in: E. Leopold und M. Kirsten (Hrsg.): Treffen der GI-Fachgruppe 1.1.3. Maschinelles Lernen.
GMD Report 114. Sankt Augustin, September 2000.

MediBook: Realisierung eines generischen Ansatzes für ein internetbasiertes Multimedia-Lernsystem am Beispiel Medizin

Cornelia Seeberg[1,2], Ivica Rimac[1], Stefan Hörmann[1], Andreas Faatz[1], Achim Steinacker[1], Abdulmotaleb El Saddik[1] und Ralf Steinmetz[1,2]

1

Industrielle Prozess- und Systemkommunikation (KOM)
Fachbereich Elektrotechnik und Informationstechnik
Technische Universität Darmstadt
Merckstr. 25 • 64283 Darmstadt

2

GMD IPSI
Forschungstentrum Informationstechnik
Institut für Integrierte Publikations- und Informationssysteme
Dolivostr. 15 • 64293 Darmstadt

{seeberg, rimac, hoermann, faatz, steinacker, el-saddik, steinmetz}@kom.tu-darmstadt.de

1. Einleitung

MediBook ist ein Kooperationsprojekt der medizinischen Fakultät der Justus-Liebig-Universität Gießen (Erstellung der Inhalte) und des Fachgebiets KOM der Technischen Universität Darmstadt (Entwicklung der technischen Plattform). Es wird vom Hessischen Ministerium für Wissenschaft und Kunst gefördert, um die traditionelle Lehre in den ersten Semestern des Medizin-Studiums um ein flexibles, zeit- und ortsunabhängiges System zum Selbstlernen zu erweitern.

Es ist ein Werkzeug zum Speichern, Verwalten und vor allen Dingen Auffinden und Kombinieren von Lernressourcen. Es bietet Hilfsmittel, um einerseits bestehende Ressourcen zu beschreiben und in einen Zusammenhang zu stellen und andererseits einzelne Ressourcen zu einer Einheit, einem Kurs zu verbinden.

MediBook ist somit eine medizinische Wissensbasis mit einem effizienten Zugriff und Werkzeugen, um aus einzelnen, unzusammenhängenden Informationseinheiten einen kohärenten Kurs zu erzeugen.

Wissensbasis

Der Wissensbasis liegt eine formale Darstellung des Gebietes der Medizin zu Grunde (siehe Abschnitt 4.). Diese formale Darstellung enthält die “Grundwahrheiten” der Medizin: Die wichtigen Begriffe (*Concepts*) - z.B. Niere, Aspirin, Bakterie - sind durch semantische Relationen miteinander verbunden - z.B. Dickdarm ist-Teil Verdauungssystem. In MediBook heißt diese formale Wissensrepräsentation *ConceptSpace*.

Den Modulen - in MediBook Medienbausteine genannt - sind Begriffe zugeordnet. Jeder einzelne ist durch Metadaten beschrieben. Unter anderem werden hier die physikalische Größe der Ressource, die Rechte, das Erstellungsdatum, aber auch pädagogische Eigenschaften gespeichert. Wir verwenden als Metadaten-Schema LOM (*Learning Object Metadata*). Dieser IEEE-Vorschlag ist ein weitverbreiteter Entwurf zu einem internationalen Standard, um Lernressourcen zu beschreiben (siehe Abschnitt 3.). Auf diese Weise können auch andere Systeme auf die MediBook-Ressourcen zugreifen. Damit wird eine Wiederverwendung möglich.

Zusätzlich werden die Medienbausteine miteinander durch rhetorisch-didaktische Relationen verbunden, so dass ein Zusammenhang zwischen ihnen (z.B. Medienbaustein A erklärt Medienbaustein B) hergestellt werden kann.

MediBook ist offen für Ressourcen unterschiedlichsten Formats. Die einzelnen Informationseinheiten können Text, Bilder, Video-Filme, Audio-Dateien oder Animationen sein. Sie können Informationen, Fallbeispiele, Thesen, Motivationen oder Aufgaben enthalten. Schon bestehende Ressourcen sollen eingebunden werden, damit eine effiziente Wiederverwendung der oft sehr aufwändig erstellten Multimedia-Elemente möglich wird.

Der Ansatz, die Medienbausteine auf den unterschiedlichen Ebenen (LOM, rhetorisch-didaktische Relationen und die Zuordnung zu einer formalen Wissensrepräsentation) zu beschreiben, ist nicht auf die Medizin beschränkt. Für ein anderes Wissensgebiet müssen die Begriffe des *ConceptSpace* definiert und gegebenenfalls die Menge der semantischen Relationen erweitert werden.

Szenario

Es gibt drei Rollen, die von MediBook unterstützt werden:

- **Autor:** Der Autor ist ein Mediziner, der in der zu lehrenden Domäne ein erfahrener Experte ist. Die Aufgabe des Autor ist es, die Wissensbasis zu erstellen. Diese Aufgabe besteht aus drei Teilen: Generierung des *ConceptSpace*, Einbinden der eigentlichen Inhalte (Medienbausteine im *MediaBrickSpace*) und Verbinden der Medienbausteine mit den entsprechenden Begriffen.
- **Lehrende:** Der Lehrende, auch ein Mediziner, trifft eine Auswahl aus allen Medienbausteinen für Studierende und bestimmt deren Reihenfolge und die Gliederungsebenen. Damit er sich in der Wissensbasis zurechtfindet, muss das System ihn in geeigneter Weise unterstützen. Durch die im Benutzerprofil gespeicherten Daten über den Lehrenden kann das System adaptiv den relevanten Ausschnitt der Wissensbasis anzeigen.
  Er hat zusätzlich die Möglichkeit, Übergangsseiten zwischen den einzelnen Medienbausteinen zu generieren.
- **Lernender:** Der Lernende hat entweder die Möglichkeit, einem Vorschlag eines der Lehrenden zu folgen oder kann selbst auf der Wissensbasis navigieren, wobei er die gleichen Hilfsmittel wie der Lehrende benutzen kann. Bei individuellen Benutzerprofilen ist eine adaptive Anpassung der Wissensbasis möglich. Der Lernende hat aber nicht die Möglichkeit, einen Kurs für andere zu erstellen.

Ausblick

Zur Zeit sind als Lernende Medizin-Studierende vorgesehen. Durch inhaltliche Erweiterungen der Wissensbasis (eine Systemerweiterung ist nicht notwendig) kann MediBook auch als Weiterbildungssystem für Ärzte oder als Informationssystem beispielsweise für Patienten und ihre Angehörigen verwendet werden.

Die Möglichkeiten für die Lernenden, Annotationen vorzunehmen, *Bookmarks* zu definieren oder interaktive Tests zu bearbeiten, können in weiteren Schritten geboten werden.

Es ist erstrebenswert, insbesondere, wenn das MediBook von einem heterogenen Lernerkreis benutzt wird, dass das System selbst die Generierung der Lektionen und Kurse vornimmt. Dazu ist ein erweitertes Benutzerprofil und ein regelbasiertes Systemmodul notwendig, das die Einträge des Benutzerprofils mit den Beschreibungen der Medienbausteine vergleicht, um die relevanten Bausteine zu finden und als einen Kurs mit Inhaltsverzeichnis zu präsentieren.

2. Architektur

Die Gesamtarchitektur von MediBook integriert den Zugriff auf Informationssysteme unterschiedlicher Ausprägung, die zudem noch an verschiedenen Standorten verwaltet werden. Dies erfordert eine *Broker* orientierte *Middleware*, die eine Zusammenführung und Integration der Metadatenbeschreibungen für die Kursmodule zusammen mit der formalen Wissensrepräsentation des Fachgebiets ermöglicht. Zusätzlich wird dadurch die verteilte Datenhaltung für die Benutzer des Systems transparent, sodass ein virtuelles Gesamtsystem entsteht. Wie man an Abbildung 1 sieht, erfolgen alle Zugriffe der *Client-Tools* auf die in MediBook verwalteten Informationen über diese Komponente. Das schließt sowohl die Zugangskontrolle und damit das Benutzerprofil, als auch alle *Retrieval*-, Erfassungs- und Kompositions-Werkzeuge ein.

Technisch wird der transparente Zugriff auf eine verteilte Wissensbasis durch eine *query*-basierte Vernetzung der einzelnen Datenbanken realisiert. Im Gegensatz zu indexbasierten Techniken, bei der die Datenbanken komplette Indizes ihrer Inhalte austauschen und die dadurch sehr performante Antwortzeiten liefern können, ermöglicht die *query*-basierte Vernetzung einen einfachen Aufbau und Erweiterung der kompletten Wissensbasis. Als mögliche Protokolle zur Kommunikation zwischen den Datenbanken kann ein auf XQL basierendes Anfrageprotokoll verwendet werden. Als Alternative hat die *Open Archives Group* ein einfaches offenes Anfrageprotokoll spezifiziert, mit dem beliebige Datenbanken abgefragt werden können. Dieses Protokoll verwendet allerdings zur Suche nur einen sehr eingeschränkten Satz von Metadaten. In MediBook wird zur Beschreibung und Verwaltung der einzelnen Kursmodule und der vollständigen Kurse der leicht erweiterte *Learning Object Metadata* (LOM) 4.1-Entwurf der IEEE verwendet. Entsprechend wurde für MediBook ein eigenes, auf LOM basierendes Anfrageprotokoll entwickelt. Datenbanken, die LOM-Daten speichern und verwalten und in die MediBook-Wissensbasis integriert werden sollen, senden eine Beschreibung der von ihnen verwalteten Daten an einen zentralen *Content Location Service*. Dieser *Service* wird von allen *Retrieval*-Diensten vor der Anfrage an die Wissensbasis kontaktiert. Der *Content Location Service* wählt anhand der Beschreibungen geeignete Datenbanken für die Anfragen aus und führt sie durch. Die einzelnen Anfragen werden von diesem Dienst gesammelt und

aufbereitet und dann an die aufrufende Komponente übermittelt. Als Datenbanken werden bei MediBook sowohl relationale Datenbanken wie Oracle 8 und DB2 von IBM, als auch die XML-basierte Datenbank Tamino der Software AG verwendet. Entsprechend muss das Anfrageprotokoll auf alle Datenbanksprachen umsetzbar sein. Zur Kodierung der Anfragen bzw. den Ergebnissen verwendet MediBook eine LOM 4.1-konforme DTD. Den *Client-Tools*, die in den folgenden Abschnitten beschrieben sind, liegt hingegen ein objektbasiertes Modell der LOM-Daten zugrunde. Das erfordert eine beliebige Umsetzung der Metadaten von bzw. zu einer XML-Repräsentation, einem Objektmodell und einem relationalen Datenschema, die von der *Middleware* realisiert wird und in Abbildung 2 skizziert ist.

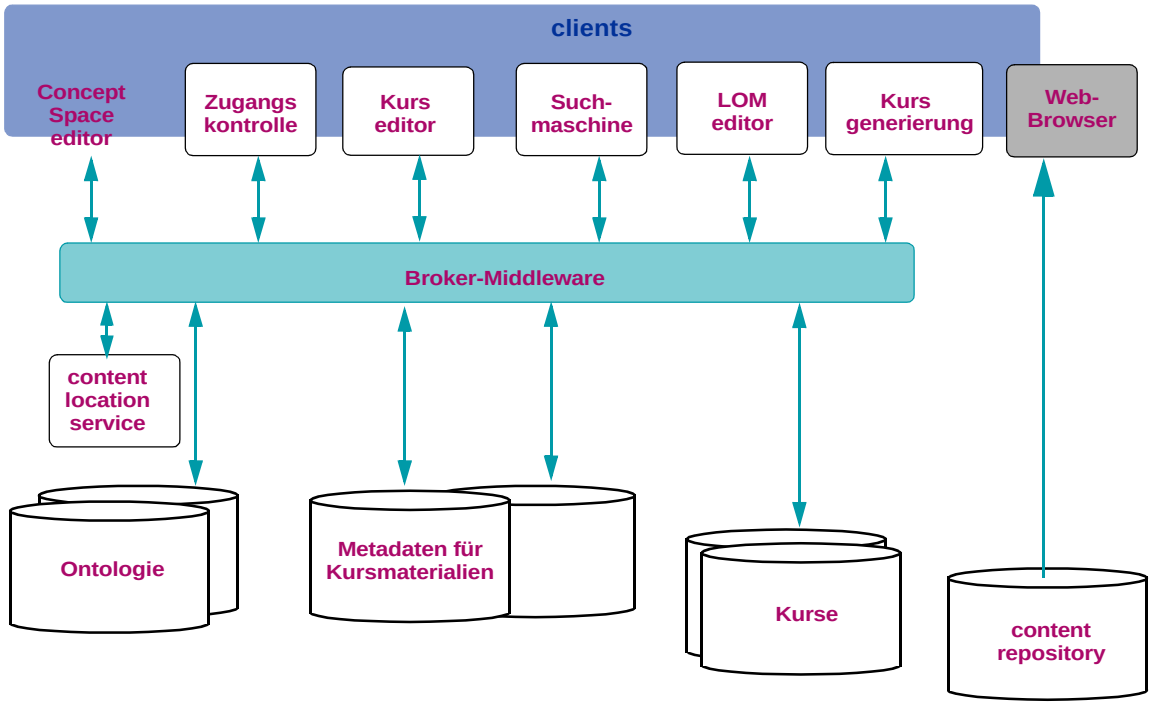

Abbildung 1: MediBook-Architektur

Zusätzlich zur Integration der verschiedenen Datenbanken mus die MediBook-*Middleware* auch die Verbindung zwischen den Metadaten der Medienbausteine und der formalen Repräsentation der Wissensbasis realisieren. Auf dem Gebiet des netzbasierten Wissensaustausch wird derzeit der *Ontology Inference Layer* (OIL) entwikkelt, der einen standardisierten Zugriff auf Ontologien ermöglichen soll. Da hier allerdings derzeit nur erste Entwürfe existieren und noch keine Implementierung verfügbar ist, verwenden wir bei MediBook ein proprietäres Verfahren zur Verbindung der einzelnen *Concepts* der Ontologie mit den Beschreibungen der Medienbausteine. Eine Integration mit dem ISO-Standard *Topic Maps* ist derzeit nicht vorgesehen.

Die Notwendigkeit, einzelne Ressourcen mit Metdaten zu beschreiben, um in der immer unüberschaubareren Informationsdichte effektiv und schnell gewünschte Informationen zu lokalisieren und zu nutzen, hat zu der Entwicklung von Metdaten-Standards für allgemeine Ressourcen wie *Dublin Core*, für Lernressourcen wie der von uns verwendete LOM-Entwurf und zu Metadatenbeschreibungen für multimediale Elemente wie MPEG7, geführt. Auch die Idee der Integration von Metadaten-*Repositories* mit formalen Beschreibungen einer Wissensdomäne findet zunehmend Verbreitung.

Ein kritischer Punkt, der die Akzeptanz der auf diesen Standards aufbauenden Systemen maßgeblich beeinflusst, ist, wie der Benutzer mit der Fülle an Datenmaterial konfrontiert wird. Obwohl eine große Zahl an Metadaten automatisch berechnet und generiert werden kann, ist die Erfassung der benötigten Beschreibungen ein signifikanter Aufwand für den Benutzer. Während bei LOM das Problem in der nötigen Sorgfalt und im Zeitaufwand des Benutzers liegt, stehen für Ontologien bisher praktisch keine Oberflächen zur Verfügung, die es einem Wissensexperten ohne technisches *Know How* er-

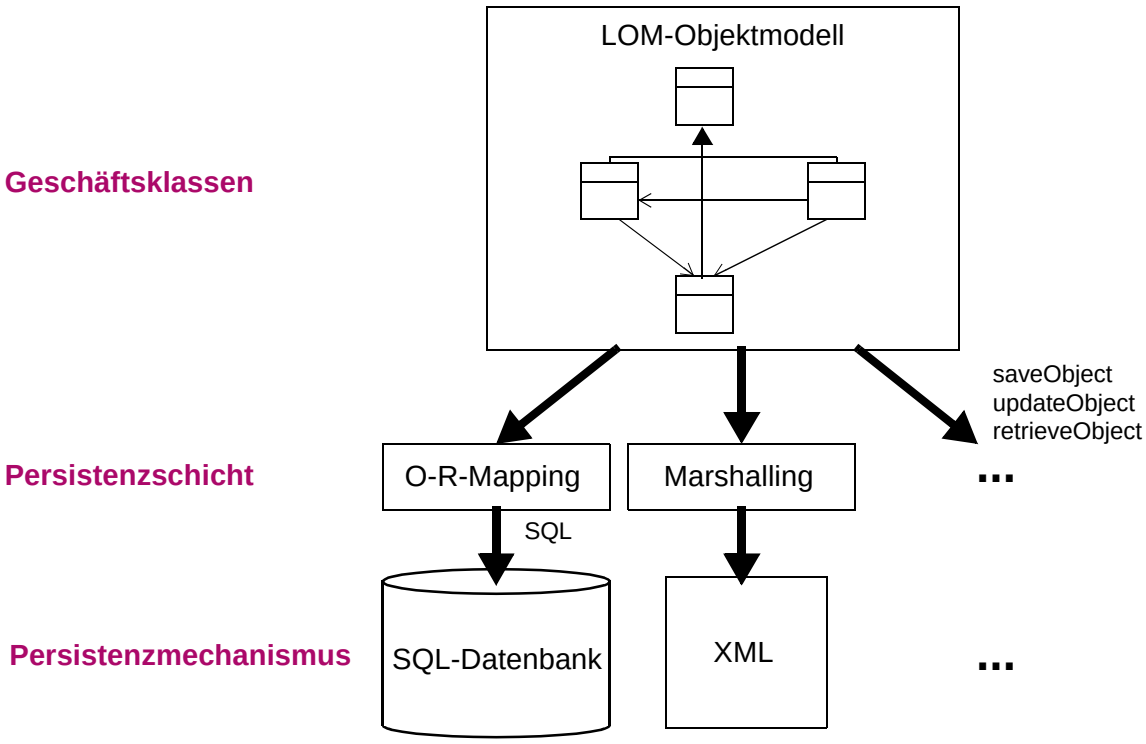

Abbildung 2: Offene Architektur: Mapping von den Geschäftsklassen auf die diversen Persistenzmechanismen

möglichen, eine Ontologie seines Wissensbereiches aufzubauen.

Anhand von drei ausgewählten Werkzeugen - *ConceptSpace-Editor*, *LOM-Editor* und Kursgenerator -, die in MediBook realisiert sind, wird im Folgenden gezeigt wie Benutzern angepasste Sichten auf das Material bereitgestellt werden, die es ermöglichen Informationen über Ressourcen einfach und schnell zu erfassen, auf ihnen zu suchen, sie untereinander und mit *Concepts* aus der Ontologie zu verbinden und sie zu adaptiven Präsentationen zusammenzustellen, die dann *online* verfügbar sind.

3. LOM-Editor

Um das gezielte Suchen und Finden von Informationen (Lernressourcen) in einer einfachen Art und Weise zu ermöglichen, wurde der *LOM-Editor* entwickelt. Dieser Metadaten-Editor basiert auf dem IEEE-LOM-Schema 4.1. LOM schlägt neun Kategorien für die Beschreibung einer Lernressource vor, die im Folgenden beschrieben werden:

- *General*
  Die Kategorie *General* stellt die grundlegenden Informationen zur Verfügung, die dazu beitragen, eine Ressource in ihrer Gesamtheit zu beschreiben. Zu dieser Informationen gehören unter anderem Titel, Sprache, Struktur.
- *LifeCycle*
  LifeCycle beschreibt die Historie und den momentanen Stand einer Ressource und nimmt Bezug auf Personen und Gruppen, die an der Entwicklung teilgenommen haben. Beispiele sind Status, Version.
- *MetaMetaData*
  In der Kategorie *MetaMetaData* werden spezielle Informationen über den Metadatensatz an sich aufgenommen. Die Datenelemente beziehen sich auf die Vorgehensweise bei der Erstellung und auf die dabei beteiligten Personen, beinhalten damit also keine Informationen über die Ressource an sich.
- *Technical*
  Die Kategorie *Technical* fasst die technischen Anforderungen und Eigenschaften der Ressource zusammen. Beispiele sind Format und Anforderungen an das Betriebsystem oder an den *Browser*.

- *Educational*
  Diese Kategorie beschreibt die didaktischen und pädagogischen Eigenschaften einer Ressource. Die pädagogischen Eigenschaften einer Ressource dienen Autoren, Lehrern und Lernenden bei der zielgerichteten Zusammenstellung von Kursen. Beispiele sind u.a. Interaktivitätsgrad, Schwierigkeitsgrad.

mehr als eine Zielressource existiert, wird jedes Ziel durch eine eigene Instanz der Kategorie *Relation* beschrieben. Beispiele sind *partOf, basedOn.*

- *Annotation*
  Die Kategorie *Annotation* stellt Informationen über die Anwendung der Ressource sowie den Verfasser des Kommentars und den Erstellungszeitpunkt zur

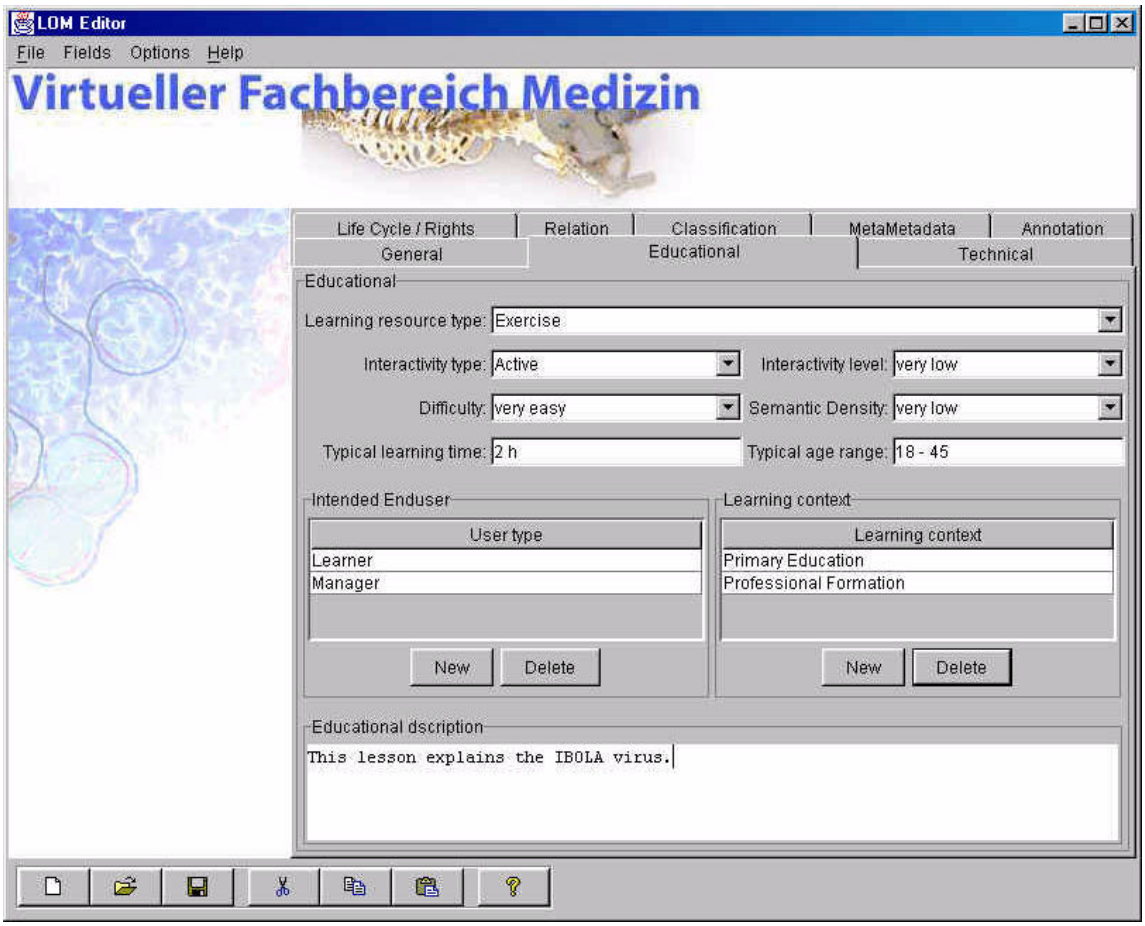

Abbildung 3: Screenshot des LOM-Editor

- *Rights*
  Die Kategorie *Rights* beschreibt die Rechte bezüglich des geistigen Eigentums einer Ressource und deren Nutzungsbedingungen. Die momentan bereitgestellten Datenfelder stellen allerdings nur eine Grundfunktionalität zur Beschreibung der Eigentumsrechte zur Verfügung.
- *Relation*
  Durch *Relation* werden die Beziehungen zwischen einzelnen, miteinander in Verbindung stehenden Ressourcen definiert. Um mehrere Beziehungen zwischen Ressourcen zu definieren, können mehrere Instanzen dieser Kategorie genutzt werden. Falls

Verfügung. Falls mehrere Anmerkungen zu einer Ressource benötigt werden, besteht die Möglichkeit, verschiedene Instanzen dieser Kategorie zu verwenden.

- *Classification*
  In der Kategorie *Classification* findet eine Einordnung der Ressource in bestimmte Klassifizierungssysteme statt. Das Wissen, das zur Klassifizierung einer Ressource notwendig ist, übersteigt den Informationsgehalt des Datenmaterials um ein Vielfaches. Die Eingliederung in eine hierarchische Struktur setzt nicht nur Kenntnisse über das zu charakterisierende Objekt voraus, sondern über alle Ressourcen eines

Klassifizierungsraums. Nur aus diesem Wissen lässt sich eine Relation zwischen Objekten definieren und eine Einordnung in ein bestimmtes System vornehmen.

Der entwickelte *LOM-Editor* (siehe Abbildung 3) kann verwendet werden, um einen Metadaten-Satz einer Lernressource zu generieren und an die *Middleware* zu übergeben, die für eine dauerhafte Speicherung zuständig ist. Er kann auch verwendet werden, um die Beschreibung vorhandener Ressourcen zu ändern.

Die Vorteile, die sich durch die Nutzung von Metadaten bei der Suche und Wiederverwendung von Lernressourcen ergeben, hängen stark von der Qualität und dem Umfang der Metainformationen ab. Ein vollständig bestimmter Metadatensatz liefert im Allgemeinen eine bessere Grundlage für die konsistente und eindeutige Beschreibung einer Lernressource. Geht man vom Einsatz des LOM *Base Scheme* mit insgesamt 58 Datenelementen aus, so ist eine manuelle Beschreibung der Ressource wenig effizient. Der Realisierungsaufwand für einen programmgestützten Generierungsmechanismus wird sich nach kurzer Zeit amortisiert haben.

Während des Einsatzes des *LOM-Editors* bei der Beschreibung verschiedenen Lernressourcen wurde festgestellt, dass im Allgemeinen einige Basis-Metadaten-Elemente, wie z.B. *author, rights of the lesson*, oder *targeted user group* unverändert bleiben. Aus dieser Beobachtung wurde festgestellt, dass der Einsatz von Schablonen (*Templates*) das Generieren und Abspeichern von Metadaten vereinfachen wird. Jedem Autor wird hierbei die Möglichkeit gegeben, sich eigenständig Vorlagen zu definieren, die eine bestimmte Menge von Datenelementen beschreiben. Eine Möglichkeit wäre die Vorgabe von fest definierten Schablonen, bei denen der Autor die Möglichkeit hat, den einzelnen Datenelementen Werte zuzuweisen und die entsprechenden Eingaben zur weiteren Verwendung zu speichern. Denkbar ist beispielsweise die Zusammenfassung der gesamten Merkmalsgruppe "*Contribute*" zu einer Schablone, in der die einzelnen Personen eines Projektes mit Namen und Tätigkeit gemeinsam erfasst werden. Alle innerhalb dieses Projektes erstellten Ressourcen lassen sich anschließend über diese Schablone mit relativ geringem Zeitaufwand eindeutig beschreiben.

4. Der ConceptSpace-Editor

Der folgende Abschnitt stellt dar, wie das Werkzeug zur Wissensrepräsentation im MediBook, der *ConceptSpace-Editor*, aufgebaut ist. Dabei gehen wir hauptsächlich auf die logischen Schichten des *ConceptSpace-Editors* ein, die innerhalb des *Smalltalk Frame Kit* (SFK) implementiert wurden.

Das MediBook verwendet als formale Wissensrepräsentation eine Ontologie. Eine Ontologie ist eine formale Konzeptualisierung eines Wissensbereiches.

Die Modellierung spezieller Wissensbereiche erfolgt nach unserem Verständnis zielgerichtet, was sich auf den Aufbau und die zulässigen Strukturen der Ontologie auswirkt. Im MediBook erfüllt die Ontologie die Funktion einer netzförmigen Navigationsstruktur für medizinische Fachbegriffe und wird zur Anordnung von Medienbausteinen herangezogen.

Ein logischer und konsistenter Aufbau der Ontologie erfordert neben den *Concept*typen, die das Wissensgebiet charakterisieren, und den Relationstypen, die die *Concepts* miteinander verbinden, die Definition von Axiomen. Axiome dienen in unserem Fall zur ständigen logischen Überwachung der Wissensmodellierung. Zwei Beispiele für den Einsatz der Axiome sind das automatische Erstellen von Umkehrrelationen zwischen *Concepts* und die Einhaltung hierarchischer Beziehungen. Das bedeutet, dass man Vorgaben formalisieren muss, die beispielsweise für eine Aussage wie “Kolibakterien verursachen Durchfall” gleichzeitig eine Aussage “Durchfall wird durch Kolibakterien erzeugt” generieren. Im Falle der hierarchischen Anordnung von *Concepts*, wie etwa bei der Aussage “Muskeln sind ein funktionaler Bestandteil des menschlichen Bewegungsapparates” ist hingegen zu verhindern, dass eine nachträgliche fehlerhafte Aussage wie “der menschliche Bewegungsapparat ist ein funktionaler Bestandteil der Muskeln” angelegt werden kann.

Eine Programmiersprache, die die Vorgabe eines zweckdienlichen Schemas für die Erstellung einer Ontologie ermöglicht, ist das *Smalltalk Frame Kit*, abgekürzt SFK. Ein solches Schema besteht aus *Concept*typen, Relationstypen und Axiomen und wird erst bei der eigentlichen Erstellung des MediBook-*ConceptSpace* durch den Mediziner mit Fachbegriffen gefüllt. Unser Ansatz sieht vor, die Arbeit des Mediziners wesentlich zu erleichtern und seinen Fokus auf die eigentliche Anordnung der Fachbegriffe zu lenken. Er soll lediglich die Inhalte der Klassifikation und Verknüpfung medizinischer Fachbegriffe durch Relationen liefern, ohne sich um formale Logik, Axiomatik oder gar Implementierungstechniken kümmern zu müssen.

Als Resultat dieser Anforderung nehmen wir eine Arbeitsteilung bei der Ontologie-Erstellung vor: einerseits erfolgt eine Schemaerstellung im SFK durch einen Software-Entwickler, andererseits wird dem Mediziner mit dem *ConceptSpace-Editor* eine graphische und intuitive Möglichkeit zur Schemafüllung geboten. Das SFK unterstützt und überwacht dabei die Schemafüllung gemäß der Axiome, ohne dass der Mediziner sich mit der Implementierung des SFK-Schemas auseinandersetzen muss. Zwischen dem SFK-Software-Entwickler und dem Mediziner muss naturgemäß eine Absprache darüber herrschen, was in der Ontologie abgebildet werden soll. Im MediBook wurden folgende *Concept*typen im SFK-Schema modelliert: KrankheitOderSymptom, Zelle, Stoff, MediBook-*Concept*. Die Anzahl der *Concept*typen wurde für die erste Version bewusst gering gehalten, da aufgrund des stark objektorientierten Ansatzes des SFK eine Erweiterung um weitere *Concept*typen und die diesbezügliche Aktualisierung der Schema-Instanziierung durch den Mediziner leichter von statten gehen kann.

Die gesamte Arbeitsweise des SFK basiert auf dem Prinzip der *Frame*-Klassen. Das sind semantische Einheiten, die mit festgelegten Attributen (*Slots*) ausgestattet sind und instanziiert werden. Die *Slots* können mit Restriktionen an ihren Wertebereich und die Anzahl ihrer Werte ausgestattet werden und fungieren gleichzeitig als Träger der Axiome des Schemas. Für unsere beiden Beispiele der inversen Relationen und der hierachischen Anordnung stehen im SFK folgende Methoden zur Verfügung: *#inverseSlot*, die den Namen eines inversen *Slots* übergibt, und *#relationalProperties*, die für die obige Relation “ist funktionaler Bestandteil von” die Eigenschaften azyklisch und transitiv übergibt. Gerade für die automatische Berechnung von Frame-Klassen und *Slots* existieren im SFK reichhaltige Inferenzmechanismen, die Berechnungen transitiver Hüllen beispielsweise sind bereits vollständig vorgegeben. Somit ist nutzt jeder Aufbau eines SFK-Schemas bereits vorhandenen Algorithmen, die das zur Einhaltung der für das Schema relevanten Axiome notwendige Neuanlegen, Löschen und Verändern von *Frame*-Klassen vornehmen.

Ein ontologisches Schema im SFK ist somit eine Hierarchie von *Frame*-Klassen, deren Bezug zueinander festgelegt werden kann, indem man gleichzeitig das SFK als Programmiersprache benutzt. Die modellierte Hierarchie wird vom SFK selbst erst dann fest installiert, wenn das komplette Schema angelegt wurde. Inkonsistenzen wie beispielsweise mehrfaches Anlegen von gleichnamigen *Slots* in einem *Frame* werden an diesem Punkt bereits abgefangen. Jede Erweiterung des Schemas, die *Frames* und *Slots* hinzufügt (vorausgesetzt, die Vererbungsstruktur, die das Schema ausmacht, wird nicht verändert) ist jederzeit durchführbar.

Das SFK exportiert fortlaufend den aktuellen Stand einer Schema-Instanziierung per XML an ein Java-Werkzeug zur Visualisierung von Graphen. Umgekehrt wird die Schema-Instanziierung durch den Mediziner - das Anlegen von benannten Knoten (*Concepts*) und aus der Menge der Kantentypen selektierten Kanten (Relationen) im Graphen - per XML an das SFK zurückgegeben. Ist eine Erweiterung des *ConceptSpace* durch ein neues *Concept* und/oder das Ziehen einer neuen Relation unzulässig, wie etwa das Anlegen kreisförmiger Beziehungen in Hierarchien, so wird diese Erweiterung nicht zugelassen.

Zulässige Operationen des Mediziners werden visualisiert, wobei der Graph durch einen auf dem Prinzip der Federkräfte basierenden Algorithmus so übersichtlich wie möglich dargestellt wird. Das SFK, die XML-Schicht und der Java-*Client* ergeben zusammengenommen den *ConceptSpace-Editor*.

Adaptive Navigationssteuerung

Navigationssteuerungen für allgemeine netzförmige Graphen, mit denen der *ConceptSpace* dargestellt werden kann, sind derzeit kaum verbreitet. Bisher sind nur spezielle Anwendungen oder Systeme für spezielle Netztopologien in kommerziellen Anwendungen zu finden. Insbesondere ist hier die Baumstruktur zu nennen. In dieser hat jedes Element eine Verknüpfung zu seinem übergeordneten Element, ausgenommen das Wurzel-Element. Beliebig viele Verknüpfungen können zu möglicherweise untergeordneten Elementen vorliegen. Benachbarte Elemente können jedoch keine Beziehung untereinander haben. Die Baumstruktur vereinfacht in vielen Fällen die tatsächlich vorliegende relationale Struktur der Objektbeziehungen, da eine Ontologie in der Regel eine vernetzte Struktur aufweist und die Baumstruktur diese als grobe Vereinfachung darstellt. Mit dem Zwang einer Einordnung für jedes Element wird der Informationsraum in disjunkte Klassen zerlegt. Mit dieser Klassifikationsstruktur werden bestimmte Objektverknüpfungen verdeckt oder redundante Zusatzinformationen in die Baumstruktur eingefügt. Dazu kommt, dass auch bei großen Baumstrukturen der Überblick verloren gehen kann

Betrachtet man die existierenden Ansätze zur Visualisierung von allgemeinen Netzen, so ist festzustellen, dass eine interaktive Navigation i.d.R. nicht vorgesehen. ist. Zudem ändert sich bei geringen Veränderungen in der Netzstruktur das komplette Layout des Graphen. Weiter-

hin ist bei der Visualisierung einer Ontologie zu beachten, dass Überschneidungen von Knoten und Kanten und schematische Ausrichtungen nicht zu einer fehlerhaften semantischen Interpretation führen. Für den bei MediBook entwickelten Ontologie-Editor sind daher die folgenden Randbedingungen von Bedeutung:

- Navigation in allgemeinen Netzen,
- Stabilität bei geringen Veränderungen der Netzstruktur,
- Visualisierung von inhaltlicher Nähe durch räumliche Nähe.

Aufgrund der beschränkten Visualisierungsfläche auf einem Monitor und der begrenzten Aufnahmefähigkeit des menschlichen Betrachters können große, komplexe Graphen nicht in ihrer Gesamtheit dargestellt bzw. erfasst werden. Folgende drei unterschiedliche Basisvarianten zur Navigation durch einen Graphen können unterschieden werden:

- Verschiebung des sichtbaren Bereichs,
- Ausschnittvergrößerung bzw. Verkleinerung (*zoomen*),
- Öffnen und Schließen von einzelnen Knoten.

Daneben existieren Kombinationen der Basisvarianten, die weitere Möglichkeiten erschließen, zum Beispiel eine Navigation mit zwei Fenstern. In dem einen Fenster wird ein Überblick des gesamten Graphen dargestellt, mit der Möglichkeit, auf einen einzelnen Bereich zu fokussieren. Dessen Detailansicht wird in einem zweiten Fenster dargestellt. Dieses Vorgehen entspricht einer Landkarte mit einer Überblicksdarstellung und einer zusätzlichen Detailansicht in einem Stadtplan. Eine weitere Möglichkeit ist die Fischaugen-Darstellung. Dabei wird die Ausschnittsvergrößerung auf einen Bereich beschränkt, ähnlich wie bei einem Blick durch eine Lupe. Bei Lösungen mit *Scrollbars* bzw. beim *Zoomen* in einen Bereich hinein, kann schnell der Kontext verlorengeht. Wie bereits erwähnt, hat sich die Navigation in hierarchischen Baumstrukturen, wie sie im Explorer praktiziert wird, bewährt. Bei dieser Navigation werden interaktiv einzelne Knoten geöffnet bzw. geschlossen. Damit ist gleichzeitig eine Übersichts- und Detaildarstellungen des Graphen möglich. Diese ist quasi das Optimum zur Darstellung und Navigation auf einem begrenzten Raum, und daher wurde beim *ConceptSpace-Editor* dieses Verfahren auf

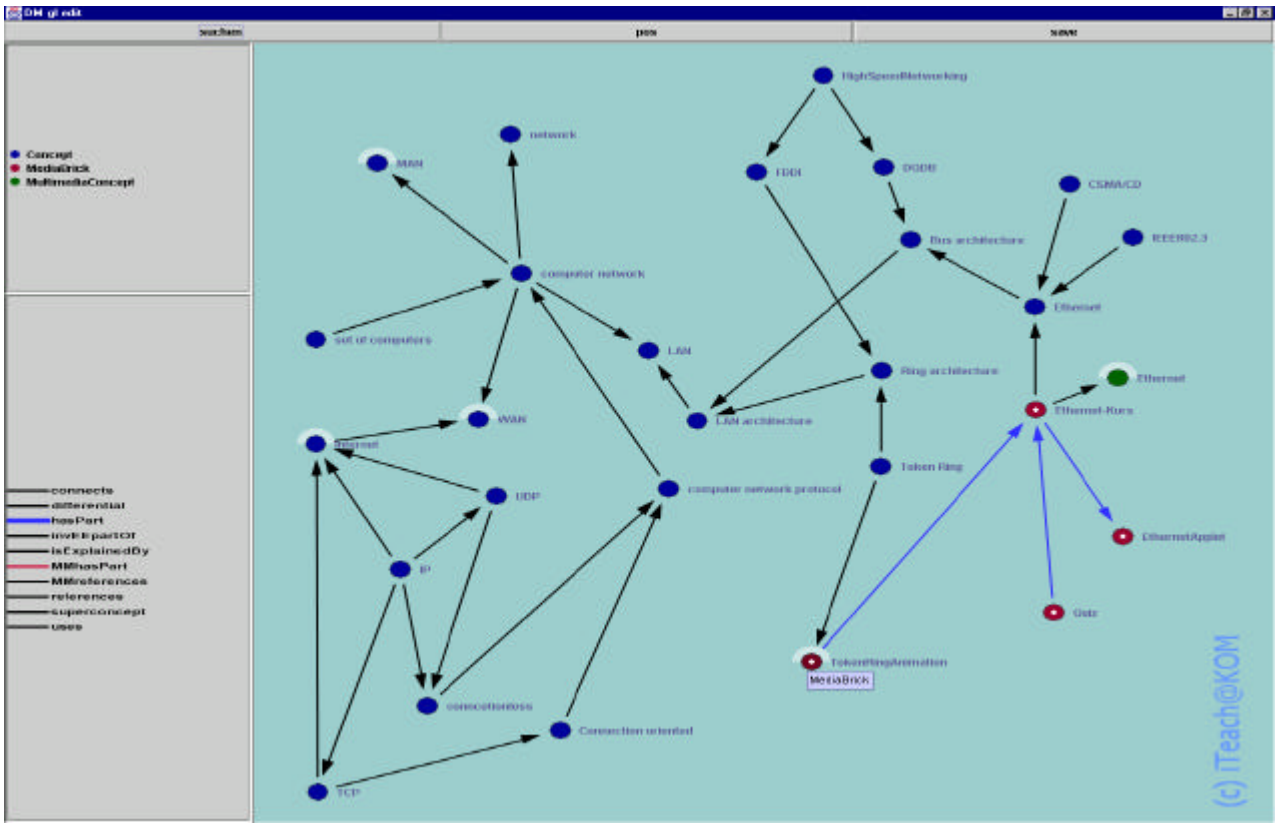

Abbildung 4: Screenshot der graphischen Oberfläche des ConceptSpace-Editors

allgemeine Graphen übertragen. Einen Screenshot, des Editors zeigt Abbildung 4.

Aus dieser Abbildung wird auch ersichtlich, wie die Verbindung zwischen einzelnen Konzepten der Ontologie und den Medienbausteinen erfolgt. Medienbausteine, die mit dem LOM-Editor erfasst und in der Wissensbasis gespeichert wurden, können auf der graphischen Oberfläche eingefügt und mit den Konzepten aus der Ontologie verbunden werden.

Das Vorgehen bei dieser Navigation entspricht einer Informationsreduktion des Graphen auf das Wesentliche. Es werden nur die Informationen angezeigt, die zur Zeit für den Benutzer zur Navigation wichtig sind. Dabei können durch das Zusammenfassen von Teil-Graphen, weitentfernte Knoten gleichzeitig dargestellt werden. Der visuelle Eindruck über die Struktur der verbliebenen Knoten und Kanten bleibt dabei im Wesentlichen erhalten.

Zukünftige Arbeiten bestehen in einer Evaluierung des derzeitigen Schemas und einer entsprechenden Erweiterung. Ein weiteres Ziel, das über das Projekt MediBook hinausgeht, ist die Vereinfachung der Schemaerstellung im SFK durch einen ebenfalls graphisch unterstützten Editor.

5. Kursgenerator

Mit dem Modul, das den Namen Kursgenerator trägt, können Präsentationen der Kurse erzeugt werden, die mit dem Kurseditor erstellt wurden (siehe Abbildung 1). Die zu präsentierenden Kurse setzen sich aus Lektionen zusammen, die eine Aufzählung mit festgelegter Reihenfolge von Medienbausteinen sind. Die Referenzierung der Medienbausteine einer Lektion wird über Relationen des Typs *HasPart* und *IsPartOf* der *Learning Object Metadata* der Kategorie *Relation* realisiert. Die baumförmige Struktur der Kurse, deren Blätter stets Medienbausteine enthalten, resultiert aus der Möglichkeit der Referenzierung von Lektionen als Teil einer Lektion, wodurch Unterlektionen definiert werden. Die auf diese Weise in Zusammenhang gebrachten Medienbausteine können komplexe Lehrangebote wie beispielsweise Vorlesungen bilden. Abbildung 5 zeigt den Teil einer leicht abstrahierten Struktur eines Kurses, aus der die logischen Zusammenhänge von Lektionen und Medienbausteinen zu entnehmen sind.

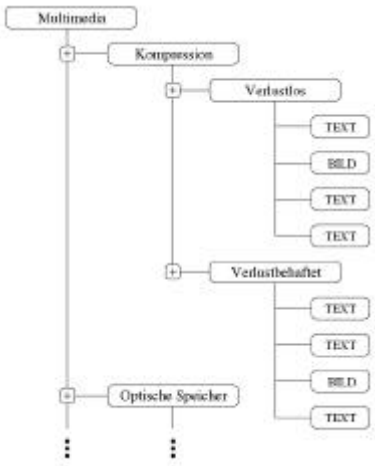

Abbildung 5: Kursstruktur

Für die automatische Erzeugung von Präsentationen dieser Kurse eignen sich in erster Linie die Dateiformate HTML und PDF. Die beiden Ausgabeformate der Kurse sind für das Bearbeiten der Kurse am Computer beziehungsweise zum Ausdrucken der Kurse geeignet. Hieraus ergeben sich zwei grundsätzlich verschiedene Anforderungen an die Präsentationen, die bei der Erzeugung von Präsentationen zu differierenden Lösungsansätzen führen.

Der Vorteil der Präsentation in Form von HTML-Seiten liegt darin, dass sie direkt von den Lernenden am Computer betrachtet werden können, ohne dass dafür weitere Software außer einem Internet-Browser installiert werden muss. Der jedoch wichtigste Vorteil bei der Erzeugung von HTML-Seiten liegt in der Verwendungsmöglichkeit von kontinuierlichen Medien, wie Klänge und Bildsequenzen, um die Präsentation der Lernressourcen multimedial zu gestalten. Es wird jedoch zusätzlich ein Mechanismus benötigt, mit dessen Hilfe die Medienbausteine zu kleinen Einheiten zusammengefasst werden können. Dadurch soll erstens das Durcharbeiten des Kurses am Computer vereinfacht werden und zweitens das schnelle Auffinden der relevanten Lernressourcen ermöglicht werden. Abbildung 6 zeigt ein Beispiel für die Präsentation des Kurses nach Abbildung 5 in HTML-Seiten.

Die Präsentation der Medienbausteine eines Kurses in Form einer PDF-Datei lässt sich zwar ebenfalls direkt am Computer betrachten, ist jedoch hauptsächlich für den Ausdruck auf Papier der Kurse gedacht. Aus diesem Grund wird die Präsentation der Kurse auf Medienbau-

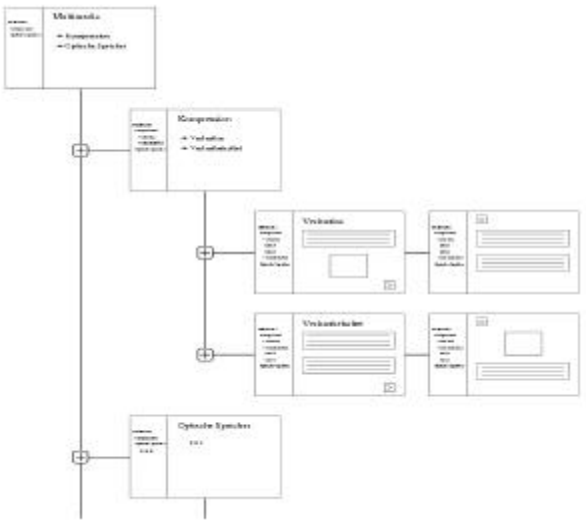

Abbildung 6: HTML-Seiten

steine mit statischem Inhalt, wie beispielsweise Texte und Bilder, beschränkt. Anders als bei der Erzeugung von HTML-Seiten werden die Medienbausteine des Kurses nicht zu kleinen Gruppen zusammengefasst, sondern in *Pre-Order* Tiefensuche des baumförmigen Kurses in der PDF-Datei hintereinander gehängt, wodurch die von Büchern gewohnte lineare Struktur der Lernressourcen entsteht (siehe Abbildung 7).

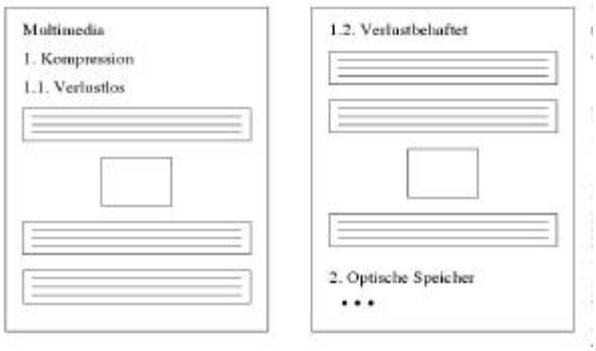

Abbildung 7: PDF-Datei

Bevor jedoch die eigentlichen Präsentationen erzeugt werden können, wird eine Beschreibung des Kurses erzeugt, die als XML-Datei exportiert werden kann. Zusammen mit dem Pfad innerhalb des Kurses zu der Lektion, von der eine Präsentation erzeugt werden soll, und dem XSL-Stylesheet, das für die Erzeugung der Präsentation verwendet werden soll, kann die Präsentation erzeugt werden. Die Generierung von Präsentationen ist unabhängig davon, ob erzeugte Dateien auf Festplatte oder CD-ROM gespeichert werden oder direkt über Internet zu den Lernenden übertragen werden. Die generierten Präsentationen können von den Lernenden mit Standardsoftware betrachtet werden. Für die PDF-Dateien kann der kostenfrei verfügbare *Viewer* der Firma Adobe verwendet werden. Die Betrachtung der HTML-Dateien erfolgt mit einem handelsüblichen Java-fähigen Internet-*Browser*. Bei der Betrachtung der HTML-Dateien werden die Lernenden durch ein Java-Applet, das im Internet-*Browser* eingebettet wird, bei der Navigation durch den Kurs unterstützt. Damit hierbei die Orientierung der Lernenden zu keinem Zeitpunkt verloren geht, wird eine Übersicht über die Struktur des Kurses angezeigt, in der die aktuell angezeigte HTML-Seite markiert ist.