschreibung: Gibt die Framerate des Videostreams an. Der Inhalt des Elements wird ohne Änderung oder Ergänzung auf der WWW-Seite, angezeigt. Der Text ist daher frei formatierbar. Soll eine Einheit angezeigt werden, z.B. `Frames/sec`, so muss diese im `Framerate`-Element mit angegeben werden.

Beispiel:

```
<Framerate>25 fps</Framerate>
```

2.1.4.14 geometry

Elternelement: Format

Beschreibung: Gibt die Auflösung des Videostreams an. Der Inhalt des Elements wird ohne Änderung oder Ergänzung auf der WWW-Seite, angezeigt. Der Text ist daher frei formatierbar. Soll eine Einheit angezeigt werden, z.B. `Pixel`, so muss diese im `geometry`-Element mit angegeben werden.

Beispiel:

```
<geometry>320x240</geometry>
```

2.1.4.15 GO

Elternelement: Lecture

Kindelemente: title, Segment*

Beschreibung: Jedes Segment einer Veranstaltung muss einem Thema zugeordnet werden. Dies erfolgt über das `GO`-Element. Der hier angegebene Titel erscheint auf der WWW-Seite im Menü. Je `GO`-Gruppe wird ein Eintrag im Menü erzeugt. Da jedem untergeordneten Segment ein `MediaObject` zugeordnet ist, bietet es sich an, das `GO` entsprechend dem umschließenden `GroupObject` zu gestalten.

Beispiel:

```
<GO>
  <title>Inhalt</title>
  <Segment seqNum="008" moId="LO1.1.0.1">
    <Video codecId="V1">
      <size>2.5 MByte</size>
      <length>1:28</length>
    </Video>
    <Audio codecId="A1">
      <size>256.7 KByte</size>
      <length>1:28</length>
    </Audio>
  </Segment>
</GO>
```

2.1.4.16 GroupObject

Elternelement: MediaObject

Kindelemente: title, MediaObject*

Attribute: `id ID #REQUIRED`

Beschreibung: Dieses Element bildet die dritte Stufe der Strukturierungshierarchie für Materialien. Es fasst eine Reihe von `MediaObject`-Elementen zusammen. Jedes `LearningModule` enthält weiterhin einen Titel. Dieser wird auf der WWW-Seite sowohl in der Darstellung nach Themen, als auch nach Terminen im Menü verwendet. Jedes `GroupObject` besitzt schließlich eine eindeutige id. Wir bilden die id aus der id des LearningModule gefolgt von einem Punkt gefolgt von einer Nummer, z.B. `LO1.1.0`, `LO2.12.1` usw.

Beispiel:

```
<GroupObject id="LO1.1.0">
  <title>Inhalt</title>
  ...
</GroupObject>
```

2.1.4.17 Info

Elternelement: Organizational

Kindelement: Segment

Attribute: `date CDATA #REQUIRED`

Beschreibung: Dieses Element fässt Segmente mit organisatorischem Inhalt eines Termins zusammen. Das Datum muss in der Form yyyy-mm-dd im date-Attribut angegeben, z.B. `date="2001-10-19"` für den 19. Oktober 2001.

Beispiel:

```
<Info date="2001-10-19">
  <Segment seqNum="001" moId="Org.DB1.1">
    <Video codecId="V1">
      <size>22.0 MByte</size>
      <length>12:58</length>
    </Video>
    <Audio codecId="A1">
      <size>2.2 MByte</size>
      <length>12:55</length>
    </Audio>
  </Segment>
</Info>
```

2.1.4.18 keyword

Elternelement: MediaObject

Beschreibung: Jeder Folie kann über dieses Element eine Reihe von Keywords zugeordnet werden, die in einer späteren Version für eine Suchfunktion dienen sollen. Derzeit werden die Keywords jedoch noch nicht ausgewertet.

Beispiel:

```
<keyword>Software-Schichten</keyword>
```

2.1.4.19 LearningModule

Elternelement: LearningUnit

Kindelemente: title, GroupObject*

Attribute: `id ID #REQUIRED`

Beschreibung: Dieses Element bildet die zweite Stufe der Strukturierungshierarchie für Materialien. Es fasst eine Reihe von `GroupObject`-Elementen zusammen und ergänzt diese um einen Titel, welcher derzeit jedoch noch nicht verwendet wird. Schließlich besitzt jedes `LearningModule` eine eindeutige id. Wir bilden diese aus der id der `LearningUnit` gefolgt von einem Punkt gefolgt von einer Nummer, z.B. `LO1.1`, `LO2.12` usw.

Beispiel:

```
<LearningModule id="LO1.1">
  <title>Motivation und Historie</title>
  ...
</LearningModule>
```

2.1.4.20 LearningUnit

Elternelement: Metadata

Kindelemente: title, LearningModule*

Attribute: `id ID #REQUIRED`

Beschreibung: Dieses Element bildet die oberste Stufe der Strukturierungshierarchie für Materialien. Es fasst eine Reihe von `LearningModule`-Elementen zusammen und ergänzt diese um einen Titel, der auf der WWW-Seite im Menü verwendet wird (Strukturierung nach Themen). Schließlich besitzt jede `LearningUnit` eine eindeutige id. Wir bilden diese aus den Buchstaben `LO` gefolgt von einer Nummer, z.B. `LO1`, `LO2` usw.

Beispiel:

```
<LearningUnit id="LO1">
  <title>Grundlegende Konzepte</title>
  ...
</LearningUnit>
```

2.1.4.21 Lecture

Elternelement: Info

Kindelement: GO*

Attribute: `date CDATA #REQUIRED`

Beschreibung: In diesem Element werden Segmente von Veranstaltungen zusammengefasst, die einen fachlichen und keinen organisatorischen Inhalt haben. Die Gruppierung erfolgt an Hand des Datums, welches im `date`-Attribut in der Form yyyy-mm-dd angegeben werden muss, z.B. `date="2001-10-19"` für den 19. Oktober 2001.

Beispiel:

```
<Lecture date="2001-10-19">
  <GO>
    <title>Inhalt</title>
    <Segment seqNum="008" moId="LO1.1.0.1">
      <Video codecId="V1">
        <size>2.5 MByte</size>
        <length>1:28</length>
      </Video>
      <Audio codecId="A1">
        <size>256.7 KByte</size>
        <length>1:28</length>
      </Audio>
```

```
    </Segment>
  </GO>
</Lecture>
```

2.1.4.22 length

Elternelemente: Video, Audio

Beschreibung: Gibt die Länge (Spieldauer) der Video- bzw. Audiodatei an. Die Angabe wird ohne Änderungen oder Ergänzungen übernommen.

Beispiel:

```
<length>12:58</length>
```

2.1.4.23 location

Elternelement: Course

Kindelemente: room, period, weekday, time

Beschreibung: In diesem Element wird angegeben wann und wo die Veranstaltung stattfindet. Die Angabe wird derzeit jedoch noch nicht ausgewertet.

Beispiel:

```
<location>
  <room>Hörsaal 2</room>
  <period>wöchentlich</period>
  <weekday>Freitag</weekday>
  <time>7:30</time>
</location>
```

2.1.4.24 MediaObject

Elternelement: GroupObject

Kindelemente: fileName, title, orientation, keyword*, reference*

Attribute: `id ID #REQUIRED`

Beschreibung: Dieses Element bildet die vierte und letzte Stufe der Strukturierungshierarchie für Materialien. Jedes `MediaObject` beschreibt genau eine Folie. Von den angegebenen Informationen werden die Keywords und der Titel derzeit nicht verwendet. Schließlich besitzt jedes `MediaObject` eine eindeutige id. Wir bilden diese aus der id des `GroupObject` gefolgt von einem Punkt gefolgt von einer Nummer, z.B. `LO1.1.1.1`.

Beispiel:

```
<MediaObject id="LO1.1.1.1">
  <fileName>DB1_01/2</fileName>
  <title>Software Schichten</title>
  <orientation>landscape</orientation>
  <reference bookId="B1">
    <section>1.1</section>
    <pages>1ff</pages>
  </reference>
  <keyword>Software-Schichten</keyword>
  <keyword>Betriebssystem</keyword>
  <keyword>System-Software</keyword>
  <keyword>Basis-Software</keyword>
  <keyword>Anwendungs-Software</keyword>
  <keyword>Individual-Software</keyword>
</MediaObject>
```

2.1.4.25 Metadata

Kindelemente: Book*, VFormat*, AFormat*, Misc?, LearningUnit*, Course*

Beschreibung: Das `Metadata`-Element dient ausschließlich der Zusammenfassung der Elemente. Die Kindelemente gliedern sich in drei Gruppen. Zum einen werden allgemeine Informationen, wie Video- und Audioformate sowie Bücher beschrieben. Diese können dann von anderen Elementen referenziert werden. Die zweite Gruppe behinhaltet die Beschreibung der Materialien. Dabei wird zwischen allgemeinen Materialien (`Misc`-Element) und thematisch eingeordneten Materialien (`LearningUnit`-Element) unterschieden. Schließlich beinhaltet die dritte Gruppe die Beschreibung konkreter Veranstaltungen, die die Materialien verwenden.

2.1.4.26 Misc

Elternelement: Metadata

Kindelement: MediaObject*

Beschreibung: Dieses Element dient der Definition von Medienobjekten, die keinen fachlichen Inhalt enthalten oder voraussichtlich nicht wiederverwendet werden. Dient u.a. zur Aufnahme der Medienobjekte für die `Organizational`-Elemente. Wird aber auch verwendet, wenn beispielsweise eine Zusammenfassung von Inhalt ohne Folie erfolgt.

Beispiel:

```
<Misc>
  <MediaObject id="Org.DB1.1">
    <fileName>DB1_Org/1</fileName>
    <title>Termine</title>
    <orientation>portrait</orientation>
  </MediaObject>
  <MediaObject id="Org.DB1.2">
    <fileName>DB1_Org/2</fileName>
    <title>Datenbanken I</title>
```

```
    <orientation>landscape</orientation>
  </MediaObject>
</Misc>
```

2.1.4.27 name

Elternelement: participants

Beschreibung: In diesem Element wird der Name eines Beteiligten der jeweiligen Gruppe von Mitarbeitern an der Veranstaltung angegeben.

Beispiel:

```
<name>Cornelius Huber</name>
```

2.1.4.28 Organizational

Elternelement: Course

Kindelement: Info+

Beschreibung: In diesem Element werden `MediaObjects` mit organisatorischen Informationen, z.B. allgemeine Informationen zur Vorlesung oder zur Prüfung zusammengefasst. Diese werden mittels des `Info`-Elements nochmals nach Datum untergliedert. Aus diesem Element und den Kindelementen wird der Eintrag „Organisatorisches“ im Menü (nur bei zeitlicher Gliederung) erzeugt.

Beispiel:

```
<Organizational>
  <Info date="2001-10-19">
    <Segment seqNum="001" moId="Org.DB1.1">
      <Video codecId="V1">
        <size>22.0 MByte</size>
        <length>12:58</length>
      </Video>
      <Audio codecId="A1">
        <size>2.2 MByte</size>
        <length>12:55</length>
      </Audio>
    </Segment>
  </Info>
</Organizational>
```

2.1.4.29 orientation

Elternelement: MediaObject

Beschreibung: benötigt, um auf der WWW-Seite den Rahmen um das Bild der Folie in den korrekten Dimensionen darzustellen. Vom XSL-Skript werden nur die Angaben „portrait“ für Hochformat und „landscape“ für Querformat korrekt ausgewertet. Andere Angaben, z.B. durch Tippfehler, führen u.U. zu falschem Layout.

Beispiel:

```
<orientation>landscape</orientation>
```

2.1.4.30 pages

Elternelement: reference

Beschreibung: Dieses Element enthält die Seiten einer Referenz im Buch. Diese Angabe wird durch das XSL-Skript um die Angabe `Seiten` erweitert. `<pages>13ff.</pages>` im XML führt somit zur Ausgabe `Seiten 13ff.` auf der WWW-Seite.

Beispiel:

```
<pages>1ff</pages>
```

2.1.4.31 participants

Elternelement: Course

Kindelement: name*

Attribute: `type CDATA #REQUIRED`

Beschreibung: Über dieses Element kann eine Reihe von Beteiligten angegeben werden, etwa Übungsleiter, Kamera, Web-Design usw. Das Attribut `type` legt dabei die Art der Beteiligung fest, z.B. Übungsleiter.

Beispiel:

```
<participants type="Webdesign, XML & XSL">
  <name>Cornelius Huber</name>
  <name>Dirk Jesko</name>
</participants>
```

2.1.4.32 period

Elternelement: location

Beschreibung: In diesem Element wird angegeben wie häufig die Veranstaltung stattfindet, beispielsweise wöchentlich. Die Angabe wird derzeit jedoch noch nicht ausgewertet.

Beispiel:

```
<period>wöchentlich</period>
```

2.1.4.33 publisher

Elternelement: Book

Beschreibung: Jedem Buch muss über dieses Element ein Verlag zugeordnet werden.

Beispiel:

```
<publisher>mitp</publisher>
```

2.1.4.34 reference

Elternelement: MediaObject

Kindelemente: section, pages

Attribute: `bookId IDREF #REQUIRED`

Beschreibung: Jeder Folie kann über dieses Element eine Reihe von Referenzen zugeordnet werden, die auf der WWW-Seite angezeigt werden. Die Definition einer Referenz beinhaltet zwei Teile. Zum Einen die Referenz zu einem `Book`-Element über das bookId-Attribut und zum Anderen den Abschnitt und die Seiten im Buch über die Kindelemente.

Beispiel:

```
<reference bookId="B1">
  <section>1.1</section>
  <pages>1ff</pages>
</reference>
```

2.1.4.35 room

Elternelement: location

Beschreibung: In diesem Element wird der Raum angegeben, in dem die Veranstaltung stattfindet. Die Angabe wird derzeit jedoch noch nicht ausgewertet.

Beispiel:

```
<room>Hörsaal 2</room>
```

2.1.4.36 Samplerate

Elternelement: AFormat

Beschreibung: Gibt die Samplerate des Audiostreams an. Der Inhalt des Elements wird ohne Änderung oder Ergänzung auf der WWW-Seite, angezeigt. Der Text ist daher frei formatierbar. Soll eine Einheit angezeigt werden, z.B. `kHz`, so muss diese im `Samplerate`-Element mit angegeben werden.

Beispiel:

```
<Samplerate>22,5 kHz</Samplerate>
```

2.1.4.37 section

Elternelement: reference

Beschreibung: Dieses Element enthält den Abschnitt im Buch. Anzugeben ist nur die Nummer. Diese Angabe wird durch das XSL-Skript um die Angabe `Kapitel` erweitert. Die Angabe `<section>1.4</section>` im XML führt somit zur Ausgabe `Kapitel 1.4` auf der WWW-Seite.

Beispiel:

```
<section>1.1</section>
```

2.1.4.38 Segment

Elternelemente: Info, GO

Kindelemente: MO, Video?, Audio?

Attribute: `seqNum CDATA #REQUIRED`, `moId CDATA #REQUIRED`

Beschreibung: Ein Segment beschreibt einen Abschnitt der Veranstaltung, der durch eine Mediendatei (je Format) repräsentiert wird und inhaltlich genau einem MediaObject entspricht. Das Segment beinhaltet daher Informationen zum zugehörigen MediaObject (zugeordnet über das Attribut `moId`), sowie zur zugehörigen Video- und Audiodatei. Weiterhin wird durch das Attribut `seqId` eine Sortierung der Segmente realisiert. Es gilt dabei zu beachten, dass die `seqNum` dem Dateinamen der Mediendateien entsprechen muss, so impliziert `seqNum="003"` beispielsweise die Dateien `003.avi` und `003.mp3`, wobei die Endungen erst durch das XSL-Skript erzeugt werden.

Beispiel:

```
<Segment seqNum="008" moId="LO1.1.0.1">
  <Video codecId="V1">
    <size>2.5 MByte</size>
    <length>1:28</length>
  </Video>
  <Audio codecId="A1">
    <size>256.7 KByte</size>
    <length>1:28</length>
  </Audio>
</Segment>
```

2.1.4.39 size

Elternelemente: Video, Audio

Beschreibung: Gibt die Größe der Video- bzw. Audiodatei an. Die Angabe wird ohne Änderungen oder Ergänzungen übernommen. Soll eine Einheit angezeigt werden, z.B. `kByte`, muss diese im XML mit angegeben werden.

Beispiel:

```
<size>22.0 MByte</size>
```

2.1.4.40 teacher

Elternelement: Course

Beschreibung: Dieses Element beinhaltet den Namen des Vorlesenden. Sollen mehrere Vorlesende angegeben werden, so muss dies beispielsweise durch Komma getrennt, in einem `teacher`-Element erfolgen, da derzeit nur ein solches Element je `Course` vorgesehen ist.

Beispiel:

```
<teacher>Dr. Ingo Schmitt</teacher>
```

2.1.4.41 term

Elternelement: Course

Beschreibung: In diesem Element wird das Semester angegeben, in dem die Veranstaltung stattfindet. Die Angabe wird derzeit jedoch noch nicht ausgewertet.

Beispiel:

```
<term>WS 2001/2002</term>
```

2.1.4.42 time

Elternelement: location

Beschreibung: In diesem Element wird die Zeit angegeben, zu der die Veranstaltung stattfindet, z.B. um 7:30. Die Angabe wird derzeit jedoch noch nicht ausgewertet. Momentan ist nur eine einzelne Angabe möglich. Findet die Veranstaltung zu unterschiedlichen Zeiten statt, müssen alle in einem `time`-Element angegeben werden.

Beispiel:

```
<time>7:30</time>
```

2.1.4.43 title

Elternelemente: Book, LearningUnit, LearningModule, GroupObject, MediaObject, Course, GO

Beschreibung: Jedem der angegebenen Elternelemente muss bzw. kann mit diesem Element ein Titel zugewiesen werden.

Beispiel:

```
<title>Datenbanken: Konzepte und Sprachen</title>
```

2.1.4.44 url

Elternelement: Course

Beschreibung: In diesem Element wird die URL einer WWW-Seite angegeben, die weiterführende Informationen zu der Veranstaltung beinhaltet. Auf der WWW-Seite wird ein Link auf diese Seite bereitgestellt.

Beispiel:

```
<url>http://wwwiti.cs.uni-magdeburg.de/iti_db/le...</url>
```

2.1.4.45 VFormat

Elternelement: Metadata

Kindelemente: Codec, CodecURL*, Bitrate, Framerate, geometry

Attribute: `id ID #REQUIRED`

Beschreibung: In diesem Element werden Informationen zu einem bei der Aufzeichnung verwendeten Videoformat angegeben. Jedes Format erhält dabei über das id-Attribut eine eindeutige Bezeichnung. Die `Segment`-Elemente referenzieren das Format über diese id. Wir verwenden als id ein `V` gefolgt von einer Nummer. Es gilt zu beachten, dass die id über alle Elemente eindeutig sein muss.

Beispiel:

```
<VFormat id="V1">
  <Codec>DivX 4.0</Codec>
  <CodecURL os="Windows 2000">http://www.divx.com/</CodecURL>
  <CodecURL os="Linux">http://www.divx.com/</CodecURL>
  <Bitrate>200 KBit/s</Bitrate>
  <Framerate>25 fps</Framerate>
  <geometry>320x240</geometry>
</VFormat>
```

2.1.4.46 Video

Elternelement: Segment

Kindelemente: size, length

Attribute: `codecId IDREF #REQUIRED`

Beschreibung: Dieses Element stellt Informationen über die Video-Datei des Segments bereit. Diese Informationen werden auf der WWW-Seite dargestellt. Angegeben werden müssen eine Referenz auf das Format des Videos (siehe id-Attribut des `VFormat`-Element) sowie Laufzeit und Größe.

Beispiel:

```
<Video codecId="V1">
  <size>22.0 MByte</size>
  <length>12:58</length>
</Video>
```

2.1.4.47 weekday

Elternelement: location

Beschreibung: In diesem Element wird der Tag angegeben, an dem die Veranstaltung stattfindet, z.B. Freitags. Die Angabe wird derzeit jedoch noch nicht ausgewertet. Momentan ist nur eine einzelne Angabe möglich. Findet die Veranstaltung häufiger als wöchentlich oder an unterschiedlichen Tagen statt, müssen alle Termine in einem `weekday`-Element angegeben werden.

Beispiel:

```
<weekday>Freitag</weekday>
```

2.1.4.48 year

Elternelement: Book

Beschreibung: Jedem Buch muss über dieses Element ein Erscheinungsjahr zugeordnet werden.

Beispiel:

```
<year>2001</year>
```

2.2 Aufbau der Webseite

Dieser Abschnitt beschreibt den Aufbau der WWW-Seite und und die Zusammenhänge mit den in der XML-Datei beschriebenen Metadaten.

2.2.1 Verzeichnisstruktur

Für die Verwaltung der verschiedenen Dateien (Video, Audio, Folien und HTML) wird die in Abbildung 2.3 dargestellte Verzeichnisstruktur verwendet. Bis auf die Unterverzeichnisse `audio` und `video` werden alle durch die in Abschnitt 2.3 beschriebenen Skripte automatisch in der korrekten Struktur erzeugt. Die verwendete Verzeichnisstruktur ist folgendermaßen gegliedert:

1. Die für die Darstellung der WWW-Seite erforderlichen Bilder, Skripte, etc. sind in den Verzeichnissen `images` und `libs` abgelegt. Dies umfasst insbesondere die JavaScript-Dateien für das Menü, die Overlays mit den technischen Informationen, usw. Die Inhalte der Verzeichnisse sind für alle bereitgestellten Veranstaltungen identisch.

2. In den Verzeichnissen `folien` und `medien` werden die Folien-, Video- und Audiodateien der einzelnen Veranstaltungen abgelegt. Die Folien werden weiter nach den verschiedenen Dateitypen (pdf und png) untergliedert. Die

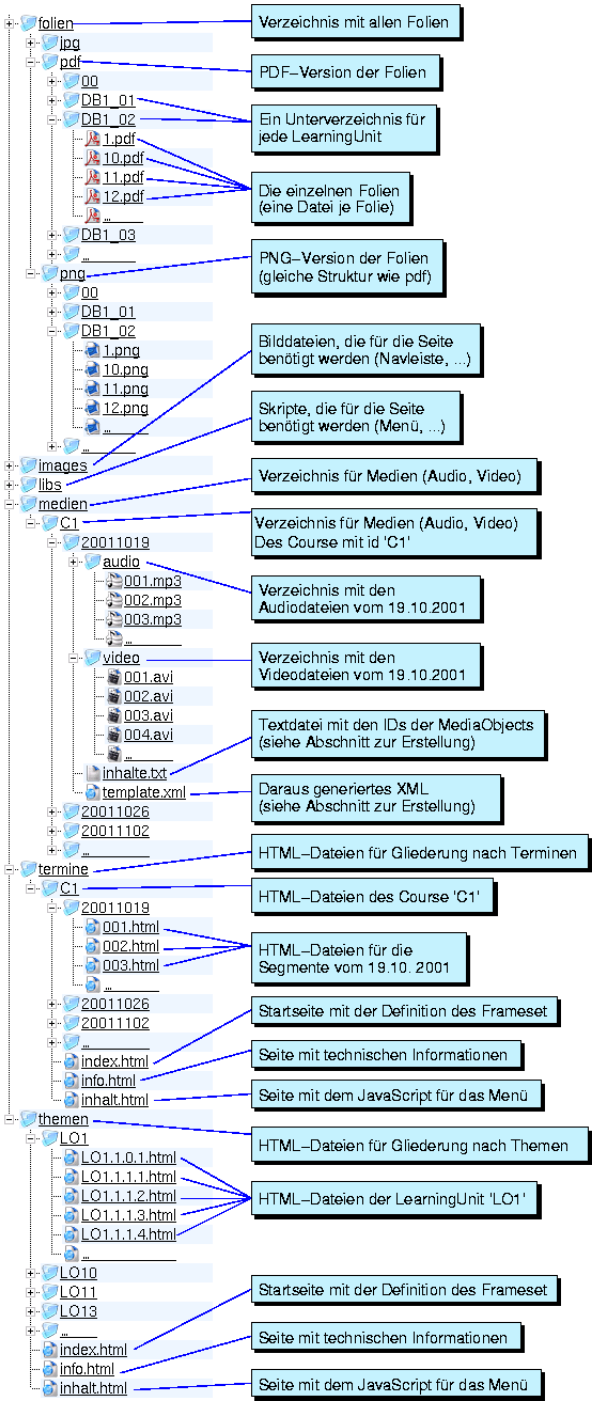


Abbildung 2.3: Verzeichnisstruktur für die WWW-Seite

weitere Strukturierung innerhalb dieser Verzeichnisse ist für alle Formate gleich und entspricht dem im Element `//MediaObject/fileName` angegebenen Dateinamen. Enthält die Angabe auch Verzeichnisse, so werden diese relativ zum Verzeichnis des jeweiligen Formats betrachtet. Die Angabe `<fileName>DB1_02/11</fileName>` im XML beschreibt beispielsweise die Dateien `11.pdf` und `11.png` im Verzeichnis `DB1_02`.

Während die Strukturierung der Folien keinen Bezug zu den einzelnen Veranstaltungen hat, da diese in mehreren Veranstaltungen verwendet werden können, ist es bei den Video- und und Audiodateien erforderlich, eine derartige Zuordnung vorzunehmen. Daher wird unterhalb des Verzeichnisses `medien` zunächst ein Verzeichnis je `Course` erzeugt, wobei als Name automatisch der Wert des Attributs `//Course/@id` aus der XML-Datei verwendet wird. Weiterhin erfolgt eine Untergliederung entsprechend dem Datum einer Veranstaltung, das automatisch aus dem Attribut `//Lecture/@date` aus der XML-Datei erzeugt, wobei das Format `jjjjmmtt` (`j`: Jahr, `m`: Monat, `t`:Tag) verwendet wird. Jedes dieser Verzeichnisse beinhaltet schließlich je ein weiteres für jeden Medientyp (Audio, Video). Die Dateinamen selbst müssen mit einem `//Segment` korrespondieren, das diesen Namen als `seqNum` enthält. Ein Segment einer `Lecture` vom 19.10.2003 mit der Sequenznummer 3 (`seqNum="003"`) korrespondiert somit einerseits zur Videodatei `20011019/video/001.avi` und andererseits zur Audiodatei `20011019/audio/001.mp3`. Werden die XML-Beschreibungen der einzelnen Vorlesungen mit Hilfe des Skripts `initXML` (siehe Abschnitt 2.3.3) erzeugt, muss in jedem Verzeichnis eines Vorlesungstermins eine Textdatei `inhalt.txt` vorhanden sein. Diese enthält die zu den Medien korrespondierenden `id`s der `MediaObject`-Elemente. Das Skript erzeugt daraus die Datei `template.xml` mit den Informationen der Medien. Diese beiden Dateien können nach der Eingabe in die XML-Datei ggf. wieder gelöscht werden.

3. Die Verzeichnisse `termine` und `themen` beinhalten schließlich die generierten HTML-Dateien für die WWW-Seite. Die Strukturierung unterhalb `termine` entspricht der der Medien, da diese ein nach Terminen gegliedertes Menü besitzt. Die Struktur in `themen` orientiert sich an der veranstaltungsunabhängigen Strukturierung der Materialien. Es wird für jede `id` einer `LearningUnit` ein Verzeichnis generiert, das für jedes darin enthaltene `MediaObject` eine HTML-Datei enthält. Diese erhält als Namen den Inhalt von`//MediaObject/@id`. Eine Abbildung der `LearningModule`- und `GroupObject`-Ebene erfolgt nicht.

   Schließlich enthalten `termine` und `themen` jeweils noch drei Dateien, die für die Darstellung des Menüs notwendig sind.

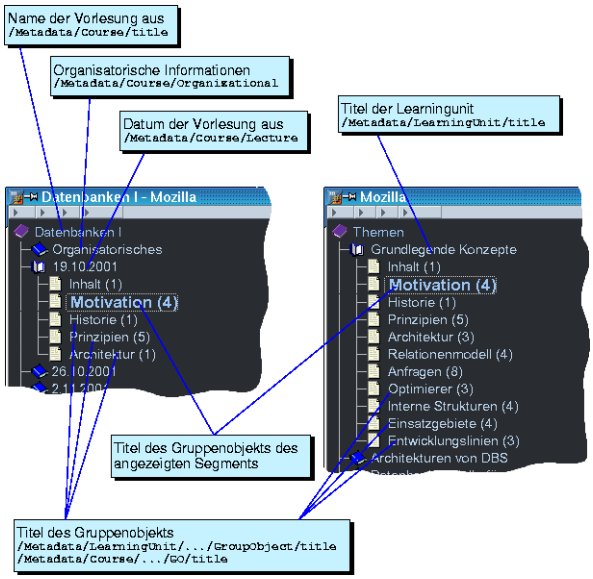


Abbildung 2.4: Aufbau des Menüs (links geordnet nach Terminen, rechts nach Themen)

2.2.2 Aufbau der HTML-Seiten

Dieser Abschnitt gibt einen Überblick über die einzelnen Elemente der HTML-Seiten und die jeweils zugehörigen Informationen aus der XML-Datei.

Das Menü

Wie in Abschnitt 1.3 dargestellt, können die Informationen auf der WWW-Seite auf zwei Arten strukturiert werden (nach Themen und nach Terminen). Dementsprechend unterscheiden sich die Menüs. Bei einer Strukturierung nach Terminen (vgl. Abbildung 2.4 links) werden der Titel der Vorlesung (`//Course/title`) und die einzelnen Vorlesungstermine (`//lecture/@date`) dargestellt. Das Menü der Strukturierung nach Themen (vgl. Abbildung 2.4 rechts) unterscheidet sich darin, dass statt des Datums der Veranstaltung der Titel der Learningunit (`//LearningUnit/title` verwendet wird. Das Menü enthält dabei einen Eintrag je `LearningUnit`-Element in der XML-Datei. Die zweite Ebene des Menüs enthält bei beiden Varianten der Strukturierung die Titel der Gruppenobjekte. Bei der Gliederung nach Terminen wird dabei `//GO/title` und bei Gliederung nach Themen `//GroupObject/title` verwendet.

Das Menü ist in JavaScript realisiert, wobei eine angepasste Version von *Morten's JavaScript Tree Menu* (`http://www.treemenu.com/`) zum Einsatz kommt. Für eine genaue Beschreibung der Funktionen, Konfiguration, etc. sei daher auf die entsprechende WWW-Seite verwiesen.

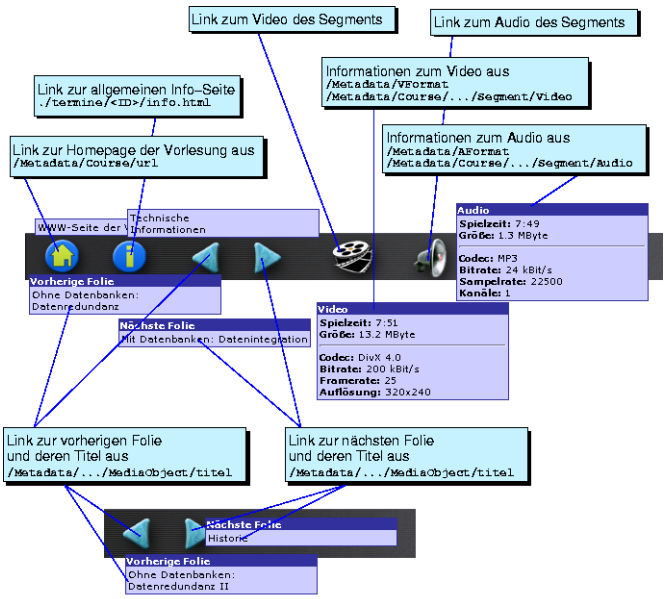


Abbildung 2.5: Die Navigationsleiste (oben strukturiert nach Terminen, unten nach Themen)

Die Navigationsleiste

Bei Strukturierung der Materialien nach Terminen, enthält die Navigationsleiste sechs Icons (vgl. Abbildung 2.5 oben). Die ersten beiden verweisen auf Seiten mit weiteren Informationen, die folgenden zwei dienen der Navigation zwischen den einzelnen Folien und die übrigen beiden verweisen auf die Video- und die Audiodatei zu dieser Folie. Im Falle der Strukturierung nach Themen enthält die Navigationsleiste lediglich die Icons für die Navigation zwischen den Folien (vgl. Abbildung 2.5 unten), da für jede Folie mehrere Video- und Audiodateien existieren können. Dies ist immer dann der Fall, wenn die entsprechende Folie in mehreren Kursen oder in mehreren Veranstaltungen eines Kurses verwendet wurde. Daher erfolgt die Verlinkung in diesem Fall in einem separaten Abschnitt unterhalb der Folie (vgl. Abbildung 2.6).

Die ersten beiden Icons verweisen auf die im XML angegebenen WWW-Seite mit Informationen zur Vorlesung. Die dort angegebene Seite ist beliebig und muss separat erstellt werden. Weiterhin wird auf eine Seite mit technischen Informationen verwiesen, die u.a. auch die Namen der an der Vorlesung beteiligten Personen beinhaltet. Diese Datei (`info.html`) wird vom XSLT-Skript automatisch generiert.

Die folgenden beiden Icons dienen der Navigation zwischen den einzelnen Segmenten eines Vorlesungstermins bzw. zwischen den Medienobjekten einer `LearningUnit`. Es kann jeweils zum vorherigen und nächsten Segment bzw.

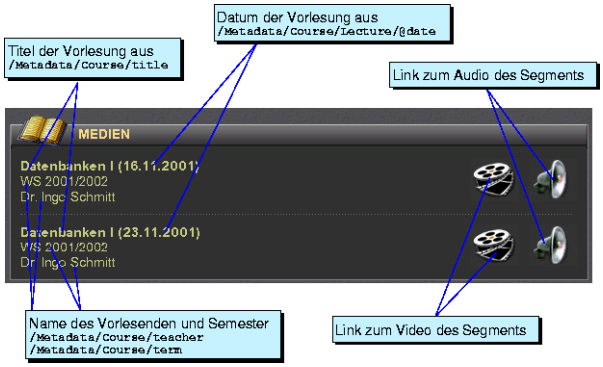


Abbildung 2.6: Verlinkung der Medien bei thematischer Darstellung

Medienobjekt gesprungen werden. Der Wechsel zum ersten Segment des folgenden Termins bzw. des letzten Segments des vorherigen Termins ist derzeit nicht möglich. Die Verweise zwischen den Seiten werden vom Skript automatisch, basierend auf der laufenden Nummer der Segmente (`//Segment/@seqNum`) bzw. der id der Medienobjekte erstellt. Es gilt dabei zu beachten, dass das Skript derzeit davon ausgeht, dass die Segmente in der XML-Datei entsprechend ihrer laufenden Nummer aufsteigend sortiert sind.

Befindet sich die Maus über einem der Icons wird zusätzlich der Titel des nächsten bzw. vorherigen Medienobjekts (und somit der zugehörigen Folie) angezeigt. Ändert sich beim Wechsel des Medienobjekts auch das Gruppenobjekt, wird im Menü (vgl. Abbildung 2.4) der entsprechende Eintrag hervorgehoben.

Der dritte Teil der Navigationsleiste enthält zwei Icons, die auf die Medien (Video- und Audio-Datei) verweisen. Befindet sich die Maus über einem der Icons, werden Informationen zur Datei angezeigt, u.a. Format und Größe.

Die Anzeige der Zusatzinformationen ist mit JavaScript unter Verwendung von *overLIB* (`http://www.bosrup.com/web/overlib/`) realisiert. Für eine genauere Beschreibung sei daher auf die entsprechende WWW-Seite verwiesen.

Die Folie

Unterhalb der Navigationsleiste wird die zum Segment gehörende Folie als Bild angezeigt, welches zu einer PDF-Datei verlinkt ist. Derzeit wird davon ausgegangen, das das Bild im PNG-Format vorliegt. Mit geringen Änderungen am XSLT-Skript sind aber auch andere Formate denkbar.

Das Bild der Folie wird durch einen Rahmen eingeschlossen, dessen Kopf zusätzliche Informationen bereitstellt (vgl. Abbildung 2.7). Dies sind der Titel der Lehrveranstaltung, der Termin der Vorlesung, zu der diese Folie gehört, sowie die Gesamtzahl der Folien dieses Termins und die aktuelle Folie. Bei thematischer

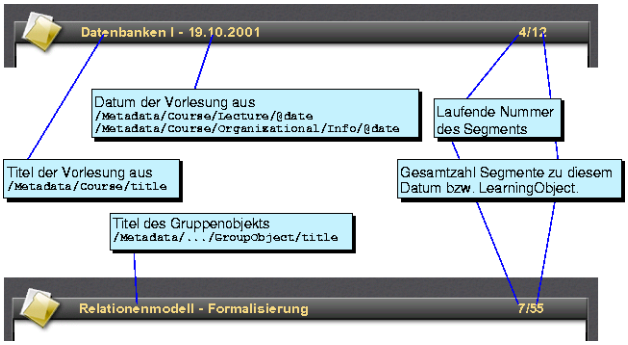


Abbildung 2.7: Aufbau der Navigationsleiste

Gliederung wird statt des Vorlesungstitels und des Datums der Titel des Gruppenobjekts angezeigt, so wie dieser auch im Menü erscheint. Die Zählung der Folien im Kopf ist unabhängig von der in der XML-Datei für die Segmente vergebenen laufenden Nummer. Sie beginnt jeweils bei der ersten Folie eines Termins mit 1. Hat ein Termin beispielsweise zunächst fünf organisatorische Segmente (`001.avi` bis `005.avi`) und anschließend zehn fachliche Segmente (`006.avi` bis `015.avi`), so haben diese die Sequenznummern `001` bis `015`. Im Kopf der Folie erscheinen jedoch die Angaben `1/5` bis `5/5` für die organisatorischen und `1/10` bis `10/10` für die fachlichen Folien.

Die Referenzen

Unterhalb der Folie werden die zugeordneten Referenzen zu Abschnitten aus Büchern dargestellt (vgl. Abbildung 2.8). Die Informationen zum Abschnitt und zur Seite einer Referenz werden aus dem `MediaObject` gewonnen, das dem `Segment` zugeordnet ist. Die Informationen zum Buch aus dem `Book`, das wiederum dem `MediaObject` zugeordnet ist. Die Darstellung ist bei beiden Strukturierungsarten (Terminen, Themen) identisch.

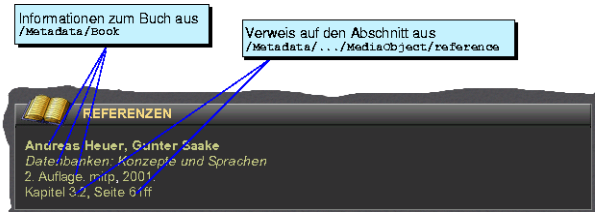


Abbildung 2.8: Darstellung der Referenzen

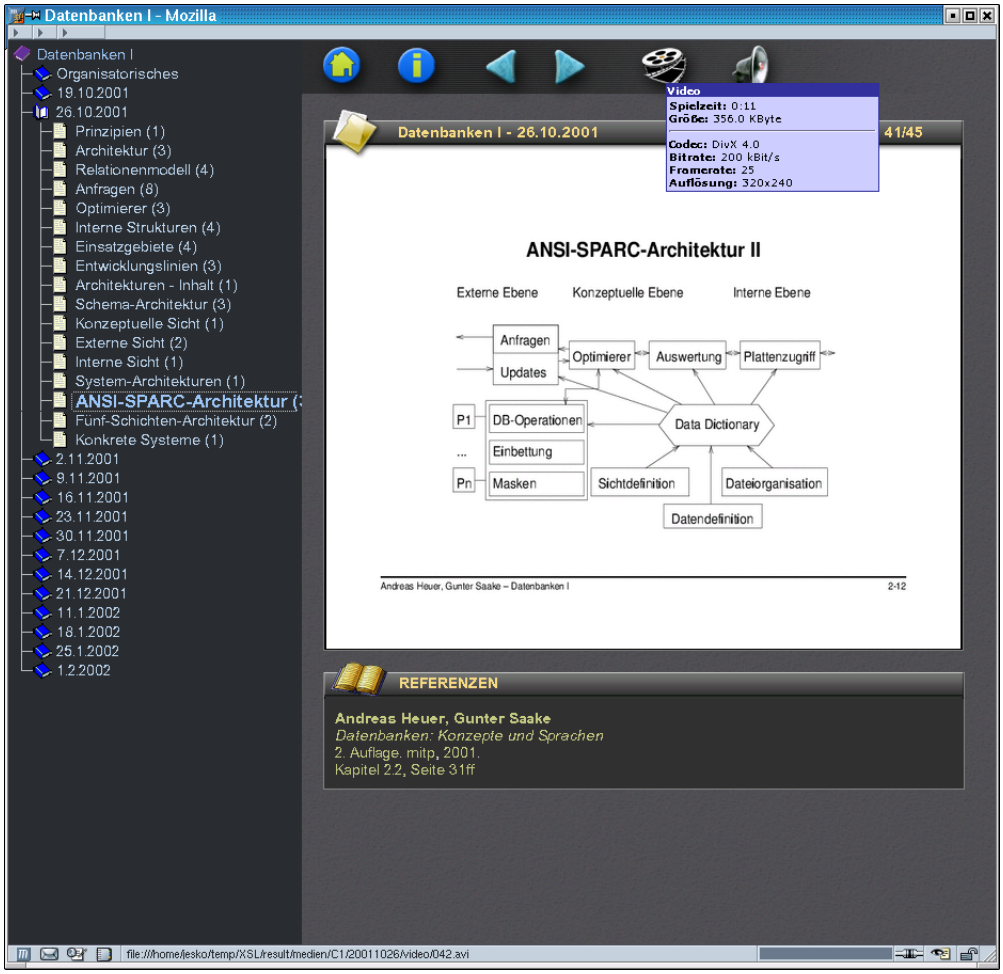


Abbildung 2.9: Gesamtansicht der Webseite (Menü nach Terminen strukturiert)

2.2.3 Gesamtansicht

In den Abbildungen 2.9 und 2.10 ist die Gesamtansicht der WWW-Seiten nochmals in beiden Strukturierungsvarianten dargestellt.

2.3 Erstellung der WWW-Seite

Nachdem die Struktur der Metadaten und der Aufbau der WWW-Seite beschrieben wurde, soll im Folgenden auf die notwendigen Arbeitsschritte zu deren Erstellung eingegangen werden. Jeder der Arbeitsschritte wird dabei zunächst kurz dargestellt. Daran schließt sich eine genauere Beschreibung an, die weiterführende Hinweise und genauere Erläuterung der Funktionsweisen der Skripte bein-

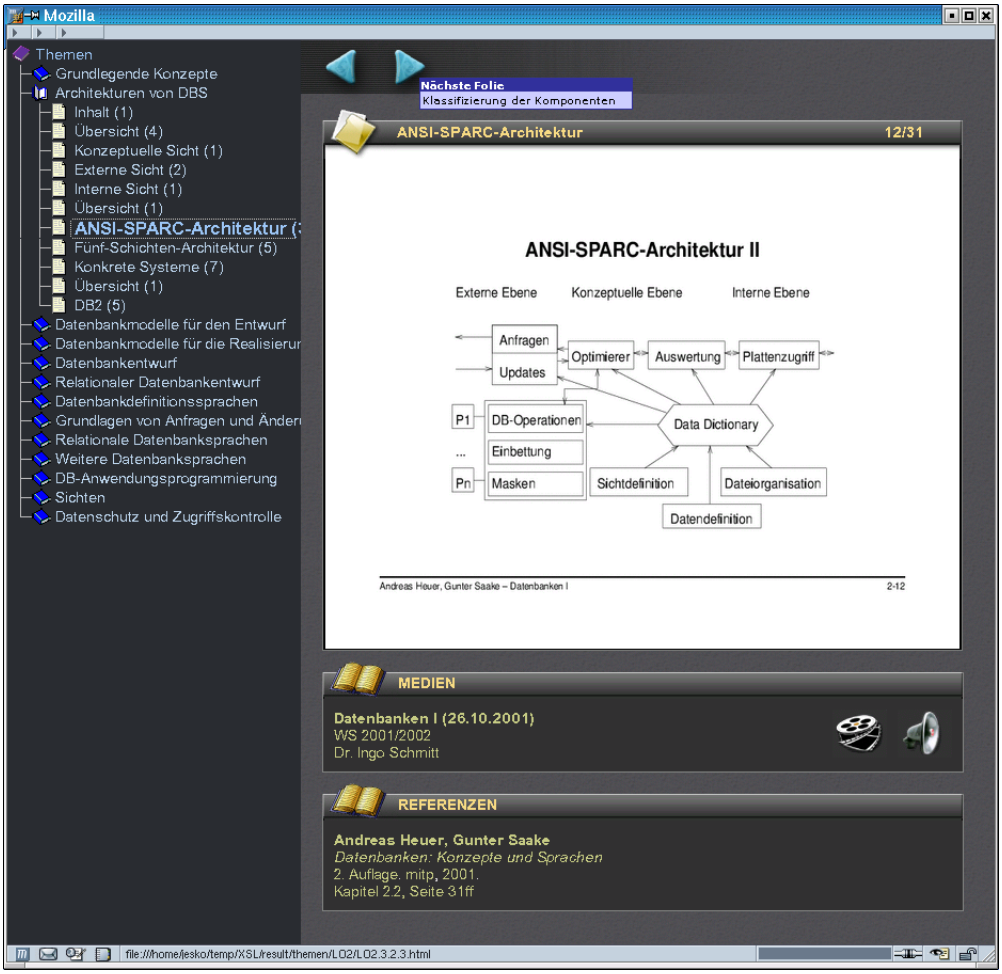


Abbildung 2.10: Gesamtansicht der Webseite (Menü nach Themen strukturiert)

haltet. Es sei erwähnt, dass die hier vorgestellten Skripte nur auf Systemen mit Bash-Shell verwendbar sind, d.h. UNIX, Linux, etc. Für andere Plattformen sind entsprechende Anpassungen und u.U. andere Werkzeuge erforderlich.

2.3.1 Aufbereiten der Folien

Der erste Schritt besteht in der Aufbereitung der Folien. Für jedes Segment ist zunächst eine PNG-Datei als Repräsentation der Folie erforderlich, die in die Seite eingebettet wird. Weiterhin wird die Folie als PDF-Datei benötigt. Diese wird nach einem Klick auf das Bild (PNG) angezeigt. Es ist durchaus möglich, einem Segment eine mehrseitige PDF-Datei zuzuordnen. Da wir aber von einer Segmentierung des Video entsprechend der verwendeten Folien ausgehen, erstellt das im

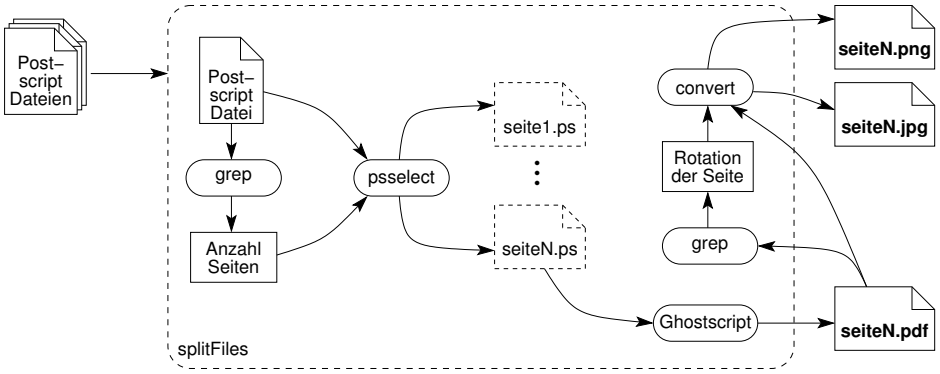


Abbildung 2.11: Erzeugung einzelner Dateien für die Folien

folgenden beschriebene Skript jeweils einseitige PDF-Dateien. Die Arbeitsweise des Skripts ist in Abbildung 2.11 angedeutet. Die Erzeugung der Dateien erfolgt in folgenden Schritten:

1. Ein Verzeichnis erstellen, in dem die Dateien erzeugt werden sollen.
2. Alle (mehrseitigen) PostScript-Dateien in dieses Verzeichnis kopieren.
3. Das Skript `splitFiles` ebenfalls in dieses Verzeichnis kopieren.
4. Das Skript starten. Es analysiert zunächst die Quelldateien und teilt sie anschließend. Sollte dabei ein Problem auftreten, wird eine Fehlermeldung ausgegeben.
5. Das Skript erzeugt nun eine Reihe von Unterverzeichnissen, in denen sich die konvertierten Dateien in den Formaten PDF, PNG und JPG befinden. Diese können, falls erforderlich, in das eigentliche Zielverzeichnis kopiert werden, auf das der Webserver Zugriff hat.

Grundlage für die Erzeugung der Dateien der Folien bilden mehrseitige Postscript-Dateien. Mittels des Shell-Skripts `splitFiles` werden daraus automatisch die notwendigen separaten Dateien erzeugt und in der im Abschnitt 2.2 beschriebenen Struktur abgelegt. Die Postscript-Dateien müssen an irgend einer Stelle die Anzahl der Seiten enthalten, damit sie geteilt werden können. Dies erfolgt in einer Zeile:

```
%%Pages: ppp
```

wobei `ppp` der Anzahl der Seite in der Datei entspricht. Dateien, die unter Windows erzeugt wurden, z.B. indem die Druckerausgabe umgeleitet wird, enthalten diese Zeile häufig nicht. Eine generelle Lösung für dieses Problem gibt es

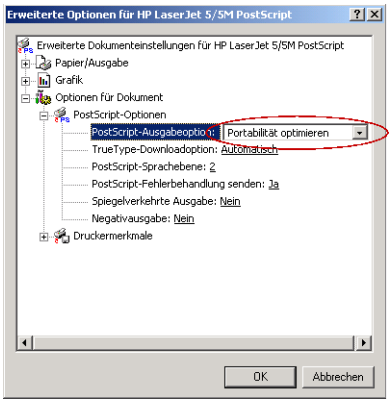


Abbildung 2.12: Erzeugung von PostScript-Dateien unter Windows

unseren Erfahrungen nach nicht. In der Regel hilft es, jedoch in den erweiterten Druckereinstellungen die Option *Postscript-Ausgabeoption* auf den Wert *Portabilität optimieren* zu setzen (vgl. Abbildung 2.12).

Nachdem alle zu bearbeitenden PostScript-Dateien und das Skript in ein Verzeichnis kopiert wurden, kann die Konvertierung duch Aufruf von `splitFiles` gestartet werden. Für jede PostScript-Datei wird ein Verzeichnis erstellt, das den selben Namen trägt wie die Originaldatei, alle Seiten der Datei `01.ps` gelangen in ein Verzeichnis `01`. Unsere Folien sind entsprechend der Kapitelnummern in [HS00] organisiert, d.h., dass beispielsweise alle Folien zum Thema „Architektur“ im Verzeichnis `02` erzeugt werden, da dieses Thema in [HS00] im zweiten Kapitel behandelt wird. Eine andere Einteilung ist aber durchaus möglich. Die Dateinamen sind nicht auf Ziffern beschränkt.

Das Skript testet zunächst, über folgendes `grep`-Kommando, ob jede Datei eine Zeile mit der Anzahl der Seiten besitzt:

```
grep -E -e "^%%Pages: [[:digit:]]*$" datei.ps
```

Liefert dieser Aufruf nichts zurück, dann ist keine entsprechende Zeile vorhanden und es wird eine Fehlermeldung ausgegeben. In diesem Fall wird keine der Dateien gesplittet.

Ist diese Zeile in jeder Datei vorhanden, erfolgt das eigentliche Aufteilen. Dazu wird mittels `psselect` aus der Datei nacheinander jede einzelne Seite extrahiert und mit `Ghostscript` in das PDF-Format konvertiert. Anschließend werden mittels `convert` aus dem `ImageMagick` Paket aus der PDF-Datei je eine PNG- und eine JPG-Datei generiert, die die korrekte Größe für die WWW-Seite besitzen.

Es hat sich gezeigt, dass es mit Ghostscript-Versionen kleiner 7 gelegentlich Probleme bei der Konvertierung gibt. Daher sollte eine entsprechend neue Version verwendet werden. Die wenigsten Probleme traten bei unseren Tests mit den

Versionen 7.x auf. Weiterhin sollte beachtet werden, dass bei der Erstellung der PostScript-Dateien keine Type 3 Fonts, sondern Type 1 Fonts verwendet werden. Anderenfalls ist die Anzeigequalität auf dem Bildschirm ungenügend. Bei Verwendung von LaTeX sei in diesem Zusammenhang auf `psnfss` (`http://www.ctan.org/tex-archive/help/Catalogue/entries/psnfss.html`) und speziell das darin enthaltene `fontenc`-Paket verwiesen. Unter Windows lässt sich dies bei den meisten PostScript-Druckertreibern über die *Erweiterten Optionen* einstellen.

Die Konvertierung in das PDF-Format erzeugt (nach unseren Erfahrungen) die Folien immer in der korrekten Ausrichtung (Hoch- oder Querformat), auch wenn diese in der Originaldatei gemischt waren. Lediglich mit Ghostscript Version 8.x funktionierte dies gelegentlich nicht. In diesem Fall sollte Ghostscript 7.x verwendet werden, mit dem es (zumindest in unseren Tests) keine Probleme gab. Die korrekte Ausrichtung der PNG-Datei gestaltet sich schwieriger, da die Dimensionen des Zielbitmap angegeben werden müssen. Die Ausrichtung der Seite in der PDF-Datei wird von `convert` nicht automatisch erkannt. Eine Möglichkeit dies zu umgehen, ist die Ermittlung der Rotation aus der PDF-Datei. Sie ist in einer Zeile der Form

```
/Rotation rrr...
```

angegeben, wobei `rrr` der Gradzahl entspricht, um die rotiert werden muss. Die drei Punkte deuten an, dass die Zeile noch weitere beliebige Zeichen enthalten kann. Das Skript `splitFiles` verwendet `grep` und `sed` um diesen Wert zu ermitteln. Bei all unseren Versuchen war dies erfolgreich. Es kann aber auch Fälle geben, in denen die Ermittlung der Rotation nicht erfolgreich ist. Sollte dieser Fall eintreten, bleibt nur das Anpassen des Skripts oder das manuelle Rotieren. Konnte der Winkel ermittelt werden, wird dieser mit der `-rotate`-Option an `convert` übergeben. Damit werden auch alle PNG- und JPG-Dateien automatisch in der korrekten Ausrichtung erzeugt.

2.3.2 Erstellen der Medien

Die Erstellung der Medien (Audio- und Videodateien) ist derzeit noch ein vorwiegend manueller Prozess und verursacht somit einen Großteil des Arbeitsaufwandes. Eine grobe Übersicht der Aufbereitung gibt Abbildung 2.13. Die Erstellung erfolgt in folgenden grundlegenden Schritten:

1. Zähler der Kamera, soweit vorhanden, zurücksetzen.
2. Aufnahme der Vorlesung mit einer DV-Kamera.
3. Während der Aufnahme sollten bereits in etwa die Zeiten der Folienwechsel und die `id`s der jeweiligen `MediaObject`s oder die Nummern der Folien

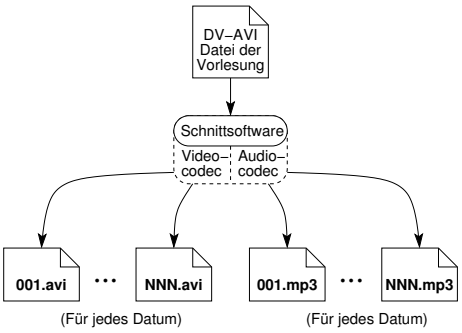


Abbildung 2.13: Aufbereitung der Video- und Audiodateien

notiert werden, da dies den späteren Schnitt vereinfacht. Als Zeitangabe bietet sich der Zählerstand der Kamera an.

4. Kamera mit Computer verbinden, z.B. über IEEE1394-Schnittstelle (Firewire, iLink, ...) und Aufnahme mittels geeigneter Software auf den Rechner übertragen.

5. Mit Videoschnittsoftware die Segmente auswählen und in das Zielformat kodieren. Die Zieldateien müssen dabei die Namen `001.xxx`, `002.xxx`, usw. erhalten, wobei `xxx` für eine beliebige Endung steht, z.B. `avi`. Die genaue Vorgehensweise unter Verwendung von VirtualDub ist weiter unten aufgeführt.

6. Aus den erzeugten Videodateien den Audio-Stream extrahieren und separat ablegen.

7. Die Medien entsprechend der in Abbildung 2.3 zur Webseite beschriebenen Struktur ablegen und ggf. in das Zielverzeichnis kopieren oder verschieben.

Der erste Schritt besteht darin, das Video aufzuzeichnen. Dafür verwendeten wir eine handelsübliche Videokamera aus dem Amateurbereich (konkret eine Sony DCR-TRV 110E). Die Aufzeichnung erfolgte im Digital8 Format. Die Investition in professionelle Technik bringt keine Vorteile, da die Bild- und Tonqualität des letztlich erzeugten Videos durch die verwendete Komprimierung relativ stark abnimmt. Je nach Verhältnissen bietet es sich jedoch an, eine zweite Kamera zu verwenden, z.B. eine speziell für die Aufnahme einer Tafel. Bei der Vorbereitung sollte u.a. Folgendes beachtet werden:

- Die Kamera unbedingt auf einem Stativ montieren und während der Aufnahme möglichst nicht bewegen. Großflächige Bewegungen des Bildinhaltes, wie sie etwa bei Schwenks auftreten, können sich durch die starke Kompression u.U. negativ auf die Bildqualität auswirken.

- Die Kamera mit dem Netzteil am Stromnetz betreiben und zusätzlich mit einem Akku versehen. Ein unerwarteter Ausfall kann so recht wirksam vermieden werden.
- Den automatische Weißabgleich der Kamera möglichst deaktivieren. Ansonsten können starke Helligkeiswechsel des Hintergrundes, z.B. Fenster oder von Projektoren beleuchtete Flächen, zu plötzlichen unerwünschten Änderungen bei der Belichtung führen. Die Kamera stellt sich meist auf den hellen Hintergrund ein, wodurch Personen im Vordergrund zu dunkel erscheinen.
- Helle Hintergründe, z.B. Fenster oder von Projektoren bestrahlte Flächen sollten sich nach Möglichkeit außerhalb des Kamerabildes befinden. Der Kontrast ist in der Regel zu hoch, so dass entweder der Vordergrund zu dunkel oder der Hintergrund zu hell wird. Lässt sich dies, wie im Fall einer von aufgezeichneten Vorlesung, nicht realisieren, sollte die Kamera entsprechend manuell eingestellt werden.
- Um einen Verlust des Videomaterials, beispielsweise durch Hardwaredefekte, Übertragungsprobleme, zu vermeiden, immer auf Kassette aufzeichnen. Falls die technische Möglichkeit besteht, kann zusätzlich eine simultane Übertragung auf eine Rechner erfolgen. Dadurch erspart man sich später die zeitaufwendige Übertragung.
- Sofern vorhanden, den Zähler der Kamera auf 0 setzen. Dies vereinfacht das spätere Auffinden der Segmente.
- Es ist zu empfehlen, bereits während der Aufzeichnung die Zählerwerte der Kamera oder den ungefähren Zeitpunkt des Beginns der einzelnen Segmente zu notieren. Dieses Vorgehen spart beim späteren Schnitt einige Zeit. Weiterhin sollten die Nummern der jeweils verwendeten Folien oder die `ids` der `MediaObjects` notiert werden. Alternativ kann auch sofort die Datei `inhalte.txt` (vgl. Abschnitt 2.3.3) erstellt werden.

Für die Audioaufzeichnung verwendeten wir ein tragbares Funk-Mikrofon. Das in die Kamera integrierte Mikrofon konnte nicht genutzt werden, da sich die Kamera im Publikum befand und somit vorrangig die Umgebungsgeräusche aufgenommen wurden. Mit dem verwendeten separaten Mikrofon wurden selbst auf kurze Entfernung kaum Nebengeräusche aufgezeichnet. Bei der Tonaufzeichnung sollte u.a. folgendes beachtet werden:

- Zunächst muss sichergestellt werden, das die Batterie bzw. der Akku des Mikrofons hinreichend geladen ist. Wir verwendeten einen Akku, der sich für eine 90 minütige Vorlesung als hinreichend erwies.

- Die Räumlichkeiten sollten vor der ersten Aufnahme auf mögliche Störungen bei der Funkübertragung getestet werden.

- Weiterhin sollte untersucht werden, ob die technischen Daten des Ausgangs des Mikrofonempfängers und des Eingangs der Kamera zueinander passen. Ansonsten kann es leicht zu sehr leisen oder stark übersteuerten Aufnahmen kommen. Ggf. muss zusätzlich ein separater Mikrofonverstärker verwendet werden.

- Von einer separaten Aufnahme des Tons, z.B. auf einem Diktiergerät oder direkt auf einen Computer muss abgeraten werden, da sich der Video- und der Audiostrom dann im Nachhinein u.U. nicht oder nur schwer synchronisieren lassen.

Nach der Aufzeichnung muss das gesamte Video zunächst auf Festplatte kopiert werden. Dies erfolgte bei unserer Kamera über die IEEE-1394-Schnittstelle (Firewire, iLink, ...). Die Schnittstelle ließ nur eine Kopie mit Originalgeschwindigkeit zu, so dass dieser Vorgang für eine Vorlesung ca. 90 Minuten dauert. Diese Zeit wurde genutzt, um bereits bekannte Metadaten einzugeben. Für die Aufzeichnung kann jede Video-Schnittsoftware verwendet werden, die das DV-Format und IEEE-1394 unterstützt. In unserem Fall kam Pinnacle Studio 7 zum Einsatz. Soll das Videomaterial lediglich auf die Festplatte übertragen werden, bietet sich alternativ auch DVIO (`http://www.carr-engineering.com/dvio.htm`) an. Es gilt beim Kopieren folgendes zu beachten:

- Die Festplatte muss hinreichend schnell sein. Aktuelle IDE- und SCSI-Festplatten bereiten keine Probleme. Netzwerklaufwerke sollten nicht verwendet werden. Dies unterbindet die Schnittsoftware in der Regel ohnehin.

- Es muss genügend Speicherplatz vorhanden sein. Für eine 90-minütige Vorlesung werden ca. 20 GB benötigt.

- Alle nicht notwendigen Hintergrundprogramme sollten abgeschaltet werden. Dies schließt auch den Bildschirmschoner ein.

- Es sollte möglichst ein Einzelplatzrechner verwendet werden, so dass keine Gefahr einer Störung über das Netzwerk besteht.

Nach der Übertragung auf die Festplatte müssen die Dateien für die einzelnen Segmente erstellt werden. Dies erfolgte vorwiegend in Handarbeit unter Verwendung von VirtualDub. Wir gehen im Weiteren davon aus, dass der Umgang mit der Software bekannt ist. Zunächst werden die Einstellungen für den Encoder vorgenommen:

- Im ersten Schritt muss über einen Filter ein Deinterlacing vorgenommen werden. Da das Bild ohnehin auf ein Viertel (360x288) verkleinert werden soll, bietet sich die Einstellung *Discard ...field* an. Es ist dabei egal ob *odd* oder *even* verwendet wird.
- Weiterhin muß die Breite des Bildes halbiert werden. Dazu wird der *Resize*-Filter mit einer Zielgröße von 360x288 verwendet.
- **Anmerkung:** Die Anwendung von *Resize* ohne vorheriges Deinterlacing ist nicht zu empfehlen, da dies (zumindest in unseren Versuchen) zu „Geisterbildern" führte. Dies liegt vermutlich daran, dass beim Verkleinern über zwei Halbbilder interpoliert wird, die sich bei Bewegungen unterscheiden.
- Schließlich müssen noch die Codecs nach Bedarf ausgewählt und eingestellt werden. Wir verwenden für den Ton das MP3-Format mit einem Kanal (Mono) bei einer Sampelrate von 24 kHz und einer Bitrate von ca. 24kBit/s. Für gesprochenen Text liefern diese Einstellungen eine gute Qualität bei geringem Datenvolumen. Für das Video verwendeten wir DivX 4 mit einer Bitrate von ca. 200kBit/sec. Eine 90-minütige Vorlesung lässt sich damit auf ca. 150MB komprimieren. Kodiert wurde im 1-Pass Verfahren. Der 2-Pass-Mode, und die damit verbundene Verdopplung der Kodierzeit, brachte keine nennenswerten Vorteile.

Anschließend werden nacheinander die einzelnen Segmente markiert, wobei die während der Aufzeichnung erstellte Liste von Zeiten als Hilfestellung verwendet wird. Jedes Segment wird als Job der Job-Liste hinzugefügt. Wurden alle Segmente markiert, kann der Job gestartet werden. Je nach Leistung des verwendeten Rechners und des Codecs, kann dies einige Stunden in Anspruch nehmen. Wir haben die Kodierung daher meist über Nacht durchgeführt. Im Letzten Schritt müssen noch, wenn dies gewünscht wird, die Audiodateien separat gespeichert werden. Dies erfolgt in VirtualDub über die Funktion *Save WAV*.

Nachdem alle Videodateien im Verzeichnis `video` und alle Audiodateien im Verzeichnis `audio` abgelegt wurden, ist die Erstellung der Medien abgeschlossen. Vor der Erstellung der HTML-Seiten ist es nun noch erforderlich, die fehlenden Metadaten (Dateigrößen und Spieldauer) zu generieren.

2.3.3 Eingabe der Metadaten

Die Eingabe der Metadaten ist teilweise durch ein Skript automatisiert. Dieses erzeugt ein Template mit einigen medienbezogenen Informationen, die noch vervollständigt und in die XML-Datei mit den übrigen Informationen integriert werden müssen. Die notwendigen Schritte bei der Eingabe sind:

1. Erstellen der Medien wie im vorherigen Abschnitt beschrieben.

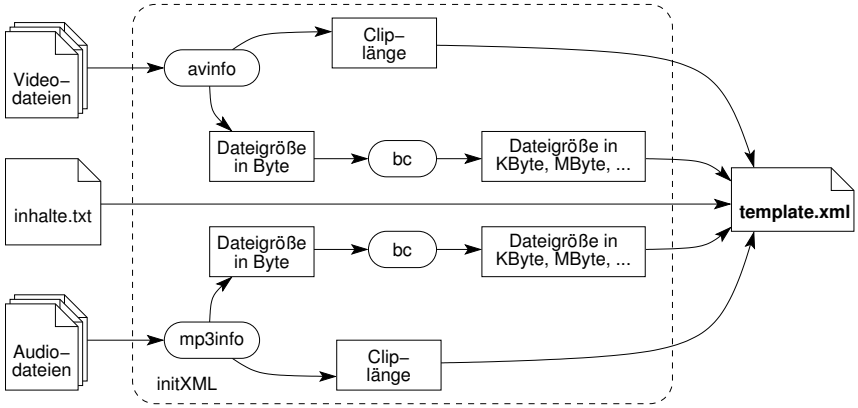


Abbildung 2.14: Generierung der Metadaten

2. Allgemeine Informationen (Bücher, Formate, etc.) und Informationen zu den Medienobjekte (Learningunits, Learningmodules, etc.) eingeben.

3. Erstellen der Datei `inhalte.txt` im entsprechenden Verzeichnis (vgl. Abschnitt 2.2.1). Der Aufbau der Datei ist weiter unten beschrieben.

4. Aufruf des Skripts `initXML` mit den unten angegebenen Optionen.

5. Ergänzen der fehlenden Informationen in der generierten XML-Datei, z.B. Titel von `GO`-Elementen.

6. Das erzeugte XML in die Datei mit den übrigen Informationen integrieren.

Zunächst müssen die allgemeinen Informationen zu Büchern, Formaten und Inhalten eingegeben werden. Eine Teilautomatisierung dieser Schritte, insbesondere die Erfassung von Informationen der Folien ist derzeit in Arbeit. Weiterhin ist die Eingabe der allgemeinen Informationen zu jeder Veranstaltung (Ort, Zeit, Teilnehmer, etc.) erforderlich.

Für die Eingabe der `Lecture`-Elemente steht ein Skript zur Verfügung, das die Tags automatisch erzeugt und die Informationen zu den Medien einfügt (vgl. Abbildung 2.14). Das Skript benötigt dazu den Kommandozeilenrechner `bc`, das Programm `mp3info` zur Ermittlung der Informationen der Audiodateien und das Programm `avinfo`, das Informationen über die Videodateien liefert. Die Version von `avinfo` wurde geringfügig verändert, damit die Ausgaben beider info-Programme gleiche Formate liefern. Werden die Medien nicht als mp3 und avi bereitgestellt, funktioniert das Skript nicht korrekt und muss ggf. angepasst werden, z.B. indem `mp3info` und `avinfo` durch andere Programme ersetzt werden, die die notwendigen Informationen liefern. Mittels des Kommandozeilenrech-

ners `bc` wird aus den Ausgaben von `mp3info` und `avinfo` die Dateigröße berechnet, wobei die Einheit (Byte, kByte, MByte, etc.) automatisch angepasst wird.

Zusätzlich zu den erwähnten Programmen werden noch die fertig kodierten Medien und eine Textdatei (`inhalte.txt`) mit der Auflistung der verwendeten `MediaObject`s (Angabe der `id`) benötigt. Die Medien müssen dabei in der vorgeschriebenen Struktur organisiert sein. Die Textdatei mit den `id`s muss sich im Verzeichnis eines Vorlesungstermins befinden, d.h. im Verzeichnis, das das Datum der Vorlesung trägt (vgl. Abschnitt 2.2.1). Die Inhalte für die Vorlesung vom 19.10.2001 der Veranstaltung „C1“ befinden sich beispielsweise in der Datei `./medien/C1/20011019/inhalte.txt`. Der Aufbau der Datei sieht beispielsweise folgendermaßen aus:

```
Org.DB1.1
Org.DB1.2
Org.DB1.3
Org.DB1.4
Org.DB1.5
Org.DB1.6
Org.DB1.7
LO1.1.0.1
LO1.1.1.1
LO1.1.1.2
LO1.1.1.3
LO1.1.1.4
LO1.1.2.1
LO1.1.3.1
LO1.1.3.2
LO1.1.3.3
LO1.1.3.4
LO1.1.3.5
LO1.2.1.1
```

Es muss beachtet werden, dass für jedes Medienobjekt, d.h. jedes Paar aus avi- und mp3-Datei genau eine Zeile in der Textdatei existiert. Die Zuordnung erfolgt an Hand der Zeilennummer, d.h. die Datei `005.avi` wird dem `MediaObject` in der fünften Zeile zugeordnet (`Org.DB1.5`). Aus diesen Angaben werden alle `Segment`-Elemente erzeugt. Weiterhin prüft das Skript, ob sich die Zahl zwischen dem zweiten und dritten Punkt der `id` geändert hat. In diesem Fall wird zusätzlich ein `GO`-Element erzeugt. Im Beispiel würden somit u.a. `LO1.1.1.1` bis `LO1.1.1.4` in einem `GO`-Element zusammengefasst. Der Titel muss später von Hand nachgetragen werden. Dies funktioniert jedoch nur dann, wenn die vorgeschlagene Identifizierung der Lernobjekte mittels ids der Form LOu.v.w.x verwendet wird. In einer Weiterentwicklung sollen auch die Titel aus den zugehörigen `GroupObject`s automatisch ergänzt werden.

Sind alle Voraussetzungen erfüllt, kann das Skript `initXML` gestartet werden. Es besitzt folgende Optionen (optionale in eckigen Klammern):

`-IN quelle`
Name des Verzeichnisses, in dem sich die Dateien (Medien) befinden. Das Skript testet zunächst, ob sich unterhalb des angegebenen Verzeichnisses zwei weitere mit den Namen `./audio` und `./video` befinden. Das würde bedeuten, es wurde als Quelle das Verzeichnis einer einzelnen Vorlesung angegeben. Ist das der Fall werden nur die Daten dieser Medien ausgegeben. Ansonsten wird davon ausgegangen, dass das Verzeichnis einer Veranstaltung mit mehreren Vorlesungsterminen übergeben wurde und es werden alle vorhandenen Unterverzeichnisse durchlaufen und die darin enthaltenen Medien ausgegeben.

`-VCODEC id`
Gibt die id des verwendeten Formats für die Videodateien an. In der XML-Datei muss ein `VFormat`-Element mit dieser id existieren.

`-ACODEC id`
Gibt die id des verwendeten Formats für die Audiodatein an. In der XML-Datei muss ein `AFormat`-Element mit dieser id existieren.

`[-VTYPE Suffix]`
Legt den Typ der Videodateien fest, d.h. den Suffix der Dateinamen ohne den Punkt. Wird der Typ nicht angegeben, wird standardmäßig „avi“ verwendet. Bei der Angabe des Suffix müssen die Groß- und Kleinschreibung beachtet werden.

`[-ATYPE Suffix]`
Legt den Typ der Audiodateien fest, d.h. den Suffix der Dateinamen ohne den Punkt. Wird der Typ nicht angegeben, wird standardmäßig „mp3“ verwendet. Bei der Angabe des Suffix müssen die Groß- und Kleinschreibung beachtet werden.

Abschließend zwei Beipiele:

`initXML -IN ./medien/C1/20011019 -ACODEC A1 -VCODEC V1`
Erzeugt XML für alle Segmente der Vorlesung vom 19. Oktober 2001, wobei die `Video`-Elemente das Format mit der id „V1“ und die `Audio`-Element das Format mit der id „A1“ zugeordnet bekommen.

`initXML -IN ./medien/C1 -ACODEC A1 -VCODEC V1`
Erzeugt alle Elemente für die Veranstaltung mit der id „C1“. Grundsätzlich entspricht der Aufruf dem aus dem vorherigen Beispiel, nur werden alle Unterverzeichnisse (Termine) ausgegeben.

Die Ausgabe des Skripts erfolgt auf der Konsole und kann nach Bedarf in eine Datei umgeleitet und anschließend vervollständigt werden. Schließlich ist noch das Einfügen in die eigentliche XML-Datei erforderlich. Nun können die HTML-Dateien für die WWW-Seite erzeugt werden.

2.3.4 Erzeugen der HTML-Dateien

Mit Hilfe der Skripte `termine` und `themen` können zwei verschiedene Varianten der WWW-Seite erzeugt werden. Diese unterscheiden sich nur in der Struktur des Menüs und der Anordnung der Links zu den Medien (vgl. Abschnitt 2.2.2). Dies hat jedoch keine Auswirkungen auf die eigentliche Ausführung der Skripte. Daher wird der Prozess im folgenden lediglich an Hand des Skripts `termine` beschrieben. Folgende Schritte sind auszuführen:

1. Ein Verzeichnis erstellen und die Skripte zusammen mit dem `src`-Verzeichnis in dieses kopieren. Das `src`-Verzeichnis beinhaltet alle notwendige Verzeichnisse, JavaScript-Dateien, Bilder, etc., die für die WWW-Seite erforderlich sind.

2. Die XML-Datei mit den Metadaten ebenfalls in dieses Verzeichnis kopieren.

3. In das erstellte Verzeichnis wechseln und das Skript `termine` und/oder `themen` mit den unten angegebenen Optionen aufrufen.

4. Das Skript kopiert alle notwendigen Dateien (alles im `src`-Verzeichnis) und startet Xalan mit den notwendigen Optionen. Anschließend befinden sich die HTML- und sonstigen Dateien in der korrekten Struktur im angegebenen Zielverzeichnis.

Der grundlegende Prozess zur Erstellung der HTML-Seiten und die verwendeten Werkzeuge sind in Abbildung 2.15 dargestellt. Grundlage ist eine korrekte XML-Datei, die der im Abschnitt 2.1 beschriebenen Struktur entspricht. Zunächst müssen daher die notwendigen Metadaten für jedes einzelne Segment und ggf. begleitende Daten wie Bücher, Abschnitte etc. hinzugefügt werden. Für diese Zwecke benutzten wir einen Texteditor und das im Abschnitt 2.3.3 vorgestellte Skript.

Für die Erstellung der HTML-Seiten haben wir ein XSLT-Skript erstellt, das mittels Xalan-Java ausgeführt wird. Prinzipiell sollten auch andere XSLT-Prozessoren einsetzbar sein, wenn die notwendigen Extensions unterstützt werden (siehe `http://exslt.org/"`) oder das Skript entsprechend angepasst wird. Benötigt werden folgende Extensions:

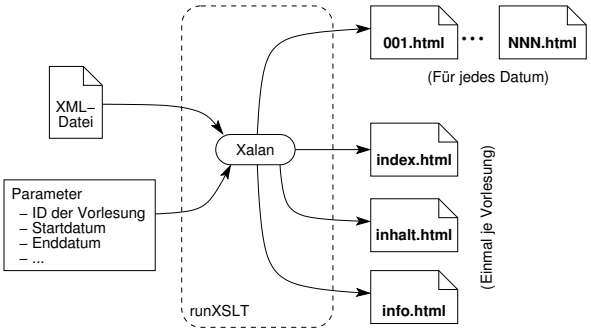


Abbildung 2.15: Erzeugung der HTML-Dateien

`redirect` Diese Extension dient dem Schreiben mehrerer Ausgabedateien (siehe dazu `http://xml.apache.org/xalan-j/extensionslib.html#redirect`). Dies ist notwendig, um mit einem Aufruf von Xalan alle Dateien schreiben zu können.

`date` Diese EXSLT-Extension stellt verschiedene Funktionen zur Arbeit mit Datumswerten bereit (siehe `http://exslt.org/date/index.html`). Sie wird verwendet, um die Seiten für einzelne Vorlesungstermine zu erzeugen und die Datumswerte zu formatieren.

`str` Diese EXSLT-Extension stellt verschiedene Funktionen zur Arbeit mit Zeichenketten zur Verfügung (siehe dazu `http://exslt.org/str/index.html`). Benötigt wird diese Extension, um evtl. vorhandene Zeilenumbrüchen in `title`-Elementen zu entfernen. Diese würden sonst zu Fehlern im erzeugten JavaScript führen.

Für die Transformation sind weiterhin verschiedene Informationen, wie die id der Vorlesung, das Zielverzeichnis und optional ein Zeitraum erforderlich. Zur Vereinfachung des Aufrufs haben wir wiederum ein Shell-Skript erstellt, welches diese Parameter als Eingabe erhält. Durch das Skript werden zunächst alle notwendigen Verzeichnisse erstellt und Dateien kopiert (vgl. Abschnitt 2.2). Letztere befinden sich im `src`-Verzeichnis, welches wir zusammen mit dem Skript bereitstellen. Die notwendigen Informationen erhält das Skript über folgende Optionen:

`-IN quelle`
Das Argument gibt die XML-Quelldatei mit den Metadaten an. Das Skript testet, ob diese Datei existiert. Falls nicht, wird eine Fehlermeldung ausgegeben. Das Skript testet jedoch nicht, ob die Datei auch eine passende XML-Datei ist. Sollte dies nicht der Fall sein, meldet Xalan nach seinem Aufruf in der Regel einen Fehler.

-OUT ziel
Gibt ein Verzeichnis an, in dem die Dateien erstellt werden sollen. Existiert dieses Verzeichnis bereits, werden vorhandene Dateien überschrieben. Falls nicht, wird eine Verzeichnisstruktur entsprechend Abschnitt 2.2.1 erzeugt.

-VORL vID
Mit diesem Argument wird angegeben, für welche Vorlesung die Dateien erzeugt werden. `vID` ist dabei die id eines `Course`-Elements aus der XML-Datei. Mit jedem Aufruf des Skripts können jeweils nur Dateien für eine Vorlesung erzeugt werden können.

[-START Datum]
Mit dieser Option kann das Startdatum eines Zeitraums angegeben werden. In diesem Fall werden nur `Segmente` betrachtet, deren `Datum` größer oder gleich dem angegeben ist. Wird zusätzlich kein Enddatum angegeben, so werden lediglich die Dateien für den angegebenen Tage erzeugt. Die Angabe erfolgt im Format `jjjj-mm-tt`. Das Skript prüft lediglich, ob das Format stimmt und ob `jjjj`, `mm` und `tt` nur Ziffern enthalten. Es wird jedoch nicht getestet, ob es sich um ein korrektes Datum handelt. Da die Werte vom Skript zunächst in einen einzelnen Integer-Wert umgerechnet werden, kommt es auch bei der Angabe ungültiger Werte zu keiner Fehlermeldung.

[-ENDE Datum]
Mit dieser Option kann das Enddatum eines Zeitraums angegeben werden. Weiterhin gilt das bereits beim Startdatum erwähnte.

[-INHALT (alles | Teil)]
Mit diesem Argument wird schließlich angegeben, ob das Menü vollständig, d.h. unabhängig vom angegebenen Start- und Enddatum für alle Termine der gewählten Vorlesung, erzeugt werden soll. Die Angabe `alles` bietet sich bei der Erstellung einer WWW-Seite an, die gelegentlich um eine Vorlesung erweitert wird. Es werden dann jeweils nur die notwendigen HTML-Dateien für den hinzukommenden Termin, aber das gesamte Menü erzeugt. Die Angabe `Teil` ist andererseits von Vorteil, wenn ein Ausschnitt erzeugt werden soll, beispielsweise für die Publikation auf CD-ROM. In diesem Fall wird ein Menü erzeugt, das nur den relevanten Teil enthält. Wurde dieses Argument nicht angegeben, so wird `alles` als default angenommen. Wurden weder Start- noch Enddatum angegeben, wird dieser Parameter ignoriert.

[-ORG (ja | nein)]
Legt fest, ob die Segmente mit organisatorischen Informationen mit in das Inhaltsverzeichnis aufgenommen werden sollen. Standardwert ist `ja`. Sollen die organisatorischen Informationen, z.B. für eine Publikation auf CD-

ROM oder DVD, nicht erscheinen, muss `nein` angegeben werden. Wenn `ja` angegeben wird, werden immer die organisatorischen Informationen aller Termine erzeugt, unabhängig von einem evtl. angegebenen Zeitraum.

`[-VTYPE Suffix]`
Legt den Typ der Videodateien fest, d.h. den Suffix der Dateinamen ohne den Punkt. Wird der Typ nicht angegeben, wird standardmäßig `avi` verwendet. Bei der Angabe des Suffix muss die Groß- und Kleinschreibung beachtet werden.

`[-ATYPE Suffix]`
Legt den Typ der Audiodateien fest, d.h. den Suffix der Dateinamen ohne den Punkt. Wird der Typ nicht angegeben, wird standardmäßig `mp3` verwendet. Bei der Angabe des Suffix muss die Groß- und Kleinschreibung beachtet werden.

Für den Fall, dass das Skript nicht verwendet und Xalan direkt aufgerufen wird, soll abschließend die verwendete Umrechnung des Datums in eine Zahl erwähnt werden. Dies erfolgt mittels folgender Formel:

$$x = ((jjjj - 2000) * 12 + mm) * 31 + tt$$

Da der Wert lediglich für Vergleiche verwendet wird, ist dieser Umrechnung hinreichend. Start- und Enddatum müssen in dieser Form übergeben werden, da das Stylesheet die Werte in `//Lecture/@date` ebenso umrechnet, bevor der Vergleich erfolgt. Abschließend einige Beispiele:

`termine -IN metadata -OUT result -VORL C1`
Erzeugt im aktuellen Verzeichnis ein neues mit dem Namen `result` und kopiert alle notwendigen Dateien dorthin. Anschließend wird Xalan aufgerufen, welches für sämtliche Termine der Vorlesung „C1“ die HTML-Dateien generiert. Auch die Segmente mit organisatorischem Inhalt werden generiert. Da weder ein Typ für die Video- noch die Audiodateien angegeben wurde, wird automatisch avi bzw. mp3 angenommen.

`termine -IN metadata -OUT result -VORL C1 -START 2001-11-23`
Dieser Aufruf hat den gleichen Effekt, mit dem Unterschied, das lediglich die Dateien für den 23. November 2001 erzeugt werden. Da das Argument `-INHALT` fehlt, wird jedoch das vollständige Menü mit allen Terminen und Themen der Vorlesung erzeugt. Auch in diesem Fall werden die Segmente mit organisatorischem Inhalt generiert.

`termine -IN metadata -OUT result -VORL C1 -START 2003-01-09 -ENDE 2003-01-23`

Dieser Aufruf erzeugt alle Dateien von Vorlesungsterminen zwischen dem 9. Januar und dem 23. Januar 2003. Da das Argument `-INHALT` wiederum fehlt, wird auch in diesem Fall das komplette Menü erzeugt.

`termine -IN metadata -OUT result -VORL C1 -START 2003-11-23 -INHALT Teil`
Dieser Aufruf erzeugt das selbe Ergebnis, wie das zweite Beispiel, allerdings mit dem Unterschied, dass das Menü diesmal ebenfalls lediglich den 23. November 2003 enthält.

`termine -IN metadata -OUT result -VORL v2 -ORG nein -VTYPE mp2 -ATYPE ogg`
Dieser Aufruf erzeugt das selbe Ergebnis, wie das erste Beispiel, allerdings werden die organisatorischen Segmente nicht generiert. Weiterhin werden in diesem Fall nicht die Standardendungen avi und mp3, sondern mp2 und ogg verwendet. Die Medien müssen dann natürlich bereits in diesen Formaten vorliegen.

Für die Generierung der Seiten mit thematischer Strukturierung wird eine ähnliches Skript (`themen`) bereitgestellt. Dieses besitzt folgende Optionen:

`-IN quelle`, `-OUT ziel`, `[-VTYPE Suffix]`, `[-ATYPE Suffix]`
Diese Optionen entsprechen denen des zuvor beschriebenen Shell-Skripts `termine`.

`[-INHALT (alles | Teil)]`
Diese Option entspricht der des zuvor beschriebenen Skripts `termine`, mit dem Unterschied, dass sich die Beschränkung nicht auf gewählte Termine, sondern Learningunits bezieht. Im Falle `ja` (Standardwert) wird ein Menü mit Einträgen aller `LearningUnit`-Elemente in der XML-Datei generiert, im Falle `nein` nur für die über die `-LU` Option gewählte. Wird die `-LU` Option nicht angegeben, wird die `-INHALT` ignoriert und alles erzeugt.

`[-LU loid]`
Optionale Id einer Learningunit (z.B. `LO2`). Wird diese Option angegeben, erstellt das Skript nur die HTML-Dateien für dieses eine Objekt. Wird die Option nicht angeben, werden alle Dateien erstellt.

2.3.5 Publikation im WWW

Wurde das Shell-Skript für die Erstellung der HTML-Dateien verwendet, so befinden sich diese anschließend zusammen mit allen anderen notwendigen Dateien in der notwendigen Struktur. Wenn erforderlich muss dieser Baum an eine für den Web-Server erreichbare Stelle kopiert werden. Wurde das Verzeichnis bereits beim Aufruf des Skripts als Zielverzeichnis angegeben, kann dieser

Schritt entfallen. Für jeden Vorlesungstermin müssen anschließend die `audio-` und `video-`Verzeichnisse erzeugt werden. Abschließend werden die erstellten Video- und Audio-Dateien hineinkopiert.

Literaturverzeichnis

[ADE+03] Klaus Alfert, Ernst-Erich Doberkat, Gregor Engels, Marc Lohmann, Johannes Magenheim, and Andy Schürr. MuSofT: Multimedia in der SoftwareTechnik. In Johannes Siedersleben and Debora Weber-Wulff, editors, *SEUH 8 Software Engineerung im Unterricht der Hochschulen, Berlin 27./28. Februar 2003*, pages 70–80, Heidelberg, February 2003. dPunkt-Verlag.

[ADL02] Advanced Distributed Learning (ADL). *Shareable Content Object Reference Model (SCORM) Version 1.2*, October 2002.

[HS00] Andreas Heuer and Gunter Saake. *Datenbanken: Konzepte und Sprachen*. Informatik Lehrbuch-Reihe. MITP-Verlag, Bonn, 2. auflage edition, 2000.

[IEE02] Institute of Electrical and Electronics Engineering, Inc., Learning Technology Standardization Committee (LTSC). *Draft Standard for Learning Object Metadata (LOM)*, July 2002. `http://ltsc.ieee.org/`.

[Mas03] Massachusetts Institute of Technology. MIT OpenCourseWare. online, 2003. `http://ocw.mit.edu/`.

[MuS03] MuSofT – Multimedia in der SoftwareTechnik. online, 2003. `http://www.musoft.org/`.

[Stu03] Universität Stuttgart. 100-online. online, 2003. `http://www.uni-stuttgart.de/100-online/`.

[The03] The University of Texas at Austin. World Lecture Hall. online, 2003. `http://www.utexas.edu/world/lecture/`.