

Kurse erstellen mit Learning Object Metadata

Stefan Hoermann und Ralf Steinmetz

Multimedia Kommunikation - KOM
Fachbereich Elektrotechnik und Informationstechnik
Technische Universität Darmstadt
Merckstr. 25 • D-64283 Darmstadt

{hoermann, steinmetz}@kom.tu-darmstadt.de

Zusammenfassung: Im Rahmen des k-MED-Projektes, eines vom BMB+F geförderten Gemeinschaftsprojektes elf medizinischer, drei technischer und eines gestalterischen Lehrstuhls, werden an der TU-Darmstadt Werkzeuge zum Speichern, Verwalten, Auffinden und Strukturieren von modularisierten Lernressourcen für das medizinische Grundstudium entwickelt. Hierfür werden technische Hilfsmittel entwickelt, um insbesondere bereits bestehende Ressourcen zu beschreiben und mit Ressourcen, die sich in einer multimedialen Wissensbasis befinden, zu verknüpfen. Ziel ist es, dadurch die Wiederverwendung von Lehrmaterialien zu optimieren. Die Nutzung modularisierter Unterrichtsmaterialien stellt Autoren vor die im didaktischen Kontext neuartige Aufgabe, eine wiederverwendbare Wissensbasis zu erstellen. Für die Erstellung und Speicherung von Kursstrukturen zur Verwendung in Lernsystemen und zum Austausch von Kursen werden Datenstrukturen benötigt, die möglichst auf weit verbreiteten Standards beruhen, um eine möglichst hohe Interoperabilität zwischen verschiedenen Systemen zu gewährleisten. Im vorliegenden Papier wird gezeigt, wie mit bereits bestehenden WBT-Standards Kursstrukturen mit modularisierten Lernressourcen erstellt werden können, und eine Erweiterung zu Learning Objects Metadata vorgestellt, auf dessen Basis das Erstellen von adaptierbaren Kursen möglich wird. Im Zusammenhang mit rhetorisch-didaktischen Relationen werden Möglichkeiten zur Adaption dieser Kurse gezeigt. Anschließend wird ein Editor beschrieben, mit dem basierend auf einer Wissensbasis adaptierbare Kurse nach der vorgestellten Methode erstellt werden können.

1 Einleitung

k-MED ist ein vom BMB+F gefördertes Gemeinschaftsprojekt elf medizinischer, drei technischer und eines gestalterischen Lehrstuhls innerhalb des Bundeslandes Hessen. Das Projekt verfolgt das Ziel, Fächer der Medizin multimedial aufzubereiten, die Inhalte Studierenden in einem flexibel konfigurierbaren web-basierten Lernsystem anzubieten und Lehrenden Material für die Lehre zur Verfügung zu stellen. Bei den medizinischen Fächern handelt es sich um Fächer der Vorklinik sowie um Fächer des klinischen Studienabschnitts.

Der in k-MED verfolgte technische Ansatz ist insofern originär, als zugleich die Modellierung des ausgewählten medizinischen Wissens und die Entwicklung multimedialer Wissensmodule betrieben werden. Auf der Basis der Wissensmodellierung wird es möglich, die Wissensmodule miteinander zu vernetzen und so ein integriertes Gesamtbild zu erzeugen, das die bisherige getrennte Betrachtungsweise überlappender Fachgebiete aufhebt.

Das für k-MED vorgesehene Konzept unterscheidet sich von vorhandenen Multimedia-Produktionen auf dem Medizinsektor. Fächerbezogenes Wissen wird vollständig modular und

multimedial aufbereitet (Wissensmodule). Mit diesem Material konstruieren die Autoren während der Projektlaufzeit instruktionale Einheiten, also Kurse Studienelemente, einerseits für unterschiedliche Zielgruppen, andererseits für unterschiedliche Lernvorgaben und Lernbedürfnisse, d.h. sowohl auf fachsystematisches Lernen bezogene als auch problem- oder fallorientierte und damit fächerübergreifende Kurse. Die technische Plattform erlaubt es den Autoren, in ihre Kurse Wissensmodule einzubauen, die in anderen Fächerzonen erzeugt wurden. Die Plattform ermöglicht auch, dass (andere) Autoren aus den produzierten Wissensmodulen später weitere Kurse mit anderen Zielsetzungen bilden, und sie unterstützt eine vergleichsweise problemlose Aktualisierung der Wissensmodule.

Obwohl der Aufwand der Modularisierung der Lernressourcen in manchen Fällen etwas höher ist, ergeben sich daraus trotzdem verschiedene Vorteile. Beispielsweise wird der Grad der Wiederverwendung von bereits erstellten Lernmodulen deutlich verbessert. Hieraus resultiert direkt auch eine Kostensenkung bei der Erstellung neuer Lernangebote durch die Verwendung bereits bestehender Lernressourcen. Hierfür werden Möglichkeiten benötigt modularisierte, multimediale Lerneinheiten zu Kursen zu strukturieren. Beim Entwurf dieses Lösungsansatzes standen besonders die Verwendung standardisierter Komponenten im Vordergrund, um die Interoperabilität zwischen verschiedenen Systemen zu gewährleisten [Referenz auf das Paper, das mir Cornelia gegeben hat]. Hierdurch, und durch die Trennung des Inhalts von der Darstellung wird die Möglichkeit der Wiederverwendung der zu Kursen und Einheiten zusammengefassten modularisierten, multimedialen Lernressourcen deutlich erhöht. So wird die Verwendung bereits erstellter Kurse in anderen Kontexten, mit verschiedenen Layouts und Formaten und sogar in anderen Lernsystemen möglich.

Im vorliegenden Papier werden in Abschnitt 2 verwandte Arbeiten diskutiert. In Abschnitt 3 wird ein Überblick über k-MED gegeben, um den Kontext, in dem die nachfolgend vorgestellten Lösungsansätze entstanden sind, zu beleuchten. Anschließend wird in Abschnitt 4 der Lösungsansatz zur Strukturierung von multimedialen Modulen feiner Granularität vorgestellt. Hierfür werden zunächst benötigte Datenelemente eingeführt. In Abschnitt 5 wird der zuvor beschriebene Lösungsansatz in Bezug auf Wiederverwendbarkeit und Adaptierbarkeit untersucht. Danach wird in Abschnitt 6 die Generierung von Präsentationen von Kursen, die nach dem in Abschnitt 4 vorgestellten Schema erstellt wurden, diskutiert. Hierbei wird die Generierung von Präsentationen in der Online- und Printversion untersucht. Schließlich wird in Abschnitt 7 eine Implementierung vorgestellt, mit der Kurse unter der Verwendung bereits bestehender Ressourcen nach dem in Abschnitt 4 vorgestellten Schema erstellt werden können. Mit Abschnitt 8 endet das vorliegende Papier mit einer Zusammenfassung und einem Ausblick.

2. Related Work

Im Folgenden werden einige Arbeiten vorgestellt, die zu dieser verwandt sind.

SCORM (Shareable Content Object Reference Model) [ADL01] ist eine Entwicklung der Advanced Distributed Learning (ADL) Initiative. SCORM stellt ein Referenzmodell für web-basierte Lernressourcen dar, bei dessen Entwicklung Ziele wie Zugänglichkeit, Interoperabilität, Dauerhaftigkeit und Wiederverwendbarkeit im Vordergrund stand. Das Referenzmodell fasst in diesem Bereich Spezifikationen und Richtlinien bereits etablierter Organisationen wie IMS Global Learning Consortium, Inc., AICC (Aviation Industry CBT Committee) und IEEE LTSC (Learning Technology Standards Committee) zusammen und verbindet sie in einem größeren Zusammenhang. Das Content Aggregation Model [ADL01] von SCORM, bestehend aus Metadaten nach IEEE und Content Structure Format, sind sehr verwandt mit der vorliegenden Arbeit.

Learning Material Markup Language (LMML) ist eine vielseitig anpassbare und erweiterbare Familie von XML-Sprachen, die speziell für die XML-basierte Auszeichnung von Lehr- und Lernmaterial im Rahmen des Passauer Knowledge Management Systems (PaKMaS) [SF00] konzipiert wurde. LMML stellt ein generisches Framework dar, das je nach Bedarf des Einsatzgebietes von der universitären Ausbildung bis hin zum firmeninternen Training beliebig angepasst werden kann. Das LMML-Framework basiert auf einem Meta-Modellierungsansatz [S00][SFB99], mit dem auf der obersten Ebene ein allgemeines Metamodell zur Beschreibung von Lehrmaterial definiert wird. Ausgehend von diesem Metamodell können Instanzen abgeleitet werden und den anwendungsspezifischen Anforderungen entsprechend angepasst werden. Das LMML-Framework ist zwar geeignet für die Strukturierung von modularisierten Lernressourcen, bedient sich aber nicht verbreiteten Standards z.B. zur Speicherung von Metadaten.

TeachML ist eine Dokumentensprache zur Auszeichnung von XML-basierten Kursdokumenten, die im Projekt Targeteam (TARgeted Reuse and GEneration of TEACHing Materials) [Tee01] entwickelt wird. TeachML soll als Grundlage für plattformunabhängige und modular aus Lernobjekten und vorstrukturierten Dokumentenvorlagen aufgebaute Dokumentenmodell dienen. TeachML bietet geeignete Konzepte für die Präsentation dieser Dokumente in unterschiedlichen Ausgabeformaten. Zur Speicherung der Metadaten zu den einzelnen Modulen greift TeachML auf IMS-Metadaten zurück.

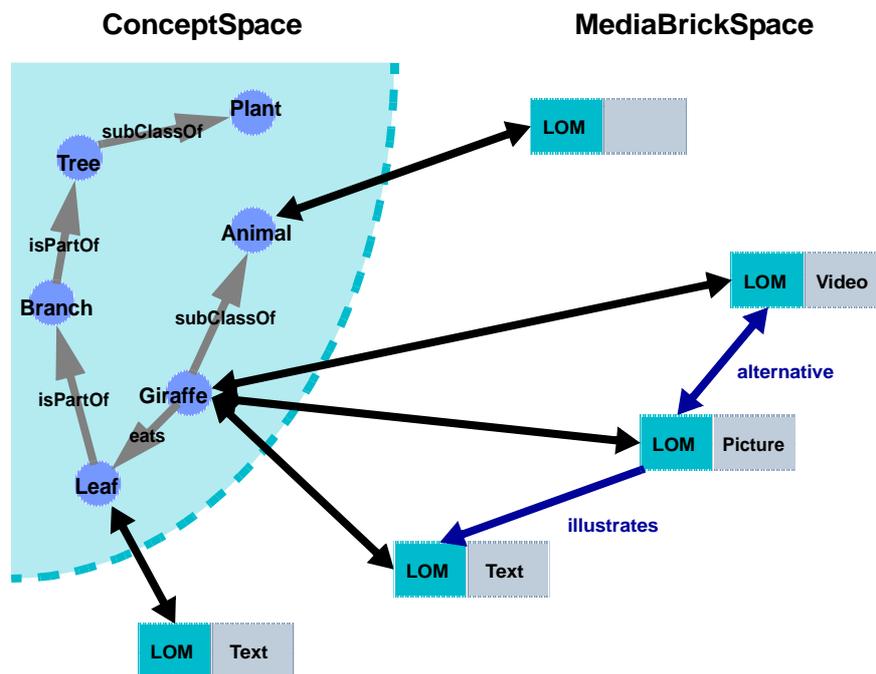
Schließlich ist noch DocBook [WM99] zu erwähnen, das zur Weiterentwicklung in der Hand des DocBook Technical Committees bei OASIS [Doc02] liegt. DocBook wurde speziell für die Erstellung von Dokumentationen für Computer im Soft- und Hardwarebereich entwickelt. Mit einem Umfang von mehr als 400 verschiedenen Datenelementen eignet sich DocBook auch für die Erstellung von Büchern, Artikeln und Online-Dokumenten in anderen Anwendungsbereichen als Computerdokumentationen. Aufgrund dieser Zielsetzung eignet sich DocBook weniger für die Strukturierung von modularisierten Lernressourcen bietet aber dennoch interessante Lösungen für die Auszeichnung von hochqualitativen Online- und Printtexten.

3. Systemüberblick

In diesem Abschnitt soll ein Überblick über die Wissensbasis und das technische Rahmensystem von k-MED gegeben werden, um das Umfeld für den in dem vorliegenden Papier dargestellten Ansatz zu klären.

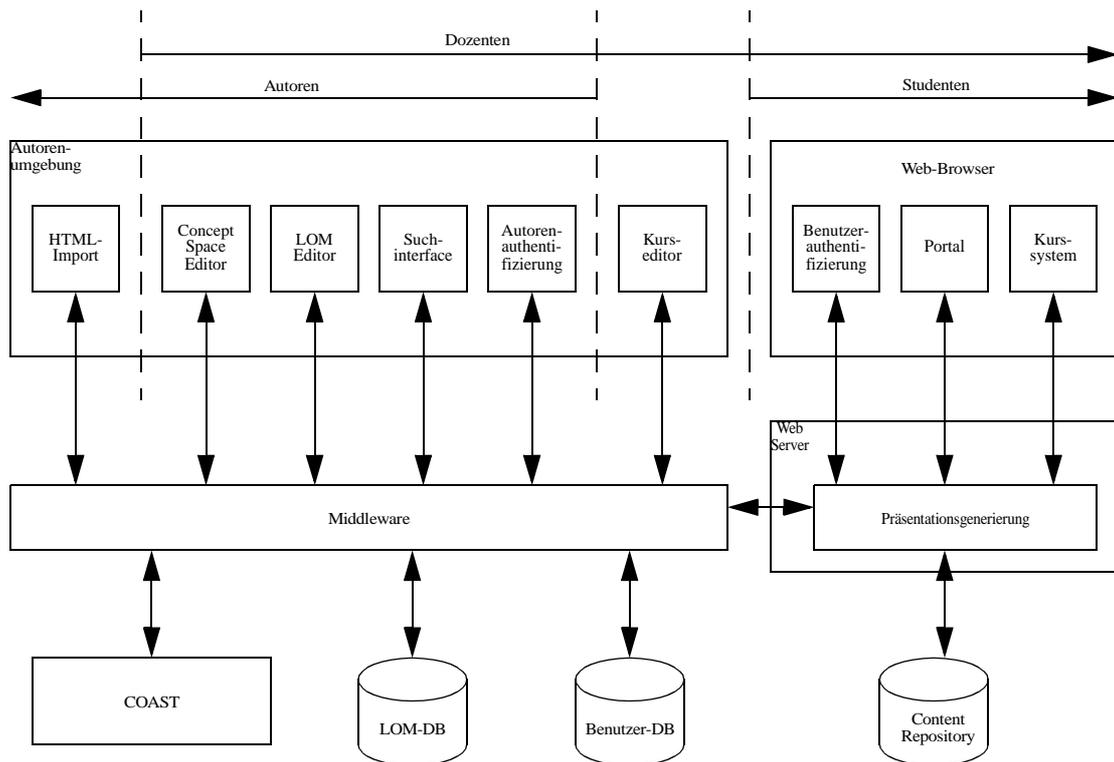
Die Wissensbasis von k-MED teilt sich in zwei Bereiche auf, dem ConceptSpace und dem MediaBrickSpace. In Abbildung n zeigt ein Ausschnitt diese beiden Bereiche. Im ConceptSpace werden medizinische Fachbegriffe gespeichert. Die semantischen Beziehungen dieser Begriffe werden unter Angabe der Art der Beziehung mit semantischen Relationen im ConceptSpace modelliert. Das semantische Netz wird durch Axiome, wie z.B. das automatische Eintragen von Umkehrrelationen oder die Transitivität der Ober- Unterbegriffsrelationen, während des Erstellungsprozesses konsistent gehalten. Die kollaborative Erstellung des semantischen Netzes erfolgt nach einer modifizierten Delphi-Methode [HJ02]. Ansätze zur semiautomatischen Erweiterung des semantischen Netzes finden sich bei [FHSS01]. Im MediaBrickSpace sind die physikalischen Daten der modularisierten Lernressourcen gespeichert. Der Name MediaBrickSpace stammt von den modularisierten Lernressourcen für die in [SSFS99] der Begriff MediaBrick geprägt wurde. Die MediaBricks stellen also die modularisierten Lernressourcen des k-MED Lernsystems dar. Damit den Benutzern von k-MED leistungsfähige Mechanismen zum Finden und Wiederverwenden der MediaBricks zur Verfügung stehen, werden diese mit Metadaten beschrieben. Hierfür verwenden wir Learning Objects Metadata (LOM), ein Metadatenschema des Learning Technology Standards

Committee (LTSC) der IEEE [LWG01]. LOM steht derzeit unmittelbar vor der Standardisierung durch die IEEE vor und ist besonders aufgrund pädagogischer Metadaten für die Attributierung von Lernressourcen geeignet. Die MediaBricks und die dazugehörigen LOM-Datensätze sind im Bild durch Rechtecke dargestellt. Im MediaBrickSpace werden ebenfalls Beziehungen inhaltlicher Art zwischen den MediaBricks festgehalten, die in [SSFS99] als rhetorisch-didaktische Relationen für MediaBricks eingeführt wurden. Diese werden durch die im LOM-Draft spezifizierte Möglichkeit der Erstellung von Relationen zwischen LOM-Datensätzen in den LOM-Datensätzen abgespeichert. Die Verbindung zwischen ConceptSpace und MediaBrickSpace wird durch Relationen zwischen Begriffen und MediaBricks hergestellt. Durch diese Relationen wird die thematische Verknüpfung der MediaBricks mit Begriffen des ConceptSpaces ausgedrückt.



ConceptSpace und MediaBrickSpace wachsen so zu einer mächtigen Wissensbasis zusammen, die durch die verwendeten Techniken dem Semantic Web [BHL01] sehr ähnlich ist, und sowohl Lehrende als auch Lernende dazu befähigt, effizient Lehr- und Lernmaterial zu finden. Hierfür werden jedoch eine Reihe neuer Hilfsmittel benötigt, da sich durch die Modularisierung Veränderungen im Autorenprozess ergeben.

Durch die Modularisierung der Lernressourcen und der konsequenten Nutzung von LOM wird die Möglichkeit adaptive Lernsysteme zu erstellen fest in dem Autorenprozess verankert [See01]. Aus diesem Grund werden für das k-MED Rahmensystem neuartige Werkzeuge benötigt, mit denen modularisierte Lernressourcen gespeichert, zu Kursen verknüpft und den Lernenden präsentiert werden können. Diese Werkzeuge sind in Abbildung n schematisch nach den Benutzergruppen [See01], die sie benutzen in der oberen Zeile angeordnet.



Zu den Werkzeugen der Lernenden gehört das Portal und das Kurssystem. Nach der erfolgreichen Benutzerauthentifizierung stehen den Lernenden Portal und Kurssystem als Informations- und Kommunikationssystem und die Präsentation der aus Modulen erstellten Kurse bereit. Diese Werkzeuge laufen im Web-Browser der Lernenden ab. Zu den Werkzeugen der Autoren gehören der ConceptSpace-Editor [SRHF00], LOM-Editor [Ste01] und das Such-Interface. Mit ConceptSpace-Editor und LOM-Editor erstellen und pflegen die Autoren die Wissensbasis, bestehend aus dem semantischen Netz und der LOM-Datenbank. Hierbei hilft ihnen das Such-Interface bei der Suche nach LOM-Datensätzen. Mitglieder der Gruppe der Lehrenden bedienen bei der Erstellung von Lehrangeboten sämtliche Werkzeuge. Hierzu benötigen sie zusätzlich den Kurseditor [HFM+01] zur Zusammenstellung von instruktionalen Einheiten aus modularisierten, multimedialen Lernressourcen. Dabei verwenden sie den ConceptSpace-Editor, den LOM-Editor und das Such-Interface lediglich zur Suche nach geeigneten Lernressourcen in der k-MED Wissensbasis. Auch für die Autoren und die Lehrenden ist ein Authentifizierungsmodul vorgesehen, damit die Zugriffsrechte auf die Wissensbasis geregelt werden können. Die Werkzeuge der Lernenden greifen auf die Präsentationsgenerierung zu, die in einem Web-Server läuft und direkten Zugriff auf das Content-Repository hat. ConceptSpace, LOM-Editor, Such-Interface, Kurseditor, Autorenauthentifizierung und Präsentationsgenerierung greifen gemeinsam auf eine Middleware-Komponente zu, die ihrerseits Zugriff auf das kollaborative Framework COAST [SSS01], welches den ConceptSpace verwaltet, LOM-Datenbank und Benutzerdatenbank hat.

4. Kursstrukturen mit LOM

In diesem Abschnitt soll die Abbildung von Kursstrukturen mit LOM gezeigt werden. Hierfür und für die in Abschnitt 7 folgende Implementierung eines Kurseditors werden zunächst alle wichtigen Datenfelder, die im LOM-Draft definiert sind, vorgestellt. Anschließend wird die

Abbildung der Kursstrukturen mit LOM beschrieben. Hierfür wird die Menge der möglichen Aggregationsstufen der einzelnen Bestandteile des Kurses, die im LOM-Draft definiert wurden, erweitert.

4.1 Benötigte Datenelemente

Für die im nächsten Abschnitt vorgestellte Realisierung der Strukturierung von modularisierten Lernressourcen eignen sich besonders Metadaten nach Learning Objects Metadata (LOM) des Learning Technologies Standards Committees der IEEE [LWG02]. LOM ist besonders für die Attributierung von Ressourcen ausgelegt, die für die Lehre vorgesehen sind, da auch pädagogische Details in den Beschreibungen der Lernressourcen berücksichtigt werden können. Im LOM-Draft werden Datenelemente und eine dazugehörige Strukturierung der Datenelemente beschrieben, mit denen Lernobjekte attribuiert werden können. Die Datenelemente eines LOM-Datensatzes sind in neun Kategorien eingeteilt. Auch innerhalb dieser Kategorien sind die Datenfelder oft noch tiefer strukturiert. Im Folgenden wird die Schreibweise *LOM.aaa.bbb[.bbb]* verwendet, die den Bezeichner der LOM-Kategorie *aaa* und des Datenelements *bbb* angibt.

In LOM-Datensätzen sind Datenfelder zur Speicherung von Identifikatoren für die weltweit eindeutige Identifikation von Medienbausteinen und LOM-Datensätzen definiert worden. Hierbei übernimmt *LOM.General.Identifier* die Identifikation des durch einen LOM-Datensatz beschriebenen Medienbausteins. Über *LOM.MetaMetadata.Identifier* können die einzelnen LOM-Datensätze weltweit eindeutig identifiziert werden. Zur Speicherung des Titels eines Medienbausteins ist das Datenfeld *LOM.General.Title* definiert worden. Dieses Datenfeld eignet sich besonders zur Anzeige des Titels im Kurseditor, ohne die Medienbausteine öffnen zu müssen. Angaben über die Granularität von Medienbausteinen werden in *LOM.General.AggregationLevel* gespeichert. Im LOM-Draft ist hierfür ein Vokabular mit den Elementen 1, 2, 3, 4 definiert worden, das Aggregationsstufen von der kleinsten hin zur größten Aggregationsstufen festlegt. Für die Speicherung des MIME-Types eines Medienbausteins ist das Feld *LOM.Technical.Format* definiert worden. Dieses Feld kann zur Assoziation von Icons verwendet werden, um dem Benutzer visuell kodierte Informationen zu dem MIME-Type eines Medienbausteins anzubieten.

Eine wichtige Rolle bei der Abbildung von Kursstrukturen auf LOM-Objekte kommt der Kategorie *LOM.Relation* zu. Sie ermöglicht das Formulieren von gerichteten Relationen ausgehend von dem LOM-Datensatz, in dem sie formuliert sind, zu einem Ziel-LOM-Datensatz. Im Datenfeld *LOM.Relation.Kind* wird der Typ einer Relation gespeichert. Im LOM-Draft sind hierfür die folgenden Typen definiert worden: *IsPartOf*, *HasPart*, *IsVersionOf*, *HasVersion*, *IsFormatOf*, *HasFormat*, *References*, *IsReferencedBy*, *IsBasedOn*, *IsBasisFor*, *Requires*, *IsRequiredBy*. Diese Relationstypen basieren auf Dublin Core [WKLW98] und liegen immer paarweise vor, damit durch zwei entgegengesetzte unidirektionale Relationen eine bidirektionale Relation zwischen zwei LOM-Datensätzen realisiert werden kann. Diese Wertemenge wird um die in [SSFS99] auf Medienbausteinen übertragenen rhetorisch-didaktischen Relationen erweitert. In *LOM.Relation.Resource-Identifier* wird der Identifikator des LOM-Datensatzes des Ziels der Relation gespeichert. Die Wertemenge dieses Feldes setzt sich aus den Identifikatoren aller LOM-Datensätze zusammen.

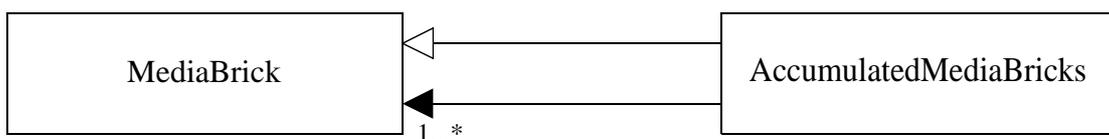
4.2 Kurse bilden mit LOM

In diesem Abschnitt soll eine Lösung vorgestellt werden, mit der modularisierte, multimediale Inhalte zu Kursen zusammengestellt werden können. Die dabei entstehenden Kurse sollen für das k-MED-Lernsystem Verwendung finden, sollten aber auch in anderen Bereichen Einsatz finden können. Beim Entwurf dieses Lösungsansatzes standen besonders die Verwendung

standardisierter Komponenten im Vordergrund, um die Interoperabilität zwischen verschiedenen Systemen zu gewährleisten. Aber auch die Trennung von Inhalt und Darstellung ist eine wichtige Eigenschaft, die der modellierte Kurs haben sollte. Mit dem Ergebnis, dass dadurch die Möglichkeit der Wiederverwendung erhöht wird, da die Inhalte in verschiedenen Kontexten mit anderen Layouts oder in anderen Formaten präsentiert werden können.

Die gespeicherten Kurse sollten eine baumförmige Struktur besitzen [gibt es hierfür eine schöne Referenz?!!!], da sie für die Lernenden eine bekannte Form der strukturierten Darstellung von Lernressourcen darstellen. Außerdem bietet die baumförmige Strukturierung der modularisierten Lernressourcen eine Möglichkeit der hierarchischen Strukturierung von Lernressourcen, ohne dabei auf eine mögliche Linearisierung der Lernressourcen verzichten zu müssen, damit die Lernenden auf einem linearen Lernpfad geführt werden können. Links, wie sie im Hypertext bekannt sind, können über Querverweise zwischen Teilen des Kursbaums realisiert werden, die je nach Darstellungsart als Hyperlinks oder als Querverweise in Form von Fußnoten dargestellt werden. Die Wahl der baumförmigen Struktur der Kurse bietet ebenfalls die Möglichkeit der Generierung einer baumförmigen Navigationsstruktur zur Navigation durch den Kurs im Web oder der Generierung eines Inhaltsverzeichnisses des Kurses für den Kurs in gedruckter Form.

Der hier vorgestellte Lösungsansatz ist besonders anschaulich an einem Beispiel aus der Objektorientierung zu erläutern. In dem Beispiel werden die Querverweise zunächst nicht betrachtet. Per Definition bestehe ein Kurs aus einer Menge von Objekten, die mittels Referenzen zu einer baumförmigen Struktur verknüpft werden. Für die Instantiierung dieser Objekte stehen zwei Klassen zu Verfügung. Die Blätter des Kursbaums sind Objekte der Klasse *MediaBrick*. Sie stellen die modularisierten multimedialen Lernressourcen dar. Die inneren Knoten des Kursbaums sind Objekte der Klasse *AccumulatedMediaBricks*. Wie der Name bereits verrät, dienen die Objekte der Klasse *AccumulatedMediaBricks*, dem Zusammenfassen von Objekten der Klasse *MediaBrick*. Sie bilden also die Wurzeln der Teilbäume im Kursbaum, indem sie Referenzen auf alle zusammengefassten Objekte der Klasse *MediaBrick* halten. Da Objekte der Klasse *AccumulatedMediaBricks* ebenfalls Lernressourcen mit einem Modulcharakter, der jedoch nicht atomar ist, darstellen, werden alle Eigenschaften der Klasse *MediaBrick* der Klasse *AccumulatedMediaBricks* vererbt. Daraus folgt, dass Objekte der Klasse *AccumulatedMediaBricks* auch Objekte derselben Klasse referenzieren können.



In Abbildung n wird dieser Sachverhalt mit Hilfe eines UML-Diagramms dargestellt. Es sind die beiden Klassen *MediaBrick* und *AccumulatedMediaBricks* zu sehen. Die Vererbung der Eigenschaften von *MediaBrick* auf *AccumulatedMediaBricks* wird mit dem oberen Pfeil ausgedrückt. Der untere Pfeil drückt aus, dass Objekte der Klasse *AccumulatedMediaBricks* Referenzen auf *MediaBrick* haben. Durch die Vererbung der Eigenschaften der Klasse *MediaBrick* auf die Klasse *AccumulatedMediaBricks*, sind auch Referenzen von Objekten der Klasse *AccumulatedMediaBricks* auf Objekte der Klasse *AccumulatedMediaBricks* möglich. Nach dieser Definition können Kursstrukturen mit den oben beschriebenen Eigenschaften erstellt werden.

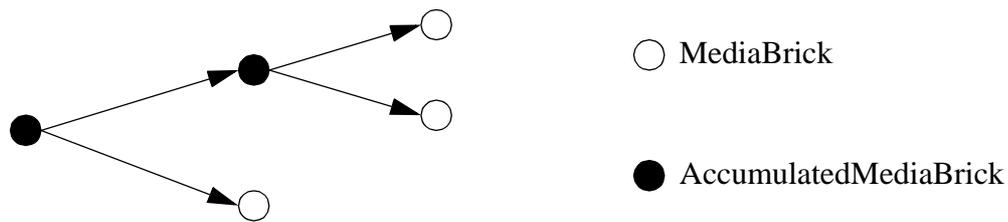


Abbildung n zeigt die Abbildung eines Kursbaums, dessen Grundlage das zuvor beschriebene Schema ist. Der dargestellte Kursbaum enthält keine Querreferenzen. Die Blätter des Kursbaums sind Objekte der Klasse *MediaBrick* und die inneren Knoten des Kursbaums sind Objekte der Klasse *AccumulatedMediaBricks*. Die Wurzel des Kursbaums zeigt auf ein Objekt der Klasse *AccumulatedMediaBricks* und ein Objekt der Klasse *MediaBrick*. Wie bereits oben erwähnt ist das möglich, da die Klasse *AccumulatedMediaBricks* von der Klasse *MediaBrick* erbt.

Für die technische Realisierung der Abbildung der Kursstrukturen auf Datenstrukturen werden Metadaten nach Learning Objects Metadata (LOM) des Learning Technologies Standards Committees der IEEE [LWG02] herangezogen. LOM eignet sich nicht nur aufgrund seiner weiten Verbreitung in Lernsystemen, in denen die Lernressourcen feingranular modularisiert vorliegen, sondern auch aufgrund der Tatsache, dass sich das oben beschriebene Schema zur Abbildung von Kursstrukturen mit LOM abbilden lässt. In Verbindung mit dem IMS Content Packaging des IMS Global Learning Consortiums wird eine Plattform- und Lernsystemübergreifende Lösung zur Speicherung und für den Austausch von Kursen gefunden, da die Kursstruktur in den Metadaten zusammen mit den physikalischen Daten der Lernressourcen in einer Datei gespeichert werden können.

Objekte der Klasse *MediaBrick* werden durch einen LOM-Datensatz und physikalische Daten einer modularisierten Lernressource repräsentiert. Gemeinsam stellen sie die Blätter im Kursbaum dar und enthalten die eigentlichen Inhalte des Kurses in Form von Texten, Bildern, Videos, Animationen und Simulationen. Abbildung n zeigt ein solches Objekt. Der LOM-Datensatz enthält eine genaue Beschreibung und weitere Metadaten der physikalischen Daten, die sich in der Regel nicht direkt aus dem Inhalt der Lernressource bestimmen lassen. Die Verbindung zwischen dem LOM-Datensatz und den tatsächlichen Daten der Lernressource wird über ein URI hergestellt, der Teil des LOM-Datensatzes ist. Hierfür wird das LOM-Datenelement *LOM.Technical.Location* benutzt, das die URIs der Lernressourcen aufnimmt. *LOM.Technical.Location* ist im LOM-Draft als eine Liste definiert worden, in der mehrere URI auf die selben physikalischen Daten an verschiedenen Stellen gespeichert werden können. Durch diese Redundanz können zeitweise nicht verfügbare Ressourcen von anderer Stelle geladen werden oder ein Load-Sharing durchgeführt werden. Durch die getrennten eindeutigen Identifikationsnummern im LOM-Datensatz wird auch das Anlegen mehrerer LOM-Datensätze für ein Ressource mit einem URI denkbar. Beide Erweiterungsmöglichkeiten/Sachverhalte sind nicht in Abbildung n dargestellt.



Objekte der Klasse *AccumulatedMediaBricks* werden ebenfalls durch LOM-Datensätze modelliert. Im Unterschied zu Objekten der Klasse *MediaBrick* enthalten sie jedoch keinen URI auf physikalische Daten. Der Inhalt der Objekte dieser Klasse setzt sich aus der Summe aller Inhalte der referenzierten Objekte der Klasse *MediaBrick* zusammen. Hierfür wird in jedem Objekt der Klasse *AccumulatedMediaBricks* eine Liste aufgebaut, in der die Objekte der Klasse *MediaBrick* gespeichert werden, aus denen sich dieses Objekt zusammensetzt. In den LOM-Datensätzen kann hierfür die Kategorie Relation verwendet werden. Nach dem LOM-Draft dürfen mehrere Instanzen der Kategorie Relation in einem LOM-Datensatz angelegt werden, wodurch die Verwaltung einer Liste von Referenzen in einem LOM-Datensatz ermöglicht wird. In den Datenfeldern LOM.Relation.Kind und LOM.Relation.Resource.Identifier werden für jede Referenz der Typ der Referenz und der Identifikator des LOM-Datensatzes gespeichert, der das Ziel der Referenz ist. Mit dieser Methode können unidirektionale Referenzen zwischen den LOM-Datensätzen aufgebaut werden. Zur Referenzierung aller Objekte der Klasse *MediaBrick*, aus denen sich ein Objekt der Klasse *AccumulatedMediaBricks* zusammensetzt, werden sie über Referenzen vom Typ HasPart mit ihren Identifikatoren referenziert. In entgegengesetzter Richtung wird in jedem Objekt der Klasse *MediaBrick*, welches durch ein Objekt der Klasse *AccumulatedMediaBricks* referenziert wird, eine Referenz vom Typ IsPartOf auf das referenzierende Objekt gespeichert. Mit Hilfe dieser Umkehrrelationen kann immer zurückverfolgt werden, an welchen Stellen ein Objekt der Klasse *MediaBrick* verwendet wird.

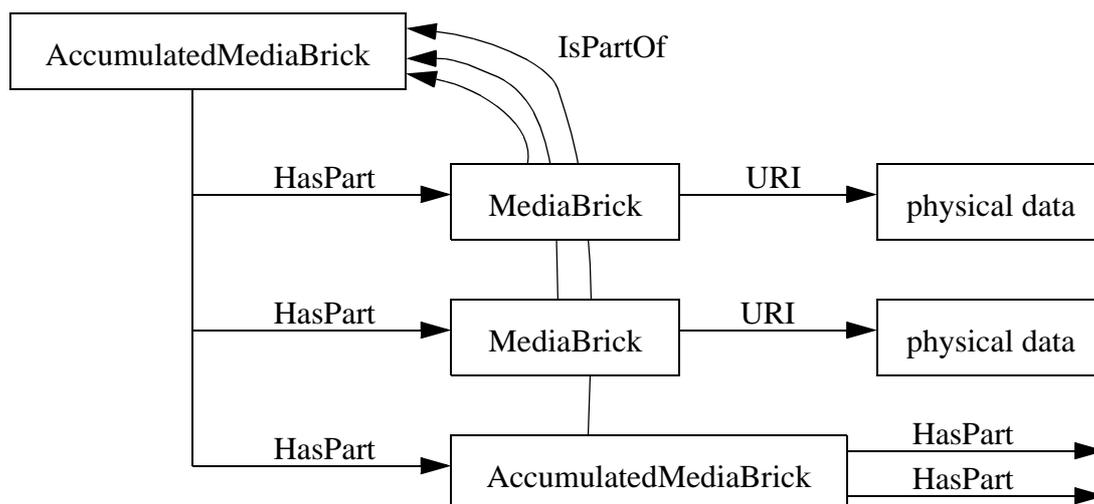


Abbildung n zeigt ein Objekt der Klasse *AccumulatedMediaBricks*, das sich aus zwei Objekten der Klasse *MediaBrick* und einem Objekt der Klasse *AccumulatedMediaBricks* zusammensetzt. Es referenziert deshalb die drei Objekte über Referenzen vom Typ *HasPart*. Jedes der drei referenzierten Objekte hat in entgegengesetzter Richtung eine Referenz auf das Objekt der Klasse *AccumulatedMediaBricks*. Die zwei referenzierten Objekte der Klasse *MediaBrick* verweisen zusätzlich jeweils mit einem *URI* auf ihre physikalischen Daten. Das referenzierte Objekt der Klasse *AccumulatedMediaBricks* setzt sich aus zwei weiteren Objekten zusammen für die in der Darstellung jeweils nur eine Referenz vom Typ *HasPart* zu sehen ist. Querverweisen zu anderen Stellen im Kurs oder zu anderen Ressourcen außerhalb eines Kurses werden ebenfalls durch das Erzeugen einer Referenz auf Objekte der Klasse *MediaBrick* realisiert. Hierfür wird an entsprechender Stelle in der Liste der Referenzen ein Objekt der Klasse *AccumulatedMediaBricks* in der Liste der Referenzen vom Typ *HasPart* eine Relation vom Typ *References* angelegt und auf den LOM-Identifikator der Zielressource verwiesen.

Auch in diesem Fall wird eine Umkehrrelation in dem referenzierten LOM-Datensatz eingetragen. Der Typ der Umkehrrelation ist *IsReferencedBy*.

Für die Auswertung der Kursstruktur stellt die Reihenfolge der Referenzen vom Typ *HasPart* und *References* eine wichtige Rolle dar. Durch die Reihenfolge dieser Referenzen wird die Abfolge der Referenzen auf die Objekte der Klasse *MediaBrick* und der Querverweise bestimmt. Aus diesem Grund muss die Reihenfolge entgegen den Empfehlungen des LOM-Drafts zumindest für die Relationen der Typen *HasPart* und *References* gewährleistet sein.

In der Praxis zeigt sich, dass für die Unterscheidung der Knoten des Kursbaums zwei Klassen nicht ausreichen. Im nächsten Abschnitt wird erläutert, welche Abstufungen hierfür notwendig sind und wie die Knoten gekennzeichnet werden.

4.3 Erweiterung der Aggregationsstufen

Die Menge der Knoten des Kursbaums lässt sich in die inneren Knoten und die Blattknoten teilen. Hierbei bildet die Menge der Blattknoten LOM-Datensätze, die Medienbausteine beschreiben, an die physikalische Daten geknüpft sind. Die Menge der inneren Knoten des Kursbaums enthält Medienbausteine, die nicht direkt physikalische Daten beschreiben. Sie sind je nach der Tiefe im Baum eine sukzessive Zusammenfassung von Medienbausteinen. Die Menge der inneren Knoten lässt sich weiterhin in die Menge von Knoten unterteilen, die Medienbausteine zusammenfassen, ohne dabei eine inhaltliche Untergliederung einzuführen, und der Menge von Knoten, die Medienbausteine zusammenfassen, mit dem Ziel eine inhaltliche Untergliederung zu schaffen.

Damit ein LOM-Datensatz einer der genannten Klassen zugeordnet werden kann, muss dieser gekennzeichnet werden. Für diesen Zweck wird im LOM-Draft das Datenfeld *LOM.General.AggregationLevel* verwendet. Die definierte Wertemenge für dieses Feld ist nicht für das oben erwähnte Klassifizierungsschema geeignet. Aus diesem Grund haben wir für das k-MED-Rahmensystem die folgende Wertemenge definiert:

Atom: LOM-Datensätze, die durch die Aggregationsstufe *Atom* gekennzeichnet sind attribuieren physikalische Daten, die über einen URI referenziert werden. Die physikalischen Daten bilden inhaltlich eine Sinneinheit und liegen in einem Format vor, das entweder direkt durch einen Web-Browser oder indirekt über Plug-Ins verarbeitet werden kann. Die physikalischen Daten, die durch diesen LOM-Datensatz beschrieben werden, liegen in einer Datei vor und sind in den meisten Fällen monomedial. Diese Art LOM-Datensätze haben keine Referenzen des Typs *HasPart* zu anderen Medienbausteinen.

Subatom: Dieser Typ von LOM-Datensätzen ist eingeführt worden, da die Bedingung, dass LOM-Datensätze des Typs *Atom* Ressourcen beschreiben, die inhaltlich eine Sinneinheit bilden und aus nur einer Datei bestehen, nicht in allen Fällen erfüllt ist. Mit LOM-Datensätzen des Typs *Subatom* werden demnach Ressourcen beschrieben, die aus einer Datei bestehen, jedoch keine inhaltliche Sinneinheit bilden. LOM-Datensätze dieses Typs werden für die Benutzer des k-MED-Rahmensystems transparent verwaltet, erfahren aber ansonsten die gleiche Behandlung wie LOM-Datensätze des Typs *Atom*.

CollectionOfSubatoms: Dieser Typ von LOM-Datensätzen ist eingeführt worden, um LOM-Datensätze vom Typ *Subatom* zu Sinneinheiten zusammenzufassen. Wobei durch die Zusammenfassung keine inhaltliche Untergliederung definiert ist. Durch diese LOM-Datensätze werden keine physikalischen Daten direkt beschrieben, sondern die Zusammenfassung von subatomaren Medienbausteinen über Relationen vom Typ *HasPart*. Die Menge der Typen von LOM-Datensätzen, die das Ziel einer Relation vom Typ *HasPart* sind,

ist auf *Subatom* begrenzt. Das Hinzufügen, Entfernen oder Vertauschen von Relationen zu subatomaren LOM-Datensätze ist bei bereits bestehenden LOM-Datensätzen dieses Typs nicht mehr erlaubt. Vom Kurseditor werden diese LOM-Datensätze wie LOM-Datensätze des Typs *Atom* behandelt.

CollectionOfAtoms: Dieser Typ ist zur Zusammenstellung größerer Sinneinheiten ohne inhaltliche Untergliederung eingeführt worden. LOM-Datensätze dieses Typs können über Relationen vom Typ *HasPart* LOM-Datensätze der Typen *Atom*, *CollectionOfSubatoms*, *CollectionOfAtoms* referenzieren. Das Hinzufügen, Entfernen oder Vertauschen von Relationen vom Typ *HasPart* ist erlaubt. Die Gruppierung der Medienbausteine in einer *CollectionOfAtoms* wird nur für Autoren und Dozenten sichtbar, für Lernende sind sie transparent.

Chapter: Dieser Typ ist zur Zusammenstellung größerer Sinneinheiten mit inhaltlicher Untergliederung eingeführt worden. LOM-Datensätze dieses Typs können über Relationen vom Typ *HasPart* LOM-Datensätze der Typen *Atom*, *CollectionOfSubatoms*, *CollectionOfAtoms* und *Chapter* referenzieren. Eine hierarchische Struktur wird durch die Möglichkeit der Referenzierung von *Chapters* in *Chapters* rekursiv definiert.

Course: Aus technischen Gründen wurde im k-MED-Rahmensystem dieser Typ von LOM-Datensätzen eingeführt. LOM-Datensätze dieses Typs unterscheiden sich nicht von LOM-Datensätzen des Typs *Chapter* außer dem technischen Unterschied, dass ein Kurs eine abgeschlossene Einheit bildet, an die Stylesheets zur Generierung von Präsentationen und Attribute geknüpft sind, über die festgestellt werden kann, ob der Kurs freigeschaltet ist.

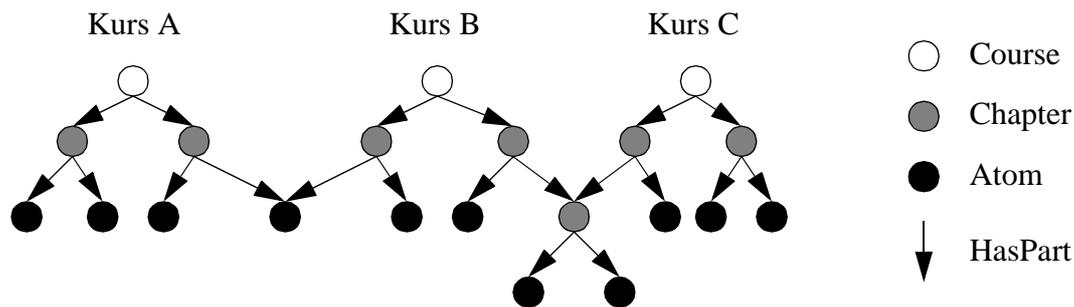
Der Mehrwert der konsequenten Nutzung von LOM zur Erstellung von Kursen liegt nicht nur in der besseren Wiederverwendbarkeit von Medienbausteinen, sondern auch in der Möglichkeit der Unterstützung von Adaptivität in den erstellten Kursen auf mehreren Ebenen. Das bezieht sich insbesondere auf das Entfernen, Tauschen oder Ergänzen von einzelnen Medienbausteinen innerhalb der Kurse.

5. Eigenschaften von Kursen mit LOM

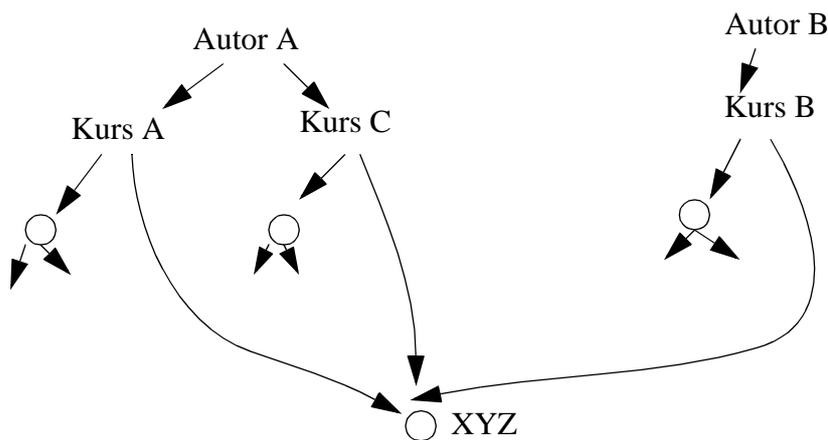
In diesem Abschnitt sollen die Eigenschaften von Kursen, die aus modularisierten, multimedialen Inhalten bestehen und mittels LOM-Datensätzen nach dem zuvor beschriebenen Verfahren strukturiert worden sind, hinsichtlich der Wiederverwendbarkeit und der Adaptierbarkeit untersucht werden.

5.1 Wiederverwendbarkeit

Der im Abschnitt n vorgestellte Lösungsansatz zur Strukturierung von Modulen mit Hilfe von LOM-Datensätzen stellt einen generischen Ansatz dar, der den Modulcharakter auf alle Hierarchieebenen eines Kurses überträgt. Hierdurch hält die Wiederverwendung von Komponenten eines Kurses auch in die Ebenen oberhalb der Module vom Typ *Atom* Einzug mit dem Ergebnis, dass ganze Kurse am Stück wiederverwendet werden können. Für die mehrfache Verwendung von einfachen Modulen, Kapiteln eines Kurses bis hin zu kompletten Kursen werden in der Wissensbasis keine Kopien der mehrfach verwendeten Ressourcen erstellt. Stattdessen werden Module, die in verschiedenen Kursen verwendet werden sollen, an den entsprechenden Stellen über eine Relation vom Typ *HasPart* referenziert.



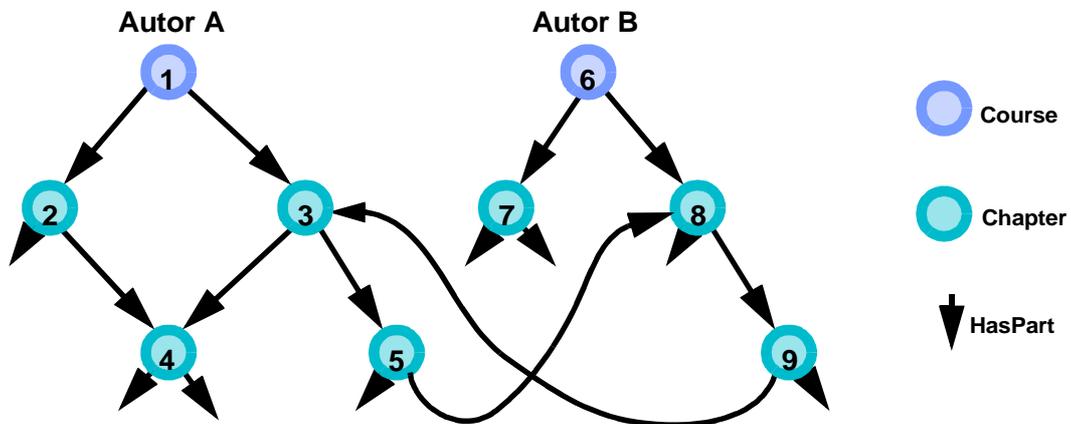
In Abbildung n ist die mehrfache Verwendung von zwei Modulen der Typen *Atom* und *Chapter* zwischen den Kursen A, B und C dargestellt. Wie bereits erwähnt, wird die mehrfache Verwendung des Moduls vom Typ *Atom* in den Kursen A und B durch die Referenzierung des Moduls aus den Kursen A und B über eine Relation vom Typ *HasPart* realisiert. Nach demselben Schema wird das Modul vom Typ *Chapter* in die Kurse B und C integriert. Hieraus ergibt sich der Vorteil, dass eine Aktualisierung der mehrfach verwendeten Module nur an einer Stelle durchgeführt werden muss. Es lassen sich aber auch Szenarien finden, in denen eine bedingungslose Aktualisierung der Module in allen Kursen, in denen das Modul Verwendung findet, nicht erwünscht ist. Gründe hierfür sind die Verwendung des aktualisierten Moduls durch verschiedene Autoren, von denen nicht jeder Autor der Änderung des Moduls zustimmt, oder, dass durch die Änderung das entsprechende Modul nicht mehr in den Kontext passt, in dem es verwendet werden soll. Ein Szenario, für das dieses zutrifft wird in Abbildung n beschrieben. An dem Szenario sind die Autoren A und B beteiligt. Nehmen wir an, Autor A beginnt mit der Erstellung des Kurses A, für den er das Modul XYZ erstellt. Nach der Fertigstellung des Kurses A beginnt Autor B mit der Erstellung des Kurses B, in dem er das Modul XYZ wiederverwendet, das er in der Wissensbasis gefunden hat. Zu diesem Zeitpunkt wird das Modul XYZ in zwei Kursen, die von zwei verschiedenen Autoren erstellt worden sind, verwendet. Nehmen wir jetzt an, dass Autor A sich entschließt einen weiteren Kurs zu erstellen. Bei der Erstellung des Kurses C entscheidet sich Autor A für die Wiederverwendung des Moduls XYZ. Bevor Autor A das Modul in Kurs C wiederverwendet möchte er es aktualisieren. Autor A möchte die Änderung des Moduls XYZ global für die gesamte Wissensbasis durchführen, da er die Änderung des Moduls XYZ auch für seinen Kurs A wirksam machen möchte. Zum Konflikt kann es bei diesem Szenario kommen, wenn Autor A nicht die Kenntnis besitzt, dass auch Autor B das Modul XYZ benutzt und die Änderung des Moduls ablehnt. Eine Lösung für dieses Problem ist die Einführung einer Versionierung für Module in der Wissensbasis. Mit Hilfe einer Versionierung für Module ist es Autor A möglich eine neue Version des Moduls XYZ zu erstellen, die er in Kurs A und C verwendet. Der Kurs B des Autors B bleibt hiervon zunächst unberührt, es wird weiterhin die erste Version des Moduls XYZ von Autor A verwendet. Über einen Mechanismus, der hier nicht weiter betrachtet werden soll, wird Autor B davon in Kenntnis gesetzt, dass für das von ihm verwendete Modul XYZ eine neue Version in der Wissensbasis zur Verfügung steht, die er bei Bedarf verwenden kann. Die Einführung der Versionierung bietet eine flexible Grundlage für Autor A zur Erstellung und Verwendung neuer Versionen seiner bereits erstellten Module und Autor B die Sicherheit, dass seine Kurse durch Veränderung wiederverwendeter Module unverändert bleiben, und die Option neue Versionen wiederverwendeter Module in seinem Kurs zu übernehmen.



Für die Veränderung eines Moduls, bei der neue Versionen entstehen, gibt es verschiedene Gründe, die in direktem Maße von der Aggregationsstufe und der Häufigkeit der Verwendung abhängen. Hierbei muss zwischen Modulen, die direkt durch physikalische Daten eines Textes, Bildes, Videos oder ähnlichem und Modulen, die eine Zusammenfassung von verschiedenen Modulen darstellen, unterschieden werden. Beispielsweise muss durch die Veränderung eines bereits in der Wissensbasis abgelegten Textes eine neue Version dieses Moduls von der Aggregationsstufe *Atom* erstellt werden. Neue Versionen für Module der Aggregationsstufen *CollectionOfAtoms*, *Chapter* und *Course* ergeben sich dann, wenn sich ein Modul, welches durch *HasPart* referenziert wird, verändert hat, die Reihenfolge der Module, die durch eine Relation vom Typ *HasPart* referenziert werden, verändert wird oder gar Module, die durch eine Relation vom Typ *HasPart* referenziert werden, gelöscht oder hinzugefügt werden. Die Veränderung eines Moduls wirkt sich also auch auf die Module in höherliegenden Hierarchien bis hin zur Kurswurzel aus, da durch die Veränderung eines Moduls des Kurses ein kompletter Teilbaum des Kurses verändert wird. Die Häufigkeit der Verwendung eines Moduls ist für die Erstellung neuer Versionen eines Moduls ebenfalls eine wichtige Größe, da nach dem zuvor beschriebenen Szenario die Erstellung einer neuen Version eines Moduls aus dem Grund der mehrfachen Verwendung des Moduls resultiert. Wird ein Modul der Wissensbasis nur in einem Kurs verwendet dann ist die Erstellung von neuen Versionen im Zuge einer Überarbeitung, die aus mehreren Schritten besteht, nicht sinnvoll. In diesem Fall kann die Erstellung neuer Versionen dieses Moduls auf Wunsch des Autoren erzeugt werden, um ein Backup des Moduls, an dem er gerade arbeitet, zu erstellen.

Durch die mehrfache Verwendung von Modulen, die in der Wissensbasis gespeichert sind, können Kursstrukturen durch Relationen vom Typ *HasPart* entstehen, die nicht mehr baumförmig sind und im Widerspruch zu den hierarchischen Relationen stehen. In Abbildung n sind zwei Kurse, in denen zwei dieser Fälle, die durch mehrfache Verwendung von Modulen der Wissensbasis entstehen können, die baumförmige Strukturierung von Modulen stören, dargestellt. Im Kurs von Autor A wird das Modul 4 mehrfach verwendet, es wird von Modul 2 und 3 durch *HasPart* referenziert. Durch die Verwendung von Modul 4 in Modul 2 und 3 liegt bereits keine reine Baumstruktur mehr vor, da Modul vier jetzt mehr als einen Vater in der Kursstruktur hat. Das kann den Lernenden beim Betrachten des Kurses verwirren. Aus diesem Grund müssen solche Strukturen für die Lernenden aufgehoben werden. Aus technischer Sicht sind Zyklen, die aus *HasPart* Relationen bestehen für die Erzeugung von Präsentationen von Kursen kritisch. Ein Beispiel für die Entstehung solcher Zyklen soll durch ein Szenario, das ebenfalls in Abbildung n gezeigt wird, beschrieben werden. Nehmen wir an Autor A erstellt einen Kurs unter Verwendung des fremden Moduls 8, welches Teil eines Kurses von Autor B

ist. Autor A referenziert Modul 8 in Modul 5. Nachdem Autor A die Arbeiten an seinem Kurs eingestellt hat, erweitert Autor B seinen Kurs. Bei der Suche nach neuem Material für seinen Kurs findet er Modul 3, das Teil des Kurses von Autor A ist. Er referenziert Modul 3 über eine Relation vom Typ HasPart in Modul 9 und schließt hiermit einen Zyklus über HasPart Relationen.



Bei der Erstellung von Präsentationen eines Kurses sind solche Zyklen insofern kritisch, da durch sie endlose Dokumente entstehen können, falls sie nicht entdeckt werden. Es müssen also Wege gefunden werden, die mehrfache Verwendung von Modulen und Zyklen durch Relationen vom Typ HasPart aufzulösen.

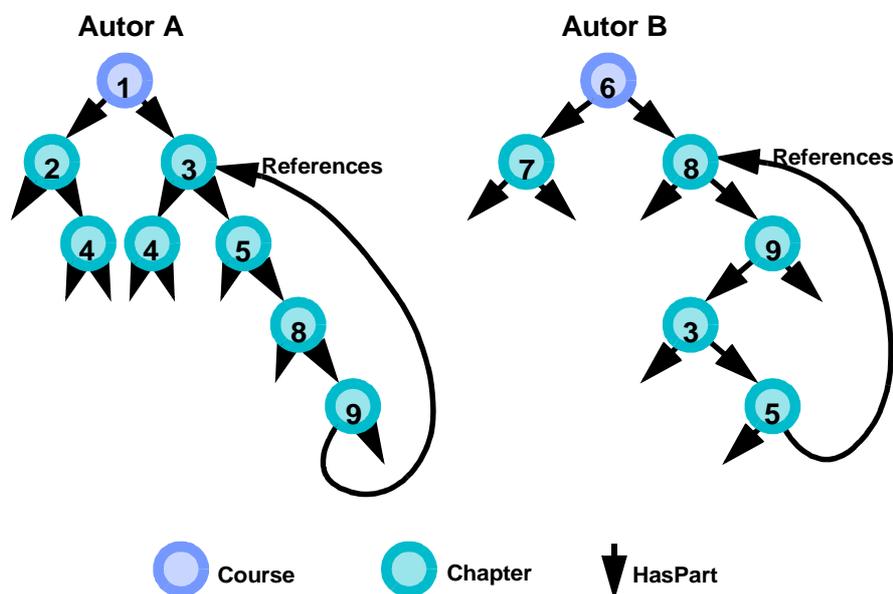


Abbildung n zeigt die Kursstrukturen der Kurse A und B nach Auflösung der mehrfach verwendeten Module und der Zyklen. Beide Kurse bestehen jetzt wieder aus einer Baumstruktur, die mit Hilfe der Relation vom Typ HasPart aufgebaut sind. Die Darstellungen der Kurse in Abbildung n spiegeln nicht den wirklichen Datenbestand der Wissensbasis wieder, sondern stellen die Interpretation durch einen Algorithmus dar, mit dessen Hilfe eine baumförmige Struktur für jeden der Kurse gebildet werden kann. Hierbei werden mehrfach verwendete Module ihrem Kontext entsprechend mehrfach in der Kursstruktur aufgenommen. Dabei handelt es sich immer um die selbe Instanz der Module. Zyklen, die aus Relationen vom

Typ HasPart aufgespannt werden, bricht der Algorithmus auf und ersetzt die Relation vom Typ HasPart, die den Zyklus in einem Kursbaum schließen würde durch eine Relation vom Typ *References*. Nach der Interpretation der Kurse der Wissensbasis nach dem vorgestellten Schema können Präsentationen der Kurse für die Lernenden erstellt werden.

5.2 Adaptierbarkeit und Adaptivität von Kursen mit LOM

Die Speicherung der Kurse mittels einer Struktur bestehend aus LOM-Datensätzen erfolgt ohne Angabe der Optik und des Layouts der Präsentation dieser Kurse. Der Kursautor legt lediglich die Reihenfolge und Struktur der Medienbausteine fest. Daraus folgt, dass für die Generierung von Präsentationen dieser Kurse Vorlagen benötigt werden, nach denen die Kurse mit einem Layout versehen werden können. Diese Vorlagen bestehen aus mikro- und makrotypographischen Vorgaben, die teilweise durch Designer bestimmt oder automatisch an die Lernenden adaptiert werden können. Bei den Teilen der Vorlagen, die durch Designer festgelegt werden, ist die Auswahl aus einer Menge durch den Benutzer denkbar, so dass die Kurse durch den Lernenden adaptierbar sind.

Bei der automatischen Adaption der Kurse an die Lernenden sind besonders die rhetorisch-didaktischen Relationen [SSFS99] zwischen den Medienbausteinen hilfreich. Sie drücken inhaltliche Beziehungen zwischen Medienbausteinen aus und werden ebenfalls wie die Verweise, mit denen Kursstrukturen aufgebaut werden können in der Kategorie Relation in den LOM-Datensätzen abgelegt. In [SSFS99] sind hierfür die Relationen *example*, *illustrates*, *instance*, *restricts*, *amplifies*, *continues*, *deepens*, *opposite* und *alternative* definiert worden. Die inhaltlichen Beziehungen, die mit Hilfe der rhetorisch-didaktischen Relationen ausgedrückt werden können, sind nicht an Kursgrenzen gebunden. Wie in Abbildung n zu sehen ist, können solche Beziehungen auch zwischen MediaBricks, die Teil eines Kurses sind, und MediaBricks, die außerhalb des Kurses oder innerhalb eines anderen Kurses sind, bestehen. Dabei hängt die mögliche Verwendung der rhetorisch-didaktischen Relationen in der Regel davon ab, ob die durch die Relationen referenzierten Medienbausteine Teil des Kurses sind oder nicht. Im Einzelfall ist die mögliche Verwendung der rhetorisch-didaktischen Relation zwischen Medienbausteinen eines Kurses von der örtliche Beziehung der Medienbausteine innerhalb der Kursstruktur oder des Typs der Medienbausteine abhängig.

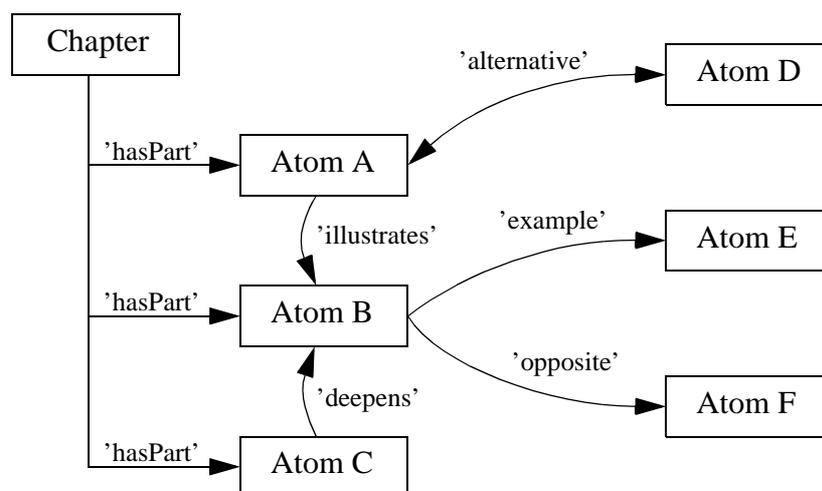


Abbildung 1: Rhetorisch-didaktische Relationen zwischen Medienbausteinen

Abbildung n zeigt einen MediaBrick von Typ *Chapter*, der sich aus drei weiteren MediaBricks vom Typ *Atom* zusammensetzt. Sie werden über Relationen vom Typ *HasPart* als Teile des Kapitels referenziert. Zusätzlich sind in Abbildung n drei weitere MediaBricks vom Typ *Atom* dargestellt, die sich nicht innerhalb des Kapitels befinden. Zwischen diesen sechs MediaBricks existieren rhetorisch-didaktische Relationen, die in der Abbildung in geschwungenen Pfeilen dargestellt sind. In der Abbildung wurden rhetorisch-didaktische Relationen der Typen *illustrates*, *deepens*, *alternative*, *example* und *opposite* verwendet.

Im Folgenden soll jeweils nach einer kurzen Erläuterung der Bedeutung der Relationen ihre mögliche Verwendung diskutiert werden.

Mit der rhetorisch-didaktischen Relation *alternative* können MediaBricks verbunden werden, die inhaltlich äquivalent sind. Bei der Generierung von Präsentationen ist diese Beziehung zwischen MediaBricks mit verschiedenen MIME-Typ von großer Bedeutung. Durch sie können alternative Module mit äquivalentem Inhalt gefunden werden, wenn bei der Generierung von Präsentationen keine kontinuierlichen MediaBricks wie Videos in Betracht kommen, weil sie nicht in das Ausgabeformat eingebettet werden können oder die Bandbreite des Internet-Anschlusses des Lernenden nicht ausreicht. Das setzt jedoch voraus, dass der MediaBrick, der stattdessen in den Kurs eingebettet wird, nicht in unmittelbarer Nähe im Kurs enthalten ist. Nehmen wir an, MediaBrick A in Abbildung n sei ein Video und MediaBrick D sei ein Text. Sie sind inhaltlich äquivalent, haben aber verschiedene MIME-Types. MediaBrick A könnte nun für Lernende mit schlechter Anbindung an das Internet durch Medienbaustein D ausgetauscht werden. Rhetorisch-didaktische Relationen vom Typ *alternative* können auch genutzt werden, um den Lernenden zusätzliche MediaBricks anzubieten. In Fällen, in denen ein Text statt eines vom Inhalt äquivalenten Videos in den Kurs eingebunden ist, könnte das Video über einen Link referenziert werden, und so dem Lernenden zusätzlich angeboten werden. Das ist natürlich nur dann sinnvoll, wenn das Video nicht in unmittelbarer Nähe zum Text Bestandteil des Kurses ist.

Mit Hilfe der rhetorisch-didaktischen Relation *illustrates* kann ausgedrückt werden, dass ein MediaBrick eine Illustration eines anderen ist. In Abbildung n ist eine solche Beziehung zwischen den MediaBricks A und B durch den Pfeil von MediaBrick A nach MediaBrick B dargestellt. Durch diese Beziehung wird ausgedrückt, dass MediaBrick A MediaBrick B illustriert. Das könnte der Fall sein, wenn es sich bei MediaBrick A um ein Bild handelt und bei MediaBrick B um einen Text. Diese Beziehung kann dann direkt durch das makrotypographische Layout ausgedrückt werden. Je nach den Vorgaben eines Designers könnte der Text um das Bild herumfließen oder die Beziehung der beiden MediaBricks an anderer Weise optisch verdeutlicht werden. Hierfür müssten sie in der Reihenfolge der MediaBricks im Kurs direkt aufeinander folgen, wie es in Abbildung n der Fall ist. Andernfalls könnte der illustrierende MediaBrick über einen Link als zusätzliches Modul angeboten werden, wenn er nicht Teil des Kurses ist, in dem der MediaBrick eingebettet ist, das er illustriert.

Die rhetorisch-didaktischen Relationen *example* und *deepens* werden eingesetzt, um auf Beispiele oder vertiefende Module zu verweisen. Diese Beziehungen unter den MediaBricks können von großer Bedeutung für die Lernenden sein. Folgen beide MediaBricks zwischen denen die Beziehung *example* oder *deepens* besteht unmittelbar aufeinander, so kann die rhetorisch-didaktische Relation dazu verwendet werden den Kontext, in dem die Module zueinander stehen dem Lernenden anzuzeigen. Das kann beispielsweise mit Piktogrammen erfolgen, die in den Satzspiegel neben den Modulen gesetzt werden. Für die MediaBricks B und C in Abbildung n könnte eine solche Regel greifen. Die rhetorisch-didaktische Relation zwischen den beiden Modulen drückt aus, dass MediaBrick C MediaBrick B vertieft. Dieser Sachverhalt kann den Lernenden durch ein Piktogramm neben MediaBrick C angedeutet werden, das auf Vertiefung hinweist. Für den Fall, dass sich die MediaBricks, zwischen denen

sich die rhetorisch-didaktischen Relationen *example* und *deepens* nicht im selben Kurs befinden, können Beispiele und vertiefende Module zusätzlich über Links angeboten werden. Ähnlich, wie die Beziehungen *example* und *deepens* genutzt werden können, lässt sich auch die rhetorisch-didaktische Relation *opposite* verwenden. Mit der rhetorisch-didaktischen Relation *opposite* können zwei MediaBricks verbunden werden, die eine unterschiedliche Aussage zu demselben Thema haben. Sie kann genutzt werden, um ein MediaBrick mit entgegengesetzter Aussage im Kontext zu markieren oder um ein zusätzliches Modul mit einer anderen Aussage über Link zur Verfügung zu stellen.

Die bis hier diskutierten Anwendungsfälle der rhetorisch-didaktischen Relationen lassen sich in den meisten Fällen statisch in Vorlagen zur Generierung von Präsentationen integrieren. Sie lassen sich jedoch auch in adaptiven Systemen einsetzen. Hierfür ist die Voraussetzung das Wissen über den Wissensstand des Lernenden. Dieses kann eingesetzt werden, um das Verhalten des Systems in Abhängigkeit des Wissensstands des Lernenden zu steuern. Zum Beispiel könnte ein adaptives System in Abhängigkeit des Wissensstands des Lernenden Module zum Kurs hinzufügen oder Module aus dem Kurs entfernen. Ein denkbare Szenario hierfür wäre das Wiederholen von bereits gelerntem Inhalt. Hierbei könnte auf die Einbindung von Beispielen verzichtet werden. Stattdessen könnten die Beispiele über Links verfügbar gemacht werden, um bei Bedarf trotzdem für den Lernenden erreichbar zu sein. Stattdessen könnten vertiefende Module in den Kurs integriert werden, die zuvor nur über Links als zusätzliche Module erreichbar waren. Im Gegensatz dazu könnte der Kurs gerade mit Beispielen angereichert werden und vertiefende Module entfernt werden, wenn ein Lernender den Kurs das erste Mal durcharbeitet.

6. Präsentation von Kursen mit LOM

In diesem Abschnitt soll gezeigt werden, wie die aus modularisierten Medienbausteinen zusammengestellten Kurse präsentiert werden können. Hierfür bieten sich zwei sehr verschiedene Medien an, die jeweils ihre Vorteile auf dem Gebiet der Lehre ausspielen: Die Präsentation von Kursen im World-Wide-Web und die Präsentation auf Papier.

6.1 Präsentation im World-Wide-Web

Zu den Vorteilen der Veranstaltung von Lehre über das Internet gehört nicht nur die Unabhängigkeit der Lernenden in Bezug auf Zeit und Ort. Sie liegt vor allem darin begründet, dass über das Web kontinuierliche und interaktive Lernressourcen genutzt werden können. Aus diesem Grund bietet sich die Generierung von Präsentationen der Kurse zur Präsentation im Web geradezu an. Aber auch die Möglichkeit der adaptiven Gestaltung der Kurse spricht für diesen Weg, da im Gegensatz zu Büchern jeder Kurs persönlich an die Lernenden angepasst werden kann. Außerdem können durch permanente Aktualisierung immer aktuelle Kurse durch die Lernenden abgerufen werden. Das alles ist möglich ohne die Installation von spezieller Software, da zum Anzeigen der Kurse nur ein Web-Browser benötigt wird.

Für die Abbildung des Kurses im Web werden die Hierarchien der Kapitel des Kurses auf Hierarchien von kurzen Web-Seiten abgebildet. Statt wie in einem Buch Unterkapitel in einem Kapitel vollständig zu integrieren, werden diese auf eigenen Web-Seiten untergebracht. Die Unterkapitel eines Kapitels sind innerhalb der Web-Seiten des Kapitels über Hyperlinks zu erreichen. Sowie auch die Web-Seiten des Kapitels über eingebettete Hyperlinks in den Web-Seiten der Unterkapitel zu erreichen sind. Hiermit wird dem Lernenden eine bereits wohlbekannt Navigationsmöglichkeit geboten, mit der er bequem und schnell durch den Kursbaum in Form von Web-Seiten navigieren können, um die gewünschten Informationen zu finden. Durch diese restriktive Navigation, die es nur erlaubt in Richtung der Wurzel oder verzweigend zu den Blättern hin zu navigieren ist das Lost-In-Hyper-Space-Syndrom nicht zu

erwarten. Lediglich Referenzen zu anderen Kursteile, die durch Hyperlinks realisiert werden bieten noch Potential für Verirrungen des Lernenden im Kursbaum.

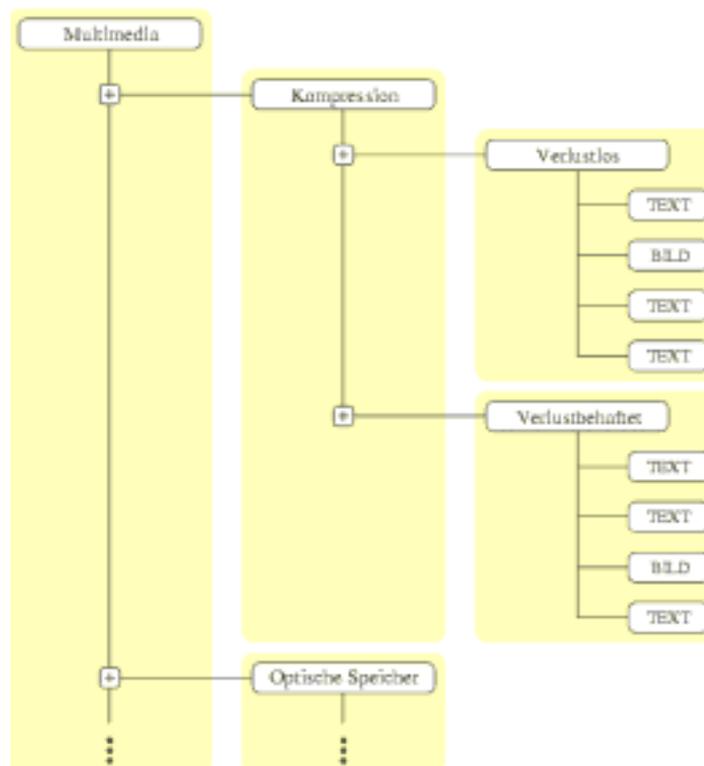
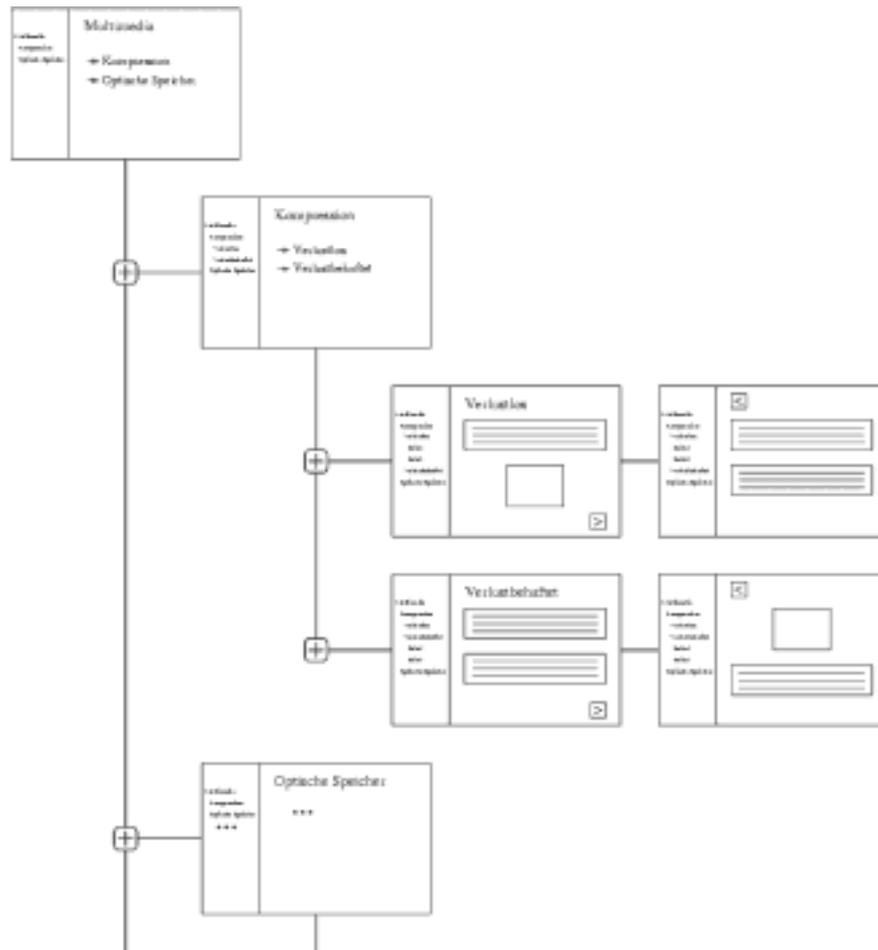


Abbildung n zeigt, wie sich die Struktur des Kurses auf HTML-Seiten abbilden lässt. Erreicht wird dies, indem von einzelnen Teilen des Kurses, den Kapiteln, einzelne HTML-Seite erzeugt werden. Dazu wird die Kursstruktur in Breitensuche durchgelaufen und stückweise in HTML-Seiten verpackt. In der Abbildung sind die einzelnen Teilkurse mit grauen Blöcken hinterlegt. Die Wurzel des Kursbaums mit dem Titel *Multimedia* beinhaltet zwei Verweise auf die Kapitel *Kompression* und *Optische Speicher*. Das Kapitel *Kompression* besteht lediglich aus zwei Verweisen zu den Kapiteln *Verlustlos* und *Verlustbehaftet*. Erst die Kapitel *Verlustlos* und *Verlustbehaftet* enthalten Medienbausteine, die nicht aus mehreren Medienbausteinen zusammengesetzt sind. In diesen beiden Kapiteln wird das eigentliche Wissen vermittelt. Zu allen diesen Kapiteln müssen HTML-Seiten erzeugt werden, von denen die, die in der Hierarchie höher liegen, lediglich zum schnellen Auffinden des gewünschten Lehrstoffes durch Navigation im Kursbaum dienen.



Selbst diese Struktur kann für den Lernenden schon nicht mehr überschaubar sein, wenn sie in Tiefe und Breite wächst. Zur Hilfe kann dem Lernenden die Struktur des Kurses wie ein Inhaltsverzeichnis des Kurses angezeigt werden. In diesem kann die aktuelle Position innerhalb des Kurses angezeigt werden. Zusätzlich bringt das Inhaltsverzeichnis die Möglichkeit, einen Überblick über den Kurs zu gewinnen. Durch Anklicken eines Eintrages des Inhaltsverzeichnisses sollte die dazugehörige HTML-Seite geladen werden und der aktuelle Eintrag im Verzeichnis farblich markiert werden. Abbildung n zeigt ein Beispiel nachdem die zu den einzelnen Lektionen gehörigen HTML-Seiten erzeugt werden könnten. Auf der linken Seite jeder HTML-Seite wird das Inhaltsverzeichnis mit vollständig expandiertem Pfad zur aktuellen Seite angezeigt. Der Rest der HTML-Seite wird für die Präsentation des Inhalts der Lektionen verwendet.

6.2 Präsentation auf Papier

Allen Vorteilen der Präsentation der Lernressourcen über das Web stehen jedoch auch einige Nachteile gegenüber. Für das Durcharbeiten der Lektionen muss ein gewisser Rahmen an technischen Mitteln vorhanden sein. Es wird ein Computer benötigt, der multimedialfähig sein muss, und ein Internet-Anschluss, der über die notwendige Bandbreite zur Anzeige von kontinuierlichen Medien wie Video verfügt. Für alle Lebenslagen, in denen dieser technische Rahmen nicht gegeben ist, bietet sich die Präsentation der Kurse auf Papier an. Zudem verfügt ein Ausdruck auf Papier über eine deutlich höhere Schriftqualität als sie Monitore von mobilen Endgeräten bieten können. Der Ausdruck eines Kurses kann aber auch aus Gründen der

Archivierung des Kurses erfolgen. Dabei muss jedoch auf kontinuierliche und interaktive MediaBricks verzichtet werden, da auf Papier nur statische Module ausgegeben werden können. Hierfür müssen kontinuierliche und interaktive MediaBricks durch äquivalente MediaBricks ersetzt werden. Diese können wie bereits in Abschnitt n gezeigt wurde über rhetorisch-didaktische Relationen vom Typ *alternative* gefunden werden.

Für Präsentationen von Kursen, die ausgedruckt werden sollen, bietet sich das Ausgabeformat PDF an. Dieses Format hat sich in den letzten Jahren im Internet für den Bezug von Dokumenten, die direkt ausgedruckt oder am Monitor betrachtet werden sollen, durchgesetzt. Die hohe Verfügbarkeit von Software zum Ausdrucken und zur Betrachtung dieser Dokumente und die Verfügbarkeit von Programmbibliotheken [FOP] zur Erzeugung von PDF-Dokumenten machen dieses Format als Grundlage für Präsentationen sehr attraktiv. Die Tatsache, dass das PDF-Format auch auf mobilen Endgeräten unterstützt wird, sprechen ebenfalls für die Verwendung dieses Formats.

Für die Präsentation auf Papier muss die Kursstruktur zunächst linearisiert werden, das heißt die Bausteine, der zu einer Baumstruktur verbundenen Lektionen, müssen in die richtige Reihenfolge gebracht werden, mit der sie dann anschließend ausgegeben werden können. Zu diesem Zweck wird die Kursstruktur in Tiefensuche durchgelaufen. Alle auf diesem Weg befindlichen Medienbausteine werden in Pre-Order ausgegeben.

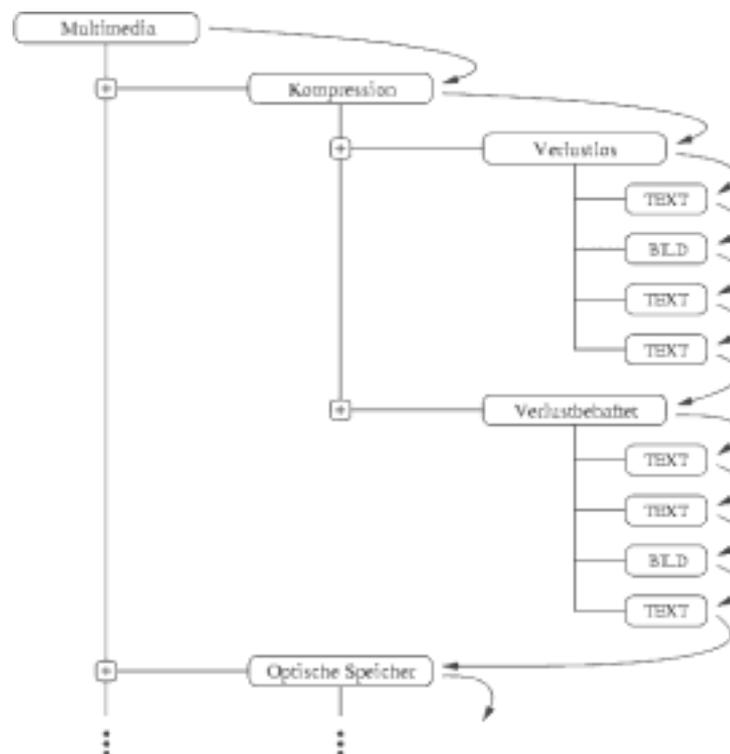


Abbildung n zeigt die Reihenfolge, nach welcher die einzelnen Medienbausteine der Kursstruktur ausgegeben werden. Das entspricht der Reihenfolge, in der auch Bücher aufgebaut sind. Probleme treten jedoch auf, wenn ein Kapitel bunt gemischt aus Unterkapiteln und einfachen Medienbausteinen zusammengesetzt ist. Wenn beispielsweise in der Kapitel *Kompression* ein Medienbaustein ist, z.B. Text zwischen den Unterkapiteln *Verlustlos* und *Verlustbehaftet*, was nach der Definition von Abschnitt n durchaus zulässig ist, dann muss eine Möglichkeit gefunden werden, mit der dem Leser verdeutlicht wird, dass der Text nicht mehr zum Unterkapitel *Verlustlos* gehört, sondern Teil des Kapitels *Kompression* ist. Derartige Zweideutigkeiten können bei der Präsentation mit HTML nach der zuvor beschriebenen

Methode nicht entstehen und können in Präsentationen auf Papier vermieden werden, wenn dem Leser mit einem Zeichen angezeigt wird, dass das aktuelle Unterkapitel zu Ende ist. Medienbausteine können nach einem solchen Zeichen dann vom Lernenden der übergeordneten Lektion zugeordnet werden.

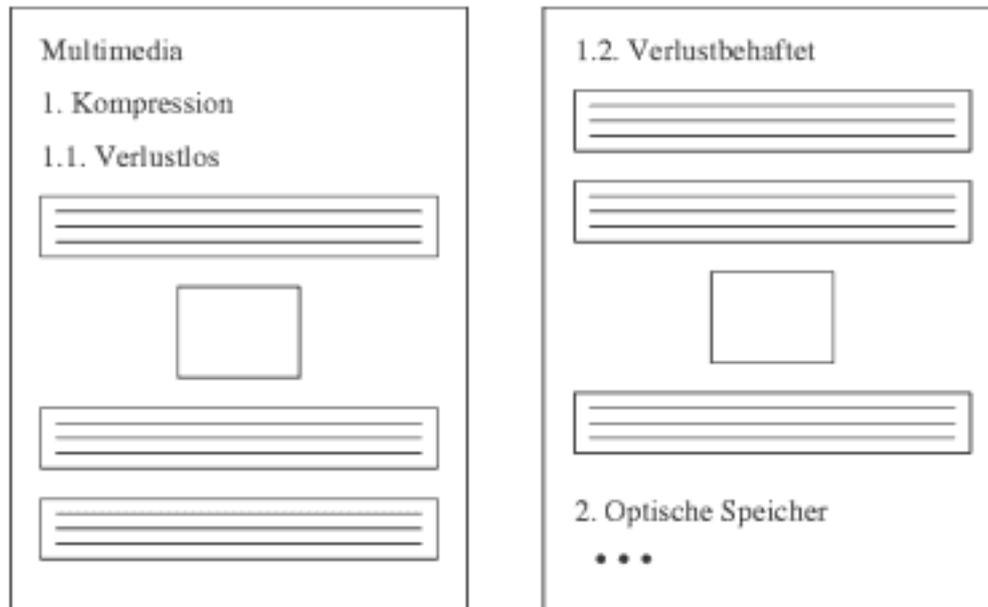


Abbildung n zeigt ein Muster, nach dem ein Kurs auf Papier präsentiert werden könnte. Es wäre auch denkbar, den Titel des Kurses mit weiteren Metadaten des Kurses wie Autor und Datum der Erstellung des Kurses zu ergänzen. Zusammen könnten diese Informationen über den Kurs mit einer Zusammenfassung des Kurses ein Deckblatt ergeben.

7. Der Kurseditor

In diesem Abschnitt wird ein Kurseditor vorgestellt, der im Vergleich zu anderen Editoren zur Erstellung von Kursen aus modularisierten Lernressourcen, die erzeugten Kurse durch Aufbau einer Objekthierarchie mit gerichteten Relationen zwischen LOM-Datensätzen nach dem in Abschnitt n gezeigten Schema speichert. Hierfür erstellt der Kurseditor transparent für Autoren und Dozenten die Relationen zwischen wiederverwendeten und neuen Medienbausteinen, die durch Kombination vorhandener Medienbausteine entstehen. Der entstandene LOM-basierte Kurseditor für k-MED verbirgt dieses alles unter einer graphischen Oberfläche, deren Benutzung sich intuitiv erschließt.

7.1 Suchschnittstelle

Zur Unterstützung des Anwenders bei der Formulierung von Suchanfragen und bei der Darstellung der Suchergebnisse wird eine Suchschnittstelle angeboten. Die Bedienung der Suchschnittstelle erfolgt in den Arbeitsschritten der Formulierung der Suchanfrage, des Startens der Suche und der Anzeige der Suchergebnisse. Die aus der Datenbank selektierten Suchergebnisse werden graphisch separiert von der aktuellen Darstellung des in der Editierung befindlichen Kurses in einer Baumstruktur dargestellt und bilden zusammen mit den bis dahin editierten Kursteilen das jeweils aktuelle Repositorium für die Bearbeitung des aktuellen Kurses im Kurseditor. Durch wiederholte Suchanfragen kann daher das aktuell angezeigte Repositorium verändert und entsprechend Benutzerwunsch angepasst werden. Dieses

Repository dient dem Dozenten und Autoren als Quelle von MediaBricks für die Zusammenstellung und Erweiterung von Kursen.

Die Baumstruktur der Suchergebnisse ist entsprechend der Medientypen der dargestellten Medienbausteine strukturiert. Der MIME-Type des Medienbausteins wird durch ein vorangestelltes Icon vor dem Titel des Medienbausteins angezeigt. Auf Wunsch der Benutzer kann die Anzeige des zum Medienbaustein gehörigen LOM-Datensatzes durch Aufruf des Metadateneditors im separaten Fenster erfolgen.

7.2 Darstellung des Kurses

In der Darstellung der Kursstruktur der Medienbausteine des zu editierenden Kurses als Baum muss beachtet werden, dass sich die Reihenfolge der Medienbausteine an der linearisierten Kursstruktur orientiert. Die Ordnung in vertikaler Richtung der Darstellung im Kurseditor entspricht demnach der geplanten linearen Bearbeitungsreihenfolge der Medienbausteine des Kurses zum Präsentationszeitpunkt ohne Traversierung von angebotenen Referenzen.

Medienbausteine, die Gruppierungselemente bilden und keine inhaltlichen Mediendaten enthalten, sind innere Knoten des Kursbaumes oberhalb der Blätter und werden im gleichen Baum mit Objekten, die inhaltliche Mediendaten enthalten und als Blätter des Kursbaumes angezeigt werden, gemeinsam dargestellt. Ausnahme bildet hierbei die Anzeige von *CollectionOfSubatoms*, wenn die Darstellung der *Subatoms* unterdrückt wird.

Die Unterscheidung des Typs der Medienbausteine erfolgt wie in der Anzeige der Suchergebnisse durch ein dem Titel des Medienbausteins vorangestelltes Icon.

7.3 Bearbeitung durch Drag-and-Drop

Neben den Bearbeitungsformen des Markierens mit anschließendem Kopieren und Einfügen durch Tastaturbedienung, menügesteuerter Bedienung oder Knopfleistenbedienbarkeit ist in der Implementierung besonders die Bearbeitung der Kursstruktur durch mausgestütztes Drag-and-Drop als intuitive und schnelle Bearbeitungsvariante der Kursstruktur hervorgehoben worden. Die Medienbausteine können per Drag-and-Drop aus der Anzeige der Suchergebnisse in die Darstellung der Kursstruktur hineingezogen und an der entsprechenden Stelle des Kurses platziert werden. Handelt es sich um Medienbausteine, die einer Gruppierung weiterer Medienbausteine dienen, so werden darin enthaltene Medienbausteine in ihren Strukturinformationen ausgewertet und mit in die Kursstruktur übernommen und traversierbar und editierbar angezeigt.

Die zweite unterstützte Drag-and-Drop-Funktionalität ergibt sich aus der Verschiebbarkeit der Medienbausteine innerhalb der aktuell editierten Kursstruktur. Dabei wurde eine von gängigen Editoren und Objektnavigationen abweichende Implementierung der Drag-and-Drop-Funktionalität erarbeitet. Dies resultiert zum einen in der Anforderung der Anzeige von Gruppierungselementen und Medieninhaltsobjekten in einem einzigen Fenster, das eine Übernahme der Drag-and-Drop-Bedienweise von File-System-Navigatoren wie etwa dem Windows Explorer widerspricht, zum anderen in der Anforderung einer schnellen und intuitiven Bedienung, die gerade durch Drag-and-Drop unterstützt werden soll. So ist es etwa im Netscape Communicator Bookmark Editor nicht möglich, in einem einzigen Bedienschnitt mittels Drag-and-Drop bei aufgeklappter Baumstruktur zu entscheiden, ob ein ausgewähltes Bauelement an das Ordnerende oder direkt nach dem Ordnerende auf einer beliebigen höheren Strukturebene, aber vor dem nächsten unter dem Ordnerende angezeigten Baumobjekt, eingefügt werden soll.

Die eindimensionale vertikale Editierung reicht daher für eine Verschiebung per Drag-and-Drop in einem Bearbeitungsschritt nicht aus. Um dies zu ermöglichen, muss in zweiter Dimension in horizontaler Richtung die Einfügetiefe des per Drag-and-Drop zu verschiebenden

Medienbausteins im Kurseditor ausgewertet und graphisch rückgemeldet werden (siehe Abbildung 2).

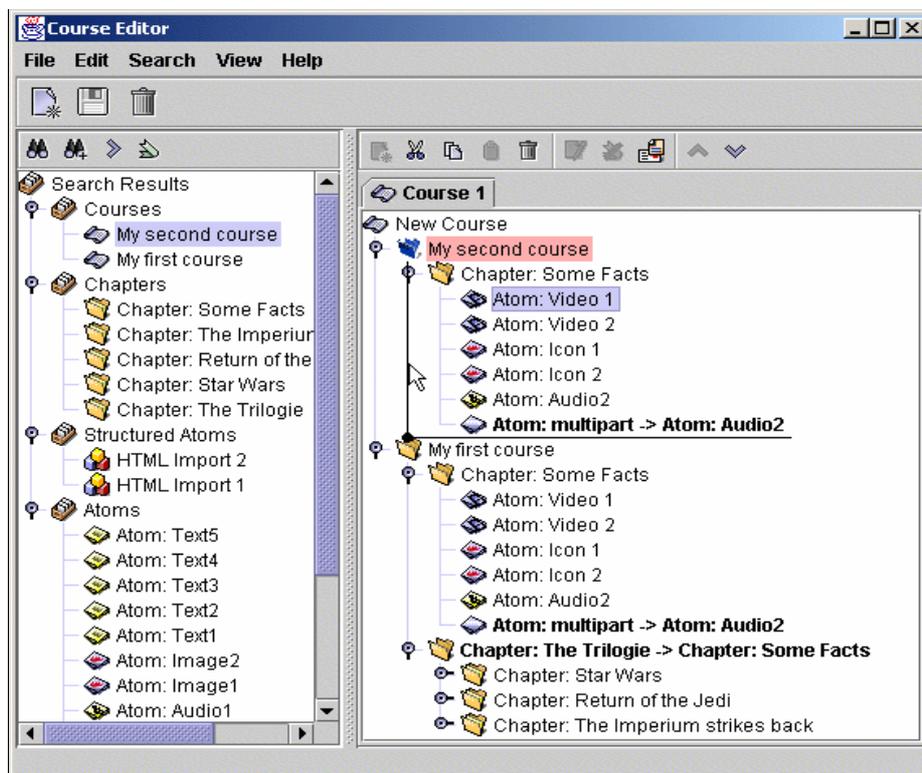


Abbildung 2: 2-D Einfügemarke bei Drag-and-Drop

7.4 Hinterlegen der Metadaten in Datenbank

In der Datenbank werden LOM-Datensätze, die sowohl modularisierte Lernressourcen als auch Kurse beschreiben, gespeichert. Hierfür haben wir basierend auf den LOM-Draft (Version 6.1) ein Datenmodell in Java umgesetzt, das als Grundlage für alle Programmkomponenten dient, die das Metadatenschema verwenden. Dieses Java-Datenmodell wurde so gestaltet, dass es mit Hilfe von Castor [ELG01] auf verschiedene relationale Datenbanken abgebildet werden kann. Castor ist ein Open-Source Framework, um Java-Objekte auf XML-Dokumente oder mittels SQL-Schnittstelle auf Datenbanken abzubilden.

8. Schluss und Ausblick

In dem vorliegenden Papier wurde ein Kurseditor zur Erstellung von adaptierbaren Kursen, die auf modularisierten Lernressourcen basieren, vorgestellt. Zunächst wurde hierfür das zugrundeliegende Datenmodell LOM beschrieben. Dieses wird sowohl als Eingabeformat in Form von Metadatenbeschreibungen für Medienbausteine als auch als Ausgabeformat für die erstellten Kurse verwendet. Hierfür wird die Hierarchie der Kursstruktur auf einen Baum von LOM-Datensätzen unter Verwendung der vorgestellten Erweiterungen abgebildet. Durch die modular abgespeicherte Kursstruktur ergeben sich nicht nur Vorteile bei der späteren Aktualisierung bereits bestehender Kurse, sondern auch verschiedene Möglichkeiten der Adaption dieser Kurse. Diese wurden anhand eines Beispiels erläutert. Anschließend wurde die Applikation des Kurseditors vorgestellt. Hier wurde insbesondere der Workflow bei der Erstellung von Kursen, die intuitiv zu bedienende Drag-and-Drop-fähige Oberfläche und das Speichern der Kurse in der LOM-Datenbank herausgehoben.

Erweiterungen für den vorgestellten Kurseditor sind insbesondere in der Pflege der Wissensbasis zu sehen. Hier ist es notwendig identische LOM-Datensätze zu finden und zu löschen oder unvollständige LOM-Datensätze zu vermeiden. Zusätzlich muss darauf geachtet werden, dass keine Zyklen durch Chapter und CollectionOfAtoms in der Datenbank entstehen. Ein weiteres Thema für Erweiterungen stellt eine Abfragemöglichkeit der Wissensbasis durch Beschreibungslogik dar. Im Laufe des Projektes wird der Kurseditor durch die Möglichkeit der Generierung von Präsentationen der erstellten Kurse erweitert und von Instruktionsdesignern evaluiert.

Literatur

- [ADL01] Advanced Distributed Learning Initiative, Shareable Content Object Reference Model, January 2001, <http://www.adlnet.org>
- [BHL01] T. Berners-Lee, J. Hendler, and O. Lassila, The semantic Web, *Scientific American* 284, 5 (2001), pg. 33-43
- [Bro01] Patria Brown, *Java Look and Feel Design Guidelines*, second edition, <http://java.sun.com/products/jlf/ed2/book/index.html>, Sun Microsystems, 2001
- [Doc02] OASIS, DocBook Technical Committee, www.oasis-open.org/committees/docbook/
- [ELG01] ExoLab Group, *Castor*, <http://castor.exolab.org>, 2001
- [FHSS01] Andreas Faatz, Stefan Hoermann, Cornelia Seeberg, and Ralf Steinmetz, Conceptual Enrichment of Ontologies by means of a generic and configurable approach, in Proceedings of the ESSLLI 2001 Workshop on Semantic Knowledge Acquisition and Categorisation, August 2001
- [HFM+01] Stefan Hoermann, Andreas Faatz, Oliver Merkel, Ansgar Hugo, and Ralf Steinmetz, Ein Kurseditor für modularisierte Lernressourcen auf der Basis von Learning Objects Metadata zur Erstellung von adaptierbaren Kursen. in LLWA 01 - Tagungsband der GI-Workshopwoche "Lernen-Lehren-Wissen-Adaptivität", pages 315-323. October 2001. Research Report #763.
- [HJ02] Clyde W. Holsapple and K.D. Joshi, A collaborative approach to ontology design, *Communications of the ACM*, Volume 45, 2, February 2002
- [LWG02] LOM working group, IEEE P1484.12/D6.4, IEEE Learning Technology Standards Committee, *Draft Standard for Learning Objects Metadata*, <http://ltsc.ieee.org/wg12/index.html>
- [S00] Christian Süß, Adaptive Knowledge Management: A Meta-Modeling Approach and its Binding to XML, in (H.-J. Klein (Ed.), 12. GI-Workshop Grundlagen von Datenbanken, Plön, TR 2005, Christian-Albrechts-Universität Kiel, Germany, 2000
- [SF00] Christian Süß, Burkhard Freitag, Entwicklung und Nutzung von Teachware: Das Passauer Knowledge Management System (PaKMaS), in Rudolf Kammerl, *Computergestütztes Lernen*, 2000. Oldenburg, München
- [SFB99] Christian Süß, Burkhard Freitag, Peter Brössler, Metamodeling for Web-Based Teachware Management, in P.P Chen, D.W. Embley, J. Kouloumdijan, S.W. Liddle and J.F. Roddick, Proc. Intl. WWWCM'99 Workshop on World-Wide Web and Conceptual Modeling in conjunction with ER'99, Nov. 15-18 1999, Paris, France, LNCS 1727, Springer Verlag
- [See01] C. Seeberg, *Modulare Wissensbasen zur Erzeugung adaptiver und kohärenter Lehrdokumente*, Dissertationsschrift, Technische Universität Darmstadt, 2001
- [SRHF00] C. Seeberg, I. Rimac, S. Hörmann, A. Faatz, A. Steinacker, A. El Saddik, R. Steinmetz, *MediBook: Realisierung eines generischen Ansatzes für ein internetbasiertes Multimedia-Lernsystem am Beispiel Medizin*, in Tagungsband: Treffen der GI-Fachgruppe 1.1.3 Maschinelles Lernen (GMD Report 114), pages 96-105, 2000
- [SSFS99] A. Steinacker, C. Seeberg, S. Fischer, R. Steinmetz, *MultiBook: Meta-data for Webbased Learning Systems*, Fachgebiet Industrielle Prozess- und Systemkommunikation, in Proceedings of the 2nd International Conference on New Learning Technologies, 1999
- [SSS01] Jan Schümmer, Till Schümmer, Christian Schuckmann, *COAST Ein Anwendungsframework für synchrone Groupware*, <http://www.opencoast.org/documentation/COAST-NOD2000.PDF>, net.objectdays 2001 conference, Erfurt, Germany, 2001
- [Ste01] A. Steinacker, *Medienbausteine für web-basierte Lernsysteme*, Dissertationsschrift, Technische Universität Darmstadt, 2001

- [Tee01] Gunnar Teege, Targeteam: TArgeted Reuse and GEneration of TEAching Materials, <http://www11.in.tum.de/forschung/projekte/targeteam/>
- [WKLW98] S.Weibel, J.Kunze, C.Lagoze, M.Wolf, *Dublin Core Metadata for Resource Discovery*, <http://purl.org/dc>, RFC2413, 1998
- [WM99] Norman Walsh and Leonard Mueller, *DocBook: The Definitive Guide*, O'Reilly & Associates, Inc., 1st Edition October 1999, <http://www.docbook.org/tdg>