Generische E-Learning-Plattform für interaktive Lehrsimulationen zum Einsatz in Selbststudium und Präsenzlehre online und offline

Dissertation

zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (Dr. rer. nat.)

Dipl.-Inform. Andreas Dieckmann

Technische Fakultät
Universität Bielefeld

Gutachter:

Prof. Dr. Robert Giegerich
Prof. Dr. Helge Ritter

31. 12. 2003

Inhaltsverzeichnis

1 Einleitung

Die vorliegende Arbeit entstand im Kontext des Projektes Monist[1], das im Rahmen des Förderprogrammes "Neue Medien in der Bildung" vom Bundesministerium für Bildung und Forschung (BMBF) gefördert wurde[2]. Die Ziele dieses Projektes[3] waren die Schaffung einer E-Learning-Umgebung für die universitäre Ausbildung im Bereich der Neuro- und Kognitionswissenschaften und den damit zusammenhängenden Fachdisziplinen (z. B. Neurobiologie, Psychologie und Informatik) sowie die "Produktion einer kritischen Masse von Inhalten in einem interdisziplinären Curriculum"[4] für diesen Themenbereich.

Der Schwerpunkt lag dabei auf dem Einsatz von Modellsimulationen in der Lehre. "Modellsimulationen nehmen in den modernen Neuro- und Kognitionswissenschaften und Informatik eine Schlüsselposition ein: Modelle sind einerseits ein entscheidendes Ergebnis experimenteller Forschung, andererseits häufig Ausgangspunkt für neue experimentelle Ansätze. Darüber hinaus gehören Modellsimulationen zu den aussichtsreichsten Neuen Medien für die Lehre, da sie Wissen transportieren, das ausschließlich mit dem Computer gelernt werden kann."[5]

Die Ziele dieser Arbeit waren der Entwurf und die anschließende Implementierung einer E-Learning-Plattform, die eine geeignete Infrastruktur für die Produktion, Verwaltung und Nutzung von auf Modellsimulationen basierenden Lehreinheiten[6] zur Verfügung stellt. Dabei sollte jedoch kein spezialisiertes E-Learning-System geschaffen werden, das lediglich für den spezifischen Einsatz im Rahmen des Projektes Monist geeignet ist, sondern eine von den konkreten Fachinhalten unabhängige Plattform für den Einsatz von Simulationssoftware in der Lehre.

Auf diese Weise wird es ermöglicht, die geschaffene E-Learning-Plattform für beliebige Fachbereiche einzusetzen und in Abhängigkeit vom jeweiligen inhaltlichen Kontext andere Applikationen als die im Projekt Monist verwendete Simulationssoftware für den Einsatz in der Lehre nutzbar zu machen.[7] Änderungen an der E-Learning-Plattform selbst sind dafür nicht erforderlich[8], lediglich die verwendeten Simulationsprogramme müssen bestimmte Schnittstellen implementieren bzw. an diese angepaßt werden.[9]

---

[1] Modellsimulation neuronaler und kognitiver Informationsverarbeitung -- Schule der Techniken

[2] Förderkennzeichen: 08NM055, Projektlaufzeit: 01.01.2001 bis 31.12.2003

[3] Siehe hierzu die *Projektskizze zur Einreichung bei dem Förderprogramm des Bundesministeriums für Bildung und Forschung: "Neue Medien in der Bildung"* unter der Internet-Adresse http://www.monist.de bzw. http://natrix.biologie.uni-bielefeld.de/monist/Archiv/monistskizze_oz.html

[4] a. a. O.

[5] a. a. O.

[6] Der Begriff "Lehreinheiten" dient im Rahmen dieser Arbeit als allgemeiner Oberbegriff für jede Art von in der Lehre verwendeten Kombinationen aus Medien und Programmen, die im Rahmen der entworfenen E-Learning-Plattform verwaltet werden können. Die Bandbreite reicht dabei von einfachen elektronischen Vorlesungsnotizen bis hin zu komplexen Online-Kursen, deren Umfang sich in der Praxis über ein ganzes Semester erstrecken kann.

[7] Ein Beispiel für ein derartiges, in die im Rahmen dieser Arbeit entwickelte Lernplattform integrierbares Simulationsprogramm stellt die Laborsimulations-Software dar, die vom Autor der vorliegenden Arbeit für den Einsatz in der Lehre in den Fachbereichen Chemie und Biochemie erstellt wurde. Diese Software wurde abgehandelt und dokumentiert in: Andreas Dieckmann, Basisfunktionen für ein generisches virtuelles Labor, Diplomarbeit an der Technischen Fakultät der Universität Bielefeld, 1998.

[8] Mit Ausnahme der Anpassung bestimmter Konfigurationsdateien, wodurch jedoch insbesondere keine Neu-Compilierung des Quellcodes erforderlich ist. Dieses Vorgehen wird in Kapitel 8 näher ausgeführt.

[9] Eine ausführliche Beschreibung dieser Schnittstellen ist Teil dieser Arbeit und erfolgt in Kapitel 8.

Die Anforderungen an die E-Learning-Plattform, die sich aus den für die vorliegende Arbeit gesetzten Zielen ergeben, sind in Kapitel 2 im einzelnen dargestellt. Dabei wird insbesondere auf die möglichen Einsatzszenarien von darauf aufbauenden E-Learning-Systemen eingegangen, woraus eine Reihe von Notwendigkeiten bezüglich der Funktionalität der zu schaffenden Softwareplattform folgt.

Kapitel 3 gibt eine Übersicht über die Funktionalität von bestehenden kommerziellen und nichtkommerziellen Softwaresystemen, die hinsichtlich ihrer Verwendbarkeit für den hier betrachteten Einsatzbereich analysiert wurden. Diese Untersuchung wurde in Form von umfangreichen Marktanalysen auf verschiedenen Messen sowohl zu Beginn der Projektlaufzeit als auch im fortgeschrittenen Projektstadium durchgeführt. Darüber hinaus wurde auch eine Reihe von E-Learning-Systemen aus dem universitären Bereich in die Untersuchung einbezogen. Auf Basis dieser Recherchen wird aufgezeigt, warum keines der betrachteten, bereits existierenden Softwaresysteme als Grundlage für die komplette Umsetzung der in Kapitel 2 aufgezeigten Anforderungen verwendet werden konnte. Diese Überlegungen waren der Grund dafür, daß die Entscheidung für eine Neuentwicklung getroffen wurde.

In Kapitel 4 wird die für diese Entwicklung entworfene Systemarchitektur dargestellt. Einen wichtigen Punkt stellt dabei die Betrachtung des Systems im Spannungsfeld zwischen traditionellen, zentralistischen Client-Server-Systemen und eigenständigeren, unabhängig von einer zentralen Kontrollinstanz lauffähigen Peer-to-Peer-Systemen dar. Im Anschluß daran werden in Kapitel 5 ausführlich die Datenstrukturen beschrieben, die für die Speicherung und Verwaltung der Daten und Metadaten entworfen wurden, welche für die zu vermittelnden elektronischen Lehreinheiten benötigten werden. Dabei wird ebenso auf die Notwendigkeit zur Verwendung von Metadaten als auch auf die strikte Trennung von Daten und Metadaten eingegangen.

Den Kern dieses Kapitels bildet der Entwurf der sogenannten Virtuellen Lernobjekte (VLO), die zur umfassenden Beschreibung jeglicher im System abgelegter Objekte (von einfachen Mediendateien über komplette Lehreinheiten bis hin zu Repräsentationen der mit dem E-Learning-System arbeitenden Benutzer) dienen. Darüber hinaus werden auch die unterschiedlichen Speichermechanismen beschrieben, die verwendet wurden, um diese Virtuellen Lernobjekte mitsamt aller für ihre Verwaltung benötigten Metadaten in einer eigenen Datenbank, die eigentlichen multimedialen Inhalte jedoch separat davon in einem über einen HTTP-Server zugänglichen Dateisystem abzulegen.

Im Anschluß an diese Betrachtung der lokalen Speicherung von Daten und Metadaten wird in Kapitel 6 aufgezeigt, auf welche Art und Weise durch die Vernetzung einzelner lokaler Installationen der geschaffenen E-Learning-Plattform eine kooperative Nutzung des Systems über das Internet ermöglicht wird und wie die Verteilung von beliebigen elektronischen Lehreinheiten an die betreffende Zielgruppe über das globale Datennetz bewerkstelligt wird.

Zu diesem Zweck wird zunächst der für die Online-Kommunikation des Programmes verwendete XML-RPC-Kommunikationsmechanismus beschrieben. Es wird erläutert, warum dieser Technologie aufgrund ihrer speziellen Eigenschaften im Vergleich mit anderen möglichen Kommunikationsmechanismen der Vorzug gegeben wurde. Dabei wird auch darauf eingegangen, auf welche Weise dieser Mechanismus im entworfenen System dazu genutzt wird, um jedem installierten "Client" auch ein erhebliches Maß an "Server"-Funktionalitäten zu ermöglichen und damit die Vorteile einer "Peer-to-Peer"-Umgebung nutzbar zu machen.

Den Abschluß des Kapitels 6 bildet eine Betrachtung der konkreten Umsetzung des XML-RPC-Kommunikationsmechanismus, wobei insbesondere auf die Erweiterungen des Standardumfanges der verwendeten XML-RPC-Softwarebibliothek eingegangen wird, die erforderlich waren, um die Übertragung der für die E-Learning-Plattform entwickelten speziellen Datenstrukturen, insbesondere der in Kapitel 5 vorgestellten Virtuellen Lernobjekte, zu ermöglichen.

Während sich die bisherigen Kapitel im Wesentlichen mit Mechanismen beschäftigten, die quasi "unter der Oberfläche" stattfinden und für den Endbenutzer[10] am Bildschirm nicht direkt sichtbar sind bzw. von diesem nicht direkt wahrgenommen werden, wird in Kapitel 7 mit dem Navigator eine Teilanwendung der E-Learning-Plattform vorgestellt, die dem Benutzer sowohl eine Sicht auf die gespeicherten Daten bzw. Metadaten als auch im Rahmen der für ihn gültigen Berechtigungen die Kontrolle über den Datentransfer zwischen verschiedenen Installationen des Systems ermöglicht.

Der Navigator dient dabei auch als ein Beispiel für eine mögliche Teilanwendung, die die von den Kernkomponenten des Systems zur Verfügung gestellten Funktionalitäten wie beispielsweise die XML-RPC-Kommunikation benutzt und über diese gemeinsame Kommunikationsschnittstelle mit anderen Anwendungen (hier in erster Linie mit der Datenbank-Applikation) kommuniziert. Im Anschluß daran wird in Kapitel 8 dargelegt, auf welche Weise weitere Komponenten, beispielsweise Simulationsprogramme für andere Fachbereiche, analog zu den im Rahmen des Projektes Monist entwickelten Anwendungen als neue Teilapplikationen in die E-Learning-Plattform eingebunden werden können.

Kapitel 9 gibt einen Ausblick auf mögliche Erweiterungen der E-Learning-Plattform, die durch die Einbeziehung zusätzlicher Online-Funktionalitäten den Nutzen für den Einsatz in der Lehre noch verstärken würden. Dabei handelt es sich einerseits um eine Kommunikationskomponente, andererseits um ein Konzept für die Nutzung der bereits bestehenden XML-RPC-Kommunikation für Tele-Teaching-Funktionalitäten. Während im Rahmen des Projektes Monist tatsächlich ein Kommunikationssystem für synchrone und asynchrone Kommunikation zwischen Lehrenden und Lernenden entwickelt wurde, das jedoch nicht Teil der vorliegenden Arbeit ist, wurde die praktische Umsetzung des geplanten Tele-Teaching-Systems mit Hinblick auf die Kürze der Projektlaufzeit nicht mehr in Angriff genommen.

In Kapitel 10 werden die in dieser Arbeit gewonnenen Erkenntnisse noch einmal kurz zusammengefaßt. Dabei wird abschließend auch auf die Rolle der E-Learning-Plattform in Abhängigkeit von darauf aufbauender Software und den benötigten dynamischen und multimedialen Inhalten eingegangen. Ein E-Learning-System, das als Grundlage für die Nutzung durch andere Programme gedacht ist, kann niemals allein stehen, sondern muß immer im gesamten Kontext der damit durchzuführenden computerunterstützten Lehre betrachtet werden.

---

[10] Selbstverständlich sind damit auch weibliche Endbenutzerinnen gemeint. Aus Gründen der besseren Lesbarkeit wird im Rahmen dieser Arbeit auf die fortgesetzte Verwendung getrennter Bezeichnungen für Personen beider Geschlechter verzichtet. Mit der verwendeten Form sind jedoch stets Menschen beiderlei Geschlechts gemeint, solange nicht explizit eine geschlechtsspezifische Unterscheidung vorgenommen wird. Die entworfene E-Learning-Plattform selbst macht (insbesondere auch in der Benutzerverwaltung) standardmäßig keinen Unterschied zwischen Benutzern unterschiedlichen Geschlechts und ist aus technischer Sicht für die Benutzung sowohl durch Frauen als auch durch Männer gleichermaßen geeignet. Zukünftigen vielleicht noch zu entwickelnden Applikationen, die die E-Learning-Plattform benutzen, ist es jedoch selbstverständlich freigestellt, bei Bedarf entsprechende Unterscheidungen zu treffen.

2 Anforderungen an die E-Learning-Plattform

Um eine generische E-Learning-Plattform für interaktive Lehrsimulationen zu schaffen, die es ermöglicht, unterschiedliche Simulationsprogramme für den jeweils betrachteten Fachbereich zum Einsatz zu bringen, ist es erforderlich, einerseits eine Infrastruktur als allgemeine Basis für den Einsatz verschiedener Lernsoftware zur Verfügung zu stellen und andererseits die erforderlichen Schnittstellen zu schaffen, die es ermöglichen, diese Programme in das bestehende System zu integrieren. Während die Schnittstellen sich im Wesentlichen aus der konkreten technischen Umsetzung ergeben[11], können für die Infrastruktur eine Reihe von allgemeinen Anforderungen formuliert werden, die sich direkt aus dem geplanten Einsatzfeld in der Lehre ergeben. Diese Vorbedingungen für die E-Learning-Plattform werden im Folgenden dargelegt.

2.1 Speicherung und Benutzung multimedialer Daten

Die technische Möglichkeit zur Speicherung und Verwaltung multimedialer Daten ist selbstverständlich eine Grundvoraussetzung für jedes elektronische System, das für den Umgang mit derartigen Daten vorgesehen ist. Deshalb muß notwendigerweise auch die E-Learning-Plattform über derartige Fähigkeiten verfügen. Lehrsimulationen können nicht als isolierte Computerprogramme betrachtet werden, die keinerlei weitere Medien zu ihrer Ausführung benötigen. Vielmehr ist für einen wirkungsvollen Einsatz in der Lehre die Verwendung unterschiedlichster Medien sowohl innerhalb als auch außerhalb der Simulationen erforderlich.

Dies betrifft einerseits Mediendaten, die innerhalb einer Simulation verwendet werden, andererseits aber auch Begleitmaterialien, Gebrauchsanweisungen, vorbereitende und nachbereitende Online-Kurse, Skripte, Übungsaufgaben usw. Im Projekt Monist werden beispielsweise neben verschiedenen statischen Medien wie Texten, Tabellen, Grafiken oder HTML-Seiten auch Audio- und Video-Daten verwendet, sowohl als Bestandteile von Simulationen als auch im Rahmen begleitender bzw. die Simulationen selbst als Bestandteil enthaltender multimedialer Kurse. Diese Lehreinheiten können wiederum auch selbst als multimediale Objekte betrachtet werden.

Ein weiterer Aspekt der Speicherung ist die einfache und effiziente Benutzung der gespeicherten Daten. Zum einen müssen die Daten leicht zugänglich sein, zum anderen spielt aber auch die Möglichkeit zur Wiederverwendung einmal gespeicherter Daten in einem anderen Kontext als ihrer ursprünglichen Verwendung eine große Rolle. Dabei ist der geringere Speicherplatz, der durch die Vermeidung einer mehrfachen, redundanten Speicherung der gleichen Daten benötigt wird, bei größeren Datenmengen trotz der immer größer werdenden Speicherkapazität der Datenträger auch heute noch von einer nicht zu unterschätzenden Bedeutung. Insbesondere in Fällen, in denen die selben Daten unnötigerweise mehrfach über das Internet übertragen werden, stellt überflüssige Redundanz gerade für private Nutzer (z. B. für Studenten, die Lehreinheiten zu Hause abarbeiten möchten) ein lästiges und unter Umständen auch teures Problem dar.

[11] Beispielsweise folgt aus der Verwendung des in Kapitel 6 beschriebenen XML-RPC-Kommunikationsmechanismus, daß einzubindende Simulationsprogramme über diese Schnittstelle mit dem Rest des Systems kommunizieren können müssen. Würde statt dessen ein E-Learning-System verwendet, das über Mechanismen wie z. B. CORBA elektronische Objekte über das Internet verschicken würde, ergäbe sich daraus eine andere technische Schnittstelle, die ebenfalls von allen Programmen implementiert werden müßte, die in das System integriert werden sollen.

Noch wichtiger allerdings ist die Wiederverwendbarkeit von multimedialen Daten für Autoren, die beispielsweise Kurse für die universitäre Lehre erstellen. Wenn bei Bedarf die einmal erstellten und danach im System abgelegten Medien wiederverwendet und neu kombiniert werden können, anstatt Skripte, Filme, Lehrsimulationen oder ganze Online-Kurse jedes Mal von Grund auf erneut erstellen zu müssen, erspart dies den Autoren eine Menge unnötiger Arbeit und ermöglicht eine erheblich effizientere Vorbereitung der computerunterstützten Lehre.

2.2 Verwendung von Metadaten zur Klassifizierung von Daten

Die Möglichkeit zur Speicherung beliebiger Daten ermöglicht für sich allein genommen noch nicht deren optimale Nutzung. Vielmehr stellt sich für die E-Learning-Plattform wie für alle elektronischen Systeme, die mit potentiell großen Datenbeständen arbeiten, das Problem, wie die vorhandenen Daten systematisch abgelegt und klassifiziert werden können, um sie bei Bedarf möglichst leicht wieder auffinden und verwenden zu können. Dieser Vorgang sollte gleichzeitig für die Benutzer des Systems möglichst komfortabel gestaltet sein, um sowohl Autoren als auch Lernenden einen einfachen Zugriff auf die für sie jeweils relevanten Inhalte zu ermöglichen.

Daß eine umständliche manuelle Suche in Verzeichnisstrukturen, in denen die Daten abgelegt sind, diese Bedingung nicht erfüllt, versteht sich von selbst. Von den Autoren redaktionell erstellte Indizes bieten in dieser Hinsicht zwar eine gewisse Verbesserung, sind jedoch sehr wartungsintensiv und bieten oftmals auch nur eine eingeschränkte Sicht der Dinge, da eine Klassifizierung und Auflistung entweder nur nach wenigen ausgewählten Gesichtspunkten erfolgen kann oder aber zu sehr unübersichtlichen Listen führt, so daß der Aufwand sowohl für eine Suche nach bestehenden Daten als auch für das Eintragen neuer Objekte wiederum erheblich zunimmt.[12]

Auch der Nutzen von Systemen zur Volltextsuche ist beschränkt. Mehrdeutigkeiten, kontextabhängige Wortbedeutungen, das Fehlen von beschreibenden Informationen über den Text im Wortlaut des Textes selbst[13] und nicht zuletzt die beispielsweise von Internet-Suchmaschinen bekannten riesigen Ergebnismengen, die ein menschlicher Benutzer oftmals nicht mit einem akzeptablen Zeitaufwand durchlesen kann, reduzieren den Wert einer Volltextsuche zur Wiederauffindung bestimmter Informationen bereits

[12] Zur Veranschaulichung diene an dieser Stelle das folgende Beispiel, das analog auf elektronische Datenbestände übertragen werden kann: Vor der Umstellung auf ein elektronisches Verwaltungssystem, das die vorhandenen Buchbestände über zahlreiche Metadaten erschließt, waren in der Stadtbibliothek der Stadt Bielefeld (wie in den meisten anderen Bibliotheken auch) alle Bücher thematisch in bestimmte Regale einsortiert (die hier den Verzeichnissen entsprechen) und daneben über ein für den menschlichen Benutzer nur schwer zu durchschauendes System von aus Buchstaben und Zahlen bestehenden Schlüsseln klassifiziert. (In elektronischen Systemen würde dies beispielsweise der Anordnung der Verweise auf einzelne Dateien im Verzeichnissystem auf einer Festplatte entsprechen.)
Außerdem gab es verschiedene Karteikartensysteme, mit denen man nach unterschiedlichen Kriterien nach bestimmten Büchern suchen konnte. Diese Indizes mußten mit hohem Aufwand manuell gepflegt werden und waren nur von begrenztem Nutzen. Verwendete man nur einen einzigen Index, beispielsweise die Titel-Kartei, so bot dieser nur relativ beschränkte Suchmöglichkeiten. Zog man hingegen parallel mehrere der vorhandenen Indizes zu Rate, beispielsweise zusätzlich die Autoren-Kartei, gestaltete sich die gleichzeitige Suche in mehreren Karteien schnell sehr unübersichtlich und verursachte oftmals einen größeren Aufwand, als die Bücher gleich direkt in den Regalen zu suchen. Die parallele Verwendung mehrerer elektronischer Indizes führt zu ähnlichen Problemen, wenngleich hier natürlich das manuelle Durchblättern der Karteikarten entfällt.

[13] Zwei einfache Beispiele sind Dateiformat und Dateigröße, die nur relativ selten im eigentlichen Inhalt von Dateien enthalten sind.

bei einer ausschließlichen Beschränkung auf Textdateien erheblich. Abgesehen davon ist eine Volltextsuche für Daten, die nicht in textueller Form vorliegen, natürlich ohne großen Nutzen. Da in einer multimedialen Umgebung jedoch auch zahlreiche andere Dateiformate verwendet werden, beispielsweise Grafiken, Videos oder Audio-Dateien, ist eine Volltextsuche denkbar ungeeignet und scheidet als mögliches Suchwerkzeug für die E-Learning-Plattform aus.

Als weitverbreitete Lösung für das Problem der Klassifizierung und Wiederauffindung von Daten hat sich inzwischen die Verwendung von Metadaten durchgesetzt, die quasi als "Daten über Daten" zusätzliche beschreibende Informationen zu jeder einzelnen Datei enthalten. Diese Metadaten werden beim Einstellen von Dateien ins System durch den Benutzer mit angegeben oder teilweise auch automatisch erzeugt und dienen fortan der Verwaltung der gespeicherten Daten. Das Spektrum der Metadaten reicht dabei von einfachen Verwaltungsinformationen wie z. B. dem Titel eines Dokumentes oder dem Namen seines Autors[14] über technische Informationen[15] bis hin zur Einordnung in ein fachspezifisches Klassifikationssystem für das jeweilige Anwendungsgebiet[16].

Welche Arten von Metadaten in einem konkreten System verwendet werden, kann nicht allgemeingültig festgelegt werden. Für E-Learning-Systeme gab es in den letzten Jahren zwar mehrfach Bestrebungen, standardisierte Metadatensätze festzulegen, die jedoch in vielen Fällen nicht den Bedürfnissen der einzelnen Lernprogramme gerecht werden.[17] Aus diesem Grund wird an dieser Stelle die Anforderung an die E-Learning-Plattform aufgestellt, ein möglichst breites Spektrum an möglichen Metadaten zu unterstützen und nicht durch die Koppelung an irgendeinen zum Zeitpunkt der Entwicklung des Systems bestehenden Standard andere Möglichkeiten von vornherein auszuschließen.

Unterschiedliche E-Learning-Anwendungen, die auf der im Rahmen der vorliegenden Arbeit entworfenen Plattform basieren, können prinzipiell sehr verschiedene Metadaten verwenden. Gerade im fachspezifischen Bereich ist davon auszugehen, daß die zur Klassifikation von Daten in unterschiedlichen Kontexten benötigten Metadaten sich erheblich unterscheiden.[18] Daraus ergibt sich als weitere Anforderung, die Verwendung unterschiedlicher Metadatensätze durch verschiedene Anwendungen zu ermöglichen. Dies soll durch eine anwendungsspezifische Konfigurationsmöglichkeit der im System zu verwaltenden Metadaten erreicht werden.

Darüber hinaus wird auch die Möglichkeit zur Angabe verschiedener Metadatensätze für unterschiedliche Datentypen gefordert. Beispielsweise sollen bei Bedarf komplette, aus vielen verschiedenen Teilobjekten zusammengesetzte Online-Kurse auch durch einen veränderten Metadatensatz von einfachen "atomaren" Dateien wie z. B. Texten oder Grafiken unterschieden werden können.

---

[14] Also von Informationen, wie sie bereits in traditionellen Karteien von Bibliotheken verwendet wurden.
[15] Beispielsweise Dateigröße oder MIME Type.
[16] In einem biologischen Kontext könnte dies beispielsweise eine Einordnung eines in einem Fachtext betrachteten Organismus in biologische Taxonomien sein.
[17] Eine nähere Betrachtung der Metadaten-Problematik folgt in Kapitel 5. Dort werden unter anderem auch verschiedene Metadaten-Standards vorgestellt und am Beispiel der im Rahmen des Projektes Monist verwendeten Daten auf ihre Anwendbarkeit überprüft.
[18] Beispielsweise ist davon auszugehen, daß Simulationsprogramme aus so verschiedenen Fachbereichen wie Neurobiologie und Laserphysik unterschiedliche Metadaten zur fachlichen Beschreibung einzelner Lehreinheiten verwenden würden. Derartige Unterschiede sind jedoch keineswegs auf die fachlichen Metadaten beschränkt, sondern können auch technische oder verwaltungsbedingte Metadaten betreffen. So könnte beispielsweise eine bestimmte E-Learning-Anwendung auf jegliche Informationen über die Größe oder das Erstellungsdatum einer Datei verzichten, während eine andere genau diese Metadaten dringend benötigt.

Der Grund dafür ist, daß für solche zusammengesetzte Objekte vielfach ganz andere Klassifizierungen getroffen werden können als für ihre einzelnen Bestandteile. Oftmals gewinnt z. B. eine bestimmte Grafik erst in einem umfassenden fachlichen Kontext eine bestimmte Bedeutung, die eine Klassifikation und die Einordnung in einen bestimmten Bereich des betrachteten Studienfaches erfordert. Lehrspezifische oder lernspezifische Aspekte wie z. B. die Bearbeitungsdauer einer umfassenden Simulation sind vielfach erst bei vollständigen, aus mehreren einzelnen Medien zusammengesetzten Kursen von Belang, während die für einzelne Bestandteile angegebenen technischen Informationen wie z. B. der MIME Type nur bei einer isolierten Betrachtung der jeweiligen Dateien benötigt werden könnten.

Ein weiterer Aspekt der Verwendung unterschiedlicher Metadatensätze für verschiedene Datentypen ergibt sich, wenn auch die Benutzer des Systems intern durch Objekte repräsentiert werden. Dies ist im Hinblick auf die beabsichtigte Funktionalität der zu schaffenden E-Learning-Plattform ausdrücklich erwünscht, da die Integration einer Benutzerverwaltung in das System ebenfalls zu den Anforderungen zählt[19] und durch die Betrachtung der Benutzer als eine besondere Art von Objekten die Notwendigkeit zur Schaffung einer zusätzlichen Verwaltungsmethode entfällt. Selbstverständlich sind für die Verwaltung der Benutzer wiederum vollkommen andere Daten bzw. Metadaten erforderlich als für Lehrsimulationen oder die darin verwendeten multimedialen Daten. Dies ist ein weiteres Argument dafür, für verschiedene Datentypen eine unterschiedliche Konfiguration der dafür jeweils zu verwaltenden Metadatensätze zu ermöglichen.

2.3 Lehreinheiten als Konfigurationen dynamischer Programme

Bislang wurden für die E-Learning-Plattform Anforderungen aufgestellt, die darauf abzielten, einzelne Dateien, aus denen sich Lehreinheiten zusammensetzen, abspeichern und auf eine geeignete Weise wieder auffinden und verwenden zu können. Dies wirft allerdings die Frage auf, worum es sich bei diesen übergeordneten, zusammengesetzten Einheiten überhaupt handelt. Die Art und Weise der Zusammensetzung von einzelnen Elemente zu einem größeren Ganzen ist für das zu konstruierende E-Learning-System von entscheidender Bedeutung.

Wie bereits zu Beginn festgestellt, ist es das Ziel dieser Arbeit, eine Plattform zu schaffen, in die verschiedene Simulationsprogramme für unterschiedliche Fachbereiche in gleicher Weise eingebunden werden können. Verschiedene Programme werden dabei aber die benötigten Daten nicht notwendigerweise auf die gleiche Weise verwenden. Als kleinsten gemeinsamen Nenner kann man jedoch die Feststellung treffen, daß man einzelne Lehreinheiten als unterschiedliche Konfigurationen betrachten kann, mit denen die verwendeten Programme aufgerufen werden.

Natürlich hat es in der Geschichte der Lernsoftware auch viele einzeln stehende, quasi "monolithische" Programme gegeben, die zwar eine teilweise beträchtliche Anzahl von vorgegebenen Simulationen enthielten, jedoch entweder nur mit großem Aufwand oder sogar überhaupt nicht erweiterbar waren.[20] Solche Programme, die es voraussichtlich auch in Zukunft weiterhin geben wird, haben für einen beschränkten Einsatzbereich, in

[19] Siehe Kapitel 2.4.

[20] Zwei Beispiele für derartige Programme aus dem Bereich der Laborsimulations-Software wurden vom Autor der vorliegenden Arbeit beschrieben in: Andreas Dieckmann, Basisfunktionen für ein generisches virtuelles Labor, Diplomarbeit an der Technischen Fakultät der Universität Bielefeld, 1998, Seiten 12-17. Die fehlende Erweiterbarkeit bestehender Programme war die Motivation für die genannte Diplomarbeit.

dem eine Erweiterung des Spektrums der enthaltenen Simulationen durch das System benutzende Lehrende oder Autoren nicht vorgesehen ist, durchaus ihre Berechtigung.

Für einen "wachsenden" Themenbereich, in dem kontinuierlich neue fachliche Inhalte zur Verfügung gestellt werden können, sind derartige Programme jedoch nur begrenzt geeignet. Zwar könnten nach und nach verschiedene unveränderliche Programme in die Lehre integriert werden, dies gibt jedoch den Lehrenden nicht die Möglichkeit, eigene Inhalte einzubringen und den Benutzern zur Verfügung zu stellen, es sei denn, daß jedes Mal eine eigene Software für diesen Zweck erstellt wird. Es liegt auf der Hand, daß eine solche Vorgehensweise mit einem hohen Aufwand verbunden ist, der in den meisten Fällen gar nicht erst geleistet werden kann oder aber zumindest einen zeitnahen Einsatz neuer Erkenntnisse in der Lehre aufgrund der einzuplanenden Zeiten und Kosten der Software-Entwicklung schwer bis unmöglich macht.

Abhilfe für dieses Problem schafft nur die Entwicklung umfassender, konfigurierbarer Softwarekomponenten, die es ermöglichen, mit einem geringen Aufwand und unter Verwendung bereits bestehender Programmteile und Medien neue Simulationen und darauf aufbauende Kurse für die elektronische Lehre zu erzeugen.[21] Der E-Learning-Plattform kommt dabei die Rolle zu, die einzelnen Bestandteile zu archivieren und den Benutzern in einer möglichst einfachen Art und Weise zur Verfügung zu stellen. Ebenso müssen komplette Lehreinheiten gespeichert und zugänglich gemacht werden.

Die E-Learning-Plattform hat dabei auch die Aufgabe, beim Aufruf einer Datei, die eine Konfiguration für eine bestimmte dynamische Simulationssoftware enthält, zu erkennen, welches die zuständige Softwarekomponente für diese Konfigurationsdatei ist, und diese Applikation dann in geeigneter Weise aufzurufen, um die Darstellung der vom Benutzer ausgewählten Konfiguration zu gewährleisten. Die Plattform muß also über einen Mechanismus verfügen, um unterschiedliche Typen von Konfigurationen zu verwalten und diese den entsprechenden Simulationsprogrammen zuzuordnen. Dabei kann es wiederum erforderlich sein, für die verschiedenen Dateitypen die Verwaltung von unterschiedliche Metadatensätzen zu ermöglichen.

Für das interne Format der von den einzelnen Simulationsprogrammen verwendeten Konfigurationsdateien sollen an dieser Stelle keine Einschränkungen gemacht werden, um ein möglichst breites Spektrum an verschiedenen Formaten benutzen zu können. Da die Handhabung der Inhalte dieser Dateien nur den jeweiligen Programmen obliegt und die E-Learning-Plattform lediglich für die Zuordnung und den korrekten Aufruf zuständig ist, muß lediglich die Forderung aufgestellt werden, daß alle Programme, die im System eingesetzt werden, eindeutig voneinander zu unterscheidende Dateitypen für ihre Konfigurationen verwenden.

Dies muß jedoch nicht notwendigerweise bedeuten, daß die Konfigurationsdateien auch intern vollkommen unterschiedliche Datenformate benutzen. Es ist sogar allgemein empfehlenswert, daß Konfigurationen in für die jeweilige Anwendung spezifischen XML-Formaten abgelegt werden, da XML ein weltweit anerkannter Standard zur

[21] Exemplarisch für den Bereich der Laborsimulations-Software wurden die Vorteile solcher generischer Ansätze aufgezeigt in: Andreas Dieckmann, Basisfunktionen für ein generisches virtuelles Labor, Diplomarbeit an der Technischen Fakultät der Universität Bielefeld, 1998. Zur Ersparnis von Zeit und Kosten siehe dabei insbesondere die Betrachtungen des erheblich geringeren Aufwandes zur Erstellung von Simulationen bei der Nutzung generischer Ansätze in Kapitel 5.4, Seiten 85-87, und Kapitel 7.1, Seite 107. Die für diesen Bereich gewonnen Erkenntnisse können auf andere Fachbereiche übertragen werden, da die beschriebenen Vorteile rein technischer Natur sind.

Speicherung von Daten in festgelegten Formaten ist und damit auch gut dazu geeignet ist, Informationen über bestimmte Konfigurationen strukturiert abzulegen. Diese XML-Dateien können dann im Rahmen der jeweiligen Simulationsanwendung unter Verwendung gebräuchlicher XML-Tools editiert, verifiziert und effizient eingelesen werden. Auch eine manuelle Bearbeitung der XML-Dateien ist bei Benutzung eines für den menschlichen Benutzer verständlichen Formates denkbar. Lediglich nach außen hin muß eine Unterscheidbarkeit der Dateien gegeben sein, die von unterschiedlichen Anwendungen benutzt werden. Dies kann und soll die E-Learning-Plattform dadurch unterstützen, daß unabhängig vom Dateiinhalt unterschiedliche Typen definiert und bestimmten Anwendungen zugeordnet werden können.

2.4 Benutzerverwaltung und Zugriffsrechte

Eine weitere wichtige Anforderung an die E-Learning-Plattform ist die Möglichkeit der Verwaltung von Benutzern. Derartige Funktionalitäten, die heutzutage in den meisten Systemen vorhanden sind, die von vielen Benutzern verwendet werden, sind auch für ein elektronisches Lernsystem von Bedeutung. Das Spektrum der Einsatzmöglichkeiten reicht dabei von der einfachen Regelung der Frage, welche Personen auf bestimmte Lehreinheiten Zugriff haben und welche nicht, über die Einteilung von Lernenden in bestimmte Gruppen gemäß ihres Studienfortschrittes und ihrer Teilnahme an gewissen Veranstaltungen bis hin zur personifizierten Präsentation bestimmter Lehrinhalte.

Daneben ermöglicht erst eine funktionierende Benutzerverwaltung einen sinnvollen Einsatz von kommunikativen Funktionalitäten eines E-Learning-Systems in der Lehre, wie z. B. Chat und E-Mail. Die Gestaltung personalisierter Fragebögen wird ebenso ermöglicht wie elektronische Tests bis hin zu Online-Prüfungen. Durch die Sammlung von benutzerspezifischen Daten könnte eine Modellierung des persönlichen Lernweges erfolgen, die es ermöglicht, individuell auf Stärken und Schwächen einzugehen und je nach Lernverhalten einzelnen Personen unterschiedliche Lernwege zu präsentieren.[22]

Das wichtigste Einsatzgebiet für eine Benutzerverwaltung ist jedoch die Kontrolle über die Zugriffsrechte auf einzelne Dateien, die im System verwaltet werden. Einerseits kann der lesende Zugriff auf bestimmte Daten beschränkt werden. Dies kann aus verschiedenen Gründen erwünscht sein. Beispielsweise kann der Zugriff auf gewisse Lehreinheiten auf die Teilnehmer einer bestimmten Veranstaltung begrenzt werden. Ebenso können die Lösungen von Übungsaufgaben vom Autor unter Verwendung von eingeschränkten Zugriffsrechten zu einer beliebigen Zeit in das System eingestellt, aber erst zu einem bestimmten Zeitpunkt für die Allgemeinheit freigegeben werden. In ähnlicher Weise könnte das Recht zum Einsehen von Protokollen oder von Lösungen zu bestimmten Übungsaufgaben auf den jeweiligen Lernenden und seinen betreuenden Tutor beschränkt werden.

Andererseits ist natürlich auch die Frage von Bedeutung, wer im System über das Recht zum schreibenden Zugriff auf welche Dateien verfügt. Es ist unmittelbar einsichtig, daß auf Lehrmaterialien, die die offizielle Grundlage eines Online-Kurses bilden, nicht jeder teilnehmende Student einen Schreibzugriff haben darf. Ebenfalls sollte natürlich auch nicht jeder Student in den persönlichen Kursunterlagen, Notizen, Übungsaufgaben und Protokollen seiner Kommilitonen herumschreiben dürfen.

[22] Selbstverständlich sind bei einem möglichen Einsatz derartiger Verfahren zur Lernwegkontrolle die für die Speicherung solcher persönlicher Daten relevanten Datenschutzbestimmungen zu beachten.

Darüber hinaus spielen aber auch die Koordination zwischen einzelnen Autoren sowie das Urheberrecht eine Rolle. Auch wenn Lernende bereits generell vom schreibenden Zugriff ausgeschlossen wurden, ist es nicht unbedingt immer wünschenswert, wenn jeder Lehrende unabhängig vom anderen die Kursunterlagen beliebig verändern kann. Das Anlegen von neuen Lehreinheiten sollte ebenfalls berechtigten Autoren vorbehalten sein, während es im System durchaus Bereiche geben kann, in denen jeder Lernende für sich persönlich beliebige Dateien ablegen und bearbeiten kann, ohne daß diese für die Allgemeinheit zugänglich sind.

Wie schon anhand der Schreib- und Leserechte deutlich wird, kann es sinnvoll sein, bestimmte Funktionalitäten des Systems nur ausgewählten Benutzern zugänglich zu machen. Beispielsweise sollte die Möglichkeit zum Einsehen persönlicher Lernwege einzelner Studenten nur den die betreffende Veranstaltung organisierenden Lehrkräften zugänglich sein. Funktionen zum Editieren von Simulationen und Kursen sollten von vornherein den dazu berechtigten Autoren und Lehrenden vorbehalten bleiben. Auch derartige Einschränkungen der zugänglichen Funktionalitäten des Systems lassen sich mit Hilfe einer Benutzerverwaltung realisieren.

Natürlich wäre es äußerst umständlich, bei jeder derartigen Einschränkung von Rechten stets für jeden einzelnen Benutzer festzulegen, ob diesem ein bestimmtes Recht gewährt wird oder nicht. Deshalb ist es eine sehr nützliche Funktionalität, Benutzer zu Gruppen zusammenzufassen, für die dann wiederum gesonderte Zugriffsrechte vergeben werden können. Einzelne Gruppen können bei Bedarf wiederum mit weiteren Benutzern zu neuen Gruppen zusammengefaßt werden, für welche dann zusätzlich wieder bestimmte andere Zugriffsrechte gelten. Auf diese Weise läßt sich bei größeren Benutzerzahlen die Übersicht über die vergebenen Rechte erhalten.

Ein letzter Aspekt der Benutzerverwaltung ist die Tatsache, daß natürlich auch nicht jeder Benutzer nach Belieben die Möglichkeit haben darf, Zugriffsrechte für Objekte zu vergeben oder andere Benutzer bestimmten Gruppen zuzuordnen oder ihnen Rechte auf fremde Objekte zu erteilen. Zu diesem Zweck muß es eine Gruppe von Benutzern mit bestimmten administrativen Rechten – im Folgenden Administratoren genannt – geben, die exklusiv über die Berechtigung verfügen, Benutzer anzulegen und zu bearbeiten, diese zu Gruppen hinzuzufügen oder aus diesen wieder zu entfernen sowie allgemeine Zugriffsrechte auf bestimmte Teilbereiche des Systems sowie bestimmte Typen von Objekten festzulegen.

Ein detailliertes System zur Verwaltung von Benutzern und Zugriffsrechten ist also von erheblicher Bedeutung für die E-Learning-Plattform, gleichermaßen aus technischer Hinsicht wie aus Gründen der Lernwegkontrolle und der Sicherstellung der Rechte der Autoren an den von ihnen ins System eingebrachten Lehrmaterialien. Je flexibler die von einem solchen System zur Verfügung gestellten Möglichkeiten sind, desto besser kann die E-Learning-Plattform auf die Bedürfnisse der computerunterstützten Lehre im Allgemeinen und auf die der einzelnen Lehrenden und Lernenden eingestellt werden.

Außerdem bietet die durch ein solches System erzielte Personalisierung des Systems den nicht zu unterschätzenden Effekt, daß dadurch die in elektronischen Systemen oft auftretende Unpersönlichkeit erheblich abgemildert wird. Dem Benutzer kann auf diese Weise ein Eindruck von Individualität vermittelt werden, der dem in der herkömmlichen Lehre nahekommt oder diesen zumindest imitiert. Dies ermöglicht eine persönlichere Interaktion zwischen den Benutzern in kommunikativen Teilen des Systems wie z. B. Chaträumen und verbessert damit auch die Akzeptanz des Systems durch die Benutzer.

2.5 Mehrsprachigkeit von Inhalten und Benutzeroberfläche

In vielen Fachbereichen ist die Lehre heutzutage nicht mehr auf nationale Grenzen beschränkt, sondern findet in einem internationalen Kontext statt. Auch und gerade für elektronische Lernsysteme, die aufgrund ihrer einfachen Verteilbarkeit und ihrer zumindest potentiell großen Unabhängigkeit von Präsenzlehre an festgelegten Standorten[23] einen internationalen Einsatz ermöglichen, ist daher eine möglichst unkomplizierte Internationalisierung von entscheidender Bedeutung. Mehrsprachigkeit ermöglicht es einem Programm, im wörtlichen Sinn "Grenzen zu überwinden", oder vereinfacht diesen Vorgang zumindest erheblich.

Die Übersetzung von Menüs und anderen Bedienungselementen in mehrere Sprachen ermöglicht es fremdsprachigen Benutzern, sich ganz auf die Bedienung des Programms zu konzentrieren, anstatt Zeit und Energie auf die Überwindung der Sprachbarriere zu verwenden.[24] Dies betrifft jedoch natürlich nicht nur die Benutzeroberfläche, sondern erst recht die durch das System zu vermittelnden Inhalte. Daher sollte das System über die Möglichkeit verfügen, die gleichen Medien und Lehreinheiten in unterschiedlichen Sprachen nebeneinander zur Verfügung zu stellen.

In Abhängigkeit von der für den aktuellen Benutzer des Systems eingestellten Sprache sollen diesem dann sowohl die Bedienungselemente des Programms als auch die Inhalte in dieser Sprache präsentiert werden, sofern die gewünschten Inhalte in seiner Sprache auch tatsächlich vorhanden sind. Anderenfalls kann entweder eine als Standard voreingestellte Sprache ausgewählt werden oder die Auswahl dem Benutzer überlassen werden. Auch bei Inhalten, die lediglich in einer einzigen Version vorliegen, spielt die Information darüber, um welche Sprache es sich dabei handelt, eine wichtige Rolle, um die Lernenden gleichermaßen wie die Lehrenden und Autoren vorab zu informieren, welche Sprachkenntnisse für die Bearbeitung erforderlich sind.

Die Forderung nach Mehrsprachigkeit gilt allerdings nicht nur für die Inhalte, sondern auch für die Metadaten, die zu ihrer Verwaltung verwendet werden. Dazu ist es nötig, zumindest textuelle Metadaten wie z. B. Titel oder Stichwörter in mehreren Sprachen im System ablegen zu können. Auch das zur fachlichen Klassifikation verwendete Vokabular ist von Sprache zu Sprache unterschiedlich, weshalb eine mehrsprachige Ablage von Metadaten unerläßlich ist. Darüber hinaus können sich auch die technischen Metadaten von Sprache zu Sprache unterscheiden, wenn beispielsweise für verschiedene Sprachen Dateien unterschiedlicher Größe im System abgelegt werden.

---

[23] Die aktuellen Entwicklungen im E-Learning-Bereich machen allerdings deutlich, daß auf denjenigen Teil der Lehre, der in Form von Präsenzlehre und mit persönlichem Kontakt zwischen den Lehrenden und den Lernenden stattfindet, nur schwer verzichtet werden kann, da darunter die Qualität der Lehre leidet. Auf Fachmessen wie z. B. der LEARNTEC 2003 in Karlsruhe gewinnt deswegen immer mehr der Begriff des sogenannten *Blended Learning* an Bedeutung. Dahinter verbirgt sich nichts anderes als die Kombination und Integration von Präsenzlehre und elektronischer Lehre. Im Rahmen des Projektes Monist und bei der Entwicklung der E-Learning-Plattform war es von vornherein ein Ziel, ein System zu schaffen, das nicht nur zum individuellen Selbststudium geeignet ist, sondern genauso auch zur Unterstützung von anderen Lehrformen wie beispielsweise computerunterstützten Vorlesungen oder in der Gruppe am Computer durchgeführten Praktika, Übungen und Seminaren.

[24] Daß sich dadurch auch die Akzeptanz eines solchen Softwaresystems bei internationalen Benutzern erhöhen kann, wenn eine Benutzeroberfläche in der jeweiligen Muttersprache vorhanden ist, liegt auf der Hand, soll hier jedoch nicht weiter untersucht werden. Statt dessen sei lediglich auf die weite Verbreitung von lokalisierten Versionen der meisten Standardprogramme verwiesen, die insbesondere im Fall von Microsoft®-Windows®-basierten Systemen vom Betriebssystem selbst über Textverarbeitungsprogramme, betriebswirtschaftliche Anwendungsprogramme und Software-Entwicklungsumgebungen bis hin zu den meisten Computerspielen reicht.

2.6 Unabhängigkeit von Hardware und Betriebssystem

Um eine weite und dabei möglichst einfache Verteilbarkeit des E-Learning-Systems zu erreichen, wird als weitere Anforderung eine möglichst weitgehende Unabhängigkeit von der verwendeten Hardware und dem Betriebssystem gefordert, auf dem das System zum Einsatz gebracht wird. Dadurch soll die Einsetzbarkeit in einem in dieser Hinsicht heterogenen Umfeld gewährleistet werden. Während an Universitäten beispielsweise in einigen Fachbereichen in erheblichem Umfang UNIX Workstations im Einsatz sind, ist die EDV anderer Einrichtungen komplett auf PC-Systemen und Netzwerken aufgebaut, die auf unterschiedlichen Versionen von Microsoft® Windows® basieren.

Die Studenten benutzen auf ihren privaten Computern teilweise ebenfalls Microsoft® Windows®, während andere die unterschiedlichsten Versionen von Linux verwenden. Daneben gibt es noch verschiedene andere Betriebssysteme. Da es äußerst umständlich wäre, für jedes existierende oder zukünftig noch zu entwerfende Betriebssystem sowie für die jeweils verwendete Hardware stets eine eigene Version der E-Learning-Plattform zu erschaffen und diese unterschiedlichen Varianten dann auch noch mit erheblichem Aufwand interoperabel zu halten, stellt die Plattformunabhängigkeit[25] des in dieser Arbeit zu entwerfenden Systems eine Kernanforderung dar.

Die Forderung nach der Plattformunabhängigkeit legte übrigens bereits frühzeitig eine Implementation des E-Learning-Systems in der Programmiersprache Java nahe, da Java von der Firma Sun Microsystems mit der Zielsetzung entwickelt wurde, daß in dieser Sprache geschriebene Programme ohne Umschreiben oder Neukompilieren des Quellcodes auf jeder Hardware- und Betriebssystem-Plattform verwendbar sein sollten.[26] Die Implementierung in einer bestimmten Programmiersprache kann natürlich nicht als eine Grundlage für den Systementwurf geltend gemacht werden, da die Umsetzung immer erst der zweite Schritt nach der Entwicklung der dafür erforderlichen Konzepte sein kann. Die zuvor aufgestellte Forderung nach der Unabhängigkeit der E-Learning-Plattform von der Hardwarearchitektur und vom Betriebssystem kann jedoch durch die Implementierung in der Programmiersprache Java erreicht werden. Prinzipiell ist diese Forderung demnach also erfüllbar.

Ein weiterer Punkt, der die Unabhängigkeit der E-Learning-Plattform betrifft, soll an dieser Stelle ebenfalls nicht unerwähnt bleiben. Für die Datenspeicherung bietet es sich prinzipiell an, ein bestehendes Datenbanksystem zu verwenden, um in dieser Hinsicht nicht quasi "das Rad neu erfinden" zu müssen. Der Entwurf eines eigenen, vollkommen neuen Datenbanksystems ist keine notwendige Teilaufgabe bei der Entwicklung einer E-Learning-Plattform. Um die Verwendbarkeit der Plattform auf beliebigen Systemen

---

[25] Die zumindest aus sprachlicher Sicht auf den ersten Blick vielleicht paradox erscheinende Forderung nach einer "plattformunabhängigen Plattform" ergibt sich aus der mehrdeutigen und teilweise ungenauen Art und Weise, in der das Wort "Plattform" in unterschiedlichen Kontexten in den Sprachgebrauch in der Informatik Eingang gefunden hat. Legt man jedoch die allen diesen Einsätzen gemeinsame Bedeutung dieses Wortes als "etwas, das als Basis für andere, darauf aufbauende Programme dient" zugrunde, läßt sich der Widerspruch dahingehend auflösen, daß das E-Learning-System vom jeweiligen Betriebssystem abstrahiert und an dessen Stelle selbst zu einer Grundlage für die darauf aufbauenden Applikationen wird, die seine Methoden zur Datenverwaltung nutzen und auch aus ihm heraus aufgerufen werden. Im Idealfall sind diese Programme ihrerseits wiederum von der (Betriebssystem-) Plattform unabhängig und bauen nur auf der Grundlage der hier entworfenen (E-Learning-) Plattform auf, in die sie über bestimmte Schnittstellen integriert sind.

[26] Zur Unabhängigkeit von in Java geschriebenen Programmen von der Hardwarearchitektur und vom Betriebssystem, auf dem sie eingesetzt werden, siehe: Cay S. Horstmann, Gary Cornell, Core Java 1.1, Volume I - Fundamentals, Sun Microsystems Press, Mountain View, California, 1997, Seiten 8-9.

zu ermöglichen, aber auch, um eine Abhängigkeit von einem bestimmten, herstellerspezifischen Datenbanksystem von vornherein zu vermeiden, ist es sinnvoll, für die Datenbankanbindung eine allgemeine Schnittstelle zu schaffen, die in abstrakter Weise die benötigten Datenbankoperationen definiert.

Bei Bedarf kann dann für jede konkrete zu benutzende Datenbank eine Implementation dieser Schnittstelle geschaffen werden, ohne daß man an der eigentlichen E-Learning-Plattform Änderungen vornehmen müßte. Die aus allen Bereichen des Systems an die Datenbankschnittstelle gerichteten, abstrakten Aufrufe werden dann an der Schnittstelle in konkrete Operationen auf der tatsächlich verwendeten Datenbank umgesetzt. Somit können für den gleichen Zweck auf verschiedenen Computern völlig unterschiedliche Datenbanken zum Einsatz gebracht werden. Ebenso könnte man bei Bedarf eine bisher verwendete Datenbank leicht durch eine neue ersetzen.

Für den Programmcode, der im Kernbereich der E-Learning-Plattform auf den Daten operiert, die in diesen Datenbanken gespeichert sind, führt dabei die Verwendung eines anderen Datenbanksystems ebensowenig zu irgendwelchen Veränderungen wie für den menschlichen Benutzer.

2.7 Offline-Einsatz in Selbststudium und Präsenzlehre

Die in diesem und dem nachfolgenden Abschnitt aufgestellten Anforderungen betreffen die Eignung des E-Learning-Systems für die beabsichtigten Einsatzszenarien. Zunächst werden die geplanten Verwendungsmöglichkeiten des Systems ohne eine Verbindung zum Internet aufgezeigt. Der nächste Abschnitt befaßt sich dann mit den zusätzlichen Fähigkeiten, die das System durch das Hinzufügen von Online-Funktionalitäten erhält.

Der Offline-Einsatz der E-Learning-Plattform ist hauptsächlich für die folgenden beiden Anwendungsfälle vorgesehen: einerseits für das Selbststudium ohne Betreuung bzw. für das eigenverantwortliche Bearbeiten elektronischer Lehreinheiten durch die Studenten außerhalb bestimmter Lehrveranstaltungen, andererseits im Rahmen von Präsenzlehre, bei der die im System enthaltenen Lehrmaterialien zur besseren Veranschaulichung des Lehrstoffes verwendet werden.

Im Falle des Selbststudiums ergibt sich für die Lernenden ein hohes Maß an Freiheiten bezüglich der Gestaltung des eigenen Lernverhaltens. So kann unabhängig von einem festen Stundenplan zu beliebigen Zeiten gelernt werden.[27] Auch der Ort des Lernens kann frei bestimmt werden: Für die Studenten zugängliche Computer in universitätseigenen Rechnerräumen kommen dafür ebenso in Frage wie der private PC zu Hause, und die Möglichkeit, das E-Learning-System auch auf tragbaren Computern ohne die Notwendigkeit irgendwelcher Anschlüsse einsetzen zu können, ermöglicht es, praktisch jeden Raum zu einer virtuellen Lernumgebung zu machen und bei Bedarf auch beispielsweise in der Eisenbahn oder sogar in der freien Natur lernen zu können.[28]

[27] Bislang unveröffentlichte Lehrevaluationen aus der ersten Testphase der im Rahmen des Projektes Monist entwickelten Lernsoftware, die auf Basis der in dieser Arbeit entworfenen E-Learning-Plattform eingesetzt wurde, haben gezeigt, daß gerade dieser Aspekt von der Mehrheit der beteiligten Studierenden als ein großer Vorteil des Systems empfunden wurde. Die Veranstaltungsteilnehmer betonten mehrfach, daß trotz dichtgedrängter und von Person zu Person teilweise erheblich differierender Stundenpläne eine Teilnahme an den computerbasierten Kursen ohne nennenswerte Probleme möglich war.

[28] Im Rahmen eines Praktikums in der Testphase des Projektes Monist wurden an Studenten Notebooks ausgegeben. Die Möglichkeit, den Ort des Lernens selbst zu bestimmen, wurde dabei intensiv genutzt.

Die E-Learning-Plattform soll aber keineswegs nur zum Selbststudium eingesetzt werden, sondern kann auch im Rahmen von Lehrveranstaltungen eine Rolle spielen, die von einem Dozenten in Form von Präsenzlehre abgehalten werden. Einerseits können die im System abgelegten Inhalte zur Unterstützung von Vorträgen und Vorlesungen verwendet werden. Beispielsweise kann anhand bestimmter Simulationen die Erklärung dynamischer Vorgänge in einer Weise erfolgen, die unter Verwendung traditioneller Lehrmittel wie Folien oder gar Tafel und Kreide überhaupt nicht möglich wäre. Somit ermöglicht der Einsatz computerbasierter Lehrmaterialien eine Dynamisierung von bis dato eher statischen Lehrformen.

Andererseits ist auch die Möglichkeit vorgesehen, die E-Learning-Plattform im Rahmen von Seminaren oder Praktika zum Einsatz zu bringen. Dabei sollen unter Anleitung und Betreuung eines Dozenten bestimmte Lehreinheiten von den Studenten einzeln oder in Gruppen ausgeführt und die Ergebnisse dann in der Veranstaltung besprochen werden können. Dadurch wird eine bessere Anleitung gegenüber dem reinen Selbststudium und eine direkte Lernwegkontrolle ermöglicht. Durch die somit gegebene Möglichkeit eines persönlichen Eingreifens des Lehrenden wird oft ein besserer Lernerfolg erzielt als durch reines Selbststudium, da auf diese Weise zusätzliche Erklärungen gegeben, Rückfragen gestellt und Mißverständnisse vermieden werden können.[29]

In jedem Fall stellt die Schaffung einer Offline-Komponente der E-Learning-Plattform eine zentrale Anforderung an das zu entwickelnde System dar, um die obengenannten Einsatzszenarien unabhängig von der ständigen Verfügbarkeit von Internet-Anschlüssen zu ermöglichen. Dafür muß unter anderem die Datenspeicherung offline funktionieren, mithin muß es eine lokale Installation des Speichermechanismus und damit praktisch notwendigerweise auch eine lokale Datenbank geben. Außerdem müssen die einzelnen Programme, die zur Ausführung der Lehrsimulationen benötigt werden, zusammen mit der lokalen Installation der E-Learning-Plattform auf den Computern aller Benutzer installiert werden und dort ausgeführt werden können.

2.8 Online-Einsatz zur Datenverteilung und Kommunikation

Zusätzlich zu der lokalen Installation bei den einzelnen Benutzern werden für einige zentrale Funktionalitäten des zu entwickelnden Systems Online-Komponenten benötigt. Dabei handelt es sich einerseits um einen Mechanismus für die Verteilung der für die Bearbeitung von Lehreinheiten erforderlichen Daten (sowohl Konfigurationen für die verwendeten Applikationen als auch die dafür benötigten Medien), andererseits um eine möglichst vielseitige Möglichkeit zur Kommunikation zwischen räumlich getrennten Lehrenden und Lernenden.

Natürlich wäre prinzipiell auch eine Verteilung von neu erstellten oder für bestimmte Veranstaltungen speziell zusammengestellten Lehreinheiten und Medien an die dafür als Adressaten vorgesehenen Benutzer per CD möglich. Dies gestaltet sich jedoch in der Regel zu umständlich, insbesondere dann, wenn für die unterschiedlichsten Systeme jeweils verschiedene selbstausführende Installationspakete erstellt werden müssen. Die Installation jedem Benutzer selbst zu überlassen, droht spätestens dann zu scheitern, wenn Benutzer teilnehmen, die nur über geringe Erfahrungen mit Computern verfügen.

[29] Die Vorteile einer derartigen Kombination aus E-Learning und Präsenzlehre wurden mittlerweile auch von den Herstellern kommerzieller E-Learning-Produkte erkannt und führten zur Entwicklung des neuen Schlagwortes *Blended Learning*, das nichts anderes als eben diese Kombination beinhaltet.

Statt dessen sollen im Rahmen der zu entwickelnden E-Learning-Plattform die Daten auf einem zentralen Server (oder ggf. auch mehreren Servern) zur Verfügung gestellt und den Lernenden zum Download angeboten werden. Autoren sollen die von ihnen neu erstellten, ggf. offline erzeugten Lehrmaterialien ebenfalls über das Internet auf den Server hochladen können und diese somit der Allgemeinheit oder ausgewählten Nutzern zugänglich machen können. Ebenso sollen Studenten die im Rahmen von bestimmten Lehrveranstaltungen erarbeiteten Ergebnisse bei Bedarf auf dem Server ablegen und auf diese Weise beispielsweise auch Protokolle oder gelöste Übungsaufgaben den dafür zuständigen Betreuern zukommen lassen können.[30]

Während die bisher dargestellte Sichtweise die E-Learning-Plattform im Prinzip als ein klassisches Client-Server-System betrachtet, ist allerdings auch eine andere mögliche Nutzungsweise wünschenswert: Durch Hinzufügen von Peer-to-Peer-Funktionalitäten, die es jeder lokalen Installation der E-Learning-Plattform ermöglichen, unabhängig von einem zentralen Server Daten mit anderen Instanzen des Programms auf den Computern anderer Benutzer austauschen zu können, wird eine kooperative Nutzung des Systems möglich, ohne daß dafür ständig ein Eingreifen an einer zentralen Stelle erforderlich ist.

Neben einer Verbesserung der Verbreitung der Lehreinheiten und multimedialen Daten, die so auch ohne die Notwendigkeit einer Verbindung mit dem zentralen Server (und sogar bei dessen zeitweisem Ausfall) von Benutzer zu Benutzer weitergegeben werden können, ist so auch ein direkter Austausch von Daten durch die Studenten untereinander möglich. Beispielsweise ermöglicht dies eine bessere Kooperation der Studenten bei der gemeinsamen Anfertigung von Protokollen oder anderen im Rahmen von bestimmten Lehrveranstaltungen anzufertigenden elektronischen Materialien.[31] Darüber hinaus kann der Austausch von nicht für die Allgemeinheit bestimmten oder relevanten Daten auf diese Weise abseits des Servers erfolgen.

Die andere wichtige Online-Funktionalität, deren praktische Umsetzbarkeit im Rahmen der zu entwerfenden E-Learning-Plattform gefordert wird, ist die Möglichkeit zur Schaffung einer Kommunikationsumgebung, die es den Lehrenden und Lernenden ermöglicht, während der Benutzung des Systems miteinander in Kontakt zu treten. Dies kann auf unterschiedlichen Wegen geschehen. Ein einfaches Beispiel sind asynchrone Kommunikationsmittel wie E-Mail oder damit vergleichbare Systeme zur Übermittlung von Nachrichten.

Von mindestens ebenso großer Bedeutung sind jedoch Möglichkeiten zur synchronen Kommunikation. Diese können von einem einfachen, textbasierten Chat über darauf aufbauende Systeme mit höheren Funktionalitäten[32] bis hin zu Teleteaching-Funktionen

[30] Letzteres kann natürlich auch per E-Mail geschehen. Die hier beschriebene Vorgehensweise hat jedoch zwei Vorteile. Zum einen können die Protokolle oder Aufgabenlösungen innerhalb des Systems direkt an die dazugehörigen Lehrmaterialien gekoppelt bleiben, indem sie beispielsweise in einem dazugehörigen Verzeichnis abgelegt werden. Zum anderen ist es auf diese Weise auch leichter möglich, derartige von den Lernenden erarbeiteten Materialien anderen Personen oder auch der Allgemeinheit zugänglich zu machen. Eine strukturierte, mit Metadaten versehene Ablage im System ermöglicht dann wiederum auch einen leichteren Zugang zu diesen Inhalten und damit auch eine weitere Benutzung in der Lehre.

[31] Natürlich kann diese Möglichkeit auch zum unerwünschten Abschreiben beispielsweise von Lösungen für Übungsaufgaben genutzt werden. Da hierfür jedoch auch ohne die innerhalb des technischen Rahmens der E-Learning-Plattform gegebenen Kooperationsmöglichkeiten leicht Mittel und Wege gefunden werden können, ist zumindest nicht davon auszugehen, daß ein solches unerwünschtes Verhalten durch die Peer-to-Peer-Funktionalitäten gefördert oder auch nur unverhältnismäßig erleichtert wird.

[32] Dabei kann es sich beispielsweise um einen sogenannten MUD handeln, in dem die Benutzer neben der reinen Kommunikation auch die Möglichkeit zur Generierung und Manipulation von virtuellen Objekten haben und sich in virtuellen Welten bewegen können. Siehe hierzu Kapitel 9.1.

reichen, die es Lehrenden ermöglichen, aktiv in den Lernprozeß der Lernenden bei der Bearbeitung von elektronischen Kursen einzugreifen.[33] Auf diese Weise können nicht nur die Lernenden untereinander kommunizieren oder die Lehrenden ihnen bei Bedarf Hilfestellung bei Problemen geben und Fragen zu den Lehrmaterialien beantworten. Es wird darüber hinaus sogar ermöglicht, komplette Online-Veranstaltungen abzuhalten, bei denen durch die Nutzung der bereitgestellten Kommunikationsmöglichkeiten der persönliche Kontakt nachgebildet wird, der im Rahmen der traditionellen Präsenzlehre besteht. Damit können die in Kapitel 2.7 beschriebenen Vorteile derartiger Lehrformen gegenüber dem ausschließlich computerbasierten Lernen zumindest teilweise in die elektronische Lehre übertragen werden.

Mindestanforderung an die E-Learning-Plattform auf dem Gebiet der Kommunikation ist es daher, die technische Möglichkeit für derartige Interaktionen zwischen Benutzern zur Verfügung zu stellen. Ein synchroner Austausch von entsprechenden Nachrichten muß durchführbar sein. Die einzelnen Installationen des Systems müssen über geeignete Kommunikationsschnittstellen verfügen, und Anwendungen wie z. B. Chats müssen in das System integriert werden können. Damit die für derartige Zwecke zur Verfügung stehenden Kommunikationswege nicht unnötig begrenzt werden, empfiehlt es sich auch aus diesem Grund, zumindest die Möglichkeit vorzusehen, die E-Learning-Plattform nicht nur als Client-Server-System einzusetzen, sondern darüber hinaus Funktionalitäten eines Peer-to-Peer-Systems nutzbar zu machen. Eine entsprechende, auf der Plattform basierende Applikation vorausgesetzt, könnte dann auch die Kommunikation zwischen den Benutzern ggf. vollkommen unabhängig von einem zentralen Server erfolgen.

2.9 Erweiterbarkeit und Open-Source-Strategie

Als letzte wichtige Bedingung, die die zu entwerfende E-Learning-Plattform erfüllen muß, soll an dieser Stelle die Forderung nach einer leichten Erweiterbarkeit des Systems aufgestellt werden. Dies betrifft alle Bereiche der Plattform, von der Definition neuer Datentypen und der für deren Verwaltung benötigten Metadaten[34] über das Einbinden verschiedener Applikationen[35], die für den Entwurf und die Ausführung von Kursen und Lehrsimulationen für unterschiedliche Fachbereiche verwendet werden, bis hin zu noch größeren und umfassenderen Erweiterungen, die zum jetzigen Zeitpunkt noch gar nicht abzusehen sind.

Daraus ergibt sich, daß es sich bei der E-Learning-Plattform um ein offenes System handeln muß, das möglichst ohne eine Neuprogrammierung der bestehenden Elemente um neue Komponenten ergänzt werden kann. Bei Bedarf sollten aber auch umfassende Erweiterungen möglich sein, um die Software an neue Probleme und Einsatzgebiete anpassen zu können. Dazu ist es erforderlich, nicht nur die notwendigen Schnittstellen, die neue Teilprogramme zur Integration in die E-Learning-Plattform implementieren müssen, offenzulegen und zu dokumentieren, sondern es empfiehlt sich, den kompletten Quellcode inklusive der zugehörigen Dokumentation der Allgemeinheit zugänglich zu machen. Eine solche Open-Source-Strategie, die einen fortschreitenden Entwicklungsprozeß durch die Möglichkeit von nutzerseitigen Neuentwicklungen ermöglicht, kann beispielsweise unter Verwendung einer Lizenzvereinbarung realisiert werden, die eine Variation der LGPL (GNU Library General Public License) darstellt.

[33] Siehe Kapitel 9.2.
[34] Siehe Kapitel 2.2.
[35] Siehe Kapitel 2.3.

Eine derartige Entscheidung bringt jedoch einige Implikationen für die Implementation der E-Learning-Plattform mit sich. Wenn das komplette System nach der Fertigstellung im Rahmen einer Open-Source-Politik verbreitet werden soll, bedeutet dies natürlich, daß zu seiner Umsetzung auch keine fremden Bestandteile benutzt werden dürfen, die nicht ebenfalls als Open-Source-Software unter einer entsprechenden Lizenz zur Verfügung stehen. Damit ist der Einsatz kommerzieller Produkte als Grundlage des Systems praktisch ausgeschlossen, solange diese nicht für diesen Zweck explizit zur Verfügung gestellt werden und eine (zumindest für den universitären Bereich) dauerhaft kostenfreie Lizensierung ausgehandelt wird, die auch für künftige Weiterentwicklungen des Systems gilt.

Selbst in diesem Fall, der angesichts des sicher vorhandenen kommerziellen Interesses der Hersteller an der Vermarktung ihrer Software eher unwahrscheinlich ist, ergeben sich mehrere Probleme. Zum einen würde die Möglichkeit der Weiterentwicklung der E-Learning-Plattform aus Urheberrechtsgründen auf diejenigen Teile beschränkt sein, die in eigener Regie entwickelt wurden und nicht von externen Unternehmen zugekauft wurden. Dies würde die Erweiterbarkeit des ganzen Systems unter Umständen erheblich einschränken. Zum anderen ergäbe sich auch eine nicht zu unterschätzende technische Abhängigkeit von bestimmten kommerziellen Produkten. Eine Garantie für die Wartung und technische Weiterentwicklung solcher Programme kann nicht gegeben werden, und eine eventuelle Inkompatibilität neuer Versionen der fremden Software kann zu großen Problemen für die zukünftigen Benutzer der E-Learning-Plattform führen. Daher ist die Verwendung kommerzieller Software als Basis für das System kritisch zu betrachten und nach Möglichkeit zu vermeiden.

Dies betrifft insbesondere die ggf. bei der Implementierung der E-Learning-Plattform einzusetzenden Datenbanken. Eine ausschließliche Verknüpfung des Systems mit einem bestimmten kommerziellen Datenbanksystem würde eine nur schwer zu überwindende Abhängigkeit schaffen. Unter Umständen könnte in einem solchen Fall die Verwendung der Plattform durch andere Benutzer an der Hürde der Lizenzbedingungen und der dafür aufzuwendenden Kosten scheitern. Eine Möglichkeit, dies zu vermeiden, ist natürlich die Benutzung von ebenfalls im Rahmen einer Open-Source-Lizenzpolitik vertriebenen Datenbanksystemen.

Andererseits kann bei der Verwendung der Plattform in anderen Einsatzbereichen unter Umständen auch eine Anbindung an dort bereits bestehende Datenbanksysteme, für die möglicherweise schon anderweitig eine Lizenz erworben wurde, wünschenswert sein. Aus diesen Gründen bietet es sich an, für die Datenbankanbindung des Systems eine allgemeine Schnittstelle vorzusehen, die bei Bedarf für unterschiedliche Datenbanken individuell implementiert werden kann. Für eine Referenz-Implementation des Systems sollte in diesem Falle eine nichtkommerzielle, frei erhältliche Datenbank eingesetzt werden, die sich möglichst eng an den bestehenden Standards für Datenbanksysteme orientiert, um damit in einem möglichst umfassenden Maße eine Vorbildfunktion für zukünftige Implementierungen der Datenbankschnittstelle zu erreichen.

Entsprechend gilt auch für andere Bereiche des Systems, daß nach Möglichkeit überall dort, wo es für eine bestimmte umzusetzende Funktionalität mehrere verschiedene technische Lösungen geben kann, offene Schnittstellen geschaffen werden sollten, um es damit zu ermöglichen, daß bei Bedarf verschiedene Implementierungen für den gleichen Zweck verwendet werden können, und daß zukünftige Erweiterungen des Systems an diesen Stellen ihre eigenen technischen Möglichkeiten mit einbringen können.

3 Evaluation bestehender Softwaresysteme

Bevor bei der Entwicklung eines Softwaresystems über eine eigene Neuentwicklung nachgedacht wird, empfiehlt es sich stets, zu evaluieren, ob die zu lösenden Aufgaben nicht schon ganz oder teilweise von bereits bestehenden Programmen erledigt werden können. Aus diesem Grunde wurde auch beim Entwurf der E-Learning-Plattform eine Vielzahl von bereits auf dem Markt erhältlichen Produkten auf ihre Anwendbarkeit für die Erfüllung der vorher aufgestellten Anforderungen untersucht. Ungeachtet der in Kapitel 2.9 geäußerten grundlegenden Probleme bei der Verwendung kommerzieller Software in Systemen, die im Rahmen einer Open-Source-Politik der Allgemeinheit zur Verfügung gestellt werden sollen, wurden dabei auch zahlreiche Programme evaluiert, die von kommerziellen Softwareunternehmen entwickelt und vertrieben werden.

Um eine Übersicht über die bestehenden Systeme zu geben, kann leider auf keinerlei allgemein verfügbare, wissenschaftliche Literatur zurückgegriffen werden, da generelle Untersuchungen über die auf dem Markt erhältlichen Produkte in den betrachteten Bereichen weder in gedruckter noch in elektronischer Form vorliegen.[36] Dennoch soll auf eine Darstellung der im Rahmen des Projektes Monist vorgenommenen Evaluation kommerzieller und anderer Programme an dieser Stelle nicht verzichtet werden, da die in diesem Evaluationsprozeß gewonnenen Erkenntnisse die Entscheidung begründeten, aufgrund der mangelnden Eignung der bislang vorhandenen Softwaresysteme für die Umsetzung der aufgestellten Anforderungen die Eigenentwicklung vorzunehmen, deren Entwurf und Implementierung Gegenstand der vorliegenden Arbeit sind.

Die Untersuchung der erhältlichen Programme wurde einerseits in Form von Marktanalysen auf verschiedenen Fachmessen durchgeführt und basiert andererseits zum Teil auf Internet-Recherchen. Daher stützen sich die folgenden Darstellungen in erheblichem Maße auf Messeunterlagen, Werbeprospekte und Datenblätter der einzelnen Hersteller sowie auf persönliche Gespräche mit Firmenvertretern auf den verschiedenen Messen.

---

[36] Im Bereich des Dokumentenmanagements (siehe Kapitel 3.1) muß an dieser Stelle die Einschränkung vorgenommen werden, daß es zwar sehr wohl Marktstudien über Dokumentenmanagementsysteme gibt, diese allerdings nur wenig Rückschlüsse auf die Verwendbarkeit in einer den hier aufgestellten Vorgaben entsprechenden E-Learning-Plattform erlauben. Aufgrund der schnellen Entwicklung auf dem Softwaremarkt sind einige der zu diesem Thema vorgenommenen Studien auch schon veraltet, beispielsweise das aufgrund mangelnder Aktualität bereits nicht mehr im Buchhandel erhältliche Werk: Hans J. Bullinger, Renate Mayer, Marktstudie Dokumentenmanagement, Dr. Th. Gabler Verlag, Wiesbaden, 1994.
Einige neuere Studien beschäftigen sich zwar mit einer größeren Anzahl teilweise nur wenig verbreiteter Dokumentenmanagementsysteme, lassen jedoch - möglicherweise aus Gründen hoher Lizenzkosten oder vom Hersteller nicht zur Verfügung gestellter Testversionen - unglücklicherweise einige der gängigsten Systeme unbeachtet. Ein Beispiel für einen solchen Fall ist die folgende Diplomarbeit: Christian Fuchs, Entwurf einer einheitlichen und formalisierbaren Beschreibung von Dokumenten-Management-Systemen (DMS) auf der Basis einer vergleichenden Untersuchung bestehender DMS, Diplomarbeit an der Fakultät für Informatik und Automatisierung der Universität Ilmenau, 2001.
Daneben gibt es einige in den letzten Jahren entstandene Studien, die sich zwar mit einigen der aktuell gängigen Dokumentenmanagementsysteme befassen. Da diese Studien jedoch zu dem Zweck geschrieben wurden, IT-Manager von kommerziellen Unternehmen bei Anschaffungsentscheidungen zu unterstützen, liegt der Fokus dabei auf betriebswirtschaftlichen Gesichtspunkten wie beispielsweise der Abbildung von Unternehmensprozessen und Workflows im System sowie Kosten-Nutzen-Rechnungen. In der Frage, ob die betrachteten Systeme für den Einsatz in einem E-Learning-System geeignet sind, führen diese Studien jedoch leider nicht weiter. Zwei Beispiele sind die folgenden Werke: Cornelia Versteegen, Dokumentenmanagement, Verlag IT Research, Sauerlach, 2000; Katharina Angerhausen, Dietmar Weiß, Holger Mertens, Dokumenten-Management-Systeme im Vergleich, Oxygen Verlag, Feldkirchen, 2002. Der Preis für ein Exemplar der beiden vorgenannten Studien liegt übrigens bei 1.450 € bzw. 738 €, was ebenfalls die kommerzielle Zielgruppe dieser Untersuchungen verdeutlicht.

Obwohl derartige Quellen natürlich in erheblichem Maße von dem Interesse der Firmen geprägt sind, ihre eigene Software zu Verkaufszwecken im besten möglichen Licht darzustellen, lassen sich daraus in den meisten Fällen doch in hinreichendem Maße die für eine Beurteilung der technischen Fähigkeiten der Systeme benötigten Informationen entnehmen.

Darüber hinaus wurden Testversionen zahlreicher Programme evaluiert und auf ihre technische Eignung für den Einsatz in der E-Learning-Plattform untersucht. Außerdem wurden die auf den Internet-Seiten der einzelnen Hersteller erhältlichen Informationen verwendet. Dies gilt sowohl für die untersuchten kommerziellen Produkte, die in den Kapiteln 3.1 (Marktübersicht zu Projektbeginn) und 3.3 (Marktübersicht gegen Ende der Projektlaufzeit) dargestellt werden, als auch für die ebenfalls in die Studie mit einbezogenen, hauptsächlich nichtkommerziellen Entwicklungen aus dem universitären Bereich (Kapitel 3.2), die zu Beginn des Projektes erstmals evaluiert und deren weitere Entwicklung danach kontinuierlich beobachtet wurde.

3.1 Marktübersicht zu Projektbeginn

Zu Projektbeginn waren E-Learning-Programme noch weitgehend Insellösungen. Zwar gab es (wie übrigens auch innerhalb des Monist-Konsortiums[37]) die unterschiedlichsten Programme für verschiedene Fachbereiche. Diese waren jedoch im allgemeinen nicht untereinander kompatibel und nicht für den Einsatz in einem gemeinsamen technischen Umfeld vorgesehen.[38] Der Begriff einer übergreifenden Plattform für computerbasiertes Lernen war nur wenig verbreitet, und die meisten derartigen Entwicklungen, die von vielen kommerziellen Unternehmen im Laufe der Projektlaufzeit geschaffen wurden[39], steckten zu diesem Zeitpunkt noch in den Kinderschuhen.

Da zu Projektbeginn also praktisch kein Produkt vorlag, das den Anspruch hatte, eine voll integrierte E-Learning-Plattform zu bieten[40], wurde bei der zu diesem Zeitpunkt vorgenommenen Marktanalyse das Hauptaugenmerk auf geeignete Möglichkeiten zur Speicherung der in einem E-Learning-System zu verwaltenden multimedialen Daten gelegt. Diejenigen Programme, die über entsprechende Fähigkeiten verfügten[41], wurden anschließend daraufhin untersucht, inwieweit die erforderlichen Möglichkeiten gegeben

---

[37] Das Monist-Konsortium wurde gebildet von den folgenden Professoren und ihren Lehrstühlen:
- Prof. Dr. M. Egelhaaf, LS Neurobiologie, Fakultät für Biologie, Universität Bielefeld (federführend)
- Prof. Dr. A. Aertsen, Neurobiologie und Biophysik, Institut für Biologie III, Universität Freiburg
- Prof. Dr. H. Cruse, Abt. Kybernetik/Theoretische Biologie, Fakultät für Biologie, Universität Bielefeld
- Prof. Dr. D. Dörner, LS Psychologie II, Fak. Pädagogik/Psychologie/Philosophie, Universität Bamberg
- Prof. Dr. H. M. Gross, FB Neuroinformatik, Fak. für Informatik und Automatisierung, TU Ilmenau
- Prof. Dr. H.A. Mallot, LS Kognitive Neurowissenschaft, Fakultät für Biologie, Universität Tübingen
- Prof. Dr. R. Menzel, Institut für Neurobiologie, FB Biologie, Freie Universität Berlin
- Prof. Dr. H. Ritter, AG Neuroinformatik, Technische Fak., Universität Bielefeld
- Prof. Dr. N. M. Seel, Seminar für Erziehungswissenschaft, Philosophische Fak., Universität Freiburg

[38] Die Verbesserung dieses Zustandes für den Fachbereich der Neuro- und Kognitionswissenschaften war gerade eine der wichtigsten Zielsetzungen bei der Konzipierung und Beantragung des Projektes Monist. Siehe hierzu die *Projektskizze zur Einreichung bei dem Förderprogramm des Bundesministeriums für Bildung und Forschung: "Neue Medien in der Bildung"* unter der Internet-Adresse http://www.monist.de bzw. http://natrix.biologie.uni-bielefeld.de/monist/Archiv/monistskizze_oz.html

[39] Siehe Kapitel 3.3.

[40] Die einzige Ausnahme war die auf dem Dokumentenmanagementsystem Hyperwave Information Server basierende Hyperwave E-Learning Plattform. Siehe hierzu die entsprechenden Ausführungen zu den Produkten der Firma Hyperwave AG im Kapitel 3.1.2.

[41] Insbesondere über die Möglichkeit zur Speicherung und Verwaltung beliebiger eigener Metadaten.

waren, um entsprechende Erweiterungen vorzunehmen, um die in Kapitel 2 dargelegten Anforderungen zu erfüllen. Wichtige Beurteilungskriterien waren dabei unter anderem die Möglichkeit zur Integration eigener Applikationen (in erster Linie der benötigten Simulationssoftware) in die bestehenden Systeme, die Plattformunabhängigkeit, die im Sinne einer leichten Verbreitung nach Möglichkeit zu vermeidende Abhängigkeit von einem bestimmten Datenbanksystem sowie die Realisierbarkeit der benötigten Offline-Komponente.[42]

Die Marktanalyse wurde im Rahmen von Messebesuchen auf den Fachmessen Internet World 2000[43], Systems 2000[44] und CeBIT 2001[45] vorgenommen. Dabei stellte sich heraus, daß die in Frage kommenden Produkte sich ausschließlich den drei miteinander verwandten Bereichen Content Management, Dokumentenmanagement und Knowledge Management zuordnen ließen.[46] Obwohl diese drei Begriffe, die auf den drei genannten Messen durchgängig den Status populärer Schlagwörter besaßen, insbesondere in den Werbebroschüren der Unternehmen oftmals miteinander vermengt und dabei in vielen Fällen schlichtweg falsch verwendet wurden, läßt sich dennoch eine klare Abgrenzung zwischen diesen drei Begriffen schaffen.

Entsprechend dieser Abgrenzung werden in den folgenden Abschnitten die betrachteten Produkte in die genannten drei Gruppen eingeordnet, denen jeweils eine Definition des

---

[42] Diesem Punkt kommt insofern eine entscheidende Bedeutung zu, als daß die hier betrachteten Systeme praktisch ausschließlich für den internetbasierten Einsatz geschaffen wurden. Da in unternehmensweiten Netzwerken normalerweise zentrale Server für die Datenspeicherung zur Verfügung stehen, wird einer lokalen Speicherung, wie sie im Rahmen der E-Learning-Plattform aus den in Kapitel 2.7 genannten Gründen benötigt wird, zumeist keine Bedeutung beigemessen, obwohl dies im Falle des Ausfalles von Servern oder Netzwerken auch im industriellen Bereich unweigerlich zu Problemen führt.

[43] Die Internet World 2000 fand vom 23.-25.05.2000 auf dem Messegelände Berlin statt. Der Besuch auf dieser Messe durch den Autor erfolgte zwar noch nicht im Rahmen des Projektes Monist, das zu diesem Zeitpunkt noch nicht begonnen hatte, sondern aufgrund einer vorherigen Tätigkeit in einer Internet-Firma, jedoch mit einer ähnlichen Zielsetzung. Nähere Informationen zu dieser Messe, die bis einschließlich zum Jahre 2002 an ihrem ursprünglichen Standort in Berlin stattfand, sind im Internet erhältlich unter der Adresse: http://www.internetworld-messe.de/deutsch/messe/default.htm. Im Jahre 2003 wurde die Messe offenbar wegen Ausstellermangels abgesagt und zeitlich und räumlich mit der Systems in München zusammengelegt. Informationen über die Messe am neuen Standort sind im Internet erhältlich unter der Adresse: http://www.internetworld-messe.de.

[44] Die Systems 2000 fand vom 06.-10.11.2000 auf dem Gelände der Messe München statt. Der Besuch auf dieser Messe durch den Autor erfolgte ebenfalls noch nicht im Rahmen des Projektes Monist, das zu diesem Zeitpunkt noch nicht begonnen hatte, sondern aufgrund einer vorherigen Tätigkeit in einer Firma, die betriebswirtschaftliche Software entwickelte, jedoch mit einer ähnlichen Zielsetzung. Informationen über die Systems sind im Internet erhältlich unter der Adresse: http://www.systems-world.de.

[45] Die CeBIT 2001 fand vom 22.-28.02.2001 auf dem Gelände der Hannover Messe statt. Der Großteil der Evaluation der in diesem Kapitel beschriebenen Software wurde auf der Grundlage eines mehrtägigen Messebesuches im Rahmen des Projektes Monist vorgenommen. Informationen über die CeBIT sind im Internet erhältlich unter der Adresse: http://www.cebit.de.

[46] Die wechselnde Verwendung der englischen und deutschen Begriffe erfolgt hier in gleichem Maße, wie dies im Sprachgebrauch auf den Messen der Fall war und auch zum gegenwärtigen Zeitpunkt noch ist. Während für den englischen Begriff "content management" keine deutsche Bezeichnung, auch nicht die wörtliche Übersetzung "Inhaltsmanagement", benutzt wird, hat sich im deutschsprachigen Raum die Bezeichnung "Dokumentenmanagement" gegenüber dem englischen "document management" praktisch ausnahmslos durchgesetzt. Dahingegen hält sich der Gebrauch der Begriffe "Wissensmanagement" und "knowledge management" ungefähr die Waage, wobei allerdings eine Tendenz zu beobachten ist, daß in theoretischeren Betrachtungen und in Abhandlungen, die einen wissenschaftlichen Anspruch für sich reklamieren, die deutsche Bezeichnung benutzt wird, während in reinen Werbebroschüren sowie in den meisten Messevorträgen und an den Messeständen zumeist der englische Begriff zum Einsatz kommt. Dieses Verbreitungsmuster der Verwendung der englischen und deutschen Bezeichnungen könnte Gegenstand einer interessanten sprachwissenschaftlichen Untersuchung sein, die an dieser Stelle natürlich nicht vorgenommen werden kann.

betreffenden Begriffes vorangestellt ist. In der Praxis sind die Übergänge zwischen den einzelnen Gruppen sind in der Funktionalität der Systeme zwar oft fließend, weil z. B. die meisten Programme aus dem Bereich Knowledge Management als Grundlage für die Speicherung von Informationen ein Dokumentenmanagementsystem benutzen und viele Dokumentenmanagementsysteme ihrerseits auch dazu verwendet werden können, Daten in ähnlicher Weise automatisiert auf WWW-Seiten zu übertragen, wie dies durch (Web) Content Management Systeme geschieht. Jedes der betrachteten Produkte kann jedoch entsprechend seiner Kernfunktionalität und seinem Haupteinsatzgebiet eindeutig einem der drei genannten Bereiche zugeordnet werden.

3.1.1 Content Management Systeme

Der Begriff "Content Management Systeme" (CMS) ist insofern etwas irreführend, weil man sich unter der "Verwaltung von Inhalten" die unterschiedlichsten Dinge vorstellen kann. Deutlicher wird die Bedeutung, wenn man berücksichtigt, daß diese Bezeichnung eigentlich als Kurzform für "Web Content Management System" verwendet wird. Bei einem (Web) Content Management System handelt es sich um ein Programm, das für die Verwaltung von Inhalten eingesetzt wird, die auf umfangreichen WWW-Seiten verwendet werden. Dabei kann es sich beispielsweise um die Homepages von (meist größeren) Unternehmen handeln, aber auch um Online-Produktkataloge, elektronische Zeitungen und im Prinzip jede beliebige Ansammlung von Internet-Seiten, die ständig aktuell gehalten werden sollen und bei denen eine manuelle Verwaltung aufgrund der Menge der enthaltenen Informationen zu aufwendig wäre. Auch für die Verwaltung des firmeninternen, nicht für die Öffentlichkeit zugänglichen Intranets, das heutzutage in vielen größeren Unternehmen zur Versorgung der eigenen Mitarbeiter mit internen Informationen dient, werden oft Content Management Systeme eingesetzt.

Charakteristisch für solche Systeme ist es, daß diese meist über ein System zur Ablage von multimedialen Daten verfügen, das von der eigentlichen, für den späteren Benutzer im Internet sichtbaren Oberfläche getrennt ist. Die Daten können durch berechtigte Autoren gesondert bearbeitet werden, ohne daß diese sich um das letztendliche Layout der daraus entstehenden WWW-Seiten Gedanken machen müssen. Dieses wird nach im System festlegbaren Kriterien um die eingestellten Daten herum erzeugt und stellt z. B. sicher, daß alle veröffentlichten Seiten in einem firmeneigenen Erscheinungsbild, dem sogenannten Corporate Design, dargestellt werden.

Die Veröffentlichung der Inhalte erfolgt dann durch eine kontrollierte Übertragung auf einen Webserver, die entweder automatisiert nach bestimmten Regeln erfolgt oder eine explizite Genehmigung durch bestimmte, dafür verantwortliche Personen erfordert. Der Datenbestand auf dem Webserver stellt immer einen zu einem bestimmten Augenblick angelegten, bestimmten Genehmigungskriterien folgenden Ausschnitt der Daten in dem nichtöffentlichen Datenbestand des Content Management Systems dar. Die meisten Systeme bieten darüber hinaus den Benutzern Werkzeuge zum vereinfachten Editieren von HTML-Seiten an, die beispielsweise die Erzeugung bestimmter Typen von Seiten nach vorgegebenen Formatvorlagen ermöglichen, ohne daß die Benutzer selbst über HTML-Kenntnisse verfügen müssen.

Im Folgenden werden die auf den genannten Messen evaluierten CMS in alphabetischer Reihenfolge der Firmennamen dargestellt, wobei zu jedem Produkt die benutzten Quellen (meist in Form von Internetseiten) angegeben werden. Es folgt jeweils die Betrachtung der Eignung der Systeme für die Verwendung in der E-Learning-Plattform.

3.1.1.1 Coextant Hyper.Net

Das Produkt Hyper.Net der Firma Coextant[47] ist ein klassisches Content Management System, dessen Hauptzweck die Bereitstellung von in einem Unternehmen anfallenden Informationen im WWW ist. Daneben können die Daten auch anderen Programmen zur Verfügung gestellt werden, was derzeit hauptsächlich im Bereich der Integration in die verschiedenen Teilanwendungen von Microsoft® Office® geschieht. "Hyper.Net ist eine Art 'Transmitter' (technisch: XML Web Service), der Dokumente und Dokumentationen vom Ort der Entstehung abholt, regelbasiert aufbereitet, zu strukturierten Daten auf XML Basis konvertiert und diese in relationalen Datenbanken zur standardisierten Nutzung durch beliebige Anwendungen und Prozesse bereitstellt."[48]

Als Quelle der zu verarbeitenden Daten kommen dabei viele unterschiedliche Verwaltungsmöglichkeiten in Frage, von Verzeichnissen auf der Festplatte eines Computers bis hin zu Dokumentenmanagementsystemen: "Hinzufügungen, Änderungen oder Löschungen an Dokumentteilen oder ganzen Dokumenten können über beliebige Dokumentenverwaltungssysteme (Bibliotheken, öffentliche Ordner, DMS, ECMS, Windows Filesystem ...) an Hyper.Net geroutet werden."[49] Der Prozeß der Umwandlung von Daten in das von Hyper.Net intern verwendete, XLM-basierte Datenformat und der anschließenden Bereitstellung wird von der Firma Coextant als "Content Automation" bezeichnet.[50]

Obwohl Hyper.Net ausreichende Möglichkeiten zu einer kooperativen Bearbeitung von Inhalten bietet, würde sich die Verwendung als Basis für die E-Learning-Plattform zumindest schwierig gestalten. Zum einen fehlt hier wie in den meisten CMS für den Lernenden als Endbenutzer eine Offline-Komponente, da das System vollständig auf die Veröffentlichung im Internet oder Intranet ausgerichtet ist. Da alle für die Allgemeinheit zugänglich gemachten Dokumente zuvor einen von bestimmten, im System verankerten Genehmigungen abhängigen Veröffentlichungsprozeß durchlaufen müssen und für den Endbenutzer aufgrund der reinen WWW-Oberfläche, in der die veröffentlichten Daten präsentiert werden, außerdem ein Rückkanal für die Einbringung eigener Informationen fehlt, ist die Möglichkeit zur aktiven Beteiligung der Lernenden stark eingeschränkt.

Die Fähigkeiten des Systems zur Verwaltung multimedialer Daten erscheinen durchaus ausreichend, die Zugänglichkeit der innerhalb des Systems verwendeten (größtenteils anhand von vorher festgelegten Kriterien automatisch generierten) Metadaten für den Endbenutzer ist jedoch standardmäßig nicht in hinreichender Weise gegeben, um die gewünschten Suchfunktionalitäten zu ermöglichen, was sich auch aus den von Coextant selbst veröffentlichten Fallstudien entnehmen läßt.[51] Um derartige Funktionalitäten für

---

[47] Homepages: http://www.coextant.de (deutsch), http://www.coextant.com (englisch)

[48] Zitat von der deutschen Homepage der Firma Coextant, http://www.coextant.de.

[49] Hyper.Net Produkt Übersicht, Informationsbroschüre der Firma Coextant, Seite 6. Diese Broschüre ist zum Download erhältlich auf der Homepage der Firma Coextant, http://www.coextant.de. Ein Link zu der Broschüre ist auf der Seite http://www.coextant.de/default.aspx?NAV=2 zu finden, die aktuelle Adresse (Stand: Oktober 2003) für den Download der Broschüre, die evtl. Änderungen unterworfen ist, lautet: http://www.coextant.de/Content/54B6995196A51982C1256D6000276972/71b92be9f54949b800256d6000279364/7c17cee002f949f900256d6000225481/Produkt%20Overview%20-%20deutsch%20final%20web.pdf

[50] A. a. O., Seite 6. Die Tatsache, daß dieser Prozeß in der jetzt nicht mehr zum Download erhältlichen Vorgängerbroschüre aus dem Jahr 2001 noch als "Knowledge Automation" bezeichnet wurde, zeigt zum einen die bereits erwähnte, damals weit verbreitete Verwischung der Grenzen zwischen den Begriffen, zum anderen jedoch die einsetzende Tendenz zu einer exakteren Benennung der verschiedenen Systeme.

[51] A. a. O., Seite 11. Seitens der Schweizer Firma sunrise wird dort explizit erwähnt, daß die "intelligente Navigation und Suche in den umfangreichen Datenmengen" nicht zum Funktionsumfang des von Coextant gelieferten Systems gehören und daher noch separat implementiert werden müssen.

die Lernenden dennoch zu gewährleisten, wären ebenso erhebliche Erweiterungen des Systemumfanges erforderlich wie für die Einbindung eigener Applikationen, die für die Benutzung von in das E-Learning-System eingestellten Lehreinheiten und insbesondere Lehrsimulationen erforderlich sind. Dies zeigt schon der hohe Aufwand, der von Coextant zusammen mit Microsoft betrieben wurde, um die im System enthaltenen Daten für die Teilprodukte von Microsoft® Office® zugänglich zu machen, was letztlich zur Entwicklung eigener, für diesen Zweck separat erhältlicher Systemkomponenten führte. Für die Einbindung eigener Software jedes Mal einen vergleichbaren Aufwand betreiben zu müssen, ist im Sinne der Forderung nach einer leichten Erweiterung des Systems nicht akzeptabel.

Während für die Endbenutzer von Hyper.Net angesichts der erzeugten Web-Oberfläche theoretisch keinerlei Einschränkungen bezüglich des zu verwendeten Betriebssystems bestehen, solange nicht die für Microsoft® Office® geschaffenen Systemerweiterungen benutzt werden sollen, ist die Kernanwendung selbst nur für Microsoft-Betriebssysteme erhältlich[52], was insbesondere für die Autoren eine erhebliche Einschränkung bedeutet. Als Datenbanksysteme können Microsoft® SQL Server®, IBM DB/2, Oracle und Lotus Domino[53] verwendet werden, was zwar keine ausschließliche technische Abhängigkeit von einem einzigen Anbieter bedeutet, jedoch nur Optionen auf die Verwendung einiger lizenzpflichtiger und damit erhebliche Kosten verursachender Produkte offenläßt. Eine frei implementierbare Schnittstelle für beliebige Datenbanken existiert nicht.

3.1.1.2 Documentum 4.1 Content Management Platform

Als traditioneller Hersteller von Systemen zur Verwaltung elektronischer Dokumente, die in ihren Anfängen eher dem Bereich Dokumentenmanagement zuzuordnen waren, entwickelt die kalifornische Firma Documentum Inc.[54] heute sehr umfangreiche Content Management Systeme mit zahlreichen Zusatzmodulen für die unterschiedlichsten Einsatzbereiche. E-Learning-Produkte gehören allerdings nicht dazu.[55] Die im Jahre 2001 zum Zeitpunkt der Marktanalyse aktuelle Produktversion 4.1 wurde mittlerweile durch das Nachfolgeprodukt, die neue Documentum 5 Enterprise Content Management Platform, ersetzt.[56]

Dieses System besteht aus einer großen Anzahl von Komponenten für unterschiedliche Einsatzbereiche. "Die Documentum-Architektur ruht auf drei Säulen: Content Repository, Content Services und Content Applications."[57] Während das Content Repository für die Speicherung von Daten in beliebigen Formaten zuständig ist, übernehmen die Content Services unter anderem "die Zugriffssteuerung, die Versionskontrolle, Such- und Workflowfunktionen"[58]. Unter dem Oberbegriff Content Applications schließlich versteht man jegliche Programme, die auf der Basis der zugrundeliegenden Services die im Repository gespeicherten Daten nutzen.

---

[52] A. a. O., Seiten 4 und 10, sowie entnehmbar aus der Beschreibung der Systemanforderungen unter der URL: http://www.coextant.de/topic.aspx?DOC_UNID=EBDFAE81615A4f959D5F145CD54D4895.
[53] A. a. O., Seite 6.
[54] Homepages: http://www.documentum.de (deutsch), http://www.documentum.com (englisch)
[55] Siehe hierzu die auf der Homepage der Firma Documentum erhältliche Produktliste unter der folgenden Adresse: http://www.documentum.de/products/glossary/product_glossary_home.htm.
[56] Siehe hierzu die Informationen der Firma Documentum zum Lieferstart des neuen Produktes unter der Internet-Adresse: http://www.documentum.de/products/launch/documentum_5.htm.
[57] Produktbeschreibung der Enterprise Content Management Platform auf der Homepage der Firma Documentum unter der Adresse: http://www.documentum.de/products/platform/index.htm.
[58] A. a. O.

Das CMS Documentum 5 verwaltet Metadaten und ermöglicht prinzipiell die Definition von unterschiedlichen Objekttypen, die auf unterschiedliche Weise verwaltet und von eigenen Applikationen benutzt und bearbeitet werden.[59] Für diese ist jedoch ggf. eine sehr aufwendige Integration in den Rahmen der Documentum-Produkte zu leisten.[60] Zudem beruht die gesamte Architektur auf der Vorstellung der Bearbeitung und Verbreitung von statischen Dokumenten wie beispielsweise technischen Zeichnungen oder Unternehmensberichten[61], was der Verwendung für die Bearbeitung dynamischer konfigurierbarer Simulationen nicht entgegenkommt.

Neben diesem nicht unerheblichen Aufwand, der zudem eine starke technische Bindung aller zu integrierenden Programme an die Documentum-Plattform bedeuten würde, stellt sich auch bei Documentum das Problem, daß das gesamte System ausschließlich für den Online-Einsatz konzipiert wurde. Die Entwicklung einer Offline-Komponente würde, falls überhaupt möglich, einen unvertretbar hohen Aufwand mit sich bringen, zumal die Anbindung an die unterstützten Datenbanksysteme ebenfalls nur über serverseitige Dienste erfolgt. Insgesamt läßt sich sagen, daß das CMS von Documentum mit seinen umfangreichen darauf aufbauenden Business-Applikationen zwar eine sehr vielseitige Software für Unternehmen darstellt, wie auch die äußerst beeindruckende Kundenliste[62] erkennen läßt, auf der die meisten namhaften weltweiten Unternehmen zu finden sind. Als Basis für die zu entwickelnde E-Learning-Plattform ist das System jedoch aufgrund des zu hohen Integrationsaufwandes nicht geeignet, zumal E-Learning-Funktionalitäten weder in Version 4.1 noch in der neuen Version 5 zu den unterstützten Anwendungsbereichen gehören.

3.1.1.3 Gauss VIP

Unter dem Kürzel VIP, das hier für "Versatile Internet Platform" steht, vertreibt die Firma Gauss Interprise AG[63] eine Reihe von aufeinander aufbauenden Produkten, deren Grundlage der VIP Content Manager[64] bildet. Wie der Name schon sagt, handelt es sich dabei um ein Content Management System, welches für die Erstellung, Speicherung, und Verwaltung von Informationsobjekten verwendet werden kann. Ziel ist dabei stets die strukturierte Veröffentlichung im Internet. "Je nach Inhalt und Funktion unterscheidet VIP ContentManager verschiedene Objekttypen, z. B. HTML-Seiten, XML-Inhalte, Themen, Grafiken, Vorlagen und Verbundobjekte. Unabhängig vom Objekttyp können die VIP-Objekte auch vom Benutzer definierten Objektkategorien zugeordnet werden, z. B. Rechnung, Lieferschein oder Preisliste. Für diese Kategorien lassen sich spezifische Attribute angeben, die über einen erweiterten Metadaten-Dialog bearbeitet und im Objektfilter als Filterkriterium herangezogen werden können."[65]

---

[59] Diese Funktionalität ist zum Teil neu und stand in Documentum 4.1 noch nicht zur Verfügung.

[60] Siehe hierzu das Documentum Technical White Paper: Developing Web Services with Documentum, das unter der Adresse http://www.documentum.de/products/collateral/platform/wp_tech_web_svcs.pdf zum Download erhältlich ist.

[61] Siehe hierzu das sogenannte Produkt-Datenblatt der Firma Documentum zu Documentum 5, das unter der Internet-Adresse http://www.documentum.de/products/collateral/platform/ds_documentum5.pdf zum Download angeboten wird, dabei insbesondere die Betrachtungen über Content-Typen auf Seite 3.

[62] http://www.documentum.de/customer_success/customer_list.html

[63] Homepage: http://www.gauss.de, zur Zeit erfolgt dort eine automatische Weiterleitung auf die Adresse: http://ham-hal.gauss.de/gaussvip_prod/gaussvip/EMEA/DE/home.jsp.

[64] Eine Produktbeschreibung ist im Internet unter der folgenden Adresse zum Download erhältlich: http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/content-manager_german.pdf

[65] Technical White Paper: Web-Content-Management mit der VIP CM Suite, zum Download erhältlich unter: http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/Whitepaper_VIP_Content_Management_Suite_8_de.pdf, Seite 53

Eine für die Zwecke der E-Learning-Plattform hinreichende Metadaten-Verwaltung im Sinne der in Kapitel 2.2 aufgestellten Anforderungen wäre mit diesem System gegeben. Weiterhin unterstützt der VIP Content Manager mehrsprachige Benutzeroberflächen und Inhalte und verwaltet außerdem "Flexible Zugangsberechtigungen auf der Basis von Benutzern, Benutzergruppen und -rollen"[66].

Darüber hinaus ist eine weitgehende Plattformunabhängigkeit des Systems gegeben: "VIP ContentManager ist vollständig in Java implementiert. Diese plattformneutrale Programmiersprache ermöglicht den Einsatz auf diversen Betriebssystemen und Hardwareplattformen."[67] In der Praxis wird das Produkt derzeit für die Betriebssysteme Microsoft Windows 2000 Server, Microsoft NT 4 Server, HP-UX 11.0, IBM AIX 4.3.3, Sun Solaris 8 (SPARC), Linux SuSE 8.0 und Linux Red Hat 7.3 (Intel) ausgeliefert[68], was auch mit der Verfügbarkeit der beiden unterstützten Datenbanken (Microsoft® SQL Server 2000® sowie Oracle 8i und 9i)[69] zusammenhängt.

Problematisch ist jedoch auch in diesem Fall die für ein Content Management System typische Sichtweise, die nach der Bearbeitung der Inhalte durch bestimmte privilegierte Autoren eine ausschließliche Veröffentlichung im Internet vorsieht, die gewissermaßen eine Einbahnstraße darstellt und keinen Rückkanal für eine Beteiligung der Endbenutzer beinhaltet. Als ausschließliches Werkzeug zum Betrachten der veröffentlichten Inhalte dient der Web-Browser, dessen Flexibilität insbesondere hinsichtlich der Betrachtung dynamischer Inhalte und der Integration eigener Software Grenzen gesetzt sind.

Die Konstruktion als reines Client-Server-System steht der für die E-Learning-Plattform erforderlichen Nutzung einer Offline-Komponente entgegen, und für die Verwendung eigener Programme für die Erstellung und Benutzung von dynamischen Simulationen wäre doppelte Arbeit erforderlich, da diese einerseits für die Autoren in die Umgebung des VIP Content Managers eingepaßt werden müßten und andererseits für die Lernenden als Endbenutzer in einer Weise zur Verfügung gestellt werden müßten, die ihre Verwendung aus dem Internet-Browser heraus ermöglicht.

Da dies für verschiedene Browser und Betriebssysteme sehr unterschiedlich gehandhabt werden muß, wäre hierfür wiederum die Erstellung einer Vielzahl von unterschiedlichen Client-Installationen erforderlich, ohne daß dies bereits eine weitergehende Beteiligung der Benutzer über den Status reiner Online-Leser hinaus ermöglichen würde. Die zu diesem Zweck erforderlichen Weiterentwicklungen, insbesondere die Erstellung einer funktionsfähigen Offline-Komponente, würden einen ähnlichen Arbeitsumfang wie für eine komplette Eigenentwicklung erfordern, die jedoch darüber hinaus den großen Vorteil der Unabhängigkeit von kostenpflichtigen Lizenzen mit sich bringen würde. Aus diesem Grund ist eine Eigenentwicklung auch in diesem Fall vorzuziehen.

---

[66] Produktbeschreibung für den VIP Content Manager, zum Download erhältlich unter der Adresse: http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/content-manager_german.pdf, Seite 4.

[67] Technical White Paper: Web-Content-Management mit der VIP CM Suite, zum Download erhältlich unter: http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/Whitepaper_VIP_Content_Management_Suite_8_de.pdf, Seite 24.

[68] Produktbeschreibung für den VIP Content Manager, zum Download erhältlich unter der Adresse: http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/content-manager_german.pdf, Seite 4.

[69] A. a. O., Seite 4.

3.1.1.4 Imperia 5.0 Web Content Management System

Die Imperia Software Solutions GmbH[70] entwickelt mit dem gleichnamigen Produkt ein reines Web Content Management System, das im Gegensatz zu anderen, beispielsweise dem Content Management System von Documentum, von Beginn an zu diesem Zweck eingesetzt wurde. Seinen Ursprung hat Imperia in einem Redaktionssystem, welches in einem Online-Projekt zu dem Zweck geschaffen wurde, doppelten Arbeitsaufwand zu vermeiden, der entstand, weil dabei verwendete Texte zuvor zuerst von Journalisten mit verschiedenen Teilapplikationen von Microsoft® Office® erstellt und anschließend von Programmierern nochmals in HTML umgesetzt werden mußten.[71] Seitdem wurde das Produkt kontinuierlich weiterentwickelt. Auf der CeBIT 2001 wurde die zu diesem Zeitpunkt brandneue Version 5.0 vorgestellt, mittlerweile ist bereits die übernächste Version Imperia 7 erhältlich.

Neben komfortablen Verwaltungsfunktionalitäten für die unterschiedlichsten Medien, die in der jeweiligen Quellapplikation editiert und anschließend ins System eingestellt werden können, bietet Imperia umfangreiche Funktionalitäten zur Verwaltung dieser Daten. Mehrsprachige Inhalte werden ebenso verwaltet wie beliebige Metadaten, hinzu kommt die automatische Erstellung von Volltext-Indizes. Versionskontrolle, Benutzerverwaltung und die Möglichkeit der Abbildung von Workflows zur Bearbeitung von Inhalten sind ebenfalls in den Grundfunktionalitäten des Systems enthalten.[72]

Imperia ist ein webbasiertes System, das über Standard-Internetbrowser zu bedienen ist. Das Produkt ist plattformunabhängig und arbeitet praktisch mit jeder beliebigen auf dem Markt erhältlichen relationalen Datenbank zusammen, insbesondere auch mit frei verfügbaren Systemen wie z. B. MySQL.[73] Für eine potentielle Erweiterbarkeit ist von besonderer Bedeutung, daß das System (zumindest an zahlende Kunden) mit komplett offengelegtem Quellcode und mit einer umfangreich dokumentierten Programmierschnittstelle[74] ausgeliefert wird. Dies ermöglicht theoretisch die Integration beliebiger darauf aufbauender Applikationen in das System, sofern eine serverseitige Bearbeitung eingehender Anfragen erfolgt und auf den angezeigten WWW-Seiten an dynamischen Code nur Java-Applets oder die in den marktüblichen Internet-Browsern ausführbaren Skriptsprachen zur Anwendung kommen.

Genau deshalb bildet auch in diesem Fall die Generierung einer Offline-Komponente das größte Problem, da diese zum einen als Applikation in die verschiedensten Browser integriert werden müßte und zum anderen eine lokale Speicherung von Daten noch nicht vorgesehen ist. Auch die offene Programmierschnittstelle für den Server ermöglicht es nicht, auf den Clients die benötigte Simulationssoftware in einfacher Weise zur Verfügung zu stellen. Diese könnte allenfalls separat installiert und von Betriebssystem zu Betriebssystem in unterschiedlicher Weise als auszuführendes Programm zum Öffnen der verschiedenen Dateitypen, in denen die Simulationen vorliegen, angegeben werden. Trotz der vergleichsweise leichten Erweiterbarkeit stellt auch hier die Trennung in den von Autoren zu bearbeitenden Datenpool und die nach einem Veröffentlichungsprozeß für andere Benutzer zugänglichen Internetseiten ein erhebliches Hindernis dar.

---

[70] Homepage: http://www.imperia.net.

[71] Quelle für diese Information ist eine unter dem Titel "Auf den Content kommt es an" herausgegebene Messebroschüre der Firma Imperia zur CeBIT 2001, die leider nicht mehr zum Download erhältlich ist.

[72] Diese Informationen können entnommen werden aus dem Produktdatenblatt zu Imperia 7, welches im Internet unter der Adresse http://www.imperia.net/imperia/md/content/folder/Imperia_7_Datenblatt.pdf zum Download erhältlich ist.

[73] A. a. O., Seite 1.

[74] A. a. O., Seite 1.

3.1.1.5 Infopark NPS Content Management System

Mit dem Content Management System NPS der Firma Infopark AG[75] "können größere Internet-, Intranet- und Extranet-Auftritte effektiv erstellt und verwaltet werden"[76]. Bei diesem System handelt es sich also um ein reines Content Management System. NPS ist serverseitig für die Betriebssysteme SUN Solaris, Linux, Microsoft® Windows® 2000 und Microsoft® Windows® XP, erhältlich.[77] NPS arbeitet mit den Datenbanken Sybase, Oracle, IBM DB/2 und MS SQL Server® zusammen.[78] Als Client "kann jeder gängige Web-Browser verwendet werden, zusätzliche Software muß nicht installiert werden"[79]. Eine clientseitige Plattformunabhängigkeit ist somit gegeben, allerdings bedeutet dies auch, daß die Generierung einer Offline-Komponente wie bei den bisher betrachteten, anderen Content Management Systemen mit erheblichen Schwierigkeiten verbunden ist.

NPS kann multimediale Inhalte beliebiger Dateitypen verwalten und bietet hinreichende Möglichkeiten zur Nutzung von Metadaten: "NPS unterscheidet zwischen fünf Typen von Objekten: Publikationen, Dokumente, Bilder, Generische Dokumente und Templates. Aus diesen fünf Objekttypen können beliebige benutzerdefinierte Objektklassen mit frei definierbaren Attributen (z.B. Metadaten) abgeleitet werden. [...] Neue Objektklassen und Attribute können bei laufendem Betrieb definiert werden." Eigene Anwendungen zum Öffnen oder Bearbeiten bestimmter Dateitypen in das System zu integrieren ist allerdings nur dann leicht möglich, wenn diese nur serverseitig verwendet werden. Für diesen Fall stehen Programmierschnittstellen für die Programmiersprachen Java und Tcl sowie XML-Schnittstellen zur Verfügung.[80] Die für den Endbenutzer sichtbaren Ausgaben können jedoch auch in diesem Fall nur im Web-Browser angezeigt werden. Für weitergehende, insbesondere dynamische und interaktive Anwendungen ist diese Erweiterungsmöglichkeit jedoch nicht geeignet.

3.1.1.6 Pironet NDH pirobase®

Die Pironet NDH AG[81] entwickelt das Content Management System pirobase®, das zum Zeitpunkt der CeBIT 2001 in Version 4.2 vorlag. Zum gegenwärtigen Zeitpunkt[82] ist die Version 5.0.3 auf dem Markt. Als klassisches CMS ist pirobase® ausschließlich darauf ausgelegt, Daten zu dem Zweck in das System einzustellen, diese anschließend in webbasierten Umgebungen (Internet, Intranet, Extranet) zur Verfügung zu stellen. Derartige Ausgaben erfolgen stets in Form von HTML-Seiten, die in den marktüblichen Browsern betrachtet werden können. Clientseitig ist auf diese Weise eine weitgehende Plattformunabhängigkeit gegeben[83], die allerdings mit sich bringt, daß dynamische Funktionalitäten auf dem Computer des Endbenutzers auf Java Applets und im Browser ausführbare Skriptsprachen beschränkt sind. Die gleiche Einschränkung gilt auch für die angebotene

---

[75] Homepages: http://www.infopark.de (deutsch), http://www.infopark.com (englisch)
[76] Produktdatenblatt zum Content Management System NPS 5, im Internet zum Download erhältlich unter der Adresse: http://www.infopark.de/de/products/nps/nps550_datasheet.pdf, Seite 1.
[77] A. a. O., Seite 2.
[78] A. a. O., Seite 2.
[79] Broschüre "NPS 5 - Technische Funktionsübersicht", in Internet zum Download erhältlich unter der Adresse: http://www.infopark.de/de/products/Funktionsuebersicht_NPS5.pdf, Seite 5.
[80] Produktdatenblatt zum Content Management System NPS 5, im Internet zum Download erhältlich unter der Adresse: http://www.infopark.de/de/products/nps/nps550_datasheet.pdf, Seite 2.
[81] Homepage: http://www.pironet-ndh.com.
[82] Stand: Mitte Oktober 2003.
[83] Siehe hierzu: pirobase® CMS 5.0.3 White Paper, im Internet zum Download erhältlich unter der Adresse: http://www.pironet-ndh.com/servlet/PB/show/1001982/whitepaper5-0_v12_de.pdf, Seite 6.

Offline-Version, mit der "Teilbereiche aus pirobase CMS® oder der Gesamtauftritt als statische HTML-Seiten in das Filesystem extrahiert werden"[84] können. Eine Integration von dynamischen Softwarekomponenten, insbesondere der im Rahmen der E-Learning-Plattform benötigten Simulationssoftware, ist damit kaum zu realisieren.

Die Datenverwaltung von pirobase® ist ebenfalls in erster Linie auf die Präsentation der Inhalte als HTML-Seiten im Web-Browser ausgerichtet. "Autoren können auf den Inhaltsseiten Abbildungen (GIF- oder JPG-Dateien) einfügen oder Links zu Dateien (Textdateien, Videos, Excel-Tabellen, PDF-Dateien etc.) setzen." Eine Integration von verschiedenen Medien ist also nur dann vorgesehen, wenn diese im Browser innerhalb von HTML-Seiten angezeigt werden können. Andere Dateien sind hingegen nur über Verknüpfungen zugänglich.

"Datenbankobjekte können mit Benutzerrechten, Rechtegültigkeiten, Stichworten und Kategorien versehen werden."[85] Ein Versionskontrollsystem ist ebenfalls enthalten. "pirobase CMS® ist im Standard dreisprachig (deutsch, englisch, französisch) und unterstützt die Verwendung beliebiger Sprachen in Content, Templates und enthaltenen Client-Oberflächen."[86] Die Verwaltung von Metadaten ist jedoch sehr beschränkt und war bis zur Einführung der Version 5.0.2 im Jahre 2002 ausschließlich Administratoren vorbehalten, während normale Autoren diese Möglichkeit bis dahin nicht hatten.[87] Als Datenbank kann nur entweder IBM DB/2 oder Oracle 8i bzw. 9i verwendet werden[88], was zusätzlich eine Abhängigkeit von einem dieser beiden Datenbanksysteme erzeugt. Das System erscheint insgesamt zu unflexibel, um einen Einsatz als Grundlage für die E-Learning-Plattform zu rechtfertigen.

3.1.1.7 Schlußfolgerungen

Die betrachteten Content Management Systeme bieten durchgängig hinreichende Funktionalitäten, um einer Gruppe von berechtigten Autoren die kooperative Bearbeitung und Veröffentlichung von Inhalten zu ermöglichen, die vom Endbenutzer in einem Web-Browser betrachtet werden können. Eine Zugriffskontrolle, die über die Definition von Benutzern und Benutzergruppen geregelt wird, ist durchgängig vorhanden. Eine Mehrsprachigkeit von Benutzeroberfläche und Inhalten ist durchweg gewährleistet. Die Benutzung von beliebigen Metadaten zur Klassifizierung, Verwaltung und Suche ist mit einer Ausnahme ebenfalls in hinreichender Weise möglich.

Probleme ergeben sich jedoch aus der grundlegenden Eigenschaft jedes CMS, daß eine Ansicht der enthaltenen Daten für den Endbenutzer erst durch einen Export in ein für die Betrachtung in einem Web-Browser geeignetes HTML-Format generiert wird. Wegen dieser Einschränkung ist die Schaffung einer Offline-Komponente mit großen Schwierigkeiten verbunden, und die Integration von Anwendungen für alle Benutzer, beispielsweise der in der E-Learning-Plattform benötigten Simulationssoftware, ist praktisch nicht möglich. Außerdem stellt der Export eine Einbahnstraße dar, die keine Möglichkeit für eine Beteiligung der Endbenutzer, beispielsweise für die Erstellung von Protokollen und Lösungen zu Übungsaufgaben durch die Lernenden, vorsieht.

---

[84] A. a. O., Seite 8.
[85] A. a. O., Seite 8.
[86] A. a. O., Seite 8.
[87] Siehe hierzu die Pressemitteilung der Firma Pironet NDH zur Messe Systems 2002, welche unter der Adresse http://www.pironet-ndh.com/servlet/PB/menu/1006180 im Internet zur Verfügung steht.
[88] Siehe hierzu: pirobase® CMS 5.0.3 White Paper, im Internet zum Download erhältlich unter der Adresse: http://www.pironet-ndh.com/servlet/PB/show/1001982/whitepaper5-0_v12_de.pdf, Seite 33.

3.1.2 Dokumentenmanagementsysteme

Im vorigen Abschnitt wurde gezeigt, daß aufgrund der strikten Trennung zwischen dem System für die Datenablage einerseits und der WWW-Oberfläche für den Endbenutzer andererseits Content Management Systeme als Grundlage für die E-Learning-Plattform nicht geeignet sind. Der nächste logische Schritt besteht deshalb in der Suche nach Systemen, in denen diese Trennung nicht besteht, sondern in der für alle Benutzer zuerst einmal ein gleichberechtigter Zugang zum System besteht. Die gegebenen technischen Möglichkeiten, insbesondere zum Erstellen, Bearbeiten und Betrachten von Inhalten, müssen dann anhand der Definition von Benutzern und Benutzergruppen, denen in Abhängigkeit vom jeweiligen Kontext bestimmte Rechte zugewiesen werden, gesteuert werden können.

Dies ist ein typisches Szenario für den Einsatz von Dokumentenmanagementsystemen (DMS). Diese wurden entwickelt, um die Ablage, Bearbeitung, Verteilung und Nutzung von Dokumenten (meist in Unternehmen oder Behörden) zu ermöglichen. Dokumente können dabei in beliebigen elektronischen Formaten vorliegen und oftmals auch aus verschiedenen Teilen zusammengesetzt sein, die jeweils unterschiedliche Medientypen enthalten. Dies reicht von der einfachen Einbettung von Grafiken in HTML-Seiten über die Zusammensetzung von Dateien aus verschiedenen Teilapplikationen von Microsoft® Office® bis hin zu sehr komplexen, beispielsweise in XML-basierten Datenformaten niedergelegten Kombinationen aus den unterschiedlichsten Medien.

Im Gegensatz zu CMS dienen DMS meist nicht der Außendarstellung von Unternehmen (wobei viele DMS allerdings zusätzliche Funktionalitäten in dieser Richtung anbieten), sondern der Verbesserung der Zusammenarbeit innerhalb des Unternehmens. Dies wird auch durch die Homogenität solcher Systeme deutlich, die keine prinzipielle Trennung zwischen dem Bearbeiten von Dokumenten durch Autoren und dem Betrachten von Dokumenten durch Leser kennen, wie dies in CMS durchgängig der Fall ist. Vielmehr ist jeder Benutzer zunächst einmal ein potentieller Autor, dessen Berechtigungen nur durch seine Zugehörigkeit zu bestimmten Benutzergruppen sowie durch die Vergabe von Zugriffsrechten für die einzelnen Dokumente gesteuert werden.

Für die Einbeziehung der Lernenden in den Prozeß der Produktion von Inhalten im Rahmen der E-Learning-Plattform in Form von Seminararbeiten, Übungen, Protokollen oder Versuchsergebnissen ist dieser Punkt von entscheidender Bedeutung. Darüber hinaus bieten die klassischen Dokumentenmanagementfunktionalitäten, die Dokumente als zentrales Element des Systems und nicht als Zwischenstation auf dem Weg zum Endprodukt "Website" betrachten, erheblich bessere Möglichkeiten für die kooperative Bearbeitung von gespeicherten Daten in den unterschiedlichsten Formaten, besonders, weil sie in der Regel über bessere Möglichkeiten zur Integration von formatspezifischen Anwendungen verfügen. Gegenüber den Beschränkungen einer reinen Web-Oberfläche stellt dies einen erheblichen Vorteil dar.

Im Folgenden werden die auf den drei zu Beginn genannten Messen evaluierten DMS in alphabetischer Reihenfolge der Firmennamen dargestellt, wobei zu jedem Produkt die verwendeten Quellen (meist in Form von Internetseiten) angegeben werden. Dabei wird jeweils auch die Eignung der Systeme für die Verwendung in der E-Learning-Plattform überprüft. Zwar gibt es natürlich erheblich mehr verschiedene DMS, als hier dargestellt werden können; die ausgewählten Systeme gehören jedoch zu den gebräuchlichsten und technisch am weitesten entwickelten, die auf dem Markt erhältlich sind.

Bewußt verzichtet wurde hier hingegen auf die Darstellung einiger ebenfalls evaluierter, teils bereits älterer Dokumentenmanagementsysteme, deren Einsatzschwerpunkt auf der Übertragung von Papierdokumenten in elektronische Systeme liegt. Derartige Systeme verfügen über teilweise hochperformante Funktionalitäten zum Einscannen der Seiten von Papierdokumenten und einer anschließenden, mehr oder weniger automatischen Verschlagwortung und Vergabe eines fest vorgegebenen Satzes von Metadaten. Zudem können Workflows, die die eingescannten Dokumente vorher in Papierform durchlaufen hätten, in diesen Systemen auf elektronischem Wege abgebildet werden. Beispiele für derartige Systeme sind die älteren Produkte der Firma Documentum[89], die inzwischen ihre strategische Ausrichtung in den Bereich des Content Managements verlagert hat[90], sowie das Programm Docuware® der Firma Docuware AG[91]. Die Produkte aus diesem Teilbereich des Dokumentenmanagements bringen jedoch für die E-Learning-Plattform keinerlei neue Erkenntnisse und werden daher an dieser Stelle nicht weiter betrachtet.

Das gleiche gilt für einen weiteren Zweig der DMS, in dem eine Spezialisierung auf die Verwaltung der Ausgabeformate von betriebswirtschaftlichen Softwaresystemen wie beispielsweise SAP R/3 vorgenommen wurde. Derartige Programme erzeugen für die unterschiedlichsten Zwecke in vielen Unternehmen Ausgabelisten mit hunderttausenden von Seiten, die beispielsweise für die Buchführung benötigt werden. Natürlich ist auch die Verwaltung solcher elektronischer Dokumente ein klassischer Anwendungsfall für Dokumentenmanagementsysteme. Allerdings gibt es aufgrund der besonderen Formate, in denen die Ausgaben von SAP R/3 und anderen derartigen Programmen vorliegen, für diesen Zweck spezialisierte DMS wie z. B. das Produkt VIDiDOC der Firma BETA Systems Software AG[92], die jedoch aufgrund dieser Spezialisierung gleichermaßen nicht als Grundlage für eine E-Learning-Plattform in Frage kommen und deshalb in der folgenden Aufstellung ebenfalls nicht mit betrachtet werden.

3.1.2.1 CEYONIQ Solutions

Die CEYONIQ Technology GmbH[93] entwickelt und vertreibt unter dem Oberbegriff "CEYONIQ Solutions" ein aus verschiedenen, aufeinander aufbauenden Komponenten bestehendes Dokumentenmanagementsystem. "Das Kernprodukt ist der CEYONIQ Server, der als plattformunabhängiges Storage Management-System in bestehende Infrastrukturen integriert werden kann. Weitere Produkte sind der CEYONIQ Archive Manager, eine flexible Dokumenten-Management-Plattform"[94] sowie unterschiedliche, spezialisierte Komponenten zur Verwaltung von Druckdaten und Ausgabedaten von verschiedenen betriebswirtschaftlichen Softwaresystemen.

Die CEYONIQ Software ist in der Lage, Inhalte in beliebigen Formaten zu verwalten. Mechanismen zur Versionskontrolle stehen ebenso zur Verfügung wie die technischen Möglichkeiten für eine dauerhafte Archivierung von Daten unter Anwendung eines leistungsfähigen Backup-Systems. Dazu kommen eine umfassende Benutzerverwaltung und ein System zur Vergabe und Kontrolle von dokumentbezogenen Zugriffsrechten.

---

[89] Homepages: http://www.documentum.de (deutsch), http://www.documentum.com (englisch).
[90] Siehe Kapitel 3.1.1.2.
[91] Homepage: http://www.docuware.com.
[92] Homepage: http://www.betasystems.com.
[93] Homepage: http://www.ceyoniq.de. Die CEYONIQ Technology GmbH führt seit dem 1. Juli 2002 die das Kerngeschäft der ehemaligen CEYONIQ AG fort, die Anfang 2002 Insolvenz anmelden mußte. Die Rechte an den hier evaluierten Programmen wurden von der neuen Firma übernommen.
[94] Kurzes Unternehmensportrait der CEYONIQ Technology GmbH, im Internet zum Download erhältlich unter der Adresse: http://www.ceyoniq.de/pages/presse/files/pdf/Unternehmensportrait.PDF, Seite 1.

Dem gegenüber steht leider ein vollkommen unzureichendes System zur Verwaltung von Metadaten[95]. Die Verwaltung von selbstdefinierten Metadaten für unterschiedliche Inhaltstypen, die in Kapitel 2.2 als Anforderung aufgestellt wurde, erscheint mit diesem System daher nicht möglich.

Außerdem ist die Schaffung einer Offline-Komponente mit erheblichen Schwierigkeiten verbunden. "Der CEYONIQ Server verfügt über eine moderne und objektorientierte Programmierschnittstelle. Über diese können eigene Anwendungen mit dem CEYONIQ Server-System verbunden werden. Diese CEYONIQ Server API ist für die Sprachwelten C++, Java und Microsoft.NET verfügbar und ermöglicht, die Fähigkeiten des CEYONIQ Server in eigenen Programmen zu nutzen."[96] Diese Möglichkeiten zur Erweiterung des Systems basieren jedoch ausschließlich auf dem Einsatz eines zentralen Servers, in den zwar relativ problemlos eigene Anwendungen in Form von Web Services integriert werden können; die clientseitige Speicherung und Nutzung von Daten ohne Kontakt zum zentralen Server ist jedoch nicht vorgesehen. Da zudem die Auswahl der einsetzbaren Datenbanken auf MS SQL Server®, Oracle und IBM DB2 beschränkt ist, bietet das System insgesamt nicht die notwendige Flexibilität, um für die Verwendung als Grundlage der E-Learning-Plattform in Frage zu kommen.

3.1.2.2 FileNET Panagon™

Die FileNET Corporation[97] brachte im Jahre 1998 unter dem Namen Panagon™ ein kombiniertes Dokumentmanagement- und Workflow-System auf den Markt, das nach der Akquisition einiger kleinerer Konkurrenten und der danach erfolgten Integration von deren Produkten jetzt unter dem Namen FileNET P8 vertrieben wird.[98] Das System kann "strukturierte und unstrukturierte Daten, elektronische Dokumente, eingescannte Dokumente, Verbunddokumente, Bilder, Audiodateien, Videodateien, Formulare, E-Mails und andere"[99] verwalten, "unterstützt das Management komplexer Dokumente z. B. durch Versionskontrolle, Check-in und Check-out und Sicherheitsmechanismen"[100] und ermöglicht ein "schnelles und einfaches Erstellen und Verwalten neuer Inhalte für Mitarbeiter der Fachabteilungen, indem die entsprechenden Metadaten für die Indizierung und Klassifizierung der Inhalte automatisch zugewiesen werden"[101].

Die Definition von eigenen Metadaten und insbesondere eigenen Objekttypen, denen dann ein beliebiger Satz von Metadaten zugewiesen werden kann, ist jedoch offenbar nicht vorgesehen.[102] Die in das System integrierte Suche "ermöglicht die einfache Lokalisierung von Ordnern, Objekten und Dokumenten über mehrere Speichersysteme

---

[95] Bezeichnenderweise findet sich bei einer Volltextsuche auf sämtlichen Unterseiten der Homepage der CEYONIQ Technology GmbH (durchgeführt am 14. Oktober 2003) überhaupt nur ein einziges Mal das Wort "Metadaten", nämlich in der Beschreibung eines Fremdproduktes (EMC Cetera).

[96] CEYONIQ Server, Der strategische Baustein für zukunftssichere Archivierung, Broschüre der CEYONIQ Technology GmbH, im Internet zum Download erhältlich unter der Adresse: http://www.ceyoniq.de/pages/produkte/files/pdf/CEYONIQ_Server.pdf, Seite 5.

[97] Homepages: http://www.filenet.de (deutsch), http://www.filenet.com (englisch).

[98] Stand: Mitte Oktober 2003.

[99] FileNET Content Manager, Broschüre der FileNET Corporation, im Internet zum Download erhältlich unter der Adresse: http://www.filenet.com/deutsch/Produkte/datenblatt/030420005.pdf, Seite 2.

[100] A. a. O., Seite 4.

[101] A. a. O., Seite 4.

[102] Auch in diesem Fall findet sich bei einer Volltextsuche auf sämtlichen Unterseiten der Homepage der FileNET Corporation (durchgeführt am 14. Oktober 2003) bezeichnenderweise nur zweimal das Wort "Metadaten", und zwar in der Beschreibung eines Fremdproduktes (Lotus Notes) und in der Vorstellung einer Partnerfirma (APUS4 AG).

und Datenbanken durch die Suche nach Eigenschaften und Inhalten, inkl. phonetische Suche, Suche mit Wortstämmen, Synonymen, Wild Cards etc."[103] Die automatisch generierten Metadaten dienen hingegen eher der internen Klassifikation und stehen nicht im Fokus der dem Benutzer angebotenen Suchfunktionalitäten. Eine Erweiterung des Systems in dieser Hinsicht würde einen erheblichen Aufwand mit sich bringen.

Das DMS von FileNET ist nur für die Betriebssysteme Microsoft® Windows® und Sun Solaris erhältlich. Als Datenbanken stehen ausschließlich Microsoft® SQL Server® und Oracle zur Verfügung. Neben dieser Abhängigkeit von wenigen Systemen fällt auf, daß keinerlei offene Schnittstellen zur Integration eigener Anwendungen zur Verfügung stehen. Während für bestimmte betriebswirtschaftliche Programme Möglichkeiten einer Zusammenarbeit vorgesehen sind, würde die Anpassung an eine eigene einzubringende Software umfassende Eingriffe in das System erfordern. Da zudem kein eigenständiger Client mit einer separaten Datenverwaltung vorgesehen ist, was die Schaffung einer Offline-Komponente ebenfalls erheblich erschwert, erweist sich auch das System von FileNET als nicht geeignet, um als Basis für die E-Learning-Plattform zu fungieren.

3.1.2.3 Hummingbird DOCSFusion / Hummingbird DM™

Die Firma Hummingbird Ltd.[104] "ist einer der weltweit führenden Anbieter von Enterprise Information Management Systemen (EIMS) [...] Heute vertrauen fünf Millionen Benutzer auf Lösungen von Hummingbird, um Inhalte im Unternehmen zu integrieren, zu verwalten, zu veröffentlichen oder zu durchsuchen und gemeinsam zu nutzen."[105] Der Ausdruck "Enterprise Information Management" wird von Hummingbird als Oberbegriff für Dokumentenmanagementsysteme und weitere von dieser Firma entwickelte Produkte verwendet, die aufeinander aufbauen und insgesamt dem Zweck dienen, in Unternehmen die Speicherung, Verteilung und Nutzung vorhandener Informationen in elektronischen Systemen zu ermöglichen.

Eine Kernanwendung dieses Softwaresystems, das unter dem Namen Hummingbird Enterprise™ vertrieben wird, ist das Dokumentenmanagementsystem, das heute unter dem Namen Hummingbird DM™ erhältlich ist. Zum Zeitpunkt der Marktanalyse wurde es noch unter dem Namen DOCSFusion vertrieben. Zwei zum damaligen Zeitpunkt separat erhältliche Zusatzprodukte waren der in Microsoft® Windows® integrierbare Client PowerDOCS und die plattformunabhängige WWW-Oberfläche CyberDOCS, die inzwischen in Hummingbird DM™ integriert wurden.[106]

Das DMS von Hummingbird bietet alle von einem derartigen System zu erwartenden Funktionalitäten, insbesondere die Speicherung von Daten beliebiger Inhaltstypen und die Verwaltung von Metadaten, die bei Bedarf frei konfiguriert werden können. Dabei kann der Metadatensatz für unterschiedliche Dateitypen und Anwendungszwecke völlig unabhängig voneinander definiert werden, was auch die Speicherung von verschiedenen

---

[103] FileNET Content Manager, Broschüre der FileNET Corporation, im Internet zum Download erhältlich unter der Adresse: http://www.filenet.com/deutsch/Produkte/datenblatt/030420005.pdf, Seite 4.

[104] Homepage: http://www.hummingbird.com; die deutschsprachige Version dieser Internet-Angebotes ist unter der Adresse http://www.hummingbird.com/international/germany/index.html zugänglich.

[105] Zitat von der deutschsprachigen Version der Internet-Homepage der Firma Hummingbird Ltd. unter der Adresse http://www.hummingbird.com/international/germany/index.html.

[106] Siehe hierzu die Übersicht über die Umbenennung und Neustrukturierung der Hummingbird-Produkte (Product Rebranding Glossary), die im Internet zur Einsicht zur Verfügung steht unter der Adresse: http://www.hummingbird.com/services/productrebranding.html.

Metadaten, die von unterschiedlichen Anwendungen benötigt werden, ermöglicht.[107] Die Systemkomponente "Hummingbird DM Designer ermöglicht die schnelle bedarfsgerechte Anpassung von Datenbankstrukturen und Formularen, indem es den Entwurf von Repositories für die Aufnahme von Metadaten vereinfacht."[108] Allerdings fordert dieser Aussage zufolge die Anpassung des Metadatensatzes jedes Mal Veränderungen in die Datenbankstruktur. Derart tiefreichende Eingriffe sollten jedoch nach Möglichkeit nicht vorgenommen werden müssen.[109]

Eine Möglichkeit zur Suche in den gespeicherten Metadaten ist ebenso im System enthalten wie eine Volltextsuche. Benutzer und Zugriffsrechte werden ebenso verwaltet wie unterschiedliche Versionen von Dokumenten. Hummingbird DM™ bietet für den Zugriff auf die gespeicherten Daten durch Benutzer einerseits eine WWW-Oberfläche, andererseits ermöglicht ein in das Betriebssystem Microsoft® Windows® integrierbarer Client, daß in Umgebungen, in denen dieses Betriebssystem verwendet wird, "der Benutzer von vertrauten Desktop-Schnittstellen aus, z. B. dem Windows-Desktop, dem Windows-Explorer, Microsoft Outlook sowie den gängigen Anwendungen zur Erstellung von Dokumenten, auf die Dokumentenmanagement-Funktionen von Hummingbird zugreifen kann."[110]

Als Besonderheit gegenüber den meisten anderen Dokumentenmanagementsystemen bietet der Windows-Client dabei sogar eine Offline-Komponente, um den Zugriff auf Dokumente auch dann zu ermöglichen, wenn ein Zugang zum Netzwerk nicht möglich ist, beispielsweise bei Server- oder Netzwerkausfällen oder wenn Benutzer mit mobilen Computern unterwegs sind. Diese Funktionalität steht allerdings nur bei Verwendung des Clients für Microsoft® Windows® zur Verfügung. Die geplante Offline-Komponente für die E-Learning-Plattform auf diese Weise zu realisieren, würde deshalb bedeuten, eine ausschließliche Lauffähigkeit auf diesem Betriebssystem in Kauf zu nehmen, was angesichts des heterogenen Umfeldes gerade im universitären Bereich nicht ratsam ist.

Auch auf der Serverseite besteht eine Abhängigkeit von den Betriebssystemen der Firma Microsoft®, da das DMS Hummingbird DM™ nur für Windows® 2000 Server bzw. Advanced Server und Windows® NT Server 4.0 zur Verfügung steht.[111] Darüber hinaus ist eine Abhängigkeit von den WWW-Servern Microsoft Internet Information Server (IIS) 5.0 oder 4.0 bzw. alternativ iPlanet 4.0.1 oder 4.0 gegeben.[112] Für die Speicherung der Daten kann nur eine der Datenbanken Microsoft SQL Server 2000 und 7.0, Oracle 9i (9.0.1) und Oracle 8i (8.1.7) oder Sybase System 12.5 und 12 verwendet werden.[113] In jedem Fall entstünde eine erhebliche Abhängigkeit von gleich mehreren kommerziellen Produkten, zusätzlich zum DMS von Hummingbird selbst.

Ein weiteres Hindernis stellt außerdem auch hier die Schwierigkeit der Einbindung der für die E-Learning-Plattform erforderlichen Softwareprogramme für die dynamischen Lehrsimulationen dar. Schnittstellen zur Integration derartiger Komponenten ins System sind nicht vorhanden, die Programme könnten allenfalls betriebssystemabhängig als für das Öffnen bestimmter Dateitypen zuständige Software definiert werden. In diesem Fall

---

[107] Siehe hierzu das Hummingbird DM™ Datenblatt, das im Internet zum Download erhältlich ist unter der Adresse http://mimage.hummingbird.com/alt_content/binary/pdf/collateral/ds/german/dmds-de.pdf.
[108] A. a. O., Seite 2.
[109] Die Kapitel 5.3 und 8.2 zeigen, auf welche Weise dies in der E-Learning-Plattform vermieden wurde.
[110] Hummingbird DM™ Datenblatt, im Internet zum Download erhältlich unter der folgenden Adresse: http://mimage.hummingbird.com/alt_content/binary/pdf/collateral/ds/german/dmds-de.pdf, Seite 2.
[111] A. a. O., Seite 2.
[112] A. a. O., Seite 2.
[113] A. a. O., Seite 2.

müßten jedoch stets von Betriebssystem zu Betriebssystem unterschiedliche Versionen der bei der Installation zu treffenden Systemeinstellungen an die Benutzer ausgeliefert werden. Außerdem wäre auf diese Weise nur eine lockere Anbindung der Software an das System gegeben, bei der nur in eine Richtung Daten von der E-Learning-Plattform an die einzelnen Programme übermittelt würden. Die Nutzung der vom System allen Anwendungen zur Verfügung gestellten Funktionalitäten müßte hingegen stets neu implementiert werden und könnte dadurch von Programm zu Programm unterschiedlich aussehen, was dem Prinzip der möglichst weitgehenden Integration gleichberechtigter Anwendungen in die Plattform widerspricht.

3.1.2.4 Hyperwave Information Server

Im Vergleich mit den anderen hier betrachteten Systemen kann Hyperwave[114] auf eine (zumindest nach Internet-Maßstäben) lange Geschichte zurückblicken. Der Hyperwave Information Server ging aus dem hypermedialen Informationssystem Hyper-G hervor, das ab 1989 am Institut für Informationsverarbeitung und Computergestützte neue Medien (IICM)[115] an der Technischen Universität Graz[116] unter der Leitung von Professor Hermann Maurer entwickelt wurde.

Dabei folgte man der Prämisse, daß eine Struktur von durch einfache Verknüpfungen verbundenen Dokumenten, wie sie im zur gleichen Zeit entwickelten WWW vorliegt, für die Speicherung großer Mengen von Informationen in Hypermedia-Systemen nicht ausreichend sei: "a large set of documents just connected by links would invariably lead to confusion, difficulty in maintenance, and information recall. Even before the real WWW boom started, the scientific community dealing with Hypermedia (as it was called by then) realized that the link-paradigm on which ordinary WWW is based would not be sufficiently powerful when confronted with huge amounts of documents. [...] Based on this observation investigations on how to prevent chaos in large networked information systems were started around 1989, slowly developing into a first prototype system called Hyper-G."[117]

Hyper-G wurde zunächst als Gegenentwurf zum WWW präsentiert[118]. Der wichtigste Unterschied liegt dabei in einer wesentlich strukturierteren Ablage der Informationen, die unter anderem auf der Verwendung von Metadaten (die im Rahmen von Hyper-G als Attribute bezeichnet wurden), bidirektionalen Links und einem Link-Management zur Verwaltung von Verknüpfungen beruht und Möglichkeiten zu einer voneinander unabhängigen Einordnung von Dokumenten nach hierarchischen und thematischen Gesichtspunkten bietet. Einen Überblick darüber gibt das folgende Zitat:

"Hyper-G combines the intuitiveness of top-down hierarchical navigation with the immediacy of associative hyperlinks and the power of focussed attribute and content searches. [...] Documents may be grouped into aggregate *collections*, which may themselves belong to other collections and which may span multiple Hyper-G servers,

---

[114] Homepages: http://www.hyperwave.de bzw. http://www.hyperwave.com, beide mehrsprachig.
[115] Homepage: http://www.iicm.edu. Konsequenterweise betreibt das IICM seine eigenen Internet-Seiten unter Verwendung des Hyperwave Information Servers.
[116] Homepage: http://www.tugraz.at.
[117] Hermann Maurer, Hyperwave and Hyper-G, in: Encyclopedia of Computers and Computer History, Vol. One (Ed.: R. Rojas), Fitzroy Dearborn Publishers, Chicago, 2001, Seiten 383-384. Dieser Beitrag ist am IICM zum Download erhältlich unter der Adresse: http://www.iicm.edu/iicm_papers/hyperwave.doc.
[118] Siehe hierzu: F. Kappe, H. Maurer, N. Scherbakov, Hyper-G - a Universal Hypermedia System, in: Journal of Educational Multimedia and Hypermedia, Vol. 2, Nr. 1, 1993, Seiten 39-66.

providing a unified view of distributed resources. A special kind of collection called a *cluster* is used to form multimedia and/or multilingual aggregates (the appropriate language version of a document to be displayed is selected according to the user' s language preference setting). Documents and collections may belong to multiple parent collections, opening up the possibility of providing multiple views of the same information"[119].

Neben diesen Möglichkeiten zur Strukturierung von Datenbeständen bildet das System zum Link-Management ein wesentliches Merkmal: "Hyperlinks in Hyper-G connect a *source anchor* within one document to either a *destination anchor* within another document, an entire document, or a collection. Links are not stored within documents but in a separate link database: hence links are not restricted to text documents, they can be followed backwards, they are updated and deleted automatically when their destination moves or is deleted (no "dangling links"), and they are easy to visualise graphically."[120] Tatsächlich ist es aufgrund dieser Speichermethode auch heute noch ausgeschlossen, durch das Verfolgen von Links innerhalb eines Hyperwave-Servers einen "HTTP Error 404: Not Found" zu erhalten.[121]

Nachdem die genannten Funktionalitäten von Hyper-G nicht wie erhofft als Grundlage für das weltweite Informationssystem benutzt wurden und sich statt dessen das weniger strukturierte WWW durchgesetzt hatte, wurde das System, das zu diesem Zeitpunkt bereits mit WWW Clients benutzt werden konnte, dahingehend weiterentwickelt, daß zumindest auf Basis einzelner Server oder Gruppen von Servern die Möglichkeiten zur strukturierten Ablage von Informationen genutzt werden konnten. Die auf diese Weise entwickelte Software kann sowohl für die Verwaltung von Intranets und kompletten Internet-Auftritten als auch als Dokumentenmanagementsystem eingesetzt werden, was in diesem Fall keinen prinzipiellen technischen Unterschied bedeutet, sondern lediglich auf einer unterschiedlichen Nutzung derselben, im System vorhanden Funktionalitäten beruht. Mit Gründung der Hyperwave AG im Jahre 1997 wurde Hyper-G in Hyperwave Information Server umbenannt.

Neben der Speicherung von beliebigen und von Dokumenttyp zu Dokumenttyp beliebig variierbaren Metadaten sowie umfassenden Suchmöglichkeiten sowohl in Metadaten als auch im Volltext praktisch aller gebräuchlichen textuellen Datenformate bietet der Hyperwave Information Server ein Benutzer- und Gruppenmanagement und ein System von Zugriffsrechten auf Dokumentebene.[122] Über 200 verschiedene Dateitypen können vom System automatisch erkannt und verarbeitet werden, was die Verwendung weiterer Typen jedoch keineswegs ausschließt.[123] Jeder Benutzer kann prinzipiell die Rolle eines Autors übernehmen, Einschränkungen dieser Möglichkeiten werden ausschließlich über die Berechtigungen der einzelnen Benutzer und Gruppen vorgenommen.

---

[119] Keith Andrews, Frank Kappe, Hermann Maurer, Serving Information to the Web with Hyper-G, in: Computer Networks and ISDN Systems, 27 (6), 1995, Seiten 919-926. Der gleiche Beitrag ist ebenfalls erschienen in: Proceedings of the Third International World Wide Web Conference, Darmstadt, 1995. Eine HTML-Version kann auf den Internet-Seiten der Fraunhofer-Gesellschaft unter der Adresse http://www.igd.fhg.de/archive/1995_www95/proceedings/papers/105/hgw3.html eingesehen werden.

[120] A. a. O.

[121] Für weitere Informationen über den wissenschaftlichen Hintergrund, die ursprüngliche Funktionalität und die Geschichte von Hyper-G siehe: Keith Andrews, Frank Kappe, Hermann Maurer, The Hyper-G Network Information System, in: Journal of Universal Computer Science, Vol. 1, Nr. 4 (Special issue: Proceedings of the Workshop on Distributed Multimedia Systems, held in Graz, Austria, Nov. 1994), April 1995, Seiten 206-220.

[122] Siehe hierzu das Datenblatt der Hyperwave AG zum Hyperwave Information Server 6, im Internet erhältlich unter der Adresse http://www.hyperwave.de/d/downloads/documents/is6_datasheet_de.pdf.

[123] A. a. O., Seite 2.

Von besonderem Interesse für die vorliegende Ausarbeitung ist die Tatsache, daß die Hyperwave AG im Jahre 1999 unter dem Namen Hyperwave eLearning Suite eine auf dem Hyperwave Information Server basierende E-Learning-Plattform auf den Markt gebracht hat. Mit seinen Möglichkeiten zur strukturierten Ablage und kombinierten Anzeige multimedialer Informationen, insbesondere in Form der im obenstehenden Zitat erwähnten sogenannten Cluster, ist der Hyperwave Information Server geradezu prädestiniert dafür, als Plattform für elektronische Lehrmaterialien zu dienen, solange diese aus statischen (Text, Bild) oder nicht-interaktiven dynamischen (Audio, Video) Inhalten bestehen, die im Rahmen eines Internet-Browsers angezeigt werden können.

Darüber hinaus wurde aber bereits frühzeitig von Seiten der Hyperwave AG erkannt, daß ein rein computerbasiertes Lernen durch die fehlende Interaktion zwischen den Lehrenden und Lernenden sowie zwischen den Lernenden untereinander auch Nachteile mit sich bringt: "Den rein 'Technologie-basierenden' Trainings fehlen oft Werkzeuge für Kommunikation und Zusammenarbeit. Diesem Nachteil begegnet die Hyperwave eLearning Suite mit der Möglichkeit, sowohl kurs- als auch ausbildungsgangsspezifisch alle beteiligten Parteien (Trainees, Tutoren und Trainer) via E-Mail, Chat oder Diskussionsforen zusammenzuführen."[124] Außerdem ist ein elektronisches "Schwarzes Brett" im Lieferumfang enthalten. Mit diesen Funktionalitäten bietet die Hyperwave eLearning Suite bereits die erforderlichen Kommunikationsmöglichkeiten, wie sie in Kapitel 2.8 als Anforderungen für die E-Learning-Plattform beschrieben wurden.

Ein entscheidendes Problem bringt jedoch die Tatsache mit sich, daß es sich bei der Hyperwave eLearning Suite um ein komplett webbasiertes System handelt, wodurch die Integration von eigenen Anwendungen zum Editieren und Ausführen dynamischer, interaktiver Inhalte wie Lehrsimulationen erheblich erschwert wird, solange diese nicht in Form von Java-Applets oder im Browser ausführbaren Skripten vorliegen. Hinzu kommt, daß auch serverseitig die Integration von Software auf Schwierigkeiten stößt, da die sogenannten Templates, über die im Hyperwave Information Server sowohl das Layout der Präsentation der Inhalte für die Benutzer als auch die in den ausgelieferten HTML-Seiten zur Verfügung gestellten Interaktionsmöglichkeiten wie z. B. Menüs und Buttons gesteuert werden, in einer von zwei proprietären Skriptsprachen programmiert werden müssen. Dabei handelt es sich um "Place", eine Eigenentwicklung der Firma Hyperwave, sowie um Server Side Java Script, eine Entwicklung der Firma Netscape[125], die von der Hyperwave AG um eine eigene Objektbibliothek für die Verwendung der im Hyperwave Information Server gespeicherten Dokumente erweitert wurde.[126]

---

[124] Datenblatt der Hyperwave AG zur Hyperwave eLearning Suite, im Internet zum Download erhältlich unter der Adresse http://www.hyperwave.de/d/downloads/documents/els_datasheet_de.pdf, Seite 2.

[125] Homepage: http://www.netscape.com.

[126] Bei der Programmierung von derartigen Templates kommt es immer wieder zu Schwierigkeiten, die dadurch entstehen, daß im selben Template clientseitiges und serverseitiges JavaScript mit ihren teilweise verschiedenen Objekten gleichzeitig verwendet werden oder sogar vielfach serverseitige JavaScripts dazu verwendet werden, beim Ausliefern von HTML-Seiten in Abhängigkeit vom jeweils aktuellen Kontext (z. B. den Zugriffsrechten des gerade eingeloggten Benutzers) dynamisch clientseitige JavaScripts zu programmieren. Werden gleichzeitig obendrein noch Befehle der Platzhalter-Sprache Place verwendet, die auf bestimmte auf dem Server vorhandene Objekte referenzieren, oder wird sogar von der im System vorhandenen Möglichkeit Gebrauch gemacht, Place-Befehle durch eine serverseitige JavaScript-Funktion ausführen zu lassen, entstehen Templates, die auch den erfahrensten Programmierer verwirren können und eine Erweiterbarkeit oder auch nur einfache Änderungen praktisch unmöglich machen. (Als Quelle für diese Aussage dienen eigene Erfahrungen des Autors mit der Programmierung von Templates für den Hyperwave Information Server während einer 16monatigen Anstellung im Hyperwave Consulting Team der CMG Industrie GmbH, München, von September 1998 bis Dezember 1999. Technische Informationen zum Hyperwave Information Server, insbesondere auch zur Template-Programmierung, stehen im Download-Bereich auf der Homepage der Hyperwave AG, http://www.hyperwave.de, zur Verfügung.)

Aufgrund der fehlenden Offline-Komponente[127], insbesondere der nicht vorhandenen Möglichkeiten zur serverunabhängigen Speicherung und Bearbeitung von Dokumenten auf dem Client, die allenfalls durch einen Download aus dem System heraus auf die lokale Festplatte möglich wäre, ist der Hyperwave Information Server in seiner derzeit an die Kunden ausgelieferten Form trotz der durch die Hyperwave eLearning Suite zur Verfügung gestellten zusätzlichen Funktionalitäten für die Umsetzung der im Rahmen der vorliegenden Arbeit aufgestellten Anforderungen (noch) nicht geeignet.

Ein etwas anderer Blickwinkel ergibt sich jedoch, wenn man einen besonderen Service der Hyperwave AG für Universitäten und andere Bildungseinrichtungen in die Betrachtungen mit einbezieht, der dieses System von allen anderen hier evaluierten Produkten abhebt. Selbst aus dem universitären Bereich stammend, stellt die Firma unter dem Titel Hyperwave Academic User Program (HAUP)[128] Benutzern aus dem akademischen Bereich die Möglichkeit zur Verfügung, kostenlose Lizenzen für die Benutzung ihrer Produkte zu erwerben, solange diese nicht für kommerzielle Zwecke eingesetzt wird und bestimmte Informationen über die damit betriebenen Projekte im Rahmen des HAUP zur Verfügung gestellt werden. Dafür werden auch Dokumentationen und weitere technische Informationen zu den Hyperwave-Produkten zur Verfügung gestellt, anhand deren auch Weiterentwicklungen für eigene Zwecke möglich sind.

Daraus ergeben sich im Prinzip zwei Möglichkeiten, wie eine Nutzung des Hyperwave Information Servers als Basis für die im Rahmen der vorliegenden Arbeit zu schaffende E-Learning-Plattform realisiert werden könnte. Die eine Möglichkeit bestünde darin, auf dem Computer jedes einzelnen Benutzers den kompletten Hyperwave Information Server zu installieren und diesen dann um eine einheitliche Integrationsmöglichkeit für die zu verwendende Simulationssoftware zu ergänzen. Auf diese Weise könnte man sowohl die Kommunikationsfunktionalitäten der Hyperwave eLearning Suite verwenden als auch von der aus der Möglichkeit Gebrauch machen, rechnerübergreifend Inhalte nach thematischen Gesichtspunkten sortiert zusammenzustellen und auch über die Grenzen einzelner Computer hinweg Suchanfragen sowohl nach Daten als auch nach Metadaten auszuführen, wie dies anfangs für den Einsatz von Hyper-G als weltweit vernetztes Informationssystem geplant war.

Eine derartige Lösung ist jedoch aus mehreren Gründen praktisch nicht durchführbar. Zum einen ist der Installationsaufwand, der sich für jeden einzelnen Benutzer ergeben würde, unvertretbar hoch, zumal nicht nur der Hyperwave Information Server selbst, sondern auch noch eine der beiden vom System unterstützten Datenbanken installiert werden müßte. Zur Auswahl stehen mit Oracle 8i (8.1.7) und Microsoft® SQL Server 2000 zwei kommerzielle Produkte, für die im Gegensatz zum Hyperwave Information Server selbst keine kostenlosen Lizenzen für den akademischen Bereich zur Verfügung stehen. Die Preise für diese Produkte liegen jedoch für den studentischen Endbenutzer in einem unerschwinglichen Bereich. Die Entwicklung und Auslieferung einer von der Hyperwave AG selbst entwickelten und in früheren Produktversionen mit enthaltenen Datenbank wurde vor einigen Jahren eingestellt, kostenlose Datenbanken wie MySQL werden (noch?) nicht unterstützt, und die Entwicklung einer eigenen Schnittstelle des Systems zu einer solchen Datenbank würde einen zu großen Aufwand bedeuten.

---

[127] Für einen Einsatz ohne Netzverbindung, beispielsweise auf Laptops, die von Außendienstmitarbeitern bestimmter Kunden in Regionen mitgenommen werden, in denen eine flächendeckende Versorgung mit Internet-Anschlüssen noch nicht gegeben ist, existiert die Möglichkeit eines Exports von Informationen auf eine CD. Allerdings fehlen hierbei die Möglichkeiten zum Bearbeiten der Dokumente, und jegliche Updates müssen wieder in Form eines neuen CD-Exportes erfolgen.

[128] Homepage: http://www.haup.org.

Die Alternative läge darin, den Hyperwave Information Server und die Hyperwave eLearning Suite lediglich als serverseitige Komponente zu verwenden und zusätzlich für den Einsatz auf den Clients eine eigene, plattformunabhängige Offline-Komponente für das System zu entwickeln, die über eine eigene Datenbank verfügen müßte und in der dann eine flexible Integrationsmöglichkeit für die zu verwendende Simulationssoftware geschaffen werden müßte. Allerdings wäre der Aufwand für eine derartige Anpassung mit dem für eine komplette Eigenentwicklung eines Systems zur Datenspeicherung vergleichbar oder läge wegen der sich dabei ergebenden Notwendigkeit zur Verwaltung der komplexen, für den Hyperwave Information Server spezifischen Datenstrukturen sogar noch höher.

Angesichts dieser Umstände ist eine Eigenentwicklung vorzuziehen, da in diesem Fall die Abhängigkeit von einem lizenzpflichtigen, kommerziellen Produkt vermieden werden kann und statt dessen das eigene System einerseits im Rahmen einer Open-Source-Politik zur Verfügung gestellt werden kann und damit andererseits auch prinzipiell die Möglichkeit erhalten bleibt, zur Finanzierung eines Dauerbetriebes und der technischen Weiterentwicklung des E-Learning-Systems ggf. selbst Lizenzen für den außeruniversitären Bereich kostenpflichtig vergeben zu können.

3.1.2.5 OpenText Livelink und BASIS

Die OpenText Corporation[129] entwickelt und vertreibt mit ihrem Produkt Livelink ein Dokumentenmanagementsystem, bei dem der zentrale Fokus auf Funktionalitäten zum elektronischen Management von Workflows und auf der Organisation der Kooperation und Kommunikation in all denjenigen Arbeitsschritten in Unternehmen liegt, bei denen Dokumente involviert sind. "Livelink verbessert die Zusammenarbeit, fördert den Informationsfluss, beschleunigt Prozesse und steigert damit die Innovationskraft und Effizienz des Unternehmens. Mit Komponenten für virtuelle Besprechungen (online Meetings), Informationsrecherche (Suche), Notifikation, Workflows sowie Dokumenten Management Funktionen, unterstützt Livelink die dynamische Zusammenarbeit und den Austausch von Wissen zwischen Einzelpersonen, Teams und Organisationen."[130]

Die im vorstehenden Zitat genannten Funktionalitäten sind für die E-Learning-Plattform bis auf die Kommunikationsmöglichkeiten und die Mechanismen zur Speicherung und zum Wiederauffinden von Dokumenten von geringer Bedeutung und werden daher hier nicht weiter betrachtet, obwohl sie einen wesentlichen Bestandteil von Livelink darstellen. Für die Datenspeicherung verfügt Livelink über alle Funktionalitäten, die von einem typischen Dokumentenmanagementsystem zu erwarten sind. "Livelink speichert und verwaltet alle Arten von Objekten - von einfachen und zusammengesetzten Dokumenten zu Suchanfragen und URLs - und bietet kontrollierten Benutzerzugriff zu diesen Objekten. Es verfolgt und verwaltet zahllose Versionen, Attribute, Dokumentenbeziehungen [...] Die Ansicht von Dokumenten mit Livelink ist unkompliziert, auch wenn Sie nicht über die Anwendung verfügen, mit der das Dokument erstellt wurde. Livelink konvertiert Dokumente automatisch in HTML, so dass Sie sie sofort in Ihrem Browser sehen können."[131] Letzteres gilt natürlich nur für diejenigen Textformate, die dem Programm bekannt sind und für die HTML-Konverter vorhanden sind.

---

[129] Homepages: http://www.opentext.de (deutsch), http://www.opentext.com (englisch).

[130] Zitat aus den Produktinformationen über Livelink auf der Homepage der OpenText Corporation unter http://www.opentext.de/de/livelink/index.html.

[131] Livelink® Produktübersicht, Version 9.0.0.1, im Internet zum Download erhältlich unter der Adresse http://www.opentext.de/livelink/download/Livelink_9_0_0_1_Produktübersicht.pdf, Seite 7.

Das Programm verfügt ebenfalls über eine Benutzerverwaltung und ein System zur Vergabe von verschiedenen Berechtigungsstufen. Die "Protokollfunktion von Livelink erfasst automatisch Datum und Zeit einer Aktion, wer die Aktion ausführte, und eine Beschreibung der Aktion [...] zum Beispiel, welche Personen an einem Dokument gearbeitet [...] haben"[132]. Livelink ermöglicht die Verwaltung beliebiger Metadaten, die hier als Attribute bezeichnet werden, wobei für unterschiedliche Dokumenttypen auch unterschiedliche Metadatensätze verwendet werden können: "Livelink-Benutzer können jetzt Kategorien und Attribute erstellen - Jede Gruppe und jede Abteilung in einer Organisation bestimmt die benutzerspezifischen Kategorien und Attribute, die sie für ihre speziellen Ansprüche erstellt haben möchte, ohne den Livelink-Administrator kontaktieren zu müssen."[133]

Für die Verwaltung großer Mengen von Dokumenten bietet die OpenText Corporation neben Livelink ein weiteres Dokumentenmanagementsystem namens BASIS[134] an, das sowohl separat betrieben als auch in Livelink integriert werden kann. BASIS bietet ebenfalls die Möglichkeit zur Verwaltung von beliebigen Dokumenttypen und verwaltet diese unter Verwendung von flexibel definierbaren Metadatensätzen. Es verfügt über Suchmechanismen anhand von Metadaten und Volltextindizes. Der Zugriff auf die im Datenspeicher enthaltenen Dokumente ist ebenfalls über die Definition von Benutzern und dokumentspezifischen Zugangsrechten geregelt. Ein Versionskontrollmechanismus ist ebenfalls enthalten.

Die beiden Dokumentenmanagementsysteme der OpenText Corporation können unter Verwendung von Datenbanken der Hersteller Microsoft, Oracle und Sybase betrieben werden. Neben dieser Abhängigkeit von lizenzpflichtigen, kommerziellen Datenbanken stellt die Tatsache ein Problem dar, daß als plattformunabhängige Benutzeroberfläche nur eine rein webbasierte Ansicht zur Verfügung steht, die keine Offline-Komponente zur Verfügung stellt und in die die für die Ausführung von Lehrsimulationen benötigten Anwendungen nur auf einem sehr niedrigen Niveau integriert werden können, indem diese separat installiert und von Betriebssystem zu Betriebssystem in unterschiedlicher Weise als auszuführende Programme zum Öffnen der verschiedenen Dateitypen, in denen die Simulationen vorliegen, angegeben werden.

Neben der Web-Oberfläche existiert auch ein separater Livelink-Client für das Betriebssystem Microsoft® Windows®. In diesem Client wurde auch eine Offline-Komponente realisiert. "Livelink Offline ermöglicht Benutzern mit einem Windows®-Betriebssystem, ausgewählte Inhalte in einem Livelink-Repository für Offline-Ansichten zu markieren. Während die Benutzer online sind, werden die ausgewählten Inhalte in Livelink in einem Cache auf dem lokalen Laufwerk des Benutzers ' repliziert' . Während die Benutzer offline sind, können sie den gespeicherten Livelink-Inhalt im Windows Explorer durchblättern, ansehen und bearbeiten."[135] Dieses auf den ersten Blick sehr praktische Modul bringt aber wiederum eine Plattformabhängigkeit mit sich und ist nicht auf andere Betriebssysteme übertragbar. Dies bedeutet, daß für andere Plattformen separate Offline-Komponenten erzeugt werden und die einzelnen Applikationen systemabhängig in diese integriert werden müßten, was einen inakzeptablen Aufwand verursachen und der Forderung nach einer plattformunabhängigen Lösung zuwiderlaufen würde.

---

[132] A. a. O., Seite 8.
[133] A. a. O., Seite 9. Die Festlegung von Attributen ohne Administrator ist eine Neuheit in Version 9.0.0.
[134] Siehe hierzu die Produktbeschreibung "Basis: Das Wissen im Griff - überall und jederzeit", im Internet erhältlich unter der Adresse http://www.opentext.de/de/livelink/download/BASIS_Flyer_(deutsch).pdf.
[135] Livelink® Produktübersicht, Version 9.0.0.1, im Internet zum Download erhältlich unter der Adresse http://www.opentext.de/livelink/download/Livelink_9_0_0_1_Produktübersicht.pdf, Seite 26.

3.1.2.6 Schlußfolgerungen

Die betrachteten Dokumentenmanagementsysteme bieten durchweg die Möglichkeit zur Beteiligung aller Benutzer an der Erstellung von Inhalten in Abhängigkeit von den für sie vorgegebenen Zugriffsrechten und erfüllen mehrheitlich alle Anforderungen, die in Kapitel 2 für die Speicherung und Wiederauffindung von Daten aufgestellt wurden. Insbesondere gehören hinreichende Suchfunktionalitäten bei den meisten Systemen zum Standard, und eine flexible Metadatenverwaltung ist zumindest bei den letzten drei der beschriebenen Programme ebenfalls im Funktionsumfang enthalten. Diese drei Systeme verfügen darüber hinaus über eine Reihe von weitergehenden Funktionalitäten, die im Rahmen der E-Learning-Plattform von großem Nutzen sein könnten.

Zu nennen sind dabei zunächst die bestehenden Offline-Komponenten von Livelink und Hummingbird, die besonders komfortablen Suchfunktionalitäten von Hummingbird und Hyperwave sowie die integrierten Kommunikationsmöglichkeiten von Hyperwave und Livelink. Der Hyperwave Information Server zeichnet sich darüber hinaus durch eine Vielzahl von nützlichen Verwaltungsmöglichkeiten für Inhalte aus, beispielsweise das einzigartige Link-Management und die einfache Möglichkeit zur Kombination von verschiedenen, multimedialen Inhalten zu zusammengesetzten Objekten. Auf dieser Basis hat die Hyperwave AG als erster Hersteller eines Dokumentenmanagementsystems den Schritt zur Entwicklung einer E-Learning-Plattform unternommen, die darüber hinaus bereits mehrere Jahre vor dem aktuellen Boom des sogenannten "Blended Learning" einen breiten Satz an Funktionalitäten zur Verfügung gestellt hat, die für die Interaktion und Kommunikation zwischen Lehrenden und Lernenden einerseits und zwischen den einzelnen Lernenden andererseits geschaffen wurden.

Zudem stehen die Produkte der Hyperwave AG als einzige der in dieser Marktanalyse untersuchten, kommerziellen Systeme für den akademischen Bereich kostenlos zur Verfügung. Dies ist darauf zurückzuführen, daß der Hyperwave Information Server im Grunde genommen selbst aus einem universitären Umfeld stammt und aufgrund seiner speziellen Geschichte wohl als einziges der hier betrachteten Programme den Anspruch erheben darf, aus einer wissenschaftlichen Betrachtung der Thematik der Verwaltung großer Mengen von multimedialen Informationen hervorgegangen zu sein. Aus den hier angeführten Gründen erscheint der Hyperwave Information Server von den betrachteten Dokumentenmanagementsystemen am besten als potentielle Grundlage für die in dieser Arbeit zu entwerfende E-Learning-Plattform geeignet.

Allen betrachteten Systemen gemeinsam ist jedoch eine Reihe von Problemen, welche zusammengenommen den Ausschlag für die Entscheidung zugunsten einer eigenen Neuentwicklung gaben. Der erste kritische Aspekt ist dabei das durchgängige Fehlen einer plattformunabhängigen Offline-Komponente. Zwar verfügen alle Programme über eine betriebssystemneutrale Web-Oberfläche und einige über spezielle Schnittstellen für Microsoft® Windows®, die in zwei Fällen sogar eine Möglichkeit zu einer lokalen Speicherung ausgewählter Daten und einer daraus folgenden Offline-Verwendbarkeit vorsehen. Da diese Funktionalität jedoch nicht in einer plattformunabhängigen Art und Weise zur Verfügung steht, sondern dafür noch umfangreiche Weiterentwicklungen für alle anderen Betriebssysteme erforderlich wären, bietet keines der getesteten Systeme eine hinreichende Umsetzung der in Kapitel 2.7 aufgestellten Anforderungen.

Der zweite problematische Punkt betrifft die Umsetzung der angestrebten vollständigen Integration von unterschiedlichen Simulationsprogrammen als technisch vollkommen gleichberechtigte Anwendungen, die in einer einheitlichen Weise auf die vorhandenen

Systemkomponenten zurückgreifen und mit diesen kommunizieren sollen, um damit eine einheitliche Schnittstelle für die Eingliederung derartiger Softwareprogramme in den Rahmen der E-Learning-Plattform zu schaffen. Im Falle der Web-Oberflächen wäre es in der Regel lediglich möglich, diese Anwendungen auf eine von Betriebssystem zu Betriebssystem unterschiedliche Weise als auszuführende Programme zum Öffnen der verschiedenen Dateitypen, in denen die Simulationen vorliegen, zu definieren, was nicht als eine wirkliche Integration betrachtet werden kann. In den übrigen Fällen wäre man entweder auf proprietäre Programmiermöglichkeiten angewiesen, oder es stehen nur unzureichende Schnittstellen für eine Erweiterung der jeweiligen Systeme zur Verfügung.

Das dritte Argument gegen die Verwendung eines der betrachteten Systeme liegt darin begründet, daß diese ausnahmslos die Nutzung einer nur sehr beschränkten Menge von möglichen Datenbanksystemen erfordern, die sämtlich nur als kommerzielle Produkte erhältlich sind und somit abgesehen von einer technischen Abhängigkeit obendrein auch noch nicht zu vernachlässigende Lizenzgebühren verursachen würden. Teilweise bringt die Verwendung dieser Produkte zusätzlich eine weitere technische Einschränkung auf bestimmte Betriebssysteme mit sich. Eine derartige, gleich mehrfache Abhängigkeit von Fremdprodukten ist im Hinblick auf offene Schnittstellen und im Sinne einer möglichst leichten Erweiterbarkeit des Systems grundsätzlich abzulehnen.

3.1.3 Knowledge Management Systeme

Die bisher beschriebenen Programme dienen in erster Linie der Speicherung von Daten. Für die Organisation der abgelegten Dateien und des in diesen enthaltenen Wissens sind dabei die menschlichen Benutzer zuständig, so daß die Möglichkeiten einer sinnvollen Nutzung dieses Wissens davon abhängig sind, daß die vorhandenen Ablagestrukturen in einer geeigneten Weise verwendet werden. Darüber hinaus gibt es jedoch Software, deren Zweck darin liegt, durch spezielle Techniken vorhandenes Wissen zu erschließen und den Benutzern leichter zugänglich zu machen. Derartige Programme bezeichnet man als Wissensmanagementsysteme bzw. Knowledge Management Systeme.

Die Grenzen zwischen Dokumentenmanagement und Wissensmanagement sind dabei allerdings fließend, zumal auch die Hersteller verschiedener Dokumentenmanagementsysteme damit werben, daß ihre Produkte Knowledge-Management-Funktionalitäten enthalten. Einige Beispiele dafür sind die detaillierten Suchstrategien, die in mehreren Systemen zur Verfügung stehen, sowie die von physikalischen Verzeichnisstrukturen abstrahierende Möglichkeit zur thematischen Einordnung von Dokumenten in beliebig viele verschiedene Kontexte im Hyperwave Information Server. Dabei handelt es sich jedoch um Funktionalitäten, die auch in klassischen Dokumentenmanagementsystemen ihre Bedeutung haben. Die im Folgenden betrachteten Systeme gehen jedoch darüber hinaus, indem sie durch neue Techniken andere Sichtweisen auf in DMS gespeicherte Daten ermöglichen.

In diesem Abschnitt soll allerdings keine umfassende Untersuchung über Methoden des Wissensmanagements vorgenommen werden[136], zumal die Systeme, die im Rahmen der auf den genannten Fachmessen durchgeführten Marktanalyse evaluiert wurden, keine

[136] Eine übersichtliche und prägnante Einführung zu diesem Thema unter Berücksichtigung einschlägiger Quellen gibt: Christoph Treude, Anforderungen an Wissensmanagement, Seminararbeit zum Proseminar Elemente des E-Learning an der Universität-GH Siegen, 2002. Diese Arbeit ist im Internet unter der folgenden Adresse zum Download erhältlich: http://www.die.informatik.uni-siegen.de/lehre/ws_2002_03/proseminar_e-learning/Anforderungen_an_Wissensmanagement__Ausarbeitung.pdf.

neuen Erkenntnisse über die Datenspeicherung im Rahmen der E-Learning-Plattform bieten. Viele Wissensmanagementsysteme bauen auf einem oder mehreren bestehenden DMS auf und überlassen diesen die eigentliche Speicherung der Daten, auf die über bestimmte Schnittstellen zugegriffen wird, um auf einer übergeordneten Ebene neue Repräsentationen des enthaltenen Wissens zu generieren. Obwohl aus diesem Grund keines der betrachteten Systeme als Grundlage für die E-Learning-Plattform in Frage kommt, sollen an dieser Stelle zwei Programme kurz beschrieben werden, da sie einige interessante Aspekte bieten, die durch mögliche zukünftige Erweiterungen des Systems auch für den Bereich der elektronischen Lehre genutzt werden könnten.

3.1.3.1 USU KnowledgeMiner

Die USU AG[137] entwickelt neben verschiedener betriebswirtschaftlicher Software ein Wissensmanagementsystem namens KnowledgeMiner, das als Datenquellen sowohl auf Internet-Seiten und die Abfragemöglichkeiten verschiedener Internet-Suchmaschinen zurückgreifen kann als auch die in verschiedenen Dokumentenmanagementsystemen, beispielsweise den im Rahmen dieser Arbeit beschriebenen Produkten der Firmen Documentum, FileNET, Hyperwave, Open Text und Pironet, gespeicherten Daten als Grundlage verwenden kann.[138]

Der USU KnowledgeMiner verwendet zur Organisation der verwalteten Informationen semantische Netze gemäß dem ISO Standard 13250 "Topic Map".[139] Dadurch soll "der effiziente Zugang zu großen unstrukturierten Informationsmengen ermöglicht"[140] werden. "Die wichtigsten Bausteine der Struktur sind ' Topics'(Themen, Knoten) und ' Topic Occurrences' (statische Dokumentenzuordnung zu einem Knoten). Verweise (Assoziazionen, Kanten) verknüpfen diese Topics und machen deren Beziehung deutlich. Daraus wird der Kontext zwischen zwei miteinander verknüpften Topics ersichtlich."[141] Dabei können im System mehrere unterschiedliche Topic Maps unabhängig voneinander angelegt werden. "Somit sind zum einen unterschiedliche Sichtweisen auf Themengebiete, zum anderen differenziertere Strukturierungen der Datenbestände möglich."[142] Dies reicht bis hin zur Anlage eigener Topic Maps durch die einzelnen Benutzer.

Die üblicherweise von einem oder mehreren "Themenverantwortlichen"[143] angelegten semantischen Netze werden im System in erster Linie für eine verbesserte Suchfunktion verwendet, die im Gegensatz zu einer Volltextsuche auch semantische Zusammenhänge berücksichtigt. Dabei "werden für jeden eingegebenen Suchbegriff passende Themen aus einem oder mehreren Themennetzen (Topic Maps) gesucht. Die Anwendereingabe

---

[137] Homepage: http://www.usu.de.

[138] Als Quelle für diese Information können leider nur ein Messegespräch auf der CeBIT 2001 sowie eine nicht mehr zum Download erhältliche Broschüre der Firma USU zum KnowledgeMiner genannt werden.

[139] Die Definition dieses ISO-Standards kann auf der Homepage der International Organization for Standardization, http://www.iso.org, kostenpflichtig heruntergeladen werden. Daneben stehen Kopien dieses Dokumentes auf zahlreichen Servern im Internet zur Verfügung, beispielsweise auf einem Server des Bildungsministeriums der USA unter der Adresse http://www.y12.doe.gov/sgml/sc34/document/0129.pdf (Version 1, 1999) bzw. http://www.y12.doe.gov/sgml/sc34/document/0322_files/iso13250-2nd-ed-v2.pdf (Version 2, 2002).

[140] Grundlagen zum USU KnowledgeMiner 2.0, Informationsbroschüre der USU AG, 2001, Seite 8. Die Broschüre steht auf der Homepage der USU AG leider nicht mehr zum Download zur Verfügung, ist aber im Internet noch unter der folgenden Adresse erhältlich: http://www.wissensmanagement-competence-center.de/offerer.nsf/F2392111DCF2F8AFC1256BD500546A2C/$File/knowledgeminer_faq.pdf.

[141] A. a. O., Seiten 8-9.

[142] A. a. O., Seite 10.

[143] Diese Rollenbezeichnung wird a. a. O., Seite 10, für mit dieser Aufgabe betraute Personen eingeführt.

wird mit den im Themennetz vorhandenen Begriffen und Synonymen verglichen."[144] Dieses kontextabhängige Suchverfahren führt in der Regel zu besseren Suchresultaten als eine reine Volltextsuche. Außerdem können Suchanfragen, die zum gewünschten Erfolg geführt haben, auch gespeichert werden. Aus den auf diese Weise gewonnenen Daten über erfolgreiche Kombinationen von Suchbegriffen sowie aus den Wegen, die Benutzer im System zur Auffindung bestimmter Inhalte zurücklegen, können dann neue Informationen über semantische Zusammenhänge von Begriffen gewonnen werden, die wiederum zur Verbesserung und Erweiterung der Topic Maps eingesetzt werden kann.

Ein derartiges Wissensmanagementverfahren unter Verwendung von Topic Maps würde unter Umständen auch eine sinnvolle Ergänzung im Rahmen der E-Learning-Plattform darstellen, um einerseits die Zusammenhänge zwischen verschiedenen Begriffen und Fachgebieten sowie den in das System eingestellten Lehrsimulationen zu verdeutlichen und andererseits über die Kontrolle von Lernwegen und Lernverhalten der Studenten neue Erkenntnisse über den Umgang der Benutzer mit dem System und den Inhalten zu gewinnen, die zu einer Verbesserung der Strukturen der bereitgestellten Lehrmaterialien genutzt werden könnten. Die Schaffung einer derartigen Komponente zur Auswertung von Lernwegen stand für die im Rahmen des Projektes Monist entwickelte Simulationssoftware für den Bereich der Neuro- und Kognitionswissenschaften auch tatsächlich zur Diskussion, konnte jedoch aus Zeitgründen nicht mehr umgesetzt werden.

3.1.3.2 Comma Soft infonea®

Einen ähnlichen Ansatz verfolgt die Comma Soft AG[145] mit ihrem Produkt infonea®[146]: "Mit infonea® bildet das Unternehmen auf seine Geschäftsprozesse abgestimmte Wissensnetzwerke ab. Die Elemente dieses Netzwerkes können in einem Metamodell frei definiert und miteinander verknüpft werden."[147] Auch diese "Wissensnetzwerke", bei denen es sich ebenfalls um Topic Maps handelt, werden in erster Linie für die Suche verwendet: "Des Weiteren kann der Benutzer in diesem Wissensnetzwerk praktisch auf Knopfdruck Informationen kontextbezogen recherchieren, analysieren und zwischen ihnen browsen."[148] "Dynamische Wissenslandkarten zeigen, wo welche Informationen vorhanden sind."[149]

Außerdem werden die gespeicherten Informationen über inhaltliche Zusammenhänge zwischen einzelnen Dokumenten und Begriffen auch zu einer Verbesserung der aktiven Navigation der Benutzer im System verwendet. Die kontextbezogene Suchfunktionalität ist in die normale Navigation integriert. Wählt der Benutzer einen Themenbereich aus, so erfolgt die weitere Anzeige unter Berücksichtigung des bereits Ausgewählten: "Nur die Elemente werden angezeigt, hinter denen sich garantiert Treffer befinden."[150] Damit ist die "angezeigte Informationsmenge abhängig vom gewählten Kontext"[151]. Außerdem können Informationen über verschiedene Wege durch Navigation gefunden werden, es gibt keine starren Ablagestrukturen in Form festgelegter Verzeichnishierarchien.

---

[144] A. a. O., Seite 4.
[145] Homepage: http://www.comma-soft.com.
[146] Produkt-Homepage: http://www.infonea.com.
[147] FAQ infonea®: Antworten auf häufig gestellte Fragen zu infonea®, Broschüre der Comma Soft AG, im Internet erhältlich unter: http://www.infonea.com/download/pdf/infonea_3.0_FAQ.zip, Seite 2.
[148] A. a. O., Seite 2.
[149] Factsheet infonea® 3.0, Broschüre der Comma Soft AG, im Internet zum Download erhältlich unter der Adresse: http://www.infonea.com/download/pdf/infonea_3.0_Factsheet.zip, Seite 3.
[150] A. a. O., Seite 4.
[151] A. a. O., Seite 4.

Eine weitere besondere Funktionalität von infonea® liegt darin, daß den Benutzern die Möglichkeit gegeben wird, vorhandene Informationen zu bewerten. Dies gibt wiederum anderen Benutzern die Möglichkeit, zu erkennen, welche Dokumente bereits zuvor als nützlich erachtet wurden und bei welchen der Informationsgehalt eher gering ist. Zudem können durch Kommentare weitere Hinweise für andere Benutzer abgegeben werden. Im Rahmen einer E-Learning-Plattform könnte eine derartige Funktionalität auch zum Zwecke der Lehrevaluation genutzt werden, um es den Studenten zu ermöglichen, die verwendeten Lehrmaterialien nach Kriterien wie z. B. Verständlichkeit zu bewerten. Ob eine derartige Funktionalität durch die scheinbar höhere Anonymität gegenüber zuvor verwendeten Evaluationsmitteln wie z. B. gedruckten Fragebögen zu unterschiedlichen Ergebnissen führen würde, bleibt allerdings durch entsprechende Untersuchungen noch zu klären.

3.2 Software-Entwicklungen im Universitätsbereich

Nach der im vorangegangenen Abschnitt dargestellten Marktanalyse der kommerziellen Softwareprodukte, die in Hinblick auf ihre potentielle Verwendbarkeit als Grundlage der E-Learning-Plattform evaluiert wurden, soll in diesem Kapitel aufgezeigt werden, welche Programme aus dem universitären Bereich zur Verfügung standen und stehen. Gerade im Sinne einer Open-Source-Politik, wie sie in Kapitel 2.9 als Anforderung an die E-Learning-Plattform aufgestellt wurde, empfiehlt es sich natürlich, die Möglichkeit einer Verwendung derartiger Produkte zu prüfen, um das darauf aufbauende System auf eine bereits frei verfügbare Plattform zu stellen.

3.2.1 CampusSource

Aus ähnlichen Erwägungen bezüglich der Bündelung von Entwicklungsinteressen und der besseren Verbreitung der entwickelten Software haben sich einige der wichtigsten Projekte, deren Zweck die Schaffung von Programmen für den Einsatz in Universitäten war bzw. ist, unter dem Namen CampusSource[152] zusammengeschlossen. "Ziel dieser vom Ministerium für Wissenschaft und Forschung des Landes Nordrhein-Westfalen unterstützten Initiative ist es, kooperative Prozesse für den Aufbau eines virtuellen Hochschulraums in Gang zu setzen. Die Anstrengungen der einzelnen Hochschulprojekte werden somit gebündelt und die Open Source-Plattformen als technische Voraussetzung einer virtuellen Universität allen InteressentInnen zur Benutzung und Weiterentwicklung zur Verfügung gestellt."[153]

Die Produkte der einzelnen Teilprojekte der Initiative CampusSource wurden im Zuge dieser Arbeit erstmals ebenfalls anläßlich der CeBIT 2001 evaluiert. Ihre Entwicklung wurde während der gesamten Laufzeit des Projektes Monist kontinuierlich beobachtet. Im Folgenden werden die einzelnen Teilprojekte dargestellt. Es handelt sich dabei um 13 verschiedene Projekte, von denen allerdings zum Zeitpunkt der ersten Evaluation erst sechs (OpenUSS, Uni Open Platform, ILIAS, WebAssign, MILESS und VU) Teile von CampusSource waren, während die übrigen sieben Programme (Javanti, litw^3, MMC, Stud.IP, EdoWorkSpace, SuperX und VirPa) erst zu späteren Zeitpunkten dazukamen.

[152] Homepage: http://www.campussource.de.
[153] Zitat von der CampusSource Homepage.

Der Vollständigkeit halber werden hier alle Projekte zumindest kurz vorgestellt, obwohl der vorgesehene Einsatzbereich einiger Programme (beispielsweise für die Abwicklung administrativer Prozesse, als Web-Oberfläche für Prüfungsämter oder als elektronisches Zahlungssystem) deren Verwendung als Basis der im Rahmen der vorliegenden Arbeit zu entwickelnden E-Learning-Plattform von vornherein ausschließt.

Ein grundsätzliches Problem dieser Entwicklungen aus dem Universitätsbereich wird allerdings bereits angesichts der Vielzahl der wechselseitig voneinander unabhängigen Programme deutlich, die im Rahmen der Initiative CampusSource zur Verfügung stehen: "In CampusSource sind verschiedene Lehr- und Lernsysteme vorhanden, die für unterschiedliche didaktische Konzepte, verschiedene Fachgebiete und auf Basis verschiedener Technologien entwickelt wurden. Diese Heterogenität macht es jedoch unmöglich, z. B. Funktionalitäten anderer Systeme zu übernehmen und nutzen zu können. Es wäre ein erheblicher Aufwand nötig. Um dies lösen zu können, wurde ein Projekt initiiert, welches zum Ziel hatte, eine allgemeine Referenzarchitektur für Lehr/Lernsysteme zu gestalten."[154]

Solange ein derartiges Projekt nicht erfolgreich zum Abschluß gebracht worden sein wird, stellen die einzelnen Programme lediglich Insellösungen dar, und eine Integration neu hinzukommender Software (wie z. B. im Rahmen der E-Learning-Plattform) mit mehreren der derzeit vorhandenen Produkte ist kaum zu realisieren. Nach Aussagen, die von Vertretern von CampusSource am Messestand auf der Fachmesse LEARNTEC 2003 in Karlsruhe getroffen wurden, liegt ein Durchbruch in dem besagten Integrationsprojekt jedoch noch in weiter Ferne[155] und wird aufgrund der großen technischen Unterschiede der einzelnen Programme auch für unwahrscheinlich erachtet.

3.2.1.1 OpenUSS (Westfälische Wilhelms-Universität Münster)

"Das **O**pen **U**niversity **S**upport **S**ystem ist eine rollenorientierte Plattform zur Abwicklung administrativer Lehr- und Lernprozesse innerhalb der Hochschulen."[156] Zu diesem Zweck stellt OpenUSS verschiedene Mechanismen zur Kommunikation und zur Ablage von Lernmaterialien zur Verfügung, darunter veranstaltungsbezogene E-Mail-Verteiler, themenspezifische Diskussionsforen, moderierte Chat-Rooms sowie eine personalisierte Lernmittelverwaltung und eine Vorlesungsarchivierung auf CD-ROM, um Lehrmedien auch offline zugänglich machen zu können.[157]

OpenUSS baut neben dem Internet auch auf mobile Kommunikation: "Da für die computergestützte Hochschullehre die Mobilität ein entscheidender Erfolgsfaktor ist, setzt OpenUSS auf innovative Kommunikationstechniken wie z. B. WAP und SMS. Auf diese Weise können Studierende just in time mit aktuellen Informationen über Lehrveranstaltungen versorgt werden - unabhängig davon, ob der Zugriff über den studentischen Arbeitsplatz daheim, via Mobiltelefon oder [...] über UMTS-Handy erfolgt."[158]

---

[154] Zitat von der CampusSource Homepage, http://www.campussource.de.

[155] Bezeichnenderweise heißt es auf der CampusSource Homepage wörtlich: "In einem ersten Schritt wird diskutiert, ob sich z.B. auf CORBA basierende Schnittstellen zu den jetzigen Systemen definieren und implementieren lassen, so dass Neuentwicklungen mit ergänzenden Funktionalitäten potentiell zu allen Systemen kompatibel sein können." Die Tatsache, daß die Initiative nach mehreren Jahren des Bestehens in diesem Punkt immer noch nicht über den Status eines "ersten Diskussionsschrittes" hinausgekommen ist, läßt leider nicht auf eine schnelle Lösung der bestehenden Kompatibilitätsprobleme schließen.

[156] Zitat von der CampusSource Homepage, http://www.campussource.de.

[157] Siehe hierzu die Projektbeschreibung unter http://www.campussource.de/software/openuss.

[158] A. a. O.

OpenUSS bietet Möglichkeiten zur Personalisierung und paßt sich den individuellen Bedürfnissen der einzelnen Benutzer an. "Studierende abonnieren selbständig die gewünschten Veranstaltungen und werden auf diese Weise bedarfsgerecht mit Informationen versorgt". Den Lehrenden ermöglicht das System die Ablage von elektronischen Unterlagen zu den Veranstaltungen: "Sämtliche Lernmaterialien werden strukturiert in einer Datenbank gespeichert, so dass Probleme bezüglich der Konsistenz und der Aktualität der Web-Präsenz der Vergangenheit angehören."[159]

Es handelt sich also nach der im vorigen Kapitel getroffenen Einteilung um ein Content Management System, das um Möglichkeiten zur Kommunikation zwischen Benutzern sowie zu einer automatischen Benachrichtigung erweitert wurde. Offline-Komponenten existieren nicht, OpenUSS wird über rein browserbasierte Benutzeroberflächen bedient. Daraus ergeben sich die gleichen Probleme wie bei den zuvor betrachteten CMS bezüglich der Integrierbarkeit benötigter Simulationsprogramme sowie einer abwechselnden Online- und Offline-Nutzung des Systems. In letzterem Punkt bietet auch die im System enthaltene Möglichkeit zur Offline-Distribution auf CD keine ausreichende Abhilfe, da dies die ständige Erstellung neuer CDs nach jedem Update erforderlich machen würde.

3.2.1.2 Javanti (Fachhochschule Wedel)

Dieses Programm ist eines der neuesten Teilprojekte im Rahmen von CampusSource, das auf der LEARNTEC 2003 in Karlsruhe erstmals als solches vorgestellt wurde. Folgt man nur der Beschreibung des Projektes auf der CampusSource Homepage, so erscheint dieses Programm auf den ersten Blick am besten als Basis einer E-Learning-Plattform für Lehrsimulationen geeignet zu sein, denn "Javanti ist eine integrierte Entwicklungsumgebung für interaktive Präsentationen und eLearning-Anwendungen. [...] Auf virtuellen Folien kann der Autor einer Javanti-Anwendung so genannte Smart Elements platzieren. Smart Elements sind Bausteine beliebiger Komplexität, von einfachem Text über Audio und Video bis hin zu Simulationen und Experimenten."[160]

Zieht man jedoch das Javanti Benutzerhandbuch[161] zu Rate, stellt man fest, daß es sich bei diesen "Simulationen und Experimenten" lediglich um einfache Tcl-Skripte handelt, die benutzt werden können, "um physikalische Gesetze abzubilden und mathematische Rechnungen durchzuführen. In jtap ist auch ein Timer eingebaut, der die onRepeat-Aktionslisten nach bestimmten Zeitintervallen ausführt. Sie können sich dies für Experimente zu Nutze machen, indem Sie beispielsweise Elemente über die Zeit bewegen. Sie könnten in der onRepeat-Aktionsliste z.B. das physikalische Verhalten eines Objektes festlegen."[162] Die Möglichkeiten zur Verwendung von Simulationen sind also nur sehr begrenzt und weit entfernt von einer Einbindung komplexer Modellsimulationen, wie sie beispielsweise im Rahmen des Projektes Monist verwendet werden.

Aus diesem Grunde scheidet Javanti als Grundlage für die E-Learning-Plattform aus, zumal dem System auch die dafür benötigten Dokumentenmanagementfunktionalitäten fehlen. Als Werkzeug zur Erstellung und Präsentation von teilweise dynamischen und interaktiven Vorlesungsmaterialien erscheint das Programm allerdings sehr geeignet und könnte vielleicht ebenso wie die Programme zum Editieren und Ausführen von Simulationen als Anwendung in die E-Learning-Plattform integriert werden.

---

[159] A. a. O.
[160] Projektbeschreibung unter http://www.campussource.de/software/javanti.
[161] http://www.campussource.de/dl/pub/javanti/jtapBenutzerhandbuch.pdf.
[162] A. a. O., Seite 44.

3.2.1.3 litw³ (Westfälische Wilhelms-Universität Münster)

Das als bisher letztes zu CampusSource hinzugekommene Programm "litw³ ist ein webbasiertes Literaturverwaltungssystem, dass eine orts- und zeitunabhängige Eingabe von Literaturdaten durch Arbeitsgruppen ermöglicht"[163]. Es eröffnet die Möglichkeit zur Speicherung von beliebigen Metadaten und die leichte Anpassung von entsprechenden Eingabeformularen. Die Metadaten werden für die eingebauten Suchfunktionalitäten genutzt, und "die angebotene Expertensuche erlaubt vielfältigste Abfragemöglichkeiten unter Ausnutzung aller bool´schen Operatoren"[164]. Als rein webbasiertes System kann litw³ aus jedem Web-Browser heraus bedient werden, die Serverkomponente läuft unter jedem Webserver, der PHP-Skripte ausführen kann.

3.2.1.4 MMC (FernUniversität in Hagen)

MMC oder "Multimedia-Catalogue ist ein System zur Verwaltung von wiederverwendbaren Multimedia-Komponenten für Lehrzwecke, wie beispielsweise Java-Applets, Animationen, Bilder, Videos, aber auch Lehrtexte wie Definitionen, Beispiele oder Übungen."[165] Es handelt sich dabei um eine Art Dokumentenmanagementsystem, das von den Anwendern zur Verwaltung und Benutzung von Lehrmaterialien verwendet werden kann: "Der Katalog verbessert den Prozess der Gestaltung von Multimedia-Kursen dadurch, dass eine Plattform zur Publikation, Wiederverwendung und der gemeinsamen Nutzung von Multimedia-Komponenten angeboten wird."[166] Die in das System eingestellten Objekte können mit Metadaten gemäß dem IEEE-Standard LOM (Learning Object Metadata) versehen werden.[167]

Die gespeicherten Daten werden nach Bedarf "zu Komponenten zusammengefasst, die selbst wiederum zu Sammlungen von Komponenten integriert werden können. Komponenten und Sammlungen werden zusammen mit den sie verknüpfenden Verweisen in der dem Katalog unterliegenden relationalen Datenbank gespeichert."[168] Das System stellt umfangreiche Suchfunktionalitäten auf Grundlage der eingegebenen Metadaten zur Verfügung, kann aber ebenfalls durch eine aktive Navigation der Benutzer in thematischen Verzeichnissen durchsucht werden.

Die Benutzeroberfläche besteht aus einem rein webbasierten Hypertextsystem, ist damit zwar plattformunabhängig, aber ausschließlich browserorientiert, woraus sich auch hier die gleichen Probleme bezüglich der Entwicklung einer Offline-Komponente und der Integration der Simulationsprogramme ergeben, die schon bei den im vorigen Abschnitt untersuchten, kommerziellen DMS festgestellt wurden. Serverseitig basiert MMC auf dem Apache Webserver und einer PostgreSQL-Datenbank, somit ausschließlich auf Technologien, die im Rahmen einer Open-Source-Politik und mithin kostenlos zur Verfügung stehen. Allerdings können auch andere Datenbanken, beispielsweise Oracle, verwendet werden. Für eine mögliche kommerzielle Nutzung von in MMC eingestellten Inhalten wurde außerdem eine Schnittstelle für die Integration von elektronischen Bezahlungssystemen geschaffen.

---

[163] Projektbeschreibung unter http://www.campussource.de/software/litw.
[164] A. a. O.
[165] Zitat von der CampusSource Homepage, http://www.campussource.de.
[166] Projektbeschreibung unter http://www.campussource.de/software/mmc.
[167] Zu diesem Standard für Metadaten von Lernobjekten siehe die zugehörige Internet-Seite der IEEE unter der Adresse http://ltsc.ieee.org/wg12 sowie Kapitel 3.2 der vorliegenden Ausarbeitung mit einer ausführlichen Beschreibung des LOM-Standards sowie entsprechenden Quellenangaben.
[168] Projektbeschreibung unter http://www.campussource.de/software/mmc.

3.2.1.5 Stud.IP (Georg-August-Universität Göttingen)

Stud.IP unterstützt Lehrveranstaltungen durch die Vereinfachung von administrativen Abläufen. "Zu jeder Veranstaltung einer Einrichtung werden Ablaufpläne, Dateiordner, Diskussionsforen, Teilnehmer- Literatur- und Linklisten, Newsticker u. v. a. bereitgestellt. Ziel ist es nicht, bestehende Veranstaltungen zu ersetzen sondern diese multimedial zu unterstützen."[169] Damit setzt Stud.IP gewissermaßen einen Gegenpol zu dem derzeit in der E-Learning-Industrie weitverbreiteten Ansatz, im Rahmen des sogenannten "Blended Learning" elektronische Lehre, die lange Zeit als Ersatz für herkömmliche Präsenzlehre angepriesen wurde, wieder mit Präsenzkomponenten zu verbinden. Beide Ansätze zielen jedoch im Grunde genommen in die gleiche Richtung, indem nämlich die Vorzüge beider Lehrmethoden gleichermaßen genutzt werden sollen.

"Der zweite Schwerpunkt liegt auf administrativer Ebene: Aktualisierung von bestehenden Institutshomepages über das Internet, Verwaltung von Ressourcen wie Räumen oder ausleihbarer Technik, Druck von Veranstaltungskommentaren oder die Verwaltung von Terminen im integrierten Personal Information Manager (PIM)."[170] Somit dient das Programm gleichzeitig auch als Content Management System und als elektronisches Verwaltungssystem für außerhalb von Computersystemen vorhandene Ressourcen. Ziel des Systems ist die "Steigerung der Medienkompetenz bei Lehrenden wie Lernenden"[171] und somit eine Verbesserung des Einsatzes von neuen Medien in der Lehre durch die Erleichterung des Zuganges. Für die Verwendung von Simulationssoftware, wie z. B. im Rahmen des Projektes Monist, ist das System allerdings nicht geeignet.

3.2.1.6 EdoWorkSpace (Universität Dortmund)

Das Programm "EdoWorkSpace ist eine internetbasierte Arbeits- und Projektumgebung für Lehrveranstaltungen, Forschungsprojekte oder Arbeitsgemeinschaften."[172] Diese bietet Kommunikationsfunktionen sowie Methoden zur Ablage und Verbreitung von Dokumenten: "Das EdoWorkSpace-System stellt den Nutzerinnen und Nutzern virtuelle Veranstaltungsräume zur Verfügung. Durch sie können Präsenzveranstaltungen unterstützt werden, indem die Veranstaltungsleitung oder die Teilnehmenden multimediale Materialien veröffentlichen oder die Kommunikations- und Kooperationsmöglichkeiten der Plattform nutzen. EdoWorkSpace kann aber auch für rein virtuelle Veranstaltungen oder in Forschungsprojekten genutzt werden."[173]

Zu diesem Zweck gibt es zu jeder im System angelegten Veranstaltung eigene Bereiche für Lehrmaterialien der Kursleitung, im Rahmen der Veranstaltung von Teilnehmern erzeugte Materialien sowie zur Veröffentlichung bestimmte Materialien. Hinzu kommt ein Kommunikationsbereich mit Diskussionsforum, Chat und einem E-Mail-System mit Adressenverwaltung und Mailinglisten. Veranstaltungen können auch in Untergruppen aufgegliedert werden, für die dann die gleichen Funktionalitäten in entsprechender Weise zur Verfügung stehen. Für die Verwaltung statischer Dokumente ist das System sicherlich geeignet, die Einbindung spezieller Software stellt jedoch ebenso ein Problem dar wie bei allen bisher betrachteten webbasierten Systemen. EdoWorkSpace läuft unter Linux und verwendet nur Open-Source-Komponenten wie Apache, MySQL und PHP.

---

[169] Projektbeschreibung unter http://www.campussource.de/software/studip.
[170] A. a. O.
[171] A. a. O.
[172] Zitat von der CampusSource Homepage, http://www.campussource.de.
[173] Projektbeschreibung unter http://www.campussource.de/software/ews.

3.2.1.7 SuperX (Universität Duisburg-Essen)

"SuperX ist ein Data-Warehouse für Hochschulen, in dem die verschiedensten Datenquellen aus dem Hochschulbereich (Studierendenzahlen, Prüfungen, Personal/Stellen, Haushalt, Kosten, etc.) zusammengestellt werden, um diese Information für Hochschulangehörige verfügbar zu machen. Es liefert über eine WWW-Schnittstelle tagesaktuelle Statistiken und Daten in verschiedenen Aggregationsstufen."[174] Als reine Verwaltungsanwendung ist es für die zu schaffende E-Learning-Plattform nicht von Bedeutung.[175]

3.2.1.8 Uni Open Platform (FernUniversität in Hagen)

Mit der Uni Open Platform wurde im Rahmen des vom Land Nordrhein-Westfalen geförderten Projektes "Virtuelle Universität" eine E-Learning-Plattform geschaffen, die im Rahmen einer Open-Source-Politik zur Verfügung gestellt wird. Dabei handelt es sich um "ein webbasiertes System für die Administration von Kursen, Studenten und Lehrpersonal"[176]. Für das dazu berechtigte Universitätspersonal wie z. B. Professoren, Sekretärinnen und Systemadministratoren steht eine Administrationsoberfläche zur Verfügung, die die Verwaltung von Daten über Kurse (z. B. Teilnehmer) und einzelne Studenten (z. B. Matrikelnummern, Adressen, Punkte für Hausaufgaben) ermöglicht.

Außerhalb des Programmes erstellte Lehrinhalte können im System abgelegt werden, wobei alle Datenformate unterstützt werden, die von einem Webserver verteilt werden können. Eine Metadatenverwaltung fehlt jedoch, ebenso Suchfunktionalitäten für die Lernenden, da davon ausgegangen wird, daß es stets einen veranstaltungsbezogenen, linearen Lernweg gibt, dessen Teilabschnitte in festgelegten Schritten nacheinander freigeschaltet werden: "Es wurde festgestellt, dass es vorteilhaft ist, alle Studierenden eines Kurses in einem einheitlichen Lernrhythmus zu halten. Ansonsten würde die Grundlage für Diskussionen und gemeinsames Lernen allmählich verschwinden, da einige Studierende schneller im Lehrmaterial fortschreiten würden als ihre Kommilitonen. Technisch kann ein gemeinsamer Lernrhythmus dadurch erreicht werden, dass das Lehrmaterial in Komponenten unterteilt wird, die in periodischen Zeitintervallen freigegeben werden. Dieser Vorgang kann in ' UniOpen Platform' automatisiert werden, indem das Freigabemodul entsprechend konfiguriert wird."[177] Inwiefern es tatsächlich sinnvoll ist, die Lernenden durch das Vorenthalten von Kursmaterialien in ihrem Studienfortschritt quasi künstlich auszubremsen, darf allerdings bezweifelt werden.

Eine interessante Systemkomponente der Uni Open Platform ist der Offline-Navigator, mit dem man ausgewählte Inhalte für eine Offline-Bearbeitung herunterladen kann. Dies ist gerade für die Lehre an einer Fernuniversität, von der dieses Programm stammt, von erheblicher Bedeutung. Allerdings ist das System im Grunde genommen nur für die Bereitstellung von nicht interaktiven Inhalten vorgesehen. "Typische Kurskomponenten sind Vorlesungsskripte im HTML- und PDF-Format, Hausaufgaben und deren Lösungen, übungen, multimediale Elemente, etc."[178] Für die Integration eigener Programme ergeben sich auch hier Probleme bezüglich der Erzeugung von Installationspaketen für unterschiedliche Plattformen sowie der ebenfalls plattformabhängigen Registrierung der Anwendungen als auszuführende Programme für die jeweils verwendeten Dateitypen.

---

[174] Zitat von der CampusSource Homepage, http://www.campussource.de.
[175] Weitere Informationen in der Projektbeschreibung unter http://www.campussource.de/software/superx.
[176] Projektbeschreibung unter http://www.campussource.de/software/uop.
[177] A. a. O.
[178] A. a. O., Rechtschreibfehler aus dem Original übernommen.

3.2.1.9 ILIAS (Universität zu Köln)

Die Bezeichnung ILIAS steht als Abkürzung für "**I**ntegriertes **L**ern-, **I**nformations- und **A**rbeitskooperations**s**ystem". Das Programm soll ein von Zeit und Ort unabhängiges Lernen ermöglichen und dabei kooperatives Arbeiten sowohl zwischen den Lernenden untereinander als auch zwischen Lernenden und Lehrenden ermöglichen. Dafür verfügt das System über Funktionalitäten zur Erstellung von Kursmaterialien durch Dozenten, personalisierte Arbeitsbereiche für Lernende, Lehrende und Autoren[179]. Nach Bedarf lassen sich Lehrende und Lernende veranstaltungsbezogen in Gruppen einteilen, denen wiederum spezielle Kursmaterialien zugeordnet werden können. Zur Kommunikation dienen Diskussionsforen sowie ein integriertes E-Mail-System.

Zum Erstellen von Lehreinheiten steht ein integriertes Autoren-Tool zur Verfügung, für dessen Bedienung keinerlei Programmierkenntnisse erforderlich sind. Einfache Texte können per "copy and paste" übernommen werden. Außerdem können Dateien in vielen gängigen Formaten verwendet werden, darunter "interne und externe Hyperlinks, Grafiken/Bilder, animierte Grafiken, Tabellen, Glossar, Multimedia-Objekte (Java-Applets, Flash Animationen, Audio- und Video-Dateien etc...)"[180]. Zur Verwaltung der Inhalte werden Metadaten verwendet, die auf verschiedene Ebenen bezogen werden können, von der kompletten Lehreinheit über einzelne Seiten bis hin zu den darin enthaltenen, einzelnen Elementen.

Der Zugang zu bestimmten Kursen kann an definierte Bedingungen geknüpft werden und zum Beispiel zeitabhängig, aufgrund bestimmter nachgewiesener Vorkenntnisse, nach einer persönlichen Anmeldung oder auch nur gegen die Zahlung eines bestimmten Geldbetrages über ein Payment-System gewährt werden. ILIAS selbst wird im Rahmen einer Open-Source-Politik vertrieben und kann somit kostenlos genutzt werden. Das Programm ist vollständig internetbasiert und wird über eine reine Browserschnittstelle bedient, wobei einzelne Dateien für eine Offline-Benutzung heruntergeladen werden können (was natürlich nur dann sinnvoll ist, wenn auch die zum Ausführen benötigte Applikation lokal zur Verfügung steht).

ILIAS ist also clientseitig plattformunabhängig. Der Server, der unter Verwendung von kostenlos erhältlichen Technologien wie der Datenbank MySQL und der Skriptsprache PHP betrieben wird, läuft ausschließlich unter den Betriebssystemen Unix oder Linux. Diese Kombination aus einer reinen Web-Oberfläche und einem plattformabhängigen Server, wie sie auch in mehreren der zuvor beschriebenen Systeme vorhanden ist, bringt hier die dabei üblichen Probleme mit sich. Die Integration von Simulationsprogrammen und anderen eigenen Applikationen für den wechselweisen Online- und Offline-Einsatz ist nur mit großem Aufwand möglich, ebenso die Realisierung der dazu benötigten plattformunabhängigen Datenspeicherung. Eine echte Offline-Komponente existiert ebenfalls nicht, da lediglich einzelne Dateien auf die lokalen Computer der Benutzer heruntergeladen werden können. Die zum Öffnen dieser Dateien benötigten Programme müßten stets separat installiert und in Abhängigkeit von der jeweiligen Plattform als für die jeweiligen Dateitypen zuständige Anwendungen registriert werden, dabei würden zusammengesetzte Lehrsimulationen auch nicht als Einheit behandelt werden können.

---

[179] Vergleiche hierzu die dem gleichen Schema folgende Aufteilung des Monist-Systems in die Module Studienschreibtisch, Lehrschreibtisch und Autorenschreibtisch, wie sie bereits im Projektantrag aus dem Jahr 2000 dargestellt wurde, in der *Projektskizze zur Einreichung bei dem Förderprogramm des Bundesministeriums für Bildung und Forschung: "Neue Medien in der Bildung"* unter der Internet-Adresse http://www.monist.de bzw. http://natrix.biologie.uni-bielefeld.de/monist/Archiv/monistskizze_oz.html.

[180] Projektbeschreibung unter http://www.campussource.de/software/ilias.

3.2.1.10 WebAssign (FernUniversität in Hagen)

WebAssign stellt eine Infrastruktur für die Durchführung von Übungsveranstaltungen im Internet zur Verfügung. "Studierende laden ihre Aufgaben herunter, bearbeiten sie offline und laden ihre Lösungen anschließend herauf. Ihre korrigierten Einsendungen und die Musterlösungen werden auf dem Server bereitgestellt. Ein vollständiger Online-Modus ist auch möglich."[181] Das System verwaltet die Aufgaben unter Verwendung einer Datenbank, verteilt nach einstellbaren Vorgaben die Veranstaltungsteilnehmer auf die zur Verfügung stehenden Korrektoren und speichert Informationen über den Leistungsstand der Teilnehmer.

Die Aufgaben werden in WebAssign erstellt und zu Übungsblättern zusammengefaßt. Der Aufgabentext wird in ein vom System generiertes HTML-Formular eingetragen. Dabei wird die Art der Korrektur spezifiziert, Korrekturhinweise können angegeben werden, und die Bewertung wird festgelegt. "WebAssign unterstützt verschiedene Aufgabentypen mit unterschiedlichen Korrekturmodi: Multiple-Choice Aufgaben, Zuordnungsaufgaben und numerische Aufgaben mit automatischer Korrektur und Bewertung; offene Aufgaben mit manueller Korrektur und Bewertung; Programmieraufgaben mit Online-Vortest und anschließender manueller Korrektur und Bewertung."[182]

WebAssign ist aufgrund der Beschränkung auf Übungsaufgaben nicht als Grundlage für die zu entwickelnde E-Learning-Plattform geeignet, könnte jedoch unter Umständen im Rahmen einer künftigen Weiterentwicklung als Teilanwendung in diese integriert werden, um die Nutzung der enthaltenen Übungsfunktionalitäten zu ermöglichen. Wie die anderen beschriebenen CampusSource-Produkte ist auch WebAssign ein Open-Source-Produkt. "Es ist in Java implementiert, läuft unter Linux und Solaris und benutzt Open-source-Software (Apache, Tomcat, MySQL) sowie die Softwarestandards JDBC und CORBA. Als Datenbank können auch ORACLE und Informix verwendet werden."[183]

3.2.1.11 MILESS (Universität Duisburg-Essen)

MILESS (Multimedialer Lehr- und Lernserver Essen) ist ein Projekt der Universität Essen zur Entwicklung einer digitalen Bibliothek. Neben textuellen Dokumenten, die im Volltext gespeichert werden und für eine entsprechende Suche zugänglich sind, kann das System ebenfalls multimediale Inhalte verwalten, beispielsweise Bilder, Audio- und Video-Dateien, Animationen und Simulationen[184]. "MILESS realisiert für digitale Dokumente die Dienste, die die klassische Bibliothek für gedrucktes Material bietet. Material, das bisher verstreut in der Hochschule und nur schwer zu finden war, wird in MILESS gesammelt, archiviert und dauerhaft verfügbar gemacht. MILESS verwaltet daher nicht nur beschreibende Daten zu den Inhalten, sondern legt auch die dazugehörenden Dateien selbst in einem Datenbanksystem ab, so dass die Verfügbarkeit für die Vor- und Nachbereitung oder den direkten Einsatz in der Lehrveranstaltung garantiert werden kann."[185]

---

[181] Projektbeschreibung unter http://www.campussource.de/software/webassign.
[182] A. a. O.
[183] A. a. O.
[184] Gemeint sind hier einzelne, eigenständige Programme, die als Ganzes heruntergeladen werden können, sofern sie für die jeweilige Betriebssystemplattform zur Verfügung stehen. Für die Verwaltung komplexer Programme wie im Rahmen des Projektes Monist, die über Konfigurationsdateien für die verschiedensten Simulationen benutzt werden und dabei zahlreiche voneinander unabhängige Einzelmedien verwenden, ist das System nicht geeignet, da die enthaltenen Medien nur als eigenständige Objekte behandelt werden.
[185] Projektbeschreibung unter http://www.campussource.de/software/miless.

Zur Verwaltung der gespeicherten Daten werden Metadaten verwendet, die nach den Anforderungen des jeweiligen Fachgebietes eingetragen werden können und in denen über verschiedene Suchmasken recherchiert werden kann. Die Benutzeroberfläche ist webbasiert, als Client dient ein Java Applet. "Als Datenbank-Backend des MILESS-Systems sind die Produkte IBM DB2 und IBM Content Manager im Einsatz, bei letzterem handelt es sich um ein Datenbanksystem zum Content Management multimedialer Bestände, das Funktionen wie Volltextsuche, Verwaltung verteilter Objekt-Server, Watermarking, Streaming von Audio- und Videodaten, automatisierte Auslagerung von Dateien vom Online-Plattenbereich auf Bandarchive etc. bietet."[186] Außerdem werden serverseitig Java Servlets verwendet.

Auffällig ist, daß in diesem Fall im Gegensatz zu den übrigen Projekten der Initiative CampusSource eine ausschließliche und tiefgreifende technische Abhängigkeit von den Produkten eines einzigen kommerziellen Softwareunternehmens gegeben ist, während die übrigen Programme überwiegend auf Open-Source-Software aufbauen. Dennoch hat das System im Universitätsbereich eine gewisse Verbreitung gefunden: "Erste Projekte zur Nachnutzung sind u. a. an den Universitäten Jena, Leipzig, Freiburg, Uppsala und bei der GWDG Göttingen gestartet. An der Universität Leipzig wurde auf Basis der MILESS-Software das Projekt Bach Digital realisiert, in dem digitalisierte Autographen und Tonbeispiele der Werke Johann Sebastian Bachs verfügbar gemacht werden. [...] Die Deutsche Forschungsgemeinschaft fördert aktuell die Weiterentwicklung zu einer verteilten digitalen Videobibliothek."[187]

Als Grundlage für die hier zu entwickelnde E-Learning-Plattform kommt MILESS aus mehreren Gründen nicht in Frage. Abgesehen von der bereits erwähnten, möglichst zu vermeidenden Abhängigkeit von lizenzpflichtigen Fremdprodukten ist die Schaffung einer Offline-Komponente mit eigener Datenspeicherung praktisch ausgeschlossen. Die Integration dynamischer und interaktiver Programme zum Gestalten und Ausführen von konfigurierbaren Lehrsimulationen ist nicht vorgesehen. Das System ist eine reine Web-Anwendung, was die bereits mehrfach geschilderten Probleme bezüglich der plattformabhängigen Installation und Registrierung von eigenen Programmen mit sich bringt.

3.2.1.12 VU (FernUniversität in Hagen)

Die Virtuelle Universität (VU) ist eine Plattform zur Verteilung von Lernmaterialien mit integrierten Kommunikations- und Interaktionsfunktionalitäten. In diesem Projekt wurde erstmals der Versuch unternommen, alle Bereiche einer Universität elektronisch abzubilden. Dies reicht von administrativen Funktionen wie z. B. der Verwaltung von Studenten und deren Anmeldung zu bestimmten Veranstaltungen über Kommunikation zwischen Lehrenden und/oder Lernenden bis hin zur Verbreitung von elektronischen Kursmaterialien und einer elektronischen Bibliothek. "Die Virtuelle Universität bietet neue Lehrformen und räumlich sowie zeitlich flexibles, individualisiertes und bedarfsorientiertes Lernen durch konsequente Nutzung neuer Medien (Multimedia- und Kommunikationstechnologie)."[188] Letztere wird "insbesondere auch zwischen den Studierenden untereinander für gemeinsames Lernen (peer-learning) und für die soziale Vernetzung, Möglichkeiten zur netzbasierten Gruppen- und Seminararbeit, neue Formen des übungs- und Praktikumsbetriebs über Netze"[189] verwendet.

---

[186] A. a. O., Rechtschreibfehler aus dem Original übernommen.
[187] A. a. O.
[188] Projektbeschreibung unter http://www.campussource.de/software/vu.
[189] A. a. O., Rechtschreibfehler aus dem Original übernommen.

Der erste Prototyp der webbasierten VU wurde im Jahre 1996 entwickelt. "Inzwischen nutzen mehr als 14.000 Studierende virtuelle Universitätskomponenten."[190] Das System befindet sich ungeachtet dessen immer noch in einer Weiterentwicklungsphase, deren Ziele ein verbesserter Datenschutz, die Integration von Verwaltungsfunktionalitäten und die Verbesserung der elektronischen Lehre auf der Basis der bisher damit gewonnenen Erfahrungen sind. Während die Integration von Kommunikationskomponenten bereits weit fortgeschritten ist und beispielsweise eine sogenannte "virtuelle Cafeteria" als Online-Treffpunkt zum Austausch von Informationen und zur Bildung und Pflege von sozialen Kontakten umfaßt, sind die austauschbaren Lehrmaterialien nach wie vor auf nichtinteraktive Medien beschränkt, und die Integration von Simulationsprogrammen stößt auf die gleichen Probleme wie bei allen anderen webbasierten Plattformen.

3.2.1.13 VirPa (Fachhochschule Bielefeld)

Die Abkürzung VirPa steht für den Begriff "Virtuelles Prüfungsamt". "Das Virtuelle Prüfungsamt ist ein System, das die Prüfungsbearbeitung sowohl für die Studierenden als auch für die MitarbeiterInnen der Prüfungsämter vereinfacht. Den Studierenden wird eine Webschnittstelle für Prüfungsangelegenheiten geboten und die MitarbeiterInnen werden durch die Automatisierung z.B. der Prüfungsan- und -abmeldungen von manueller Arbeit entlastet."[191] Es handelt sich also um ein reines Verwaltungssystem, das für die zu schaffende E-Learning-Plattform nicht von Bedeutung ist[192], zumal die darin verwalteten Vorgänge, namentlich die Prüfungen, weder innerhalb elektronischer Systeme noch mit deren inhaltlicher Unterstützung stattfinden.

3.2.1.14 Schlußfolgerungen

Insgesamt erscheint der Aufbau der zu entwerfenden E-Learning-Plattform auf einem oder mehreren der hier betrachteten Systeme aus technischen Gründen kaum möglich, da einerseits von diesen in der Regel keine offenen Schnittstellen zur Verfügung gestellt werden, die eine technische Anbindung ermöglichen würden, und andererseits bislang nach wie vor keine gemeinsame Plattform für die Produkte der Initiative CampusSource zur Verfügung steht. Die einzige Gemeinsamkeit, die von allen Programmen geteilt wird, ist, daß es sich dabei um reine Web-Anwendungen ohne eine Offline-Komponente mit der Möglichkeit einer eigenen Datenspeicherung handelt. Während sich mehrere Überschneidungen in den Funktionalitäten der einzelnen Systeme ergeben, insbesondere bei der (obendrein wechselseitig nicht kompatiblen) Speicherung von Daten sowie von Benutzern, Gruppen und Zugriffsrechten, fehlt in allen Programmen die Möglichkeit zu der im Rahmen dieser Arbeit beabsichtigten Integration interaktiver Lehrsimulationen.

Aus diesem Grund wurde die Möglichkeit, die E-Learning-Plattform auf einem oder mehreren Produkten aus der Initiative CampusSource aufzubauen, bereits frühzeitig verworfen, zumal es auch nicht die Aufgabe dieses neuen Systems sein kann, die Integration aller dieser Programme selbst zu leisten, besonders, weil eine ganze Reihe von Funktionen der einzelnen Systeme eher in den Bereich der Universitätsverwaltung fallen. Die Entwicklung von derartigen elektronischen administrativen Funktionalitäten hat zwar durchaus ihre Berechtigung, sie sind aber für die Erfordernisse der im Rahmen dieser Arbeit zu schaffenden E-Learning-Plattform technisch und inhaltlich irrelevant.

---

[190] A. a. O.
[191] Zitat von der CampusSource Homepage, http://www.campussource.de.
[192] Weitere Informationen in der Projektbeschreibung unter http://www.campussource.de/software/virpa.

3.3 Marktübersicht gegen Ende der Projektlaufzeit

Im letzten Projektjahr[193] wurde auf der Fachmesse LEARNTEC 2003[194] in Karlsruhe, der jährlich stattfindenden, größten und wichtigsten europäischen Messe für Bildungs- und Informationstechnologie, erneut eine Marktanalyse vorgenommen, deren Schwerpunkt auf den zu diesem Zeitpunkt zur Verfügung stehenden E-Learning-Plattformen gelegt wurde. Parallel zu der im Rahmen dieser Arbeit vorgenommenen Entwicklung wurden von zahlreichen kommerziellen Unternehmen E-Learning-Systeme entwickelt oder weiterentwickelt. Die auf der Messe vorgestellten Produkte wurden zum Vergleich auf den jeweiligen Grad der Erfüllung der in Kapitel 2 aufgestellten Anforderungen hin untersucht.

Im Folgenden wird zunächst ein allgemeiner Überblick über die wichtigsten der auf der LEARNTEC 2003 feststellbaren Trends im E-Learning-Bereich gegeben. Im Anschluß daran wird eine Reihe von einzelnen Produkten vorgestellt. Das Auswahlkriterium war dabei eine möglichst umfassende Funktionalität sowie der jeweilige Grad der Erfüllung der bereits zu Projektbeginn formulierten Anforderungen. Eine Beschränkung auf die hier dargestellten Programme war notwendig, da zum einen eine gesamte Darstellung der 307 Aussteller, die auf der LEARNTEC mit ihren Produkten vertreten waren, den Rahmen dieser Arbeit sprengen würde und zum anderen die überwiegende Anzahl der auf dieser Messe vorgestellten Produkte für die hier verfolgten Zwecke keine technische Relevanz besaßen.[195]

3.3.1 Trends auf der LEARNTEC 2003

Im Laufe der letzten Jahre zeichnet sich im E-Learning-Bereich eine deutliche Tendenz zu vielseitig einsetzbaren, inhaltsunabhängigen Plattformen ab. Anstelle der zuvor weit verbreiteten, speziellen Entwicklungen für einzelne Einsatzbereiche, bei denen es sich zumeist um technische Insellösungen handelte, die zu anderen Systemen in der Regel nicht kompatibel waren, dominieren jetzt Systemlösungen den Markt.

Herkömmliches "computer based training" (CBT), das beispielsweise durch Verteilung von multimedialen CDs realisiert wurde, ist weitgehend durch "web based training" (WBT) ersetzt worden, bei dem die benötigten Lehrmaterialien über das Internet verbreitet werden und die Teilnehmer zumeist die Möglichkeit haben, untereinander oder mit den Lehrenden auf verschiedene Art und Weise auf elektronischem Wege zu kommunizieren, einerseits in asynchroner Form per E-Mail oder über Diskussionsforen und elektronische "Schwarze Bretter", andererseits synchron in Form von Chats oder auch in darüber hinausgehenden virtuellen Lernwelten.

---

[193] Diese Angabe bezieht sich auf die dreijährige Laufzeit des Projektes Monist. Die in der vorliegenden Arbeit entworfene E-Learning-Plattform war zu diesem Zeitpunkt bereits im Wesentlichen fertiggestellt.

[194] Die LEARNTEC 2003 fand vom 04.-07.02.2003 im Kongresszentrum Karlsruhe statt. Der Großteil der Evaluation der in diesem Kapitel beschriebenen Software wurde auf der Grundlage eines mehrtägigen Messebesuches im Rahmen des Projektes Monist vorgenommen. Informationen über die LEARNTEC sind im Internet erhältlich unter der Adresse: http://www.learntec.de. Neben dieser zur jährlichen Messe jeweils komplett aktualisierten Seite ist die Homepage zur LEARNTEC 2003 in ihrem archivierten Originalzustand noch unter der Adresse http://feuerflitzer.de/lt2003website erhältlich.

[195] Die Palette der hier nicht dargestellten Produkte reichte dabei von Software für die Verwaltung von Universitäten und Unternehmen über Hardware für elektronische Präsentationen und Konferenzen bis hin zu traditionellen, nicht elektronischen Hilfsmitteln wie Wandtafeln und Flip Charts.

Die Lehrmaterialien werden dabei von den Autoren auf einem zentralen Server zum Download bereitgestellt, nachdem sie vorher entweder am eigenen Computer oder unter Verwendung bestimmter Werkzeuge direkt im System erstellt wurden. Dabei kommen alle Typen von Medien zum Einsatz, die in einem Webbrowser entweder direkt (HTML, Grafiken etc.) oder unter Verwendung marktüblicher Browser-Plugins (Adobe Acrobat, Shockwave Flash etc.) angezeigt werden können. Eine über diese Standardkomponenten hinausgehende Installation von Programmen auf dem Client ist dabei nicht vorgesehen, was allerdings auch die Integration von Simulationsprogrammen, wie sie beispielsweise im Rahmen des Projektes Monist verwendet werden, zumindest erheblich erschwert.

Eine Möglichkeit zur Speicherung und Verwaltung von Daten auf dem Client ist in den webbasierten Systemen nicht gegeben, da offenbar allgemein davon ausgegangen wird, daß bei der Verwendung von WBT-Software durchgängig ein Internet-Anschluß zur Verfügung steht. Für kommerzielle Unternehmen, die die Hauptzielgruppe derartiger Anwendungen bilden, oder auch für professionelle Schulungsagenturen, die im Auftrag verschiedener Firmen Kurse zu bestimmten Themen durchführen, ist diese Annahme durchaus berechtigt. Im universitären Bereich kann dies jedoch allenfalls für reine Präsenzveranstaltungen vorausgesetzt werden, die in Räumlichkeiten stattfinden, die mit der notwendigen Infrastruktur ausgestattet sind. Derartige Einschränkungen würden jedoch einen Verzicht auf die in Kapitel 2.7 beschriebenen Vorteile bedeuten, die sich durch die Möglichkeit eines Offline-Einsatzes der Lernsoftware ergeben.

Ein weiterer, auffälliger Trend ist die auf breiter Front stattfindende Abkehr von dem Versuch, Präsenzlehre und durch Lehrpersonal angeleitetes Training komplett durch die elektronische Lehre zu ersetzen. Nachdem zuvor jahrelang computerbasiertes Training als die beste Lösung zur Vermittlung von Wissen angepriesen wurde und dabei stets auf die reduzierten Kosten für Unternehmen durch den Verzicht auf die Beschäftigung von Lehrkräften hingewiesen wurde, hat sich diese Sichtweise, offenbar aufgrund schlechter Erfahrungen und unerwartet geringer Lernerfolge, inzwischen gewandelt. Nachdem der Wert einer persönlichen Ansprache und insbesondere der Möglichkeit zur Rückfrage und zur Diskussion mit Experten über das vermittelte Wissen erkannt wurde, wird jetzt unter dem neuen, die LEARNTEC 2003 beherrschenden Stichwort "Blended Learning" die Kombination von Präsenzlehre, elektronischer Kommunikation und CBT/WBT als bestimmender Faktor für den Lernerfolg in den Mittelpunkt gestellt.[196]

3.3.2 E-Learning-Plattformen

Im Folgenden werden die auf der LEARNTEC evaluierten E-Learning-Plattformen in alphabetischer Reihenfolge der Firmennamen dargestellt, wobei zu jedem Produkt die verwendeten Quellen (meist in Form von Internetseiten) angegeben werden. Dabei wird in erster Linie auf die Besonderheiten der einzelnen Systeme eingegangen. Da praktisch alle hier betrachteten Produkte den im vorangehenden Abschnitt beschriebenen Trends folgen, es sich insbesondere durchgängig um WBT-Lösungen handelt, die zum Einsatz im Rahmen von "Blended Learning" Konzepten vorgesehen sind, wird diese Tatsache nicht bei jedem einzelnen Produkt gesondert beschrieben.

[196] Diese Erkenntnis ist nicht neu, sondern beschreibt eine Lehrform, wie sie im universitären und außeruniversitären Bereich schon bekannt war, lange bevor das neue Schlagwort "Blended Learning" erfunden wurde. Man vergleiche hierzu z. B. den Projektantrag zum Projekt Monist, in dem bereits im Jahr 2000 eine solche Kombination gefordert wurde, in der *Projektskizze zur Einreichung bei dem Förderprogramm des Bundesministeriums für Bildung und Forschung: "Neue Medien in der Bildung",* erhältlich unter: http://www.monist.de bzw. http://natrix.biologie.uni-bielefeld.de/monist/Archiv/monistskizze_oz.html.

3.3.2.1 bureau42: broker42, trainer42 und author42

Die bureau42 GmbH[197], ein Spin-Off-Unternehmen der Fraunhofer-Gesellschaft[198], entwickelt Produkte aus den Bereichen Wissensmanagement und E-Learning. Technische Grundlage der Produkte der Firma sind ein OpenBase™ Datenbankserver (wobei auch andere Datenbanken verwendet werden können), ein WebObjects™ Applikationsserver sowie der frei verfügbare Apache Webserver (an dessen Stelle ebenfalls auch andere Webserver verwendet werden können). Neben dem Dokumentenmanagementsystem broker42[199], das durch zusätzliche Knowledge-Management-Funktionalitäten auf Basis von für jeden einzelnen Benutzer personalisierten semantischen Netzwerken die Suche nach benötigten Informationen vereinfacht, entwickelt und vertreibt die bureau42 GmbH die E-Learning-Plattform trainer42 und das Autorenwerkzeug author42.

Ebenso wie die Informationsrecherche in broker42 werden auch die Lerninhalte in trainer42 individuell den Bedürfnissen des jeweiligen Benutzers angepaßt, wobei das Vorwissen ebenso berücksichtigt wird wie der individuelle Lernstil: "Jeder Mensch lernt anders. Jeder hat sein eigenes Lerntempo und seinen eigenen Lernstil. Strikt vorgegebene Lernpfade oder die Möglichkeit zum explorativen Lernen - trainer42 unterstützt hierzu sämtliche Rollen im Bildungsprozess, um dem Lerner den Zugang zur Ressource Wissen zu optimieren."[200] Das System ist eingeteilt in verschiedene Umgebungen für Lerner, Tutoren, Autoren und Administratoren, die jeweils unterschiedliche Funktionen zur Verfügung stellen. Daneben stehen als Kommunikationsmöglichkeiten Diskussionsforen sowie die Möglichkeit zu allgemein zugänglichen Anmerkungen zu enthaltenen Dokumenten zur Verfügung.

Für die Autoren existiert eine spezielle Umgebung zum Erstellen von Lehrmaterialien: "Ohne jegliche Programmierkenntnisse können Kursautoren komplexe, vernetzte Lerninhalte erstellen. Die Lernumgebung bietet dazu einen Kurseditor zur Bearbeitung von Inhalten, einen Strukturedi tor zur Organisation der Inhalte und einen Indexeditor zur Erstellung von Kursindizes und Glossaren."[201] In das System eingestellte Lernobjekte können in anderen Kontexten wiederverwendet werden. "Zur weiteren Entlastung der Autoren kann der in jedem Kurs vorhandene Kursindex automatisch aus den Kursmaterialien erstellt werden. Es entsteht eine ' semantische Wolke' der Kursmaterialien."[202]

Die Erstellung von multimedialen Lerninhalten wird zusätzlich von dem Autorentool author42 unterstützt, das außerdem mehrere zusätzliche Funktionalitäten bietet: "Lassen Sie sich bei der Testerstellung durch den author42 Testgenerator führen. Arbeiten Sie beim Erstellen von Lerninhalten in Teams und mischen Sie unterschiedliche fachliche und didaktische Kompetenzen. Und schließlich: Exportieren Sie das Ganze in Ihr gewünschtes Design"[203]. Zur Vereinfachung des Arbeitsprozesses bietet das Programm eine Reihe von Vorlagen für Inhaltsseiten und verschiedene Formen von elektronischen Tests. Lehrmaterialien können wahlweise in HTML, XML oder PDF exportiert werden.

---

[197] Homepage: http://www.bureau42.de.
[198] Homepage: http://www.fraunhofer.de.
[199] Für weitere Informationen siehe: broker42 - Personalisierte Information Just-in-time und Just-enough, Produktbroschüre der bureau42 GmbH zum DMS broker42, im Internet zum Download erhältlich unter der Adresse http://www.bureau42.de/download/broker42_Produktblatt.pdf.
[200] trainer42 - Personalisiertes Learning on demand, Produktbroschüre, im Internet zum Download erhältlich unter der Adresse: http://www.bureau42.de/download/trainer42_Produktblatt.pdf, Seite 1.
[201] A. a. O., Seite 2.
[202] A. a. O., Seite 2.
[203] author42 - Einfaches Erstellen wieder verwendbarer Lernobjekte, Produktbroschüre, zum Download erhältlich unter der Adresse: http://www.bureau42.de/download/author42_Produktblatt.pdf, Seite 1.

3.3.2.2 EEDO: FORCE TEN, FORCE TEN Simulator und WebClass

Das E-Learning-System FORCE TEN der EEDO Knowledgeware Corporation[204] bietet den Lernenden eine vollkommen browserbasierte Oberfläche. Autoren können hingegen ihre Inhalte auch zuerst in verschiedenen Anwendungen erstellen, beispielsweise unter Verwendung von Microsoft® Office®. Beim Einbringen in FORCE TEN werden diese Dateien dann automatisch in webkonforme Formate konvertiert. Neben der Erstellung und Bereitstellung von Kursen bietet das Programm Möglichkeiten zur Verwaltung von Benutzern, für die personalisierte, dem individuellen Leistungsstand entsprechende Trainingspläne erstellt werden können. "FORCE TEN´s ' Personalisiertes Wissensportal' stellt den Lernenden verschiedene Zugangswege zu den Inhalten zur Verfügung. Dazu gehören ein profilgesteuerter, virtueller Lernraum, eine komplexe Suchmaschine sowie gemeinsame Kommunikationstools."[205]

Außerdem bietet EEDO als Zubehör noch zwei weitere Produkte an. Der FORCE TEN Simulator ist ein Werkzeug zum simulationsbasierten Training für Anwendersoftware. "Die Oberfläche der zu schulenden Software kann einfach eingebunden werden. Auf dieser Grundlage lassen sich Szenarios erstellen, welche die typischen Arbeitsschritte abbilden, die mit der Software verbunden sind."[206] Derartige Simulationen bestehender Programme werden in erster Linie dazu angewendet, Anwender im Umgang mit Standardsoftware wie z. B. Office-Produkten oder SAP zu unterweisen. EEDO bietet außerdem unter dem Namen WebClass ein Echtzeit-Konferenzsystem an, das sowohl für Online-Konferenzen als auch für Teleteaching verwendet werden kann. "Wenn Sie WebClass nutzen, können Sie beispielsweise Software schulen und anderen Anwendern den Blick auf Ihre Bildschirmoberfläche gewähren. Dokumente und Anwendungen werden allein über die Nutzung eines Webbrowsers für jeden sichtbar und nutzbar."[207]

3.3.2.3 ets: CourseFactoryWeb® und DistanceLearningSystem®

Die ets e-learning corporation[208] bietet unter dem Namen CourseFactoryWeb® (CFW) ein Programm zur Erstellung von webbasierten Trainingskursen an. "Die Systemfamilie umfasst unter anderem ein leistungsfähiges Autorensystem, Instrumente zur Integration von Text-, Graphik- oder Soundelementen, ein Organisationssystem zur Archivierung von Ressourcen und einen integrierten Generator für interaktive Tests."[209] In die mit CFW erstellten, seitenorientierten Kurse, die über in das System eingebaute Werkzeuge ohne Programmierkenntnisse angelegt werden können, lassen sich verschiedene Typen von Medien (Text, Grafik, Sound, Animation etc.) einbauen. Lehrinhalte können auch von mehreren Autoren per Internet gemeinsam erstellt werden. Die Verwaltung der Daten erfolgt in einem eingebauten Content Management System.

Fertige Kurse können entweder direkt in CFW ausgeführt oder in das ebenfalls von ets angebotene DistanceLearningSystem® (DLS) eingestellt werden, bei dem es sich um ein webbasiertes System zur Verwaltung und Verteilung von Lehrinhalten handelt, das

---

[204] Homepage: http://www.eedo.com, deutsche Version unter: http://www.eedo.com/deutsch.
[205] FORCE TEN: Die schnelle und mächtige eLearning Plattform, Produktbroschüre, im Internet erhältlich unter der Adresse: http://www.eedo.com/deutsch/produkte/ForceTen_brochure_german.pdf, Seite 2.
[206] EEDO Produktübersicht, Broschüre der Firma EEDO, im Internet zum Download erhältlich unter der Adresse: http://www.eedo.com/deutsch/produkte/Overview_german.pdf, Seite 2.
[207] EEDO WebClass, Produktbroschüre der Firma EEDO, im Internet zum Download erhältlich unter der Adresse: http://www.eedo.com/deutsch/produkte/webclass_german.pdf, Seite 2.
[208] Homepage: http://www.ets-online.de.
[209] Produktübersicht unter der Internet-Adresse http://www.ets-online.de/etso01q02/10000.html.

zusätzlich über Administrations- und Kommunikationsfunktionalitäten verfügt. DLS kann in Form einer elektronischen Akademie betrieben werden, in der verschiedenste elektronische Lehrveranstaltungen mit Teilnehmern und Tutoren besetzt werden, die jeweils eigene Sichten der Kurse mit auf ihre Rollen zugeschnittenen Funktionalitäten erhalten. Außerdem stehen zur asynchronen Kommunikation Diskussionsforen und zur synchronen Kommunikation Chats zur Verfügung. "Komplett in DLS integriert ist außerdem das Konferenzsystem CentraOne, welches als eines der führenden Systeme auf diesem Gebiet Funktionen wie Multipoint-Audio- und -Videokonferenzen, Whiteboard oder Application-Sharing zur Verfügung stellt."[210] CFW läuft unter mehreren UNIX-Betriebssystemen (z. B. HP-UX, Solaris, Linux) und verwendet mit dem Apache Web-Server, Perl und der Datenbank MySQL frei erhältliche Software als Grundlage.

3.3.2.4 Hyperwave eLearning Suite

Die bereits in Kapitel 3.1.2.4 im Zusammenhang mit dem Hyperwave Information Server beschriebene eLearning Suite der Hyperwave AG[211] wurde in den letzten Jahren kontinuierlich weiterentwickelt. Als erste der hier betrachteten E-Learning-Plattformen auf den Markt gekommen, verfügt die eLearning Suite auch heute noch gegenüber den meisten anderen Produkten über einen technologischen Vorsprung, der insbesondere auf die technischen Eigenschaften des zugrundeliegenden Hyperwave Information Servers zurückzuführen ist. Besonders nützlich ist, daß bereits im DMS enthaltene Dokumente unmittelbar für die Gestaltung elektronischer Lehreinheiten verwendet werden können, was gerade in Unternehmen oftmals den Aufwand für die Autoren erheblich verringert.

Dazu kommen die bereits bewährten Kommunikationsfunktionalitäten, Möglichkeiten zur kollaborativen Erstellung und Bearbeitung von Lehrmaterialien und die bereits im Hyperwave Information Server realisierte, auf Benutzern und Gruppen basierende Personalisierung der angezeigten Inhalte. Außerdem gibt es Tools zur Überwachung des Lernerfolges, und sogenannte "aktive Dokumente" sammeln selbständig dazu gestellte Fragen und die darauf gegebenen Antworten, so daß die eingestellten Informationen im Laufe der Zeit in immer höherem Maße selbsterklärenden Charakter annehmen.

3.3.2.5 imc: CLIX® Campus

Die imc information multimedia communication AG[212] entwickelt und vertreibt mit CLIX® ein E-Learning-System, das in unterschiedlichen Versionen für Unternehmen (CLIX® Enterprise) und für Hochschulen (CLIX® Campus) erhältlich ist. Dabei ist ein wesentliches Merkmal die auf einer durchgängigen Personalisierung aufbauende Benutzerführung. "Alle in CLIX Campus angebotenen Inhalte und Dienste können auf die spezifischen Bedürfnisse der unterschiedlichen Benutzergruppen angepasst werden. Fakultäten, Fachbereiche, Vordiplom- sowie Hauptdiplomangebote, Forschungsprojekte oder Gasthörer - jeder kann die für ihn relevanten Angebote direkt nutzen."[213] CLIX Campus kann universitäre Verwaltungsstrukturen abbilden: "Durch Schnittstellen zu Verwaltungssystemen von SAP, Oracle, Peoplesoft oder HIS können z. B. Personal- und Studierendenverwaltung oder Ressourcenmanagement integriert werden."[214]

---

[210] Produktbeschreibung unter der Internet-Adresse: http://www.ets-online.de/etso01q02/11200.html.
[211] Homepages: http://www.hyperwave.de bzw. http://www.hyperwave.com, beide mehrsprachig.
[212] Homepage: http://www.im-c.de (mehrsprachig).
[213] Produktbeschreibung unter der Internet-Adresse: http://www.im-c.de/homepage/clix_campus.htm.
[214] A. a. O.

Neben diesen administrativen Funktionalitäten bietet das System auch die Möglichkeit zur Einspeicherung und Verbreitung elektronischer Lehrmaterialien, die bestimmten Kursen zugeordnet werden und nach Art eines Content Management Systems verwaltet werden. Eine Besonderheit bietet dabei das ebenfalls von der imc AG entwickelte Aufzeichnungsprogramm Lecturnity. "Vor der Aufzeichnung erstellt ein Trainer [...] seine MS Powerpoint® Präsentation als Ausgangsbasis. Im Rahmen seiner Lehrveranstaltung wird dann die MS Powerpoint® - basierte Präsentation mit LECTURNITY® aufgezeichnet. Dabei verwendet der Trainer zusätzlich zu seinen Folien ein Headset mit Mikrofon und - falls gewünscht - ein White- oder Smartboard, um die Präsentation lebendig zu gestalten und im Präsentationsverlauf zu verändern. Auch eine Videoaufzeichnung kann zusätzlich erfolgen."[215] Direkt nach der Veranstaltung kann die komplette Vorlesung unter Verwendung aller Aufzeichnungen und Kommentare als multimediales Dokument ins Internet gestellt werden.

3.3.2.6 ORBIS NetCoach

Das von der ORBIS AG[216] entwickelte Produkt NetCoach ist webbasiertes E-Learning-System mit einem integrierten Autorensystem. Unter Verwendung üblicher Medientypen aus den Bereichen Text, Grafik, Animation, Audio und Video können Kurse für den Einsatz im Internet erstellt werden, wobei die Erstellung der Webseiten nach den Vorgaben der Autoren vom System unterstützt wird, so daß keinerlei Programmierung erforderlich ist. "Entsprechend der vorgegebenen Struktur wird automatisch das Inhaltsverzeichnis und die Guided Tour erzeugt."[217] Auch Glossare und Literaturverzeichnisse können vom Autor zur Verfügung gestellt werden. Eine Volltextsuche (aber keine Suche in Metadaten, da diese vom System nicht verwaltet werden) ist ebenso enthalten wie Chats und ein integriertes E-Mail-System zur Kommunikation.

Für die einzelnen Benutzer kann ein personalisierter Zugang in Abhängigkeit von den bisherigen Lernfortschritten und richtig beantworteten Testfragen erzeugt werden. Noch vorhandene Wissenslücken werden vom System erkannt und berücksichtigt. Für die Erstellung von Übungsaufgaben und Tests enthält das System eine spezielle Oberfläche für die Autoren. "Dem Autor stehen u.a. Multiple-Choice-Fragen, Freitext-Fragen und Lückentext-Aufgaben zur Verfügung. Die Antworten werden automatisch korrigiert und können mit einer Begründung für die Korrektur versehen werden."[218] Lernenden wird die Möglichkeit gegeben, entweder linear dem Kursverlauf zu folgen oder frei innerhalb der Lernumgebung zu navigieren.

Während für die Bedienung des Programmes clientseitig nur ein Webbrowser benötigt wird, wird serverseitig ein ungewöhnliches System verwendet: "NetCoach ist in einem speziellen Server implementiert, dem Common Lisp Hypermedia Server (CL-HTTP). CL-HTTP wird vom MIT (Massachusetts Institute of Technology) für das Amerikanische Weiße Haus entwickelt und ist für Künstliche Intelligenz-Anwendungen im World Wide Web konzipiert. Die Serversoftware steht für die Betriebssysteme MS Windows NT, MS Windows 2000 und MacOS zur Verfügung."[219]

---

[215] Produktbeschreibung unter der Internet-Adresse:
http://www.im-c.de/homepage/lecturnity/funktionen_ueberblick.htm
[216] Homepage: http://www.orbis.de (mehrsprachig).
[217] Produktbeschreibung unter der Internet-Adresse:
http://www.orbis.de/products/elearn_uebersicht.cfm?StaID=elearn3
[218] Produktbeschreibung unter der Internet-Adresse:
http://www.orbis.de/products/elearn_uebersicht.cfm?StaID=elearn2
[219] Produktbeschreibung unter der Internet-Adresse: http://www.orbis.de/products/elearn_technologie.cfm

3.3.2.7 WebCT VISTA und Campus Edition

Die E-Learning-Plattform der Firma WebCT Inc.[220] wird in zwei unterschiedlichen Varianten herausgegeben: WebCT VISTA[221] ist auf die Bedürfnisse von kommerziellen Unternehmen ausgerichtet, während WebCT Campus Edition[222] für den universitären Bereich vorgesehen ist. Die Campus Edition ist kompatibel zu mehreren Programmen, die international in Universitäten zu Verwaltungszwecken eingesetzt werden, und wird weltweit in vielen Bildungseinrichtungen verwendet.[223] Wie der Name schon vermuten läßt, handelt es sich um ein komplett webbasiertes System, das "tools to help students communicate and collaborate, manage their assignments, move effectively through course material, and continually monitor their own progress"[224] zur Verfügung stellt.

Dies umfaßt die Bereitstellung von multimedialen Inhalten ebenso wie Systeme zur Lernzielkontrolle, die sowohl den Lernenden als auch den Lehrenden einen Überblick über den individuellen Lernfortschritt verschaffen können. Online-Tests und Umfragen ergänzen diese Auswertungen. Zur Kommunikation existieren Chats, Diskussionsforen und ein integriertes E-Mail-System, das die Verwaltung kursspezifischer Mailinglisten ermöglicht. Für die Bereitstellung von Inhalten stehen den Autoren Werkzeuge zur Verfügung, über die einzelne Dateien genauso wie komplexe Kurse in WebCT eingestellt und automatisch mit von der jeweiligen Einrichtung vorgegebenen Designelementen versehen werden können, ohne daß dafür Programmierkenntnisse erforderlich sind. Allerdings ist auch dieses relativ weit verbreitete System nur auf die Verteilung von statischen (Text, Bild) oder nicht-interaktiven dynamischen (Audio, Video) Inhalten ausgelegt, die im Rahmen eines Internet-Browsers oder unter Nutzung von Plugins (Adobe Acrobat, Shockwave Flash etc.) angezeigt werden können.

3.3.2.8 Schlußfolgerungen

Die untersuchten E-Learning-Plattformen, die exemplarisch für weitere Systeme mit ähnlichen Funktionalitäten stehen, bieten durchgängig leistungsfähige Funktionalitäten zur webbasierten Verbreitung von Lehrmaterialien und verfügen darüber hinaus zumeist über ein breites Spektrum an Kommunikationsmöglichkeiten, die wertvolle Hilfsmittel für Lehrende und Lernende darstellen. Keines der Programme verfügt jedoch über eine Offline-Komponente, wie sie zur Umsetzung der in Kapitel 2.7 aufgestellten Anforderungen benötigt wird, und diese ist auf Basis rein webbasierter Systeme auch kaum zu realisieren. Außerdem sind die am Markt erhältlichen Produkte nicht zur Integration von Programmen zur Erstellung und Ausführung konfigurierbarer Modellsimulationen wie z. B. im Rahmen des Projektes Monist geeignet, da entsprechende Schnittstellen für die Einbindung dynamischer, interaktiver Inhalte fehlen, und sind somit nicht als Basis für die im Rahmen dieser Arbeit zu entwerfende E-Learning-Plattform verwendbar.

---

[220] Homepage: http://www.webct.com.

[221] Siehe hierzu die Informationsbroschüre zum Produkt WebCT VISTA, die im Internet zum Download erhältlich ist unter der Adresse: http://www.webct.com/service/ViewContent?contentID=9102515.

[222] Siehe hierzu die Informationsbroschüre zum Produkt WebCT Campus Edition, die zum Download erhältlich ist unter der Adresse: http://www.webct.com/service/ViewContent?contentID=6205796.

[223] Die Verbreitung in Deutschland ist jedoch relativ gering. WebCT wird bisher nur an den Universitäten Konstanz, Potsdam, Hamburg und Kaiserslautern eingesetzt. Eine Liste von Bildungseinrichtungen, die die WebCT Campus Edition verwenden, findet sich in der von WebCT herausgegebenen Broschüre "Transforming the Educational Experience", die im Internet zum Download erhältlich ist unter der Adresse: http://www.webct.com/service/ViewContent?contentID=6205875, Seiten 2 und 11.

[224] Informationsbroschüre zum Produkt WebCT Campus Edition, im Internet zum Download erhältlich ist unter der Adresse: http://www.webct.com/service/ViewContent?contentID=6205796, Seite 1.

4 Systemarchitektur der E-Learning-Plattform

Nachdem in den beiden vorangehenden Kapiteln die Anforderungen an die zu entwickelnde E-Learning-Plattform formuliert wurden und gezeigt wurde, warum sich keines der auf dem Software-Markt erhältlichen Programme als Grundlage für das neue System eignet, soll in diesem und den darauffolgenden Kapiteln präsentiert werden, auf welche Weise die im Rahmen dieser Arbeit vorgenommene Eigenentwicklung die zuvor aufgestellten Anforderungen umsetzt. Als Grundlage dafür wird im Folgenden zunächst die Systemarchitektur vorgestellt, die die Basis für den weiteren Systementwurf und die praktische Umsetzung dieser Überlegungen bildet.

Die erste grundlegende Entwurfsentscheidung beruht auf der in Kapitel 2.7 aufgestellten Anforderung nach einer Möglichkeit, das System offline einsetzen zu können, und der in Kapitel 2.8 beschriebenen Mechanismen zur Datenverteilung. Wie bereits dargelegt wurde, ist es zur Realisierung eines Offline-Betriebes erforderlich, eine Möglichkeit zur Datenspeicherung auf dem Computer jedes einzelnen Benutzers vorzusehen, die ebenso wie ihr Gegenstück auf dem Server zur Ablage beliebiger multimedialer Daten und der zu deren Verwaltung verwendeten Metadaten geeignet sein muß. Da auf dem Client und dem Server also praktisch die gleichen Funktionalitäten vorhanden sein müssen[225], liegt es nahe, für die Datenspeicherung einen gemeinsamen Mechanismus zu schaffen, zumal dadurch auch keine doppelte Entwicklungsarbeit anfällt. Dies betrifft insbesondere auch die Anbindung an die verwendete Datenbank.[226]

Es ist unmittelbar einsichtig, daß das zu entwickelnde System nicht auf der Basis einer Zwei-Schichten-Architektur (siehe Abbildung 4.1) aufgebaut werden kann[227], bei der die Anwendungen direkt auf die Datenbank zugreifen. Dies würde eine gemeinsame Komponente zur einheitlichen Datenspeicherung für alle in die E-Learning-Plattform zu integrierenden Programme unmöglich machen. Außerdem müßte in diesem Fall jede

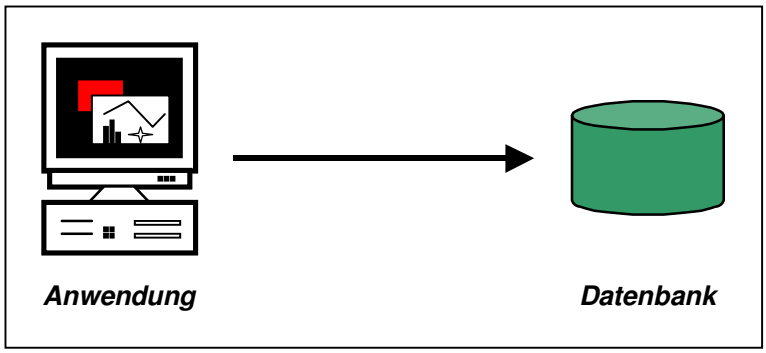

Abbildung 4.1: Zwei-Schichten-Architektur

[225] Diese Feststellung gilt natürlich nur in qualitativer Hinsicht. Quantitativ gesehen muß der Server eine erheblich größere Datenmenge speichern, verarbeiten und vor allem auf entsprechende Anfragen von den Clients zum Download ausliefern. Dieses Problem läßt sich allerdings weitgehend durch die Verwendung entsprechend leistungsfähigerer Hardware und einer auf das höhere Datentransfervolumen ausgelegten Internetverbindung lösen.

[226] Frei verfügbare Datenbanken wie beispielsweise MySQL (Homepage: http://www.mysql.com), die eine Installation des gesamten Datenspeicherungssystems bei jedem Benutzer überhaupt erst realisierbar machen, sind heute in hinreichender Leistungsfähigkeit erhältlich, um den Server-Betrieb zu ermöglichen, und gleichzeitig mit einem vertretbar geringen Installationsaufwand und Ressourcenverbrauch verbunden, um auch einen Einsatz im Client zu rechtfertigen.

[227] Zu Schichtenarchitekturen siehe: Helmut Balzert, Lehrbuch der Software-Technik, Band 1, Spektrum Akademischer Verlag, Heidelberg, Berlin, 2. Auflage, 2000, Seiten 696-699.

Anwendung die zum Laden oder Speichern von Objekten benötigten Datenbankbefehle (z. B. SQL-Statements) selbst erzeugen, was einen großen und zudem redundanten Programmieraufwand darstellen würde.[228]

Aus diesem Grund ist es erforderlich, zwischen den Programmen und der Datenbank zumindest noch eine weitere Ebene einzufügen, in welcher die in der Datenbank abgelegten Strukturen auf ein Datenmodell im Rahmen der Datenstrukturen der verwendeten Programmiersprache abgebildet werden, auf denen dann die Applikationen operieren können. Damit wird diesen eine Sichtweise auf die Daten zur Verfügung gestellt, die es ermöglicht, Veränderungen rein auf den Datenstrukturen und ihren Zusammenhängen vorzunehmen, ohne dabei die technischen Details der Datenspeicherung berücksichtigen zu müssen. Abbildung 4.2 zeigt eine derartige Drei-Schichten-Architektur[229], wie sie in der Regel in reinen Desktop-Anwendungen verwendet wird.[230]

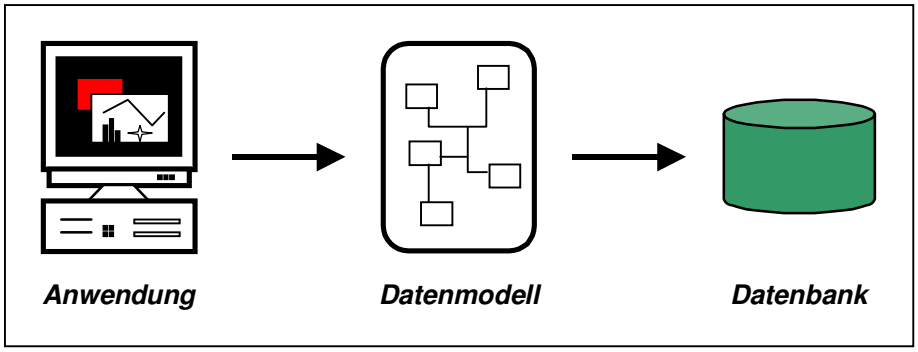

Abbildung 4.2: Drei-Schichten-Architektur

Damit ist jedoch ebenfalls noch kein hinreichender Abstraktionsgrad erreicht, der eine allgemeine und einfache zu verwendende Schnittstelle für Speicherung und Austausch von Lernobjekten und den mit diesen verbundenen Metadaten ermöglichen würde. Aus der Sicht der einzelnen Programme ist es beispielsweise nur von Interesse, durch welche Metadaten ein bestimmtes Objekt klassifiziert und beschrieben wird, nicht jedoch, in welche Datenstrukturen einer bestimmten Programmiersprache diese Metadaten aus der Datenbank heraus eingelesen wurden.

Um eine vereinfachte Schnittstelle für Operationen auf den Lernobjekten zur Verfügung zu stellen, wird deshalb zwischen dem Datenmodell und den Anwendungen noch eine vierte Schicht eingefügt, die die Datenstrukturen und Methoden der darunterliegenden Schicht in abstrakter Weise kapselt. Die Objekte dieser Schicht (im nachfolgenden Kapitel 5 als "virtuelle Lernobjekte" bezeichnet) entsprechen dem in der objektorientierten Softwareentwicklung verbreiteten Entwurfsmuster einer Fassade.[231] "Eine Fassade ist ein Entwurfsmuster, das ein Subsystem hinter einer einfachen Schnittstelle verbirgt."[232]

---

[228] Dies bringt ebenfalls einen nicht zu unterschätzenden Aufwand für die Fehlerbehandlung möglicher Datenbankfehler mit sich und bedeutet außerdem für Entwickler, daß im gleichen Programm mit einer zusätzlichen Sprache programmiert werden muß, was die Fehleranfälligkeit deutlich erhöht.

[229] Aufbau und Eigenschaften einer solchen Drei-Schichten-Architektur werden beispielsweise dargestellt in: Helmut Balzert, Lehrbuch der Software-Technik, Band 1, Spektrum Akademischer Verlag, Heidelberg, Berlin, 2. Auflage, 2000, Seiten 696-699.

[230] A. a. O., Seite 987.

[231] Zu Fassaden siehe: Erich Gamma, Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software, Addison-Wesley-Longman, Bonn, 1996, Seiten 212-217.

[232] Helmut Balzert, Lehrbuch der Software-Technik, Band 1, Spektrum Akademischer Verlag, Heidelberg, Berlin, 2. Auflage, 2000, Seite 1005.

Durch die Einführung von Fassadenobjekten werden direkte Abhängigkeiten zwischen Subsystemen minimiert. Für den hier vorliegenden Fall der Anwendungsschicht und der Datenmodellschicht bedeutet dies, daß die Anwendungen durch die von der Fassade zur Verfügung gestellten Methoden über eine abstrakte Schnittstelle auf die verwalteten Lernobjekte zugreifen können, ohne dabei die tatsächliche technische Implementation in der zugrundeliegenden Datenmodellschicht berücksichtigen zu müssen. "Die Klienten kommunizieren mit dem Subsystem, indem sie Anfragen an die Fassade schicken, welche sie dann an das zuständige Subsystemobjekt oder die zuständigen Subsystemobjekte weiterleitet. Obwohl die Subsystemobjekte die eigentliche Arbeit ausführen, muß die Fassade möglicherweise zusätzliche Arbeit ausführen, um seine Schnittstelle auf die Subsystemschnittstellen abzubilden. Die Klienten, welche die Fassade benutzen, müssen ihre Subsystemobjekte nicht direkt benutzen."[233] Durch das Einfügen der Fassadenschicht entsteht eine Vier-Schichten-Architektur (siehe Abbildung 4.3).

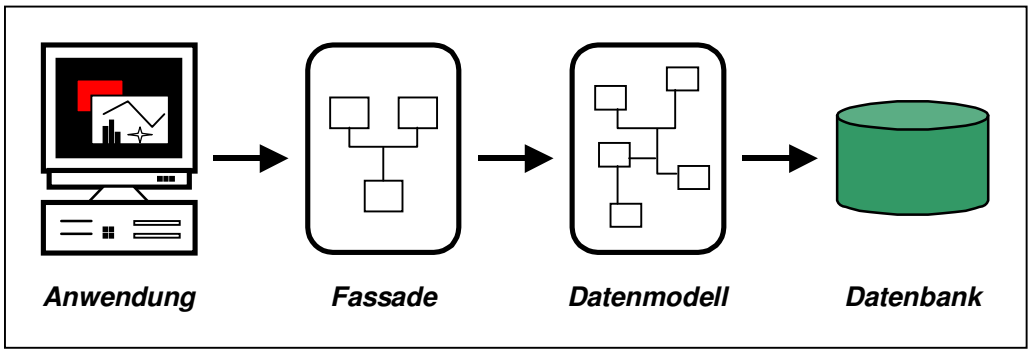

Abbildung 4.3: Vier-Schichten-Architektur

Über die Fassade können jetzt beliebige Anwendungen in einheitlicher Weise auf die Lernobjekte zugreifen. Dies kann innerhalb des Systems auf einem einzelnen Computer geschehen, ebenso jedoch zwischen unterschiedlichen Installationen der Software auf räumlich getrennten Computern über das Internet, sofern ein geeigneter Mechanismus für die Übertragung der Methodenaufrufe und der Rückgabewerte zur Verfügung steht. Wird ein geeignetes, plattformunabhängiges[234] Austauschformat (z. B. XML) gewählt, dann können sogar Anwendungen, die in vollkommen anderen Programmiersprachen geschrieben sind, die Dienste der Datenspeicherungskomponente nutzen.

Auf diese Weise steht eine Möglichkeit zur Kommunikation zwischen Anwendungen und der Datenspeicherung zur Verfügung. Betrachtet man jedoch letztere ebenfalls als eine spezielle Anwendung, kann man im nächsten Schritt dahingehend verallgemeinern, daß damit im Prinzip auch ein universeller Mechanismus für die Interaktion zwischen unterschiedlichen Anwendungen geschaffen werden kann. Zu diesem Zweck muß lediglich das Austauschformat flexibel genug gewählt werden, um den Anforderungen von beliebigen Applikationen zu genügen. Über ein entsprechendes Kommunikationsmodul können dann Anwendungen unterschiedslos sowohl innerhalb eines Computers als auch rechnerübergreifend Informationen austauschen. Für die E-Learning-Plattform wurde diese Funktionalität unter Verwendung eines XML-RPC-Kommunikationsmechanismus implementiert, der in Kapitel 6 ausführlich beschrieben wird.

---

[233] Erich Gamma, Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software, Addison-Wesley-Longman, Bonn, 1996, Seite 215.

[234] In diesem Fall bezieht sich der Begriff "Plattform" auch auf die verwendete Programmiersprache als technische Basis für die einzelnen Applikationen. Zur Mehrdeutigkeit der Begriffe "Plattform" bzw. "plattformunabhängig" in unterschiedlichen Kontexten im Sprachgebrauch der Informatik vergleiche die diesbezüglichen Anmerkungen in Kapitel 2.6.

Fügt man noch eine gemeinsame Benutzeroberfläche hinzu, in die die unterschiedlichen Anwendungen integriert werden können, erhält man das folgende Schema für die Architektur der E-Learning-Plattform:

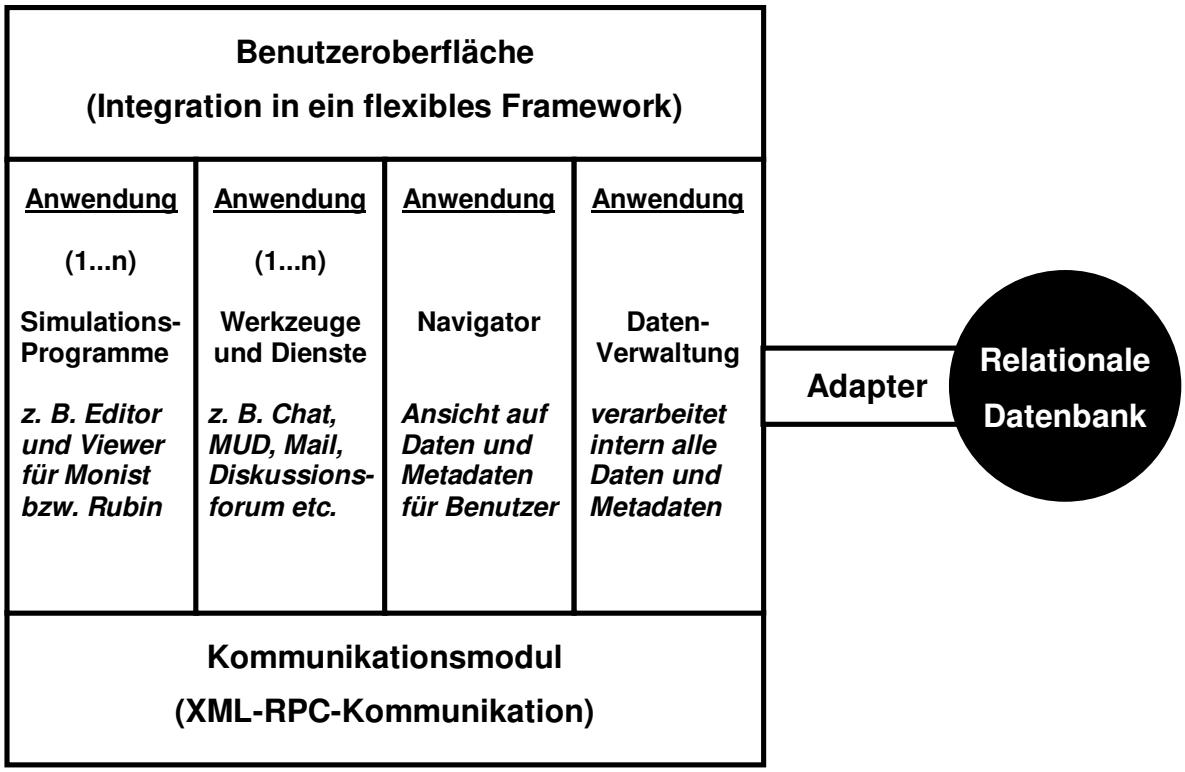

Abbildung 4.4: Architekturschema der E-Learning-Plattform

Die Teilapplikation für die Datenverwaltung, die im nachfolgenden Kapitel 5 dargestellt wird, stellt in diesem Schema lediglich einen Sonderfall einer Anwendung dar, die sich dadurch auszeichnet, daß sie über einen Adapter mit einer Datenbank verbunden ist. Die von der Datenverwaltungskomponente generierten, abstrakten Operationen auf den in der Datenbank enthaltenen Daten (beispielsweise "speichere dieses Objekt") werden vom Datenbankadapter in konkrete, datenbankspezifische Aufrufe (z. B. SQL-Befehle) übersetzt. Auf diese Weise ist es möglich, die verwendete Datenbank auszutauschen, ohne die geringste Änderung an der Datenverwaltung oder den übrigen Komponenten der E-Learning-Plattform vornehmen zu müssen. Nur ein neuer Adapter für die andere Datenbank muß bereitgestellt werden. Diese Datenbankschnittstelle wird in Kapitel 5.6 ausführlich beschrieben.

Einen weiteren Sonderfall stellt der sogenannte Navigator dar, der für den Benutzer eine Ansicht auf die im System enthaltenen Lernobjekte und die dazugehörigen Metadaten bereitstellt. Seine Benutzeroberfläche ermöglicht die Navigation in den Verzeichnisstrukturen sowie die Verwendung metadatenbezogener Suchfunktionalitäten, die von der Datenverwaltungskomponente zur Verfügung gestellt werden. Auch der Aufruf von bestimmten Simulationsprogrammen in Abhängigkeit vom Typ der vom Benutzer zum Öffnen ausgewählten Lehreinheiten wird vom Navigator geregelt. Insofern spielt der Navigator eine besondere Rolle als eine Kernapplikation der E-Learning-Plattform und dient im Rahmen dieser Arbeit gleichzeitig als Beispiel dafür, wie Applikationen über den gemeinsamen Kommunikationsmechanismus auf die von anderen Anwendungen zur Verfügung gestellten Dienste zugreifen. Die Funktionalitäten des Navigators werden in Kapitel 7 vorgestellt.

Der hier vorgestellte Aufbau der E-Learning-Plattform gilt gleichermaßen für den Client als auch für den Server, wobei sich der Server nur dadurch vom Client unterscheidet, daß er normalerweise eine größere Menge an bereitgestellten Daten enthält[235], und daß bestimmte Anwendungen dort nicht direkt ausgeführt werden, da üblicherweise kein Benutzer direkt an dem Computer arbeitet, der als Server dient. Die Benutzeroberfläche kann daher am Server ggf. abgeschaltet werden. Dies kann jedoch bereits durch eine entsprechende Konfiguration der Software geschehen, indem beispielsweise bestimmte Konfigurationsdateien für Client und Server unterschiedliche Einträge enthalten.

Darüber hinaus stellt der Server jedoch lediglich eine aus technische Sicht vollkommen gleichberechtigte Komponente in einem Netzwerk von verschiedenen Installationen der gleichen Software dar. Dieses gesamte Gefüge von Instanzen der E-Learning-Plattform kann daher im Grunde genommen auch als ein Peer-to-Peer-System betrachtet werden. Abbildung 4.5 verdeutlicht die dezentrale Kommunikation zwischen unterschiedlichen Clients sowie die in gleicher Weise hergestellte Verbindung der Clients mit dem Server.

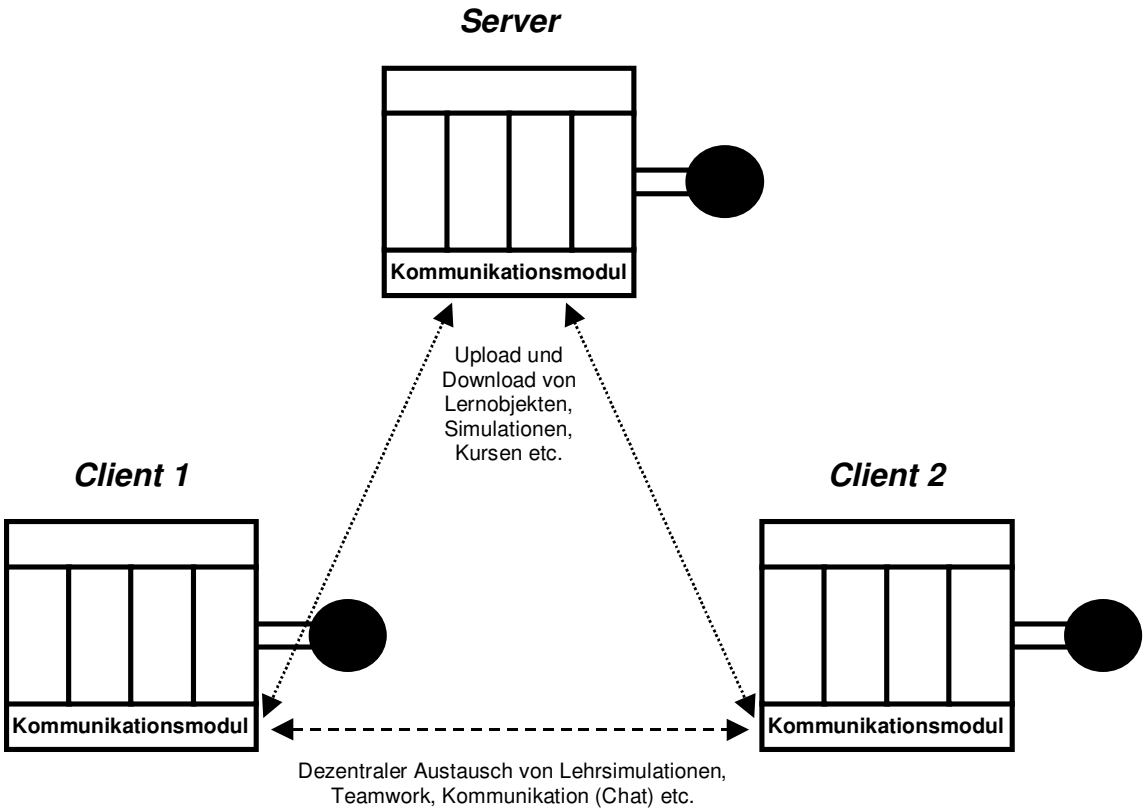

Abbildung 4.5: Kommunikation zwischen Client und Server und zwischen den Clients

Im Folgenden werden die Datenspeicherung (Kapitel 5), der Datenübertragungsmechanismus (Kapitel 6) und der Navigator (Kapitel 7) näher beschrieben. Kapitel 8 zeigt, wie Simulationsprogramme in die E-Learning-Plattform eingebunden werden, bevor in Kapitel 9 auf zusätzliche Funktionalitäten z. B. für die Kommunikation eingegangen wird.

[235] Dies äußert sich im Allgemeinen in einer größeren Datenspeicherungskapazität, die zumeist noch mit besonderen Mechanismen zur Datensicherung verbunden ist, da der Server als zentrale Anlaufstelle für die Benutzer auch die dauerhafte Verfügbarkeit der Daten sicherstellen soll. Der im Projekt Monist verwendete Server ist beispielsweise mit einem RAID-Festplattensystem ausgerüstet, um eine verbesserte Ausfallsicherheit herzustellen. Außerdem werden täglich automatische Datensicherungen über ein Bandlaufwerk vorgenommen. Diese besonderen Maßnahmen betreffen jedoch ausschließlich die Hardware und die Betriebssystemumgebung, in der die E-Learning-Plattform auf dem Server eingesetzt wird. Für die Software spielen diese Unterschiede jedoch keinerlei Rolle.

5 Speicherung von Daten und Metadaten

In diesem Kapitel wird dargestellt, auf welche Weise die in der E-Learning-Plattform verwendeten Inhalte bearbeitet und gespeichert werden. Das Spektrum der vom System zu verwaltenden Daten reicht dabei von einfachen Mediendateien (Texte, Grafiken, Audio, Video etc.) über komplette, unter Verwendung dieser Medien zusammengesetzte Lehreinheiten bis hin zu elektronischen Repräsentationen der mit dem E-Learning-System arbeitenden Benutzer. Alle diese Daten werden im E-Learning-Bereich unter dem Oberbegriff der sogenannten Lernobjekte zusammengefaßt und müssen von der E-Learning-Plattform in geeigneter Weise verwaltet werden können.

Den Kern der folgenden Betrachtungen bildet der Entwurf der sogenannten Virtuellen Lernobjekte (VLO), die zur internen Repräsentation von im System abgelegten (Lern-) Objekten dienen. Ein VLO beschreibt dabei entweder ein atomares Objekt oder ein aus anderen Objekten zusammengesetztes Objekt und enthält unter anderem die zu dessen Verwaltung verwendeten Metadaten. Da die Metadaten hierbei eine entscheidende Rolle spielen, soll an dieser Stelle zunächst auf den Stellenwert eingegangen werden, den sie im Rahmen von elektronischen Datenspeicherungssystemen im Allgemeinen und in der E-Learning-Plattform im Besonderen besitzen.

5.1 Verwendung von Metadaten zur Verwaltung von Lernobjekten

Die Gewährleistung der Wiederauffindbarkeit von Daten in großen Datenbeständen ist ein Problem, das nicht erst durch die Einführung von Computern oder mit dem Wachstum des Internet entstanden ist. Bereits in traditionellen Bibliotheken ergab sich die Notwendigkeit, die vorhandenen Bücher zu verwalten, um die Benutzbarkeit ihrer Inhalte sicherzustellen. Zu diesem Zweck wurden üblicherweise Kataloge angelegt, die beschreibende Informationen über die einzelnen Bücher enthielten und so ihre Wiederauffindbarkeit nach bestimmten Kriterien (z. B. Autor, Verlag, Titel) sicherstellten.

In heutigen elektronischen Datenbeständen besteht im Gegensatz zu einer Bibliothek oft die Möglichkeit zu einer automatischen Volltextsuche. Diese hat jedoch mehrere Nachteile: Einerseits beansprucht sie mit zunehmender Datenmenge immer größere Ressourcen. Andererseits sind manche Informationen über Daten, die zu ihrer Verwaltung oder Verarbeitung benötigt werden (z. B. der Name des Autors), nicht unbedingt immer im Inhalt einer Datei enthalten oder können zumindest nur selten nach einheitlichen Verarbeitungsmechanismen daraus extrahiert werden. Darüber hinaus ist es bei manchen Typen von Dateien (z. B. Grafik-, Video- und Audio-Dateien) schlichtweg sinnlos, Dateien anhand des Auftretens bestimmter Zeichenketten im Inhalt zu klassifizieren, da die Beziehung zwischen der reinen Abfolge der Bits und dem tatsächlich visualisierbaren Inhalt für einen menschlichen Benutzer nur in den seltensten Fällen intuitiv nachvollziehbar ist.

Abhilfe schafft die Verwendung zusätzlicher beschreibender Informationen über die zu verwaltenden Daten, die in der Regel bei der Einspeicherung der Daten in ein elektronisches System automatisch oder durch einen menschlichen Benutzer erzeugt werden. Derartige "Daten über Daten" werden im allgemeinen als Metadaten bezeichnet. Diese werden bei der Katalogisierung, Sortierung und Ablage von Informationen sowie bei der Wiederauffindung gespeicherter Daten verwendet. Darüber hinaus werden durch ihre Verwendung zusätzliche Möglichkeiten geschaffen, bei einer elektronischen

Verarbeitung von Dateien automatische Entscheidungen für die Durchführung unterschiedlicher Aktionen anhand von Merkmalen zu treffen, die über den reinen Inhalt der Dateien hinausgehen. Typische Metadaten in elektronischen Systemen sind z. B. der Titel, der Name des Autors, der Zeitpunkt der Erstellung, beim Einspeichern vom Autor angegebene Stichworte oder der Dateityp.

Zur Verwaltung von Lernobjekten in der E-Learning-Plattform werden ebenfalls Metadaten benötigt, die eine geordnete Speicherung, das Wiederauffinden und die richtige Verwendung der Lernobjekte sicherstellen. Hierbei ist zwischen zwei verschiedenen Arten von Metadaten zu unterscheiden: Allgemeine Metadaten, die unabhängig vom fachlichen Inhalt der jeweiligen Objekte sind (z. B. Datum der Erstellung, Dateityp, Name des Autors), sowie inhaltsbezogene Metadaten (z. B. Titel, Schlüsselwörter, Themengebiet).

Anwendungsprogramme, die für unterschiedliche Zwecke benutzt werden, stellen oft sehr unterschiedliche Anforderungen an die Verwaltung der von ihnen zu bearbeitenden Daten und damit auch an die dafür benötigten Metadaten. Insbesondere inhaltsbezogene Metadaten können von Programm zu Programm stark variieren. Daher gibt es bisher keinen allgemein gültigen, standardisierten Satz von Metadaten, der für jede Software anwendbar wäre und somit als Vorgabe für die in der E-Learning-Plattform zu verwendenden Metadaten dienen könnte. Es gibt jedoch mehrere Initiativen zur Standardisierung von Metadaten, deren Vorschläge bei der Spezifikation der E-Learning-Plattform als Orientierungshilfe benutzt wurden.

Im Folgenden werden zwei Metadaten-Standards vorgestellt, die von unterschiedlichen Autorengruppen vorgeschlagen wurden und über das Internet eine weite Verbreitung gefunden haben. Es handelt sich dabei zum einen um den "Dublin Core" Metadatensatz, der als allgemeine Grundlage für Metadaten gedacht ist, und zum anderen um den speziell für Lernobjekte geschaffenen, allerdings vergleichsweise komplexen Standard "Learning Object Metadata" (LOM). Die wichtigsten Elemente dieser beiden vorgeschlagenen Standards werden kurz beschrieben und auf ihre Anwendbarkeit im Rahmen der E-Learning-Plattform hin untersucht.

Dabei wird besonders berücksichtigt, daß den Benutzern der E-Learning-Plattform die Eingabe der benötigten Metadaten so leicht wie möglich gemacht werden soll. Daher darf weder die Eingabe einer zu großen Anzahl an unterschiedlichen Informationen verlangt werden, noch dürfen diese den Benutzer durch ihre Komplexität oder eventuelle Mehrdeutigkeiten verwirren. Darüber hinaus sollte jede Möglichkeit einer automatischen Generierung von Metadaten genutzt werden. Beispielsweise kann bei einem bereits beim System angemeldeten Benutzer sein Name automatisch ermittelt werden, und der Zeitpunkt der Erstellung eines Lernobjektes kann aus der aktuellen Systemzeit abgeleitet werden.

5.1.1 Metadaten nach dem "Dublin Core" Standard

Im März 1995 veranstalteten das Online Computer Library Center (OCLC)[236] und das National Center for Supercomputing Applications (NCSA)[237] in Dublin (Ohio) einen Metadata Workshop, auf dem ein Kernsatz ("core") von 13 Elementen zur Beschreibung

[236] Homepage: http:// www.oclc.org.
[237] Homepage: http://www.ncsa.uiuc.edu.

von digitalen Dokumenten definiert wurde. Dieser sogenannte "Dublin Core" wurde später auf 15 Elemente erweitert, die als Metadaten-Standard für die Beschreibung von Objekten im Internet vorgeschlagen wurden. Zur weiteren Betreuung wurde die Dublin Core Metadata Initiative (DCMI) gegründet.[238]

Der Metadatensatz des Dublin Core ist nicht als allumfassende Vorschrift zu verstehen, die jede mögliche Art von Metadaten abdecken soll, sondern als gemeinsame Basis zur Beschreibung von elektronischen Dokumenten aus unterschiedlichen Anwendungsgebieten. Sämtliche 15 Elemente sind fakultativ. Außerdem steht es Entwicklern frei, zusätzlich eigene Metadaten zu verwenden, die von den eigenen Anwendungen benötigt werden.

Die folgende Aufstellung der 15 Elemente des "Dublin Core" folgt der offiziellen deutschen Übersetzung von Diann Rusch-Feja auf dem Stand vom 18. 12. 1996.[239] Der Beschreibung dieser Elemente wurde jeweils eine Betrachtung darüber hinzugefügt, inwiefern sie für den Einsatz im Rahmen der E-Learning-Plattform relevant sind. Dabei dient das Projekt Monist, das hier als Referenzprojekt anzusehen ist, als Beispiel für die praktische Verwendung der jeweiligen Metadaten.

5.1.1.1 Titel

Dublin Core: DC.TITLE

Der Titel oder Name des Objektes wird in der Regel vom Verfasser festgelegt. Es handelt sich dabei um eine wichtige Eigenschaft des Objektes, die in der E-Learning-Plattform unter anderem dazu verwendet werden kann, um es bei Kenntnis des Titels durch eine gezielte Suchanfrage direkt aufzufinden. Der Titel ist ein wichtiges und weit verbreitetes Metadaten-Element und wird daher auch im Rahmen der Monist-Projektes berücksichtigt. Von Bedeutung ist dabei die Feststellung, daß es nicht ausgeschlossen ist, daß mehrere verschiedene Objekte eventuell den gleichen Titel haben können. Zur eindeutigen Identifikation von Objekten kann der Titel daher nicht verwendet werden. Diesem Zweck dient statt dessen ein eindeutiger Identifikationsschlüssel (Identifier).[240]

5.1.1.2 Autor

Dublin Core: DC.CREATOR

Die Angabe des Autors wird benötigt, um ein Objekt seinem Ersteller zuordnen zu können.[241] Dies ist insbesondere dann von Bedeutung, wenn es Rückfragen zu einem betrachteten Objekt gibt, eine Überarbeitung erforderlich ist oder die Urheberschaft rechtliche Bedeutung hat. Im Rahmen der E-Learning-Plattform sind Informationen darüber, wer welche Lernobjekte erstellt hat, insbesondere für die kooperative Erstellung von Kursen und Lehrsimulationen von Bedeutung. Daher muß der verwendete Metadatensatz notwendigerweise Angaben über den Autor enthalten.

---

[238] Homepage: http://www.dublincore.org. Die jeweils aktuellste Version des Standards ist stets unter der Adresse http://www.dublincore.org/documents/dces zu finden.
[239] Die Übersetzung ist im Internet erhältlich unter: http://www.mpib-berlin.mpg.de/DOK/metatagd.htm.
[240] Siehe hierzu Abschnitt 5.1.1.10: Ressourcen-Identifikation.
[241] An dieser Stelle ist zu beachten, daß der Ersteller eines elektronischen Dokumentes nicht notwendigerweise mit dem Autor identisch sein muß, da jemand anderes die Datei in seinem Auftrag in das System einstellen kann. Im Monist-System werden aus diesem Grunde AUTHOR und CREATOR unterschieden.

Die gängigste Variante in bestehenden Softwaresystemen ist es, den Benutzernamen des Autors mit dem elektronischen Dokument zu speichern. Benutzernamen müssen jedoch nicht in jedem Fall den realen Namen der betreffenden Person wiederspiegeln. Selbst in Systemen, in denen die Benutzer ihren Benutzernamen nicht frei wählen können, sondern diese von einer zentralen Stelle vergeben werden, werden Benutzernamen in vielen Fällen unter Verwendung von Initialen oder abgekürzten Namensbestandteilen gebildet. Die Angabe des vollen realen Namens als Benutzername ist oftmals unhandlich. Zudem können Mehrdeutigkeiten in diesem Fall nicht ausgeschlossen werden, weil es eventuell zwei Benutzer mit dem gleichen Namen geben kann.

In einem verteilten System wie der E-Learning-Plattform, das zum Teil auch offline betrieben werden kann, ist es jedoch nicht unbedingt immer garantiert, daß zu jedem Zeitpunkt die Zuordnung des Benutzernamens zum realen Namen des Benutzers bekannt ist. Beispielsweise könnten die auf dem Server vorhandenen Informationen über einen Benutzer einem zeitweise isolierten Client noch nicht mitgeteilt worden sein, oder der Benutzer wurde vom Server abgemeldet, bevor die Informationen zum Client übertragen wurden.[242]

Die mögliche Alternative, statt des Benutzernamens nur den Realnamen des Autors mit dem Objekt zu speichern, könnte wiederum zu Problemen führen, wenn dieser Name nicht eindeutig ist (also zwei Personen mit gleichem Vor- und Nachnamen existieren). Außerdem kann von der Eigenschaft einer Person, Autor eines bestimmten Objektes zu sein, unter Umständen das Recht zur Bearbeitung dieses Objektes abhängen, wofür wiederum der Benutzername ausschlaggebend ist, so daß dieser ebenfalls festgehalten werden muß.

Daher kann es sinnvoll sein, sowohl den Realnamen als auch den Benutzernamen des Autors mit dem Objekt zu speichern. Zum Zeitpunkt der Erstellung des Objektes ist beides bekannt, sofern sichergestellt ist, daß nur im System eingeloggte Benutzer Objekte erstellen können. Sowohl der Benutzername als auch der Realname können zu diesem Zeitpunkt automatisch ohne Eingriff des Benutzers erzeugt werden. (In diesem Fall wird diejenige Person als Autor betrachtet, die ein Objekt in das elektronische System einbringt. Diese muß allerdings nicht unbedingt der ursprüngliche Autor sein. Ein Weg, diese Möglichkeit zu berücksichtigen, ist die Schaffung von Stellvertreter-Regelungen, auf die hier nicht weiter eingegangen werden soll.)

Darüber hinaus bietet es sich an, zwei getrennte Datenfelder für den Vornamen und den Nachnamen des Autors vorzusehen. Diese Maßnahme bietet den Vorteil, daß die beiden Bestandteile des Namens getrennt verarbeitet werden können. Beispielsweise kann eine alphabetische Sortierung nach dem Nachnamen vorgenommen werden, ohne dafür den Namen in einem Datenformat abspeichern zu müssen, das der natürlichen menschlichen Sprechweise widerspricht (z. B. mit dem Vornamen hinter dem Nachnamen, durch ein Komma getrennt).

[242] Im Monist-System wird die Möglichkeit der Zuordnung dadurch erzwungen, daß beim Aufbau einer Verbindung zwischen zwei Instanzen der E-Learning-Plattform auf verschiedenen Computern zuerst die auf den beiden Systemen vorhandenen Benutzerinformationen aktualisiert werden. Diese Vorgehensweise ist allerdings relativ rigide und möglicherweise nicht für jeden möglichen zukünftigen Einsatzbereich der E-Learning-Plattform praktikabel. Der Fall, daß als Autoren eingetragene Benutzer im System nicht mehr existieren, wird im Monist-System dadurch ausgeschlossen, daß Benutzer zwar deaktiviert, aber nicht gelöscht werden können. Dies bedarf in anderen Systeme eventuell ebenfalls einer anderen Lösung.

5.1.1.3 Thema und Stichwörter

Dublin Core: DC.SUBJECT

Dieses Element dient der Speicherungen von Informationen, die das Thema oder den Inhalt des Objektes beschreiben. Dies kann zum Beispiel in Form von Stichwörtern geschehen. Bei der Entwicklung des "Dublin Core" Standards ging man von der Idee aus, daß für bestimmte Anwendungsbereiche jeweils ein spezifisches Vokabular geschaffen werden könne, mit dessen Hilfe die thematische Beschreibung formalisiert werden könnte.

Derartige Bestrebungen waren allerdings bisher nicht erfolgreich, da es sich als sehr schwierig herausgestellt hat, für einen Fachbereich ein allgemein akzeptiertes Vokabular zu vereinbaren. Dennoch gibt es einige vorgeschlagene Standards für bestimmte Bereiche, die jedoch in der Regel von einzelnen Institutionen stammen. Einige Beispiele dafür sind die "Library of Congress Subject Headings" (LCSH) und die "Library of Congress Classification" (LCC) der U. S. Library of Congress[243] sowie die "Medical Subject Headings" (MESH) der National Library of Medicine[244] am U. S. National Institute of Health[245].

Für den Bereich der Lernsoftware ebenso wie speziell für die im Rahmen des Projektes Monist behandelten Fachgebiete existiert noch kein derartiger Standard. Daher wird in diesem Fall statt dessen eine Beschreibung des Themas durch Stichworte (keywords) vorgesehen, die der Benutzer beim Einbringen der Objekte in das System angibt. Zur thematischen Klassifizierung dient darüber hinaus die Einsortierung in verschiedene Fachbereiche, die durch (virtuelle) Verzeichnisse repräsentiert werden. Dabei kann sich ein Objekt gleichzeitig in mehreren Verzeichnissen befinden, wenn es mehreren Fachbereichen angehört.[246] Weitergehende Informationen über die Inhalte von Objekten können innerhalb des Metadatensatzes außerdem in der textuellen Beschreibung der Objekte abgelegt werden. Siehe hierzu den nachfolgenden Abschnitt 5.1.1.4: Inhaltliche Beschreibung.

5.1.1.4 Inhaltliche Beschreibung

Dublin Core: DC.DESCRIPTION

Die inhaltliche Beschreibung bietet die Möglichkeit, weitergehende Informationen über ein Objekt abzulegen, die über Thema und Stichwörter hinausgehen. Dieses Element beinhaltet in der E-Learning-Plattform Freitext und kann z. B. bei Textobjekten dazu genutzt werden, eine kurze Zusammenfassung (abstract) des Inhaltes anzugeben. Anstelle einer Volltextsuche über den gesamten Inhalt kann eine Suche über die inhaltliche Beschreibung ausgeführt werden, die im der Regel weniger aufwendig und damit schneller ist sowie bei sorgfältiger Verwendung dieses Metadaten-Elements auch zu besseren Ergebnissen führt. Aus praktischen Erwägungen sollte die Länge der Beschreibung möglichst kurz gehalten werden, weil ihre Verwendung bei zunehmender Länge immer weniger Vorteile gegenüber einer Volltextsuche bieten würde.

---

[243] Homepage: http://www.loc.gov.
[244] Homepage: http://www.nlm.nih.gov.
[245] Homepage: http://www.nih.gov.
[246] Zu den in der E-Learning-Plattform verwendeten virtuellen Verzeichnissen siehe Kapitel 5.4.

5.1.1.5 Verleger bzw. Herausgeber

Dublin Core: DC.PUBLISHER

Dieses Element ist in erster Linie für Bibliotheken und für die Verwaltung elektronischer Publikationen von Bedeutung. Da es im Rahmen des Projektes Monist keinen Verleger im eigentlichen Sinne gibt, ist diese Angabe hier nur von geringer Bedeutung. Aus Gründen der Konformität mit dem "Dublin Core" Standard ist es jedoch ratsam, zumindest die Möglichkeit zur Verwaltung dieses Metadaten-Elements vorzusehen und für alle Objekte sinnvolle Werte einzutragen. Mögliche Erweiterungen des Systems könnten das "Publisher"-Element nutzen. Natürlich ist anderen Anwendungen, die auf Basis der E-Learning-Plattform geschaffen werden, die Benutzung dieses Elementes bei Bedarf ebenfalls freigestellt.

Im Rahmen des Projektes Monist spielt in gewisser Weise das Monist-System selbst die Rolle eines Verlegers, weil die eingestellten Objekte quasi durch das System veröffentlicht werden. Hinzugefügt werden könnten Informationen darüber, welcher beteiligten Institution[247] der Autor des Objektes angehört. Diese Angaben können vom System beim Generieren von Objekten bei Bedarf eventuell automatisch erzeugt werden, sofern sichergestellt ist, daß nur im System eingeloggte Benutzer Objekte erstellen können und daß die benötigten Informationen in den Benutzerdaten gespeichert sind. Eine direkte Einflußnahme des Benutzers auf das "Publisher"-Element ist vorerst nicht vorgesehen.

5.1.1.6 Weitere beteiligte Personen und Körperschaften

Dublin Core: DC.CONTRIBUTORS

Dieses Metadaten-Element dient zur Nennung von Personen, die an der Erstellung eines Objektes mitgewirkt haben. Im Gegensatz zum Hauptbeteiligten, der als Autor genannt wird (siehe Abschnitt 5.1.1.2: Autor), ist der Anteil dieser weiteren Personen an der Erstellung jedoch im allgemeinen wesentlich geringer einzuschätzen. Bei gemeinschaftlichen Arbeiten könnte unter Umständen auch der Hauptverantwortliche als Autor angegeben werden, während die übrigen Beteiligten als "Contributors" genannt werden.

Im Rahmen der E-Learning-Plattform ist es durchaus möglich, daß an der Erstellung von Objekten mehrere Personen beteiligt sind. Im Projekt Monist wird im Regelfall derjenige, der ein Objekt in das System einbringt, als Autor des Objektes eingetragen. Die Angabe weiterer beteiligter Personen wird prinzipiell ermöglicht, wobei dieses Element als Voreinstellung leer ist und vom Benutzer beim Erstellen von Objekten nicht zwingend ausgefüllt werden muß.

5.1.1.7 Datum

Dublin Core: DC.DATE

Für das Datums-Element gibt es mehrere mögliche Interpretationen, die im Rahmen des "Dublin Core" Standards durch sogenannte "Qualifier" näher bezeichnet werden können. Die gängigsten Varianten sind das Erstellungsdatum ("Created") und das Datum

[247] Zu den am Projekt Monist beteiligten Lehrstühlen siehe die diesbezüglichen Anmerkungen zu Beginn von Kapitel 3.1.

der letzten Änderung ("Modified"). Da im Rahmen der E-Learning-Plattform keine Versionskontrolle von Lernobjekten beabsichtigt wird[248], genügt die Angabe eines einzigen Datums-Elementes. Dieses wird bei der Erstellung eines Objektes mit dessen Erstellungsdatum vorbelegt und bei jeder Änderung des Objektes auf das Änderungsdatum gesetzt. Diese Aktionen können automatisch ohne Eingriff des Benutzers erfolgen.

Das Format für die Datums-Angabe entspricht dem als ISO 8601 festgelegten Standard. Dabei wird die Variante mit komplettem Datum und Uhrzeitangabe mit Stunden, Minuten und Sekunden (aber ohne Sekundenbruchteile) gewählt. Die Angabe des Datums, die auch im Projekt Monist verwendet wird, folgt also dem Format:

`YYYY-MM-DDThh:mm:ssTZD`

Die einzelnen Bestandteile dieses Datumsformates haben dabei folgende Bedeutungen:

`YYYY` : Jahresangabe (vierstellig)
`MM` : Monatsangabe (zweistellig, `01-12`)
`DD` : Tagesangabe (zweistellig, `01-31`)
`T` : Der Buchstabe `T` dient nach ISO 8601 als Begrenzer zwischen Datum und Uhrzeit
`hh` : Stundenangabe (zweistellig, `00-23`)
`mm` : Minutenangabe (zweistellig, `00-59`)
`ss` : Sekundenangabe (zweistellig, `00-59`)
`TZD` : Zeitzonenangabe relativ zu UTC (`+hh:mm` oder `-hh:mm`)

5.1.1.8 Ressourcenart

Dublin Core: DC.TYPE

Dieses Element bezeichnet die Art des Objektes. Nach dem "Dublin Core" Standard ist dabei eine Auswahl aus einer fest vorgegebenen Liste von Objekttypen vorgesehen. Im Rahmen des Projektes Monist enthält diese Auswahlliste Bezeichnungen für unterschiedliche Typen von Lernobjekten (z. B. Lehrsimulation, Kurs, Bild, Glossareintrag). Die Auswahl des Objekttyps muß vom Benutzer beim Einbringen des Objektes in das Monist-System vorgenommen werden.

5.1.1.9 Format

Dublin Core: DC.FORMAT

Im Gegensatz zum Element "Type" enthält das Element "Format" Informationen über das datentechnische Format des Objektes. Üblicherweise werden diese der Liste der angemeldeten Internet Media Types (MIME Types) entnommen. Die E-Learning-Plattform folgt dieser Praxis. Der MIME Type wird - soweit möglich - beim Einbringen des Objektes in das System automatisch ermittelt. Eine Beteiligung des Benutzers ist nur dann notwendig, wenn der MIME Type nicht direkt aus dem Namen bzw. der Namensendung der eingebrachten Datei ermittelt werden kann.

[248] Diese Entscheidung beruht auf der folgenden Überlegung: Handelt es sich bei einer Änderung einer Lehreinheit um eine Korrektur, sollte der vorherige, fehlerhafte Zustand sowieso nicht mehr zugänglich sein. Wird jedoch eine Variante erstellt, kann dafür auch eine Kopie des Originals angelegt werden.

5.1.1.10 Ressourcen-Identifikation

Dublin Core: DC.IDENTIFIER

Dieses Element soll laut Spezifikation eine Zeichenkette oder eine Zahl enthalten, die das Objekt eindeutig identifiziert. Diese Rolle übernimmt in der E-Learning-Plattform ein systemweit eindeutiger Objekt-Identifier. Dieser wird in Kapitel 5.2 in einer gesonderten Spezifikation definiert.

5.1.1.11 Quelle

Dublin Core: DC.SOURCE

Als Quelle eines Objektes kann nach dem "Dublin Core" Standard ein elektronisches oder gedrucktes Dokument angegeben werden, aus dem dieses Objekt ursprünglich stammt. Für die E-Learning-Plattform ist diese Angabe nur dann von Bedeutung, wenn bei der Erstellung von Lernobjekten von den jeweiligen Autoren externe Medien herangezogen werden. In solchen Fällen kann die betreffende Quelle in dieses Metadaten-Element eingetragen werden, es muß jedoch nicht zwingend beim Erstellen jedes Objektes vom Benutzer ausgefüllt werden. Im Monist-System wird als Voreinstellung für das "Source"-Element ohne Eingriff des Benutzers automatisch "Monist" eingetragen.

5.1.1.12 Sprache

Dublin Core: DC.LANGUAGE

Dieses Element soll nach dem "Dublin Core" Standard eine Zeichenkette enthalten, die die Sprache des Objektes spezifiziert. Die Bezeichnung der Sprachen soll dabei dem Internet-Standard zur Identifikation von Sprachen folgen, der in RFC 3066 definiert wurde.[249] Die nach diesem Standard offiziell definierten Sprachbezeichnungen werden von der Internet Assigned Numbers Authority (IANA)[250] verwaltet.

Die Sprachangabe des "Dublin Core" Standards ist darauf ausgelegt, daß ein bestimmtes Objekt immer in genau einer Sprache vorliegt, so daß eine eindeutige Sprachangabe möglich ist. Dies ist jedoch ungünstig, wenn in einem mehrsprachigen System Objekte über Inhalte in mehreren verschiedenen Sprachen verfügen. Eine eindeutige Sprachangabe ist in diesem Fall nicht möglich.

Ein mehrsprachiges System, wie es im Fall der E-Learning-Plattform geschaffen wurde, kann prinzipiell auf zwei unterschiedliche Weisen konstruiert werden. Eine Möglichkeit ist, daß jedes Objekt nur Inhalte und Metadaten in einer einzigen Sprache enthält. In diesem Fall muß man für jede neue Sprache ein neues Objekt erzeugen, dessen Sprache dann eindeutig angegeben werden kann. Die gleichen Inhalte in unterschiedlichen Sprachen liegen in diesem Fall in getrennten Objekten vor. Ein aus mehreren Objekten zusammengesetztes Objekt (wie z. B. eine Lehrsimulation) enthält jeweils nur diejenigen Objekte, die in der gewünschten Sprache vorliegen. Demzufolge müssen auch zusammengesetzte Objekte für jede Sprache neu erzeugt werden.

---

[249] Dieser "Internet Best Current Practices" Quasi-Standard kann eingesehen werden unter der Adresse: http://www.ietf.org/rfc/rfc3066.txt.
[250] Homepage: http://www.iana.org.

Der Nachteil einer solchen Lösung ist, daß alle Objekte mehrfach erzeugt werden müssen, selbst wenn sie sich in ihren Inhalten und in den Metadaten kaum unterscheiden. Zudem kann das System im Falle des Fehlens eines Objektes in einer bestimmten Sprache nur schwer ersatzweise auf eine Version des gleichen Objektes in einer anderen Sprache zurückgreifen, da zwischen diesen Objekten keine direkte Beziehung besteht bzw. solche Beziehungen gegebenenfalls aufwendig verwaltet werden müßten.

Diese Probleme treten nicht auf, wenn man statt dessen die Speicherung von mehrsprachigen Objekten vorsieht. Der Inhalt und die Metadaten werden in diesem Fall in mehreren Sprachen innerhalb des selben Objektes gespeichert. An die Stelle einfacher Metadaten-Felder treten in diesem Fall Tabellen, die für jede verwendete Sprache die entsprechenden Werte enthalten. Die Inhalte der Objekte werden ebenfalls auf diese Weise verwaltet. Bei der Verarbeitung derartiger mehrsprachiger Objekte können in Abhängigkeit von einer vom Benutzer eingestellten Sprache die Inhalte und Metadaten in dieser Sprache verwendet werden. Beim Fehlen bestimmter Daten in einer Sprache können ersatzweise die Inhalte verwendet werden, die für eine andere Sprache angegeben wurden, ohne auf andere Objekte zurückgreifen zu müssen.

Mehrsprachige Objekte verursachen durch die Speicherung von Metadaten und Inhalten in Tabellen einen höheren Verwaltungsaufwand gegenüber einsprachigen Objekten. Dies wird jedoch dadurch aufgewogen, daß insgesamt weniger Objekte verwaltet werden müssen und alle logisch zusammengehörigen Inhalte und Metadaten zusammen gespeichert sind und so bei einem Wechsel der Sprache nicht aus anderen Objekten ermittelt werden müssen. Daher wurde für die E-Learning-Plattform die Speicherung von mehrsprachigen Objekten vorgesehen.[251] Dies führt zu keiner Inkompatibilität mit dem "Dublin Core" Standard, da abhängig von der aktuell eingestellten Sprache immer ein eindeutiger Satz von Metadaten für diese Sprache ausgegeben werden kann und damit auch das Element "Language" in jedem Fall definiert ist.

5.1.1.13 Beziehung zu anderen Ressourcen

Dublin Core: DC.RELATION

Dieses Element dient der Darstellung von Beziehungen verschiedener Objekte untereinander. Im Rahmen des "Dublin Core" Standards sind zwölf verschiedene Arten solcher Beziehungen definiert, die durch sogenannte "Qualifier" näher bezeichnet werden können. Diese lassen sich grob in drei Gruppen einteilen:

a) Versionierung: Unter diesen Oberbegriff fallen neben den Qualifiern für die eigentliche Versionierung ("Is Version Of", "Has Version") auch die Qualifier für die Ersetzung von bestehenden Objekten durch neue Objekte ("Is Replaced By", "Replaces") sowie für die Darstellung gleicher Objektinhalte in einem neuen Format ("Is Format Of", "Has Format"). Da im Rahmen der E-Learning-Plattform keine Versionskontrolle von Lernobjekten vorgesehen ist, wird auf die Verwaltung dieser Typen von Beziehungen verzichtet.

b) Abhängigkeiten: Zu dieser Gruppe zählen Qualifier, die Beziehungen zwischen Teilen und den aus ihnen zusammengesetzten Objekten bezeichnen ("Is Part Of", "Has Part"), sowie Qualifier für sonstige Beziehungen von Objekten untereinander, bei denen

[251] Siehe hierzu nachfolgend Kapitel 5.3.3 (für die Speicherung mehrsprachiger Metadaten) und 5.8 (für die Speicherung mehrsprachiger Inhalte).

ein Objekt von der Existenz des jeweils anderen Objektes abhängig ist ("Is Required By", "Requires"). Derartige Beziehungen werden im Monist-System durch (virtuelle) Verzeichnisse repräsentiert. Die für ein zusammengesetztes Lernobjekt (z. B. einen Kurs oder eine Lehrsimulation) benötigten einzelnen Objekte werden in einem Verzeichnis abgelegt, welches das übergeordnete Objekt repräsentiert. Dabei kann sich ein Objekt gleichzeitig in mehreren Verzeichnissen befinden, wenn es beispielsweise für mehrere verschiedene Lehrsimulationen benötigt wird. [252]

c) Referenzen: Bei dieser Gruppe handelt es sich um Qualifier, die Verweise zwischen Objekten bezeichnen ("Is Referenced By", "References"). Beispiele dafür sind Quellenangaben für Zitate sowie Hyperlinks. Innerhalb der E-Learning-Plattform sind als Verweise nur Hyperlinks zwischen Objekten von Bedeutung. Ein Beispiel dafür aus dem Monist-System sind Links, die aus Texten innerhalb von Lehrsimulationen ins Glossar verweisen. Links sind jedoch nicht notwendigerweise nur einfache Verweise von einem Objekt auf ein anderes, sondern können zusätzlich verschiedene eigene Eigenschaften haben, wie zum Beispiel Zugriffsrechte oder einen Besitzer. (Der Ersteller eines Links wird dabei automatisch als Autor eingetragen.) Daher können sie als eigenständige Objekte behandelt und gesondert verwaltet werden. Der Startpunkt und das Ziel eines Links sind ebenfalls Eigenschaften dieser Objekte.[253] Die im Monist-System umgesetzte Verwaltung von Links geht über die nach dem "Dublin Core" Standard vorgesehene Speicherung von Referenz-Beziehungen als Metadaten hinaus. Bei Bedarf können die entsprechenden Angaben jedoch leicht aus den gespeicherten Eigenschaften der Links erzeugt werden. Damit ist beispielsweise ein Export dieser Metadaten in einer dem "Dublin Core" Standard entsprechenden Form möglich.

5.1.1.14 Räumliche und zeitliche Maßangaben

Dublin Core: DC.COVERAGE

Dieses Element dient der räumlichen und zeitlichen Eingrenzung eines Objektes. Zur Zeit hat es allerdings noch experimentellen Charakter. Es gibt einige Vorschläge für eine formelle Notierung räumlicher und zeitlicher Angaben. Beispiele dafür sind die von der Dublin Core Metadata Initiative (DCMI) vorgeschlagenen Schemata "DCMI Point" für geographische Angaben sowie "DCMI Period" zur Angabe von zeitlichen Intervallen. Allgemein anerkannte Spezifikationen für Raum- und Zeitangaben existieren derzeit jedoch nicht. Im Projekt Monist sind derartige Metadaten nicht von Bedeutung und werden daher auch nicht verwaltet.

---

[252] Zu den in der E-Learning-Plattform verwendeten virtuellen Verzeichnissen siehe Kapitel 5.4.

[253] Diese Vorstellung von Links als eigenen Objekten, welche im Rahmen des Monist-Systems auf einer in diesem Projekt selbst entwickelten technischen Basis umgesetzt wurde, ist ursprünglich aus den theoretischen Grundlagen des Systems Hyper-G entnommen, das die technische und konzeptionelle Grundlage des Dokumentenmanagementsystems Hyperwave Information Server (Kapitel 3.1.2.4 und 3.3.2.4) bildete. Siehe hierzu die folgenden Quellen (Informationen zum Link-Schema jeweils ab der zweiten Seite):
- Keith Andrews, Frank Kappe, Hermann Maurer, The Hyper-G Network Information System, in: Journal of Universal Computer Science, Vol. 1, Nr. 4 (Special issue: Proceedings of the Workshop on Distributed Multimedia Systems, held in Graz, Austria, Nov. 1994), April 1995, Seiten 206-220.
- Keith Andrews, Frank Kappe, Hermann Maurer, Serving Information to the Web with Hyper-G, in: Computer Networks and ISDN Systems, 27 (6), 1995, Seiten 919-926. Der gleiche Beitrag ist ebenfalls erschienen in: Proceedings of the Third International World Wide Web Conference, Darmstadt, 1995. Eine HTML-Version kann auf den Internet-Seiten der Fraunhofer-Gesellschaft unter der Adresse http://www.igd.fhg.de/archive/1995_www95/proceedings/papers/105/hgw3.html eingesehen werden.

5.1.1.15 Rechtliche Bedingungen

Dublin Core: DC.RIGHTS

Dieses Element dient zur Angabe von Urheberrechten oder anderen für die Benutzung eines Objektes relevanten rechtlichen Bestimmungen. Für die E-Learning-Plattform ist davon auszugehen, daß die Urheberrechte in der Regel entweder bei derjenigen Person liegen, die unter Verwendung der Autorenfunktionalität einer bestimmten, in das System integrierten Simulationssoftware eine Lehrsimulation erstellt, oder zumindest von dieser Person in einem Eingabeformular korrekt angegeben werden können. Im Rahmen des Projektes Monist liegen die Urheberrechte von Objekten in der Regel bei der Monist-Projektgruppe. Es ist jedoch durchaus möglich, daß auch außerhalb des Projektes erstellte Objekte im Monist-System Verwendung finden. Beispiele dafür wären Grafiken oder Videos, die im Rahmen bestimmter Lehrsimulationen benutzt werden. Daher muß das System eine Möglichkeit zur Angabe von Urheberrechten vorsehen.

Beim Einbringen neuer Objekte in das Monist-System wird dem Benutzer die Möglichkeit eingeräumt, eventuell vorhandene fremde Urheberrechte anzugeben. Dieses Datenfeld muß jedoch vom Benutzer nicht zwingend ausgefüllt werden. Falls für ein Objekt keine Angaben zum Urheberrecht gemacht werden, wird als Voreinstellung eingetragen, daß die Rechte für dieses Objekt bei der Monist-Projektgruppe liegen. Für andere in die E-Learning-Plattform zu integrierende Simulationsanwendungen können entsprechende Vorkehrungen getroffen werden.

5.1.2 Metadaten nach dem "Learning Object Metadata" (LOM) Standard

Der LOM-Standard wurde von einer im Jahre 1997 eigens zu diesem Zweck eingerichteten Arbeitsgruppe[254] des Learning Technology Standardization Committee (LTSC)[255] am Institute of Electrical and Electronics Engineers (IEEE)[256] vorgeschlagen. Zum Zeitpunkt des Projektbeginns im Jahre 2001 lag dieser Standard in der Version 6.1 vor, die noch als Entwurf (draft) gekennzeichnet war und noch nicht endgültig vom IEEE als Standard anerkannt wurde. Am 12. Juni 2002 wurde der neue Standard unter dem Titel *1484.12.1 LOM data model standard* offiziell vom IEEE anerkannt.[257]

Ziel der LOM-Initiative war bzw. ist die Schaffung eines Standards für Metadaten zur Beschreibung von Lernobjekten. Dieser Standard umfaßt neben allgemeinen Angaben, wie sie auch im "Dublin Core" Standard definiert sind, eine Vielzahl von Elementen, die als spezifische Metadaten zur Verwaltung von Lernobjekten betrachtet werden. Im Gegensatz zum "Dublin Core", der lediglich als ein erweiterbarer Satz von Basis-Attributen gedacht ist, sollen die Metadaten des LOM-Standards nicht durch zusätzliche Elemente ergänzt werden.

Der Metadatensatz des LOM-Standards umfaßt in seiner endgültigen Version insgesamt 77 Elemente und Unterelemente. Diese sind in neun Gruppen zusammengefaßt, die wiederum Untergruppen enthalten können. Der LOM-Standard bietet somit zwar einen sehr umfassenden Satz von Metadaten für Lernobjekte, ist aber aufgrund der Vielzahl

[254] Homepage der Arbeitsgruppe: http://ltsc.ieee.org/wg12.
[255] Homepage des LTSC: http://ltsc.ieee.org.
[256] Homepage des IEEE: http://www.ieee.org.
[257] Die genehmigte, finale Version des LOM-Standards ist im Internet zum Download erhältlich unter der Adresse: http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf.

seiner Elemente kaum handhabbar. Die Verwaltung vieler Metadaten stellt zwar an sich für die heutigen elektronische Systeme kein gravierendes Problem dar. Man muß jedoch beachten, daß diese Metadaten nicht nur verwaltet, sondern zuvor auch von einem menschlichen Benutzer erzeugt werden müssen. Die manuelle Eingabe von bis zu 77 verschiedenen Metadaten beim Hinzufügen jedes einzelnen neuen Lernobjektes würde jeder Autor zu Recht als unzumutbar betrachten. Die Akzeptanz eines derartigen Systems wäre gleich Null. Daher muß für ein Softwaresystem wie z. B. Monist eine sorgfältige Auswahl der zu verwendenden Metadaten getroffen werden. Gleichwohl muß die E-Learning-Plattform als Basis die Möglichkeiten zur Speicherung beliebiger Metadaten ermöglichen.

Die folgende Aufstellung bietet eine Übersicht über den LOM-Standard, wobei nicht jedes einzelne Element beschrieben wird, sondern lediglich auf die Funktionen der verschiedenen Gruppen von Metadaten eingegangen wird. Der Beschreibung dieser Elemente wurde jeweils eine Betrachtung darüber hinzugefügt, inwiefern sie für den Einsatz im Rahmen des Monist-Projektes, das als Referenzprojekt für die E-Learning-Plattform dient, relevant sind.

5.1.2.1 Allgemeine Metadaten

LOM-Kategorie: General

Diese Kategorie von Metadaten umfaßt allgemeine Informationen über Lernobjekte. Dazu zählen unter anderem einige der aus dem "Dublin Core" Standard bekannten Elemente: Titel (im "Dublin Core": DC.TITLE), Ressourcen-Identifikation (DC.IDENTIFIER), Beschreibung (DC.DESCRIPTION), Sprache (DC.LANGUAGE) sowie räumliche und zeitliche Maßangaben (DC.COVERAGE). Auch das Thema (DC.SUBJECT) wird in Form von Stichwörtern dieser Kategorie zugeordnet. Auf diese Metadaten-Elemente wurde bei der Beschreibung des "Dublin Core" Standards bereits ausführlich eingegangen. Für das Monist-System sind sie mit Ausnahme der räumlichen und zeitlichen Maßangaben ausnahmslos von großer Bedeutung und werden daher für alle Lernobjekte verwaltet.

Die LOM-Kategorie General enthält außerdem Elemente, die Angaben darüber ermöglichen, ob das bezeichnete Lernobjekt bereits in einem bestehenden Katalog-System erfaßt ist, um welchen Katalog es sich dabei handelt und welcher Eintrag in diesem Katalog das Lernobjekt beinhaltet. Ein solcher Katalog-Eintrag kann zum Beispiel die ISBN-Nummer eines Buches oder eine Eintragsnummer in einer Medien-Datenbank sein. Auf die Verwaltung derartiger Informationen wird im Monist-System verzichtet, da Verweise auf externe Kataloge für die Funktionsweise von Monist keinerlei Bedeutung haben und zudem keine größere Anzahl von Lernobjekten im System zu erwarten ist, für die Katalog-Informationen verfügbar sind.

Dazu kommen Metadaten-Elemente für Angaben über die Struktur des Lernobjektes und über seine Aggregationsebene. Für die Struktur wurde ein festgelegtes Vokabular definiert, das es unter anderem ermöglicht, anzugeben, ob ein Lernobjekt atomar ist oder aber aus mehreren Teilobjekten zusammengesetzt ist, und welcher Art diese Zusammensetzung ist (z. B. linear oder hierarchisch). Darüber hinaus sind vier Aggregationsebenen für Lernobjekte definiert, die folgendermaßen beschrieben werden können: Rohdaten (z. B. einzelne Mediendateien), Kollektionen von Rohdaten (z. B. zusammengesetzte Seiten), Netzwerke solcher Kollektionen (z. B. Kurse) sowie ein ganzes Curriculum (entspricht z. B. dem gesamten Monist-System).

Informationen über die Zusammensetzung von Lernobjekten sind in den im Rahmen der E-Learning-Plattform verwendeten Datenstrukturen implizit enthalten, werden jedoch nicht durch Metadaten-Elemente dargestellt. Zusammengesetzte Lernobjekte werden durch (virtuelle) Verzeichnisse repräsentiert.[258] Dabei kann sich ein Objekt gleichzeitig in mehreren Verzeichnissen befinden, wenn es mehreren zusammengesetzten Objekten angehört. Weitergehende Informationen über Struktur und Aggregation werden vom System zur Verwaltung von Lernobjekten nicht benötigt. Die dynamische Zusammensetzung von Objekten zu Lehrsimulationen und Kursen erfolgt im Projekt Monist anhand eines in XML definierten Datenformates. Die Informationen über Aufbau und Ablauf einer bestimmten Lehrsimulation gehören somit zu deren Inhalt und stellen keine Metadaten dar.

5.1.2.2 Bearbeitungsverlauf

LOM-Kategorie: Life_Cycle

Diese Kategorie ermöglicht die Angabe von Informationen zur Bearbeitungsgeschichte eines Lernobjektes. Dazu zählen eine Versionsbezeichnung (als Freitext, um keine Festlegung auf ein bestimmtes System zur Verwaltung von unterschiedlichen Versionen von Objekten zu treffen), ein Bearbeitungsstatus (mit den vier vorgegebenen möglichen Werten: Draft, Final, Revised und Unavailable) sowie Angaben darüber, welche Person oder Einrichtung zu welchem Zeitpunkt und in welcher Rolle (z. B. als Autor, Herausgeber, Verleger, Prüfer etc.) an der Bearbeitung des Objektes beteiligt war. In Abhängigkeit von der jeweiligen Rolle entsprechen diese Angaben den aus dem "Dublin Core" Standard bekannten Elementen Autor (DC.CREATOR), Verleger oder Herausgeber (DC.PUBLISHER) sowie weitere beteiligte Personen und Körperschaften (DC.CONTRIBUTORS). Daneben werden auch Datumsangaben (DC.DATE) durch diese Elemente zum Ausdruck gebracht.

Die Angabe eines Status ist im Monist-System erforderlich, um zu kennzeichnen, ob die enthaltenen Lernobjekte - insbesondere komplette Lehrsimulationen oder Kurse - sich noch in Bearbeitung befinden, fertiggestellt (aber noch nicht freigegeben) sind oder bereits erfolgreich eine Qualitätssicherung nach dem Peer-Review-Verfahren durchlaufen haben und damit für den Einsatz in der Lehre freigegeben wurden. Der Status wird in einem gleichnamigen Metadaten-Element eingetragen, dessen mögliche Inhalte sich an den im LOM-Standard definierten Werten orientieren. Ein festgelegter Workflow bestimmt, durch welche Aktionen ein Lernobjekt eine Änderung seines Status erfährt. Kommentare, die beispielsweise während der Begutachtung zu einem Objekt abgegeben werden, können zusammen mit dem Objekt gespeichert werden, wobei der Name des Autors der Anmerkung festgehalten wird.

Eine darüber hinausgehende Verwaltung von Versionen von Objekten ist im Monist-System nicht erforderlich. Während sich ein Lernobjekt in Bearbeitung befindet, sind Änderungen nur für den Autor von Bedeutung. Es ist für die Benutzung des Systems nicht erforderlich, über jede noch so kleine Änderung in der Bearbeitungsphase Buch zu führen und jeweils eine neue Version des gleichen Objektes abzuspeichern. Nach der Freigabe fertiggestellter Lernobjekte erfolgen allenfalls noch kleinere Fehlerkorrekturen, bei denen es ebenfalls nicht notwendig ist, die ersetzte, fehlerhafte Version im System aufzubewahren. Statt dessen wird sie einfach durch die neue, korrigierte Version überschrieben.

[258] Zu den in der E-Learning-Plattform verwendeten virtuellen Verzeichnissen siehe Kapitel 5.4.

Die für die Benutzung in der Lehre freigegebenen Lehrsimulationen und Kurse stehen der Allgemeinheit zur Verfügung und werden über die E-Learning-Plattform als Kopien auf viele verschiedene Computer verteilt. Die Einführung von einander ersetzenden verschiedenen Versionen der selben Lernobjekte (z. B. Version 2.0 gilt ab einem bestimmten Datum anstelle von Version 1.0) würde in einem solchen verteilten System zu einem unvertretbar hohen Verwaltungsaufwand führen, da ständig geprüft werden müßte, ob auf jedem einzelnen Computer jede Datei in der richtigen Version vorliegt. Dennoch muß es möglich sein, neue Varianten der gleichen Lehrsimulation zu erzeugen, wenn beispielsweise ein bestimmter Kurs nach Ablauf eines Jahres erneut stattfindet, dabei aber neue Erkenntnisse aus der aktuellen Forschung eingebracht werden sollen. Dies kann jedoch ebensogut dadurch erreicht werden, daß die neue Lehrsimulation zunächst als Kopie der alten angelegt und dann verändert wird.

Dieses Vorgehen bietet den Vorteil, daß man jederzeit auf die Version zurückgreifen kann, die zu einem früheren Zeitpunkt gültig war, um festzustellen, auf welche Art und Weise die Lerninhalte zu diesem Zeitpunkt vermittelt wurden. Die neue Version ist dabei von der alten vollkommen unabhängig. Alternativ können veraltete Lernobjekte bei Bedarf auch gelöscht werden, ebenso kann z. B. eine veraltete Lehrsimulation, die in der bisherigen Form im Monist-System nicht mehr erwünscht ist, auch komplett durch eine neue ersetzt werden. Die Einrichtung eines aufwendigen Versionskontrollsystems ist zu diesem Zweck jedoch nicht erforderlich.

Die im LOM-Standard vorgesehenen Angaben über an der Erstellung beteiligte Personen sind in dieser Form für den Bedarf eines normalen E-Learning-Systems auf der Basis der hier entwickelten Plattform ebenfalls zu komplex. Es ist vollkommen ausreichend, Informationen über den Autor eines Lernobjektes abzuspeichern und zusätzlich die Möglichkeit zur Angabe weiterer beteiligter Personen zu bieten. Solange nicht eine explizite Versionskontrolle von Lernobjekten beabsichtigt wird, muß nicht jede kleine Änderung unter Angabe der ausführenden Person abgespeichert werden. Aus dem gleichen Grund wird darauf verzichtet, die Uhrzeit und das Datum jeder einzelnen Änderung separat festzuhalten. Es genügt die Angabe eines einzigen Datums-Elementes. Dieses wird bei der Erstellung eines Objektes mit dessen Erstellungsdatum vorbelegt und bei jeder Änderung des Objektes auf das Änderungsdatum gesetzt. Diese Aktionen können automatisch ohne Eingriff des Benutzers erfolgen.

5.1.2.3 Meta-Metadaten

LOM-Kategorie: Meta-Metadata

Diese Kategorie enthält Elemente zur Beschreibung des Metadatensatzes. Diese Elemente beziehen sich also nicht auf das Lernobjekt selbst, sondern beschreiben die dazu angegebenen Metadaten. Dazu gehören ein global eindeutiger Identifier für den Metadatensatz, Angaben darüber, ob der Metadatensatz in einem Katalog erfaßt ist, um welchen Katalog es sich dabei handelt, und über den betreffenden Katalogeintrag. Weiterhin soll angegeben werden, welche Personen oder Einrichtungen an der Eingabe der Metadaten beteiligt waren, wann die Eingabe erfolgte, welches Schema für die Metadaten verwendet wurde und in welcher Sprache der Metadatensatz vorliegt.

Im Monist-System werden die ersten Metadaten für Lernobjekte erzeugt, wenn diese in das System eingebracht werden. Dabei wird ein Teil der Elemente automatisch generiert, der Rest muß vom Autor angegeben werden. Spätere Änderungen werden

ebenfalls vom Autor oder automatisch vom System vorgenommen. Damit erübrigen sich gesonderte Angaben darüber, wer die Metadaten eingegeben hat, ebenso wie die Angabe von Katalog-Informationen, da ein gerade neu eingegebener Metadatensatz in keinem externen Katalog enthalten sein kann.[259] Des weiteren sind die vom System verwalteten Metadaten-Elemente für alle Lernobjekte des gleichen Typs im Monist-System gleich (unabhängig davon, ob bei einem bestimmten Objekt auch tatsächlich alle Elemente verwendet werden), so daß die Angabe eines speziellen Schemas für jedes Lernobjekt ebenfalls überflüssig ist.

Darüber hinaus können Metadaten in der E-Learning-Plattform prinzipiell mehrsprachig vorliegen. Je nach der vom Benutzer eingestellten Sprache werden diesem die Metadaten in dieser Sprache oder, sofern in dieser Sprache kein Metadatensatz vorliegt, in einer voreingestellten Standard-Sprache angezeigt. Die Angabe einer Sprache für einen Metadatensatz ist im Rahmen der E-Learning-Plattform daher aus technischer Sicht nicht erforderlich und aufgrund der inhärenten Mehrsprachigkeit der Metadaten im allgemeinen auch gar nicht möglich. Die Sprachangabe des LOM-Standards ist darauf ausgelegt, daß ein bestimmtes Objekt immer in genau einer Sprache vorliegt. Diese Einschränkung ist in der E-Learning-Plattform nicht gegeben.

Aus diesen Betrachtungen folgt, daß die Angabe von Meta-Metadaten im Rahmen der E-Learning-Plattform im Grunde genommen nicht notwendig ist. Die Berücksichtigung dieser Kategorie von Metadaten würde lediglich zu einem höheren Verwaltungsaufwand für die Metadaten führen und das System für den menschlichen Benutzer weniger überschaubar machen. Aus diesen Gründen wurde die komplette Kategorie der Meta-Metadaten des LOM-Standards im Projekt Monist nicht berücksichtigt.

5.1.2.4 Technische Eigenschaften

LOM-Kategorie: Technical

Diese Kategorie umfaßt Angaben über technische Eigenschaften von Lernobjekten sowie über die zu ihrer Benutzung erforderlichen technischen Voraussetzungen (z. B. Hardware und installierte Software des Benutzers). Im einzelnen werden Metadaten-Elemente zur Angabe des Datenformates (MIME-Type, entspricht DC.FORMAT aus dem "Dublin Core" Standard), der Dateigröße und der physikalischen Adresse eines Lernobjektes definiert. Dazu kommen komplexe Angaben über benötigte Hardware- und Software-Komponenten, Anweisungen zur Installation, sonstige Anforderungen (als Freitext) sowie eine Angabe über die technische Abspieldauer z. B. von Filmen oder Videos. (Dieses Metadaten-Element sollte nicht mit der beabsichtigten Bearbeitungsdauer des Lernobjektes während des Einsatzes in der Lehre verwechselt werden, die als eigenes Element in der LOM-Kategorie Educational definiert ist.)

Die wichtigsten Elemente dieser Kategorie sind das Datenformat (MIME Type) und die Größe des Lernobjektes. Beide Elemente sind im Rahmen der E-Learning-Plattform von großer Bedeutung und werden daher auch vom Monist-System berücksichtigt. Weitere technische Eigenschaften werden dagegen nicht verwaltet, da sie nicht benötigt werden. Nach erfolgter Installation der E-Learning-Plattform benötigt der Benutzer keinerlei

[259] Natürlich könnte ein Autor Metadaten aus einem externen Katalog manuell eingeben. Die Verwaltung derartiger Angaben würde allerdings schnell die Frage nach einer automatischen Überprüfung aufwerfen, die einen erheblichen technischen Aufwand mit sich bringen würde. Das Format der im Monist-System verwendeten Metadaten entspricht jedoch ohnehin keiner in einem bekannten Katalog benutzten Struktur.

technische Informationen über die internen Komponenten des Systems mehr, da diese alle auf der gleichen in Java entwickelten Plattform lauffähig sind. Angaben über dafür benötigte Hardware und Software erübrigen sich innerhalb des Systems, denn ohne diese Hardware und Software wäre es nicht möglich, die E-Learning-Plattform zu benutzen und derartige Metadaten überhaupt zu lesen. Daher müssen diese Angaben außerhalb des Systems zusammen mit dem Installationspaket abgelegt werden.

Die im Rahmen des Projektes Monist ebenfalls verwendeten externen Komponenten werden dagegen auf eine andere Weise behandelt. Bei diesen Komponenten handelt es sich beispielsweise um bestehende Software, die in anderen Programmiersprachen geschrieben ist und daher nicht direkt in das Monist-System integriert werden kann. Sofern der Benutzer zur Installation derartiger Programme Informationen benötigt, erhält er diese innerhalb des Monist-Systems in Form eigener Lehreinheiten. Eine Installationsanweisung oder eine Bedienungsanleitung für ein bestimmtes externes Programm ist im Monist-System somit selbst ein Teil des Inhaltes und keine Meta-Information.

Das letzte verbleibende Element der LOM-Kategorie Technical, die Abspieldauer, kann ebenfalls vernachlässigt werden. Die Dualität der technischen Benutzungsdauer und der Benutzungsdauer als Lernobjekt ist wenig hilfreich. Eine separate Angabe der technischen Abspieldauer wird für die meisten Lernobjekte nicht benötigt oder kann erst gar nicht angegeben werden. (Wie definiert man z. B. die Abspieldauer eines statischen Textes?) Die Verwaltung zweier verschiedener Zeitspannen würde beim Benutzer nur zu Verwirrung führen, da die Unterscheidung zwischen den beiden Zeiten gerade bei denjenigen Lernobjekten, bei denen tatsächlich eine technische Dauer angegeben werden kann (z. B. bei Videos), nur schwer nachvollziehbar ist. Außerdem wäre die Eingabe von verschiedenen Zeitangaben für das gleiche Objekt durch den Benutzer sehr fehlerträchtig, so daß zugunsten der einfacheren Bedienbarkeit des Systems auf eine separate technische Zeitangabe verzichtet werden sollte, wie es in Monist der Fall ist.

5.1.2.5 Lernspezifische Eigenschaften

LOM-Kategorie: Educational

Diese Kategorie enthält eine Reihe von Elementen, die für den Einsatz von Lernobjekten in der Lehre von Bedeutung sind. Diese Elemente dienen der pädagogischen Klassifikation nach unterschiedlichen Gesichtspunkten und sind von zentraler Bedeutung für die Entscheidung, welche Inhalte zu welchem Zeitpunkt welcher Zielgruppe zur Verfügung gestellt werden. Sie sind insbesondere für Lehrende eine wichtige Orientierungshilfe, da anhand dieser Metadaten durch eine gezielte Suche Lernobjekte mit bestimmten Charakteristiken aufgefunden werden können, die dann beispielsweise zu Kursen zusammengestellt werden können. Im Folgenden werden die einzelnen lernspezifischen Metadaten-Elemente unter spezieller Berücksichtigung ihrer Relevanz für das Monist-System betrachtet.

Der Interaktivitätstyp betrachtet den Informationsfluß zwischen dem Lernobjekt und dem Benutzer. Dabei wird zwischen aktiven Objekten (mit Benutzer-Interaktion), expositiven Objekten (ohne Benutzer-Interaktion) sowie Objekten mit gemischtem bzw. undefiniertem Interaktivitätstyp unterschieden. Zusätzlich ist ein weiteres Metadaten-Element zur Angabe des Interaktivitätsgrades definiert, dessen Wertebereich von "sehr hoch" bis "sehr niedrig" reicht. Man beachte, daß die Angabe des Interaktivitätsgrades nur dann sinnvoll ist, wenn der Interaktivitätstyp das Objekt als "aktiv" kennzeichnet.

Angaben über die Interaktivität von Lernobjekten können im Rahmen der E-Learning-Plattform von Bedeutung sein, wenn beispielsweise ein Lehrender einen Kurs zusammenstellt und dabei entscheiden muß, ob die vorliegenden Lehrsimulationen eher für eine Präsentation in einer Vorlesung (Frontalunterricht ohne Beteiligung der Lernenden, niedriger Interaktivitätsgrad) oder für ein Praktikum (aktives selbständiges Lernen, hoher Interaktivitätsgrad) geeignet sind.

Der Hauptzweck des Monist-Systems ist die Vermittlung von Lehrsimulationen. Diese zeichnen sich dadurch aus, daß sie durchgängig interaktiv sind bzw. in aller Regel zumindest interaktive Bestandteile haben. Daher bietet es sich an, auf ein separates Metadaten-Element zur Angabe des Interaktivitätstyps zu verzichten, da dieser ohnehin praktisch immer den selben Wert ("aktiv") haben würde. Für Angaben über den Interaktivitätsgrad kann ein Metadaten-Element vorgesehen werden, für das der Autor eines Lernobjektes einen Wert aus einer fest vorgegebenen Liste wählen kann.

Angaben über die Interaktivität von Lernobjekten sind nur für zusammengesetzte Objekte wie Kurse oder Lehrsimulationen sinnvoll, während z. B. Bilder oder Texte überhaupt keine Möglichkeit zur Interaktion mit dem Benutzer bieten. (Das Anklicken eines Links in einem Hypertext-Dokument ist keine Interaktion im Sinne des LOM-Standards.) Daher wird für derartige statische Medien auf die Verwaltung dieses Metadaten-Elementes verzichtet. Beim Einbringen von Texten oder Grafiken in das Monist-System wird dem Benutzer demzufolge keine Angabe über deren Interaktivitätsgrad abverlangt. Diese Angabe ergibt lediglich beim Anlegen neuer Lehrsimulationen und beim Zusammenstellen von Kursen theoretisch einen gewissen Sinn.[260]

Ein weiteres Element der LOM-Kategorie Educational ist der Typ des Lernobjektes. Dabei ist eine Auswahl aus einer fest vorgegebenen Liste von Objekttypen vorgesehen. Dieser Eintrag entspricht dem Element DC.TYPE aus dem "Dublin Core" Standard. Im Rahmen des Projektes Monist enthält diese Auswahlliste Bezeichnungen für unterschiedliche Typen von Lernobjekten (z. B. Lehrsimulation, Kurs, Bild). Die Auswahl der möglichen Listenelemente folgt dabei nur teilweise dem LOM-Standard, da dieser einerseits einige Elemente enthält, die im Monist-System nicht benötigt werden (z. B. "Narrative Text" oder "Self Assessment") und andererseits einige in Monist benötigte Typen von Lernobjekten nicht kennt (z. B. Kurs, Instruktionstext, Modellsimulation).

Die Angabe des Objekttyps muß vom Benutzer beim Einbringen des Objektes in das System vorgenommen werden. Dies kann teilweise automatisch geschehen. So kann das System beispielsweise beim Hochladen einer Grafikdatei dieser den entsprechenden Typ zuweisen. Ebenso kann das integrierte Autorenwerkzeug beim Anlegen eines neuen zusammengesetzten Objektes, z. B. einer Lehrsimulation, diese gleich durch einen entsprechenden Metadaten-Eintrag als solche kennzeichnen.[261]

---

[260] Da sich in der Praxis im Laufe des Testbetriebes bereits frühzeitig erwies, daß dieses Metadatum für die Anwender im täglichen Betrieb keinen nennenswerten praktischen Nutzen mit sich brachte, wurde letztendlich in der zum Einsatz in der Lehre freigegebenen Version des Monist-Systems auf derartige Angaben verzichtet.

[261] In der Praxis stellt sich die Situation in Monist so dar, daß der Autor den Typ überhaupt nicht direkt in einem Metadaten-Formular eingeben muß. Entweder geschieht die Auswahl des Typs automatisch durch den Editor, oder beim Hochladen von Dateien im Navigator erhält der Benutzer in Abhängigkeit davon, in welchem Typ von Verzeichnis er sich gerade befindet, über ein Menü genau diejenigen Möglichkeiten zum Erzeugen neuer Objekte zur Auswahl, die an dieser Stelle des Systems gerade möglich sind. So ist es beispielsweise innerhalb des Inhaltes einer Lehreinheit zwar möglich, neue Seiten anzulegen, nicht jedoch z. B. Kurse (die ein übergeordnetes Strukturelement darstellen) oder Benutzer und Gruppen (die einem ganz anderen Zweig der Objekthierarchie angehören).

Ebenfalls im Rahmen des Projektes Monist anwendbar ist das Metadaten-Element zur Angabe des Schwierigkeitsgrades. Hierdurch erhalten Autoren die Möglichkeit, Lernobjekte danach zu beurteilen, wie schwierig ihre Benutzung für den Lernenden ist. Diese Information kann nicht nur für ganze Lehrsimulationen oder Kurse von Interesse sein, sondern auch dazu dienen, z. B. ins System eingebrachte Grafiken oder Texte danach zu klassifizieren, wie leicht oder schwer sie für einen durchschnittlichen Lernenden verständlich sind. Derartige Informationen können wiederum den Lehrenden dabei behilflich sein, wenn es darum geht, Kurse für Anfänger oder für Fortgeschrittene zu gestalten. Die Angabe des Schwierigkeitsgrades sollte jedoch nicht zwingend vorgeschrieben, sondern optional sein.[262]

Ein in ähnlichem Maße der Einschätzung des Autors unterliegendes Metadaten-Element ist die geschätzte Bearbeitungsdauer eines Lernobjektes. Diese Angabe ist gleich in mehrfacher Hinsicht von Nutzen: Der Lernende erhält die Möglichkeit, einzuschätzen, wie lange er für die Bearbeitung einer bestimmten Lehrsimulation benötigt. Lehrende können aufgrund dieser Zeitangaben die Gestaltung von Kursen und ganzen Lehrveranstaltungen einfacher planen. Autoren können aufgrund der Erfahrungen mit bestehenden Lehrsimulationen besser einschätzen, wie lange die Lernenden zur Abarbeitung von bestimmten Objekte benötigen, und dadurch neue Lernobjekte besser auf beabsichtigte Bearbeitungszeiten abstimmen.

Im Gegensatz zum Schwierigkeitsgrad ist die Angabe der Dauer für bestimmte Typen von statischen Lernobjekten wie z. B. Grafiken oder Texten nicht unbedingt erforderlich, während für Videos und Animationen durchaus eine Bearbeitungsdauer angegeben werden kann, die in der Regel der Abspieldauer entspricht. Für Lehrsimulationen und Kurse könnte die typische Zeitspanne, die ein Benutzer zu deren Ausführung und zur Verinnerlichung der darin vermittelten Lerninhalte benötigt, angegeben werden.[263]

Zur LOM-Kategorie Educational gehören außerdem einige weitere Elemente, die im Monist-System nicht benötigt werden. So ist beispielsweise die Angabe der semantischen Dichte vorgesehen, die beschreibt, wieviele Informationen ein Lernobjekt im Verhältnis zu seiner Größe enthält. Derartige Angaben sind nur sehr schwer quantifizierbar, da es keine allgemein anerkannte Methode zur Messung der Informationsmenge gibt. Tatsächlich gibt es sogar viele verschiedene Definitionen des Begriffes "Information". Im Rahmen des Projektes Monist wird kein Metadaten-Element zur Angabe der semantischen Dichte benutzt, da dies für die Funktionalität des Systems nicht erforderlich ist und bei den Benutzern eher zu Verwirrung durch unterschiedliche subjektive Einschätzungen des Informationsgehaltes führen würde.

Ebenfalls nicht verwendet werden die im LOM-Standard vorgesehenen Metadaten-Elemente zur Klassifizierung der Zielgruppe. Diese wird auf drei verschiedene Weisen beschrieben: Durch Angabe der Rolle des Endbenutzers, des institutionellen Kontextes

[262] Nach den Erfahrungen des Testbetriebes, die keine nennenswerte Nutzung dieses Metadatums erkennen ließen, wurde im Monist-System letztendlich im Rahmen der Vereinfachung des Metadatensatzes auf derartige Angaben verzichtet, zumal eine absolute Skala für den Schwierigkeitsgrad gerade in einem so hochgradig interdisziplinären Projekt wie Monist praktisch nicht aufgestellt werden kann.

[263] Im Zuge der Reduzierung des Metadatensatzes, die gegen Ende der Laufzeit des Projektes Monist und nach Abschluß der Phase des ersten Testbetriebes vorgenommen wurde, wurde auch dieses Element aus der Liste der zu verwaltenden Metadaten gestrichen. Ausschlaggebend dafür war neben der zur Erhaltung der Übersichtlichkeit notwendigen Vereinfachung des Metadatensatzes in erster Linie die Subjektivität dieses Elementes, da die einzelnen Benutzer in Abhängigkeit von ihren Vorkenntnissen sehr verschiedene Bearbeitungszeiten für die gleichen Inhalte benötigen können.

und des typischen Alters des Benutzers. Während über das typische Alter im universitären Bereich weder bei den Lernenden noch bei den Lehrenden oder den Autoren irgendwelche allgemeingültigen Aussagen gemacht werden können (bzw. diese nicht sehr spezifisch wären) und derartige Angaben auch im System nicht benötigt werden, können die beiden anderen Angaben weggelassen werden, weil für das gesamte Monist-System in einem Fall nur ein einziger der im LOM-Standard vorgesehenen möglichen Werte in Frage kommt, im anderen Fall dagegen nur schwer sinnvolle Einschränkungen auf einen oder mehrere der möglichen Werte vorgenommen werden können.

Bei der Rolle des Endbenutzers wird zwischen Lehrenden, Autoren, Lernenden und Managern unterschieden. Für sämtliche in der E-Learning-Plattform verwalteten Lernobjekte gilt, daß nur der Lernende Endbenutzer im Sinne des LOM-Standards ist. Ein Lernobjekt, dessen Endbenutzer ein Autor ist, wäre zum Beispiel ein Autorenwerkzeug. Ein entsprechendes Objekt stellt für den Lehrenden ein Werkzeug zur Zusammenstellung von Kursen dar. Ein Manager dagegen wird in diesem Zusammenhang als eine Person oder Institution betrachtet, die die Aufgabe hat, die Benutzung und Verbreitung von Lernobjekten zu steuern. Damit ist zum Beispiel die Verteilung einer Software wie z. B. des Monist-Systems durch das Personal eines Lehrstuhles an einer Universität gemeint. Sowohl für Manager als auch für Lehrende und Autoren stellt somit die gesamte E-Learning-Plattform bzw. der jeweils von ihnen verwendete Teilbereich einer darauf aufbauenden Software das von ihnen benutzte Lernobjekt dar. Im System selbst werden nur Objekte verwaltet, die sich an den Lernenden als Endbenutzer richten. Daher ist eine Unterscheidung der Rolle des Endbenutzers nicht erforderlich, weshalb kein entsprechendes Metadaten-Element benutzt wird.

Bei Angaben über den institutionellen Kontext handelt es sich dagegen um Informationen darüber, an welcher Art von Bildungseinrichtungen und auf welcher Stufe der Ausbildung der Lernenden an diesen Instituten ein Lernobjekt benutzt werden soll. Das Monist-System beispielsweise ist in seiner Gesamtheit in erster Linie auf den universitären Einsatz ausgelegt, wodurch bestimmte Typen von Bildungseinrichtungen (z. B. Grundschulen) gar nicht erst berücksichtigt werden müssen. Andererseits ist innerhalb der universitären Ausbildung nicht eindeutig feststellbar, zu welcher Studienphase (in erster Linie: Grundstudium oder Hauptstudium) ein bestimmtes Lernobjekt gehört. Durch die Interdisziplinarität des Monist-Projektes könnte beispielsweise die gleiche Lehrsimulation in einem der beteiligten Studiengänge im Grundstudium, in einem anderen dagegen erst im Hauptstudium verwendet werden. Daher wäre nur eine generelle Zuordnung von Lernobjekten zum Hochschulbereich möglich, die dann aber wiederum für praktisch alle Objekte in gleicher Weise gelten würde und somit als Unterscheidungskriterium überflüssig wäre. Deshalb wird auf ein Metadaten-Element zur Angabe des institutionellen Kontextes ebenfalls verzichtet.[264]

Allgemeine Angaben zur Zielgruppe können natürlich auch innerhalb der textuellen Beschreibung (entsprechend dem Element DC.DESCRIPTION des "Dublin Core" Standards) aufgeführt werden. Diese können jedoch nicht hinreichend formalisiert werden, um eine für das ganze Monist-System anwendbare Klassifizierung und eine darauf basierende automatisierte unterschiedliche Verarbeitung zu ermöglichen. Die in der LOM-Kategorie Educational definierte zusätzliche textuelle Beschreibung, die Informationen darüber enthalten soll, wie ein Lernobjekt benutzt werden soll, wird im Monist-System ebenfalls mit der allgemeinen Beschreibung zusammengefaßt.

[264] Bei einem eventuellen Export von im Monist-System angelegten Lernobjekten in andere Systeme, die einen kompletten Metadatensatz nach Maßgabe des LOM verlangen, könnten diese allgemeinen Angaben jedoch leicht automatisch generiert werden, weil ja gerade für alle Objekte die gleiche Eigenschaft gilt.

Das letzte Metadaten-Element der LOM-Kategorie Educational bezieht sich auf die Sprache des Benutzers. Man beachte, daß in der Kategorie General bereits ein Element definiert wird, das die Sprache des Lernobjektes angibt. Eine Unterscheidung dieser beiden Elemente ist im Prinzip nur in Systemen von Bedeutung, deren Zweck es ist, ihre Benutzer beim Erlernen von Fremdsprachen zu unterstützen. Im Rahmen des Monist-Projektes ist diese Unterscheidung nicht von Bedeutung. Daher wird auf die zweite Sprachangabe verzichtet. Dessen ungeachtet können Lernobjekte im Monist-System grundsätzlich mehrsprachig gestaltet werden. Auf den dafür vorgesehenen Mechanismus zur Verwaltung von mehrsprachigen Objekten wurde bereits bei der Beschreibung des Elementes DC.LANGUAGE aus dem "Dublin Core" Standard hingewiesen, er wird im weiteren Verlauf dieses Kapitels umfassend beschrieben.

5.1.2.6 Rechtliche Bestimmungen

LOM-Kategorie: Rights

Diese Kategorie umfaßt Angaben dazu, ob die Benutzung eines Lernobjektes irgendwelchen rechtlichen Bestimmungen und Einschränkungen unterliegt, insbesondere dem Urheberrecht, unter welchen Bedingungen es genutzt werden darf und welche Kosten dabei ggf. anfallen. Im wesentlichen entspricht diese Kategorie dem Element DC.RIGHTS aus dem "Dublin Core" Standard, das hier nur in verschiedene Unterelemente zerlegt wird, die getrennt voneinander verarbeitet werden. Im Rahmen des Monist-Systems ist eine derartige Unterteilung nicht erforderlich.

Dennoch ist in Monist natürlich auch eine Möglichkeit zur Angabe von Urheberrechten vorgesehen. Beim Einbringen neuer Objekte in das System wird dem Benutzer die Gelegenheit gegeben, eventuell vorhandene fremde Urheberrechte anzugeben. Dieses Datenfeld muß jedoch vom Benutzer nicht zwingend ausgefüllt werden. Falls für ein Objekt keine Angaben zum Urheberrecht gemacht werden, wird als Voreinstellung eingetragen, daß die Rechte für dieses Objekt bei der Monist-Projektgruppe liegen. In anderen Anwendungen, die auf der E-Learning-Plattform basieren, könnte an dieser Stelle beispielsweise einfach eine entsprechende Voreinstellung angegeben werden, sofern keine aufwendigeren und komplexeren Methoden zur Angabe von rechtlichen Bestimmungen benötigt werden.

5.1.2.7 Beziehungen zu anderen Lernobjekten

LOM-Kategorie: Relation

Diese Kategorie entspricht im Wesentlichen dem Element DC.RELATION aus dem "Dublin Core" Standard, ist jedoch in mehrere Unterelemente aufgeteilt. Diese bezeichnen die Art der Beziehung (in der gleichen Art und Weise, in der dies im "Dublin Core" Standard durch die entsprechenden "Qualifier" geschieht), das Zielobjekt des Verweises, den Identifier dieses Zielobjektes, seine Beschreibung sowie einen eventuellen Katalog-Eintrag zu diesem Objekt. Darauf, welche Art von Beziehungen zwischen Objekten im Monist-System verwaltet wird, wurde bereits bei der Beschreibung des Elementes DC.RELATION aus dem "Dublin Core" Standard näher eingegangen. Man beachte, daß die Beschreibung des Zielobjektes direkt über das Zielobjekt zugänglich ist und daher im Monist-System nicht noch einmal zusätzlich bei dem den Verweis repräsentierenden Objekt gespeichert wird.

5.1.2.8 Anmerkungen

LOM-Kategorie: Annotation

Diese Kategorie von Metadaten dient der Verwaltung von Anmerkungen zu Lernobjekten. Zu jeder Anmerkung umfaßt sie Angaben über ihren Autor, das Erstellungsdatum und den Inhalt der Anmerkung. Man beachte, daß die Inhalte von Anmerkungen strenggenommen eigentlich keine Metadaten sind! Im Monist-System werden deshalb Anmerkungen als eigenständige Objekte verwaltet, die selbst wiederum über Metadaten wie z. B. den Autor, Datum und Zeit der Erstellung, Typ, Format (MIME Type), Zugriffsrechte und einen Identifier verfügen. Die Speicherung von Anmerkungen und ihre Zuordnung zu den betreffenden Lernobjekten geschieht im Monist-System durch den gleichen Mechanismus, der auch Verweise (Links) zwischen Lernobjekten verwaltet. Für andere Anwendungen auf Basis der E-Learning-Plattform wird ein entsprechendes Vorgehen empfohlen, da auf diese Weise die Anmerkungen leichter zugänglich sind und von Benutzern auch unabhängig von den Zugriffsrechten auf das Objekt, zu dem sie gehören, erstellt und bearbeitet werden können.

5.1.2.9 Klassifikation

LOM-Kategorie: Classification

Diese Kategorie dient dazu, Lernobjekte in bestimmte Klassifikationssysteme einzuordnen. Dabei können Objekte auch nach mehreren unterschiedlichen Systemen klassifiziert werden, was durch mehrfache separate Angaben der entsprechenden Metadaten zum Ausdruck gebracht wird. Ein solcher Metadatensatz zur Klassifikation besteht aus mehreren Elementen, deren erstes zunächst einmal den Zweck (purpose) der Klassifikation zum Ausdruck bringen soll. Dabei kann zwischen mehreren vordefinierten Werten ausgewählt werden. Diese geben letztendlich mögliche Kriterien wieder, nach denen Lernobjekte klassifiziert werden können: Fachrichtung (discipline), Grundbegriff (idea), fachliche Voraussetzungen (prerequisite), Lernziel (educational objective), Zugangsbeschränkungen (accessibility restrictions), Ausbildungsniveau (educational level), Niveau der fachlichen Fähigkeiten (skill level) und Sicherheitsstufe (security level).

Weitere Metadaten-Elemente dienen der Einordnung in eine bestehende Taxonomie. Dabei wird zunächst der Name des Klassifikationssystems angegeben, wobei irgendeine beliebige Taxonomie verwendet werden kann, die einem definierten Standard oder einer irgendwo veröffentlichten Systematik entsprechen kann, ebensogut aber auch für den Gebrauch in einem eigenen Softwaresystem frei definiert sein kann. Zur Einordnung wird die Position des Lernobjektes in der betreffenden Taxonomie angegeben. Zusätzlich sind noch Elemente für eine Beschreibung des Lernobjektes relativ zum angegebenen Zweck der Klassifikation durch Freitext und durch Stichwörter vorgesehen.

Im Rahmen der E-Learning-Plattform und ebenso exemplarisch im Monist-System als deren Referenzanwendung wird ein anderer Ansatz zur Klassifikation von Lernobjekten und zu ihrer Einordnung nach unterschiedlichen Taxonomien verfolgt. Diese können dem Benutzer in Form von virtuellen Verzeichnisstrukturen präsentiert werden, wobei verschiedene benutzte Systematiken zur Klassifikation durch unterschiedliche Verzeichnisbäume dargestellt werden können. Der Begriff "virtuelle Verzeichnisse"[265] bedeutet dabei, daß sich das gleiche Objekt aus Sicht des Benutzers in mehreren verschiedenen

[265] Zu den in der E-Learning-Plattform verwendeten virtuellen Verzeichnissen siehe Kapitel 5.4.

Verzeichnissen befinden kann, unabhängig von der tatsächlichen physikalischen Speicherstruktur. Zur thematischen Klassifizierung dient beispielsweise die Einsortierung in verschiedene Fachbereiche, die durch derartige virtuelle Verzeichnisse repräsentiert werden. Dabei kann sich ein Objekt (z. B. eine Lehrsimulation) gleichzeitig in mehreren Verzeichnissen befinden, wenn es mehreren Fachbereichen angehört.

Eine anderes mögliches Klassifizierungssystem kann sich beispielsweise danach richten, an welcher Universität, an welcher Fakultät, in welchem Fachbereich und für welche spezielle Lehrveranstaltung ein Lernobjekt benutzt wurde. Man findet das betreffende Objekt dann in einem virtuellen Verzeichnis, das der betreffenden Veranstaltung entspricht. Wenn das gleiche Objekt auch für eine andere Lehrveranstaltung (möglicherweise an einer anderen Universität) benutzt wurde, findet man es auch in dem virtuellen Verzeichnis, das dieser Veranstaltung entspricht.

Eine Lehrsimulation, die gleichermaßen in Lehrveranstaltungen an der Fakultät für Biologie in Bielefeld, an der Fakultät für Informatik in Ilmenau und an der Fakultät für Psychologie in Bamberg eingesetzt würde, könnte auf diese Weise leicht von jedem interessierten Benutzer gefunden werden. Sie wäre einerseits über einen Pfad im virtuellen Verzeichnisbaum jeder der beteiligten Universitäten zugänglich, könnte andererseits aber auch genauso von einem Studenten gefunden werden, der entsprechend seiner persönlichen fachlichen Ausrichtung im virtuellen Verzeichnisbaum einer der drei Disziplinen Biologie, Informatik und Psychologie nach Lehrsimulationen stöbert.

Der Mechanismus zur Verwaltung dieser virtuellen Verzeichnisse bzw. Verzeichnisbäume wird in der E-Learning-Plattform so konstruiert, daß er jederzeit um geeignete Kategorien erweiterbar ist. Auf diese Weise kann zum Beispiel eine neu hinzukommende fachliche Disziplin oder ein weiterer teilnehmender Lehrstuhl auf einfache Weise dem System hinzugefügt werden und bereits vorhandene Lernobjekte mit benutzen. Ebenso können bei Bedarf vollkommen neue Klassifikationssysteme verwendet werden. Aufgrund der flexiblen Konzeption des virtuellen Verzeichnissystems kann die Handhabung so gestaltet werden, daß es dazu keiner Programmierung bedarf, sondern die neuen Verzeichnisse lediglich der bestehenden Struktur hinzugefügt werden müssen. Diese Aktion ist für den Benutzer nicht schwieriger als das Anlegen neuer Verzeichnisse in einem graphischen Betriebssystem wie Microsoft® Windows® oder Mac OS.

5.1.3 Schlußfolgerungen

Nach Betrachtung der beiden Metadaten-Standards "Dublin Core" und "LOM" ergeben sich für das Monist-System, das als Referenzanwendung für die E-Learning-Plattform dient, insgesamt 17 zu verwaltende Metadaten-Elemente, die wie folgt in drei Gruppen unterteilt werden können:

1.) Metadaten zur Herkunft des Lernobjektes:

- Autor (*DC.CREATOR*), eventuell zusätzlich Vorname und Nachname
- Herausgeber (*DC.PUBLISHER*)
- Weitere beteiligte Personen (*DC.CONTRIBUTORS*)
- Quelle (*DC.SOURCE*) für Medien, die nicht ursprünglich dem Monist-System entstammen
- Rechtliche Bestimmungen (*DC.RIGHTS*), insbesondere Copyright-Angaben

2.) Metadaten zur inhaltlichen Beschreibung des Lernobjektes:

- Titel (*DC.TITLE*)
- Stichwörter (Keywords, übernehmen die Aufgabe von *DC.SUBJECT*)
- Beschreibung (*DC.DESCRIPTION*)
- Objekttyp (*DC.TYPE*), dient zur logischen Unterscheidung von Lernobjekten
- Interaktivitätsgrad (nicht für Medien-Rohdaten)[266]
- Schwierigkeitsgrad (optional)[267]
- Bearbeitungsdauer[268]

3.) Metadaten zur technischen Verwaltung des Lernobjektes:

- Ressourcen-Identifikation (*DC.IDENTIFIER*)
- Datenformat (*DC.FORMAT*), dient zur technischen Unterscheidung von Dateitypen
- Dateigröße
- Erstellungsdatum bzw. Datum der letzten Änderung (*DC.DATE*)
- Status (im Qualitätssicherungsverfahren)

Alle textuellen Metadaten-Elemente können grundsätzlich mehrsprachig eingetragen werden. Die Liste der Metadaten zur inhaltlichen Beschreibung kann noch um weitere Elemente ergänzt werden (z. B. Abstraktionsgrad). Dazu kommen die Mechanismen zur Verwaltung von Verweisen (Links), Anmerkungen und virtuellen Verzeichnissen, die getrennt von der Metadatenverwaltung implementiert werden.

Natürlich kann diese für das Projekt Monist geltende Auflistung nicht allgemein auf jede mögliche E-Learning-Anwendung übertragen werden, die künftig auf der Basis des hier entwickelten Systems geschaffen wird. Vielmehr ist diese Zusammenstellung als ein Beispiel zu verstehen, das in anderen Anwendungen um weitere Metadaten ergänzt oder ggf. auch punktuell reduziert wird, etwa wenn in einer anderen Applikation keine Angabe über den Status benötigt wird.

Neben den beiden in diesem Abschnitt exemplarisch betrachteten Metadatensätzen gibt es außerdem noch mehrere weitere miteinander konkurrierende Standards, die in vielen heutigen E-Learning-Systemen Anwendung finden. Beispiele dafür sind der Metadatenstandard des Aviation Industry CBT Committee (AICC)[269] sowie das Sharable Content Object Reference Model (SCORM) der Advanced Distributed Learning (ADL)[270] Initiative des US-amerikanischen Verteidigungsministeriums. Die Metadaten zu Inhalten, die unter Verwendung derartiger Standards angelegt wurden, müssen von der E-Learning-Plattform ebenfalls verwaltet werden können, um die Möglichkeit des Einstellens dieser Inhalte in das System zu erhalten.

Abschließend läßt sich sagen, daß die E-Learning-Plattform zur Verwaltung beliebiger Metadaten in der Lage sein muß. Nach Betrachtung der vorgestellten Standards müssen dabei numerische und textuelle Metadatentypen unterstützt werden, wobei Texte mehrsprachig verwaltet werden müssen. Für bestimmte Texte, z. B. für Stichwörter, wird die Angabe mehrerer Werte benötigt. Außerdem muß eine Möglichkeit gegeben sein, in den Metadaten auf andere, im System abgelegte Objekte eindeutig zu referenzieren.

---

[266] Nach der ersten Testphase gestrichen.
[267] Nach der ersten Testphase gestrichen.
[268] Nach der ersten Testphase gestrichen.
[269] Homepage: http://www.aicc.org.
[270] Homepage: http://www.adlnet.org.

5.2 Eindeutige Identifizierung von Objekten in verteilten Systemen

Bevor im nächsten Abschnitt die Definition der sogenannten Virtuellen Lernobjekte (VLO) erfolgt, die zur Repräsentation von Objekten und zur Verwaltung der mit ihnen assoziierten Metadaten dienen, soll an dieser Stelle auf die Frage eingegangen werden, auf welche Weise eine exakte Identifikation von Objekten in der E-Learning-Plattform möglich ist. In einem verteilten Client-Server-System oder Peer-to-Peer-System, in dem Autoren zu verschiedenen Zeiten und an verschiedenen Orten an Inhalten für das selbe Informationssystem arbeiten, ist es erforderlich, Objekte (z. B. Texte oder Grafiken) unabhängig von ihrem Ursprung eindeutig identifizieren und referenzieren zu können.

Dateinamen sind dafür im allgemeinen wenig geeignet, da es naturgemäß sehr leicht möglich ist, daß Benutzer an unterschiedlichen Computern für ähnliche Inhalte gleiche Bezeichnungen wählen. Auch das Hinzufügen eines Pfades in einer Verzeichnisstruktur hilft hier wenig und führt eher noch zu Problemen, wenn die Daten zum Beispiel auf dem Client im Dateisystem und auf dem Server in einer Datenbank gespeichert werden.

In denjenigen Client-Server-Systemen, bei denen die Datenhaltung ausschließlich auf dem Server erfolgt[271] und die Benutzer zum Hinzufügen von Daten immer mit diesem verbunden sein müssen, kann man dieses Problem durch Einführung eines eindeutigen Schlüssels als Identifizierungsmerkmal (engl. identifier) für Objekte lösen, der beim Einstellen der Daten auf den Server automatisch vergeben wird. Dabei muß der Server sicherstellen, daß die Schlüssel für verschiedene Objekte auch tatsächlich in jedem Fall unterschiedlich sind. Wenn jedoch die Autoren nicht zu jedem Zeitpunkt mit dem Server verbunden sind, sondern auch offline am eigenen Rechner Dateien erstellen können, kann diese zentrale Vergabe und Kontrolle des Schlüssels nicht gewährleistet werden.

Abhilfe schafft hier die Einführung eines globalen Objektschlüssels, bei dem der Ursprung eines Objektes in den Schlüssel mit hineinkodiert wird. Jedes Objekt, das in einem an das verteilte System anschließbaren Programm erstellt wird, erhält einen Schlüssel zugewiesen, der zum einen innerhalb des jeweiligen Programms auf dem Rechner des Benutzers als eindeutiges Identifizierungsmerkmal dienen kann und zum anderen ein eindeutiges Identifizierungsmerkmal für das erstellende System beinhaltet. Darin ist genau festgehalten, welche installierte Instanz des Programmes zur Erstellung des Objektes benutzt wurde.

Objekte behalten diesen Schlüssel unabhängig vom Speicherort und auch beim Transport über Rechnergrenzen hinweg, so daß zu jedem Zeitpunkt und auf jedem Rechner (insbesondere auch auf dem Server) die Eindeutigkeit des Schlüssels gewährleistet ist. Dieser Schlüssel kann dann beispielsweise dazu dienen, anstelle des gesamten Dateiinhaltes in einer Datenbank abgelegt zu werden. Der eigentliche Dateiinhalt kann dann in einer separaten Datei abgelegt werden, die über den Schlüssel angesprochen werden kann, was zu einer Verbesserung der Performance der Datenbank führt.[272]

Verwendet man anstelle von Dateinamen und Verzeichnisstrukturen einen Schlüssel zur Identifikation von Dateien, ist es außerdem leicht möglich, dem Benutzer bestimmte Dateien unabhängig von der tatsächlichen Verzeichnisstruktur in virtuellen Ordnern zu präsentieren, die nach thematischen Kriterien anhand von Meta-Informationen aus der Datenbank dynamisch zusammengesetzt werden.

---

[271] Hierzu zählen insbesondere webbasierte Systeme, die über eine Browserschnittstelle bedient werden.
[272] Diese Möglichkeit wird in der E-Learning-Plattform genutzt. Siehe Kapitel 5.7.

Die globale Eindeutigkeit des Schlüssels kann nicht an der Identität des Autors eines Objektes festgemacht werden, da hierzu stets eine zentrale Identifizierung am Server notwendig wäre. Anderenfalls wäre eine Verwaltung aller Benutzer auf jedem Client erforderlich, die jedoch nicht durch Synchronisation stets auf dem selben Stand gehalten werden kann, da hierzu eine ständige Verbindung mit dem Server und den anderen Clients erforderlich wäre.[273] Außerdem könnte der selbe Autor auf verschiedenen Rechnern Dateien mit dem gleichen, jeweils lokal eindeutigen Objektschlüssel erzeugen, womit die globale Eindeutigkeit nicht mehr gewährleistet wäre. Diese Möglichkeit ist wahrscheinlicher, als es auf den ersten Blick erscheinen mag, denn sobald derselbe Autor beispielsweise auf einem privaten und einem dienstlichen Rechner die E-Learning-Plattform installiert und auf beiden Computern als erster Benutzer neue Objekte erzeugt, wären bereits die ersten Identifizierungsschlüssel doppelt vergeben.

Statt dessen muß die Eindeutigkeit aus der jeweiligen Instanz des Client-Programmes hervorgehen, mit der die Objekte erzeugt werden. Hierzu gibt es mehrere Möglichkeiten. Von einer zentralen Verteilung der Client-Software mit Vergabe eines eindeutigen Lizenzschlüssels durch eine zentrale Lizensierungsautorität ist abzuraten, da nicht gewährleistet werden kann, daß derselbe Schlüssel nicht für mehrere Installationen auf unterschiedlichen Rechnern benutzt wird. Außerdem erfordert dieser Weg eine ständige manuelle Pflege des Systems.

Die IP-Adresse des jeweiligen Clients kann ebenfalls nicht als Identifizierungsschlüssel für den Client verwendet werden, da diese sich im Laufe der Zeit ändern kann, wenn ein Computer an ein DHCP-Netzwerk angeschlossen ist[274] oder die Verbindung mit dem Internet stets durch Einwahl bei einem Internet-Provider erfolgt. Zudem ist in diesem Fall im Offline-Betrieb nicht sichergestellt, daß der Client überhaupt eine IP-Adresse zur Verfügung hat.

Statt dessen kann die eindeutige Identifizierung sichergestellt werden, indem der Client nach seiner Installation bei der ersten Einwahl auf dem Server von diesem einen Schlüssel zugewiesen bekommt. Durch diese nur ein einziges Mal erfolgende Maßnahme wird die Eindeutigkeit der Schlüssel sichergestellt. Es ist nicht erforderlich, zur Ermittlung des Identifikationsschlüssels für den Client jedes Mal eine Verbindung mit dem Server herzustellen. Damit können auf dem Client auch offline global eindeutige Schlüssel für neue Objekte erstellt werden.

Diese Vorgehensweise bedeutet jedoch wiederum die Notwendigkeit des Aufbaus einer Verbindung mit dem Server vor der ersten Erzeugung eines Objektes auf dem Client und ist auch nicht dazu geeignet, damit dem Server selbst einen Identifikationsschlüssel zuzuweisen. Außerdem erfordert ein solcher Mechanismus die Schaffung spezialisierter Funktionalitäten auf dem Server und stellt eine Hindernis dafür dar, das gesamte System der E-Learning-Plattform auch als Peer-to-Peer-System betrachten zu können, bei dem die einzelnen Installationen ein Netzwerk von voneinander unabhängigen, vollkommen gleichberechtigten Komponenten bilden.

---

[273] Gleichwohl wurde in die E-Learning-Plattform ein Mechanismus eingebaut, der jeweils beim Aufbau einer Verbindung zwischen zwei Instanzen des Systems auf verschiedenen Computern zuerst die auf den beiden Systemen vorhandenen Benutzerinformationen aktualisiert. Allerdings können sich auf diese Art und Weise trotzdem noch erhebliche Verzögerungen bei der Verbreitung geänderter Daten über Benutzer ergeben, wenn einzelne Installationen nur selten im Online-Betrieb benutzt werden.

[274] Entsprechende Tests, die im Rahmen des Projektes Monist vorgenommen wurden, führten bereits bei der Verwendung von ca. einem Dutzend Laptops, auf denen nacheinander im selben DHCP-Netzwerk die E-Learning-Plattform installiert wurde, zu doppelten Schlüsseln.

Abhilfe schafft nur ein Mechanismus, der die einzelnen Installationen der E-Learning-Plattform dazu befähigt, jeweils autonom und insbesondere auch offline einen eigenen Identifikationsschlüssel zu erzeugen. In diesem Fall liegen bei der Erzeugung natürlich keinerlei Informationen über die auf anderen Computern bereits vorliegenden oder noch zu erzeugenden Schlüssel vor. Theoretisch kann daher zwar nicht garantiert werden, daß ein auf einem Rechner bei der Installation erzeugter Identifikationsschlüssel wirklich global eindeutig ist. Jedoch kann die Vorschrift zur Erstellung so gestaltet werden, daß der Fall der Erzeugung gleicher Schlüssel so unwahrscheinlich ist, daß er in der Praxis ausgeschlossen ist.

In der E-Learning-Plattform wird dies auf folgende Weise gehandhabt: Der Schlüssel für jede neue Installation wird aus der aktuellen Systemzeit des betreffenden Computers (angegeben in Millisekunden seit dem 1. Januar 1970, 00:00 Uhr UTC) generiert. Damit ist bereits ausgeschlossen, daß zu einem anderen Zeitpunkt irgendwo der gleiche Schlüssel erneut erzeugt werden kann. Für den bereits äußerst unwahrscheinlichen Fall, daß in der selben Millisekunde auf verschiedenen Computern mehrere Installationen der E-Learning-Plattform erzeugt werden, wird der Schlüssel außerdem um eine vierstellige Zufallszahl ergänzt. Damit ist die Eindeutigkeit des Schlüssels für jede einzelne Instanz des Programmes in hinreichender Weise sichergestellt.

Der eindeutige Identifizierungsschlüssel für jedes Objekt wird im Rahmen der E-Learning-Plattform durch die Klasse *UniversalObjectIdentifier* dargestellt. Darin sind zwei private Datenfelder namens *serverIdentifier* und *objectIdentifier* definiert, deren ersteres jeweils die ID der Instanz der E-Learning-Plattform enthält, auf der ein Objekt erzeugt wurde, während letzteres die ID enthält, mit der das Objekt innerhalb dieser Installation eindeutig identifiziert werden kann. Entsprechend dem objektorientierten Prinzip der Datenkapselung sind die beiden Datenfelder nur über entsprechende Zugriffsmethoden zum Setzen und auslesen zugänglich. Damit wird auch die Erzeugung ungültiger IDs verhindert.

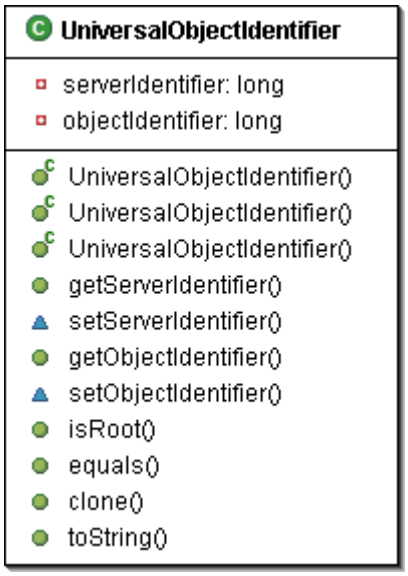

Abbildung 5.1: Klassendiagramm der Klasse UniversalObjectIdentifier

Die Klasse *UniversalObjectIdentifier* (abgekürzt: UOI) verfügt über drei verschiedene Konstruktoren. Der erste dient der Erzeugung eines neuen Identifiers für ein neues Objekt. Der zweite wird zur Rekonstruktion eines bereits zuvor bestehenden Identifiers benötigt, wenn dessen numerische Bestandteile beispielsweise aus einer Datenbank eingelesen wurden. Eine ähnliche Aufgabe erfüllt auch der dritte Konstruktor, der einen Identifier aus dessen textueller Repräsentation als String wiederherstellen kann, wie sie von der Methode *toString* erzeugt wird. Dies ist insbesondere für die Übertragung von Objekten über den XML-RPC-Kommunikationsmechanismus von Bedeutung, der in Kapitel 6 beschrieben wird. Ein *UniversalObjectIdentifier*, dessen *serverIdentifier* den Wert 12345 und dessen objectIdentifier den Wert 67890 hätte, würde beispielsweise durch den String 12345:67890 repräsentiert. Daneben verfügt diese Klasse noch über eine Methode *equals* zum Vergleich von IDs sowie über eine Methode *isRoot*, die für einen gegebenen *UniversalObjectIdentifier* feststellt, ob sich diese ID auf das Wurzelverzeichnis der E-Learning-Plattform bezieht. Dieses hat unabhängig vom jeweiligen Computer stets die ID 0:0 und stellt eine für alle Installationen des Systems gemeinsame Basis für die virtuelle Verzeichnisstruktur dar.

5.3 Repräsentation von Daten durch Virtuelle Lernobjekte

Wie in Kapitel 5.1 ausführlich dargelegt wurde, werden zur Verwaltung von Lernobjekten im Rahmen der E-Learning-Plattform Datenstrukturen benötigt, die zur Aufnahme beliebiger Metadaten geeignet sind. Darüber hinaus müssen für die einzelnen Objekte aber noch weitere Informationen gespeichert werden. Zum einen handelt es sich dabei um die Zugriffsrechte, durch die festgelegt wird, welchen Benutzern der schreibende oder lesende Zugriff auf die einzelnen Objekte erlaubt ist. Zum anderen ist die Ablage struktureller Informationen zur Einordnung der Lernobjekte in Verzeichnisstrukturen erforderlich. Alles in allem ergibt sich für jede im System gespeicherte Datei ein recht umfangreicher Satz von Verwaltungsinformationen.

Zur Aufnahme dieser Informationen werden in der E-Learning-Plattform sogenannte Virtuelle Lernobjekte (VLO) verwendet. Die Verwendung der Bezeichnung "virtuell" bezieht sich dabei auf die an dieser Stelle vorgenommene, strikte Trennung zwischen Daten und Metadaten. Ein VLO umfaßt ausschließlich die Metadaten und sonstigen Verwaltungsinformationen eines Lernobjektes, nicht jedoch dessen Inhalt, der separat in einem über HTTP zugänglichen Dateisystem abgelegt wird.[275] Es repräsentiert somit die eigentlichen Daten und definiert verschiedene Operationen, die auf den Metadaten vorgenommen werden können, ohne dabei auf die Inhalte zugreifen zu müssen.

Dies ermöglicht einerseits einen modularen Aufbau des Systems, da der Mechanismus zur Ablage der eigentlichen Daten aufgrund der Trennung von den Metadaten von deren Speicherung unabhängig gestaltet werden kann. Damit wäre es prinzipiell möglich, die in der E-Learning-Plattform realisierte Art und Weise der Datenspeicherung komplett durch eine andere Vorgehensweise zu ersetzen, ohne dafür gleichzeitig Änderungen an der Verwaltung der Metadaten vornehmen zu müssen.

Andererseits können die Verwaltungsinformationen damit ihrerseits unabhängig von den zugehörigen Inhalten in einer Datenbank verwaltet werden, was für die Operationen auf dieser Datenbank einen erheblichen Effizienzgewinn bedeutet, da die im Vergleich mit den Metadaten zumeist erheblich umfangreicheren Dateiinhalte in den internen Strukturen (z. B. relationalen Tabellen) der Datenbank nicht mit gespeichert und somit bei Anfragen auch nicht mit durchsucht werden müssen, wodurch sich jedes Mal ein erheblich geringeres zu verarbeitendes Datenvolumen ergibt.

Die hier vorgestellten Virtuellen Lernobjekte dienen bei der Verwaltung der Metadaten zur Abstraktion sowohl von den zugrundeliegenden Datenstrukturen der verwendeten Programmiersprache als auch von den Speicherstrukturen in der Datenbank. Damit spielen sie in diesem Fall die Rolle der in der Vorstellung der Systemarchitektur in Kapitel 4 beschriebenen Fassadenobjekte. Da Virtuelle Lernobjekte darüber hinaus in einem XML-basierten, plattform- und programmiersprachenunabhängigen Datenformat darstellbar sind, das über den in Kapitel 6 beschriebenen XML-RPC-Kommunikationsmechanismus übertragen werden kann[276], ist ihre Verwendbarkeit in beliebigen, künftig noch in die E-Learning-Plattform zu integrierenden Applikationen sichergestellt, selbst wenn diese in anderen Programmiersprachen geschrieben sein sollten.

Im Folgenden wird dargestellt, wie Virtuelle Lernobjekte aufgebaut sind und welche spezifischen Datenstrukturen für die Verwaltung von Metadaten sie enthalten können.

---

[275] Siehe Kapitel 5.7.
[276] Siehe Kapitel 6.3.

5.3.1 Aufbau eines Virtuellen Lernobjektes (VLO)

Virtuelle Lernobjekte werden im Rahmen der E-Learning-Plattform durch Objekte der Klasse *VirtualLearningObject* dargestellt. Diese Klasse definiert zunächst mehrere private Datenfelder, auf die über entsprechende Zugriffsmethoden operiert werden kann. Das erste dieser Datenfelder, *id*, enthält den eindeutigen Identifikationsschlüssel des Objektes, der in einer Variablen vom Typ *UniversalObjectIdentifier* enthalten ist.

Abbildung 5.2: Klassendiagramm der Klasse VirtualLearningObject (ohne Parameter und Rückgabetypen)

Zur Speicherung der Metadaten des durch ein VLO repräsentierten Lernobjektes dient die Metadatentabelle *metadata*, die durch ein Objekt der im nachfolgenden Abschnitt 5.3.2 dargestellten Klasse *MetadataTable* realisiert wird. In dieser Tabelle können verschiedene Typen von Metadaten gespeichert werden. Neben einfachen Methoden zum Setzen und Zurückgeben dieses Datenfeldes gibt es eine Methode zum Auslesen eines einzelnen Metadatums unter Angabe von dessen Namen sowie mehrere Methoden zum Hinzufügen, Ändern und Löschen von einzelnen Metadaten. Aufrufe dieser Methoden werden an die entsprechenden Methoden der Metadatentabelle weitergeleitet.

Ein weiteres Datenfeld, das den Namen *content* trägt, ist für Angaben darüber vorgesehen, wo die zu dem Lernobjekt gehörenden Inhalte gespeichert sind. Da diese Inhalte grundsätzlich in mehreren Sprachen vorliegen können, ist dieses Datenfeld als mehrsprachiger String ausgelegt. Die dafür benutzte Klasse *MultiLanguageString* wird in Abschnitt 5.3.3 ausführlich beschrieben. Derzeit wird dieses Datenfeld im Rahmen der E-Learning-Plattform allerdings nicht benutzt, da sich im jetzt verwendeten Speichersystem der Ablageort für die Inhalte eines VLO direkt aus seinem eindeutigen Identifier, dem Dateiformat (MIME Type) und der jeweiligen Sprache ergibt. Um jedoch künftigen Weiterentwicklungen des Systems auch die Nutzung anderer Speichermechanismen zu ermöglichen, die ggf. eine Angabe des Speicherortes im VLO benötigen könnten, wurde das Datenfeld *content* sowohl in der Klasse *VirtualLearningObject* vorgesehen als auch in der für die Speicherung von VLOs verwendeten Datenbankstruktur eingefügt.

Zur Verwaltung der Zugriffsrechte zum Lesen und Schreiben von Objekten dienen die beiden Datenfelder *readAccessRights* und *writeAccessRights*, in denen diese beiden Rechte getrennt voneinander auf Objektebene verwaltet werden. Da in beiden Fällen

mehrere Angaben erforderlich sein können, sind diese beiden Datenfelder vom Typ *MultiValueString*, der für die Verwaltung bestimmter Metadaten geschaffen wurde und eine gleichzeitig mit mehreren Werten belegbare, textuelle Angabe darstellt. Im Fall der Zugriffsrechte stellen die darin enthaltenen Texte jeweils die String-Repräsentation eines *UniversalObjectIdentifiers* (UOI) dar, der auf einen Benutzer oder eine Benutzergruppe verweist.

Das letzte Datenfeld namens *parents* enthält einen sogenannten Vektor von virtuellen Verzeichnissen, in denen das VLO enthalten ist. Der z. B. in der Programmiersprache Java vorhandene Datentyp Vektor kann als verkettete Liste von beliebigen Objekten betrachtet werden. Seine Zugriffsmethoden geben ihm jedoch auch die Eigenschaften eines Arrays variabler Größe, auf dessen Felder sehr effizient zugegriffen werden kann. Der Vektor *parents* enthält für jedes Elternverzeichnis dessen UOI, so daß das VLO zur Laufzeit in alle virtuellen Verzeichnisse, denen es angehört, eingeordnet werden kann, was eine flexible Navigation in den im System abgelegten Strukturen ermöglicht. Die virtuelle Verzeichnisstruktur der E-Learning-Plattform wird in Kapitel 5.4 beschrieben. Eltern können dem Vektor einzeln hinzugefügt oder auch daraus entfernt werden.

Neben den genannten Methoden zur Bearbeitung der hier aufgelisteten Datenfelder verfügt die Klasse VirtualLearningObject außerdem über zwei verschiedene Konstruktoren. Der erste dient zur Erzeugung eines komplett neuen VLO mit einem automatisch generierten, neuen *UniversalObjectIdentifier*, leerer Metadatentabelle, leerer Elternliste sowie leeren Datenfeldern für Zugriffsrechte und Ortsangabe des Inhaltes. Man beachte, daß vor dem Speichern des neuen Objektes in der Datenbank zumindest die Leserechte gesetzt sowie mindestens ein Eintrag in der Elternliste hinzugefügt werden muß, da das Objekt ansonsten von niemandem gelesen bzw. im System nicht wieder aufgefunden werden kann.[277] Beim Anlegen von neuen VLOs unter Benutzung des in Kapitel 7 vorgestellten Navigators erfolgen diese Schritte bereits automatisch.

Der zweite Konstruktor dient zur Rekonstruktion eines VLO aus seinen Datenfeldern und wird beispielsweise dazu verwendet, ein VLO aus seinen in der Datenbank gespeicherten Elementen zu erzeugen. Auch bei der Wiederherstellung von VLOs, die über den in Kapitel 6 beschriebenen XML-RPC-Kommunikationsmechanismus versendet wurden, wird dieser Konstruktor verwendet. Da das Objekt bereits in jedem Fall bereits bestanden hat, wird kein neuer UOI erzeugt. Der Konstruktor verlangt daher die Angabe des UOI dieses Objektes, der in den genannten Fällen entweder der Datenbank bekannt ist oder beim Datentransfer von Rechner zu Rechner mit übermittelt wurde.

Außerdem enthält die Klasse *VirtualLearningObject* noch drei weitere Methoden. Die erste, *hashCode*, dient zum Erzeugen eines geeigneten Hashcodes (zum Einsortieren in Hashtables). Mit der Methode *equals* kann überprüft werden, ob zwischen dem aktuell betrachteten VLO und einem anderen Gleichheit besteht, die nur dann gegeben ist, wenn sich bei gleicher ID auch die übrigen Datenfelder nicht unterscheiden. Nach der Änderung von Metadaten wäre ein VLO also beispielsweise nicht mehr äquivalent zu einer vorher in einer temporären Variablen abgelegten Kopie.

Die letzte Methode, *clone*, erzeugt eine vollständige Kopie eines VLO, d. h. auch die in seinen Datenfeldern enthaltenen Objekte werden mit allen ihren Inhalten kopiert und

[277] Durch eine Suche wäre es erst nach dem Eintragen mindestens eines Metadatums zugänglich, würde dann aber bei einer geeigneten Anfrage auch gefunden, ohne daß ein Eintrag in der Elternliste vorhanden ist. Um Objekte im System für Benutzer zugänglich zu machen, empfiehlt sich aber stets die Einordnung in mindestens ein virtuelles Verzeichnis.

nicht nur als Referenz weitergegeben. Das auf diese Weise neu erzeugte VLO verfügt allerdings noch nicht über einen *UniversalObjectIdentifier*, der ihm daher noch explizit zugewiesen werden muß. Der Grund für diese Einschränkung liegt darin, daß bei jedem Erzeugen eines neuen UOI der Zähler, über den innerhalb einer Installation des Systems der eindeutigen Objektschlüssel generiert wird, hochgezählt wird. Sollte das kopierte Objekt wieder verworfen oder aus anderen Gründen nicht gespeichert werden, würde sich eine unnötige Lücke in der Numerierung der erzeugten Objekte ergeben, was durch das zu einem späteren Zeitpunkt erfolgende Hinzufügen des UOI zum kopierten Objekt zumindest in einigen Fällen vermieden wird.

5.3.2 Metadatentabelle zur Speicherung beliebiger Metadaten

Die Speicherung von beliebigen Metadaten innerhalb eines Virtuellen Lernobjekts wird durch eine Metadatentabelle realisiert, deren Verhalten durch die Klasse *MetadataTable* definiert wird. Diese Klasse entspricht im Wesentlichen einer Hashtable, deren Einträge allerdings nur bestimmten Datentypen angehören dürfen, die als Inhalte von Metadatenfeldern zulässig sind, während es sich bei den Schlüsseln ausschließlich um Strings handeln darf, die den Namen der jeweiligen Felder entsprechen. Tatsächlich werden die Metadaten in der Klasse *MetadataTable* intern in einer privaten Hashtable verwaltet, auf die gemäß dem objektorientierten Prinzip der Datenkapselung nur über spezielle Methoden schreibend und lesend zugegriffen werden kann und die dabei sicherstellen, daß nur zulässige Datentypen in die Metadatentabelle eingetragen werden.

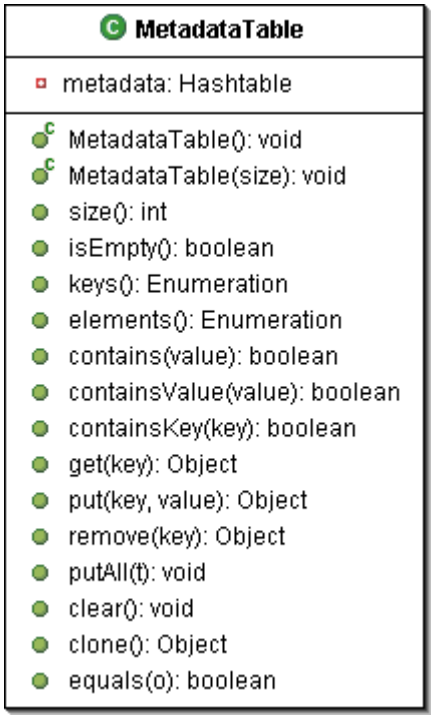

Abbildung 5.3: Klassendiagramm der Klasse MetadataTable (mit Parametern und Rückgabetypen)

Die Klasse *MetadataTable* verfügt über zwei verschiedene Konstruktoren, die beide jeweils eine leere Metadatentabelle zurückliefern und sich nur dadurch unterscheiden, daß im einen Fall eine Größe für die zu erzeugende Tabelle angegeben wird, während im anderen Fall eine anfängliche Tabellengröße von 30 Elementen voreingestellt wird. Dies sollte für die meisten Anwendungen ausreichen, solange nicht beispielsweise ein vollständiger Metadatensatz für Lernobjekte nach Maßgabe eines so umfangreichen Standards wie des LOM verwaltet werden muß.

Ansonsten entsprechen die Methoden der Klasse *MetadataTable* denen einer typischen Hashtable. Die Methode *size* liefert die augenblickliche Anzahl der Einträge in der Tabelle zurück, während mit *isEmpty* festgestellt werden kann, ob die Metadatentabelle leer ist. Die Methode *keys* liefert eine Aufzählung der Schlüssel, also der Namen aller ausgefüllten Metadatenfelder, zurück, während *elements* eine Aufzählung der Inhalte dieser Felder ausgibt. Mit *containsKey* kann festgestellt werden, ob ein bestimmter String als Schlüssel in der Tabelle enthalten ist, was gleichbedeutend damit ist, daß das entsprechend benannte Metadatum für das zugehörige VLO angegeben wurde, während mit *containsValue* bzw. kürzer mit *contains* überprüft werden kann, ob ein bestimmtes Objekt als Wert eines Metadatums in der Tabelle vorhanden ist.

Die Methode *get* liefert für einen bestimmten Schlüssel, also den Namen eines Feldes, den dafür in der Metadatentabelle gespeicherten Wert zurück. Umgekehrt kann mit der Methode *put* für einen anzugebenden Feldnamen ein Objekt in der Tabelle gespeichert werden, solange sein Format einem der für Einträge in Metadatenfeldern zulässigen Datentypen entspricht. Anderenfalls wird ein von dieser Methode ein Ausnahmefehler (Exception) vom Typ IllegalValueTypeException erzeugt (siehe Abbildung 5.4), der von aufrufenden Klassen ggf. abgefangen werden muß. Die Methoden zum Hinzufügen von Metadaten, die in der Klasse *VirtualLearningObject* definiert sind, berücksichtigen dies bereits.

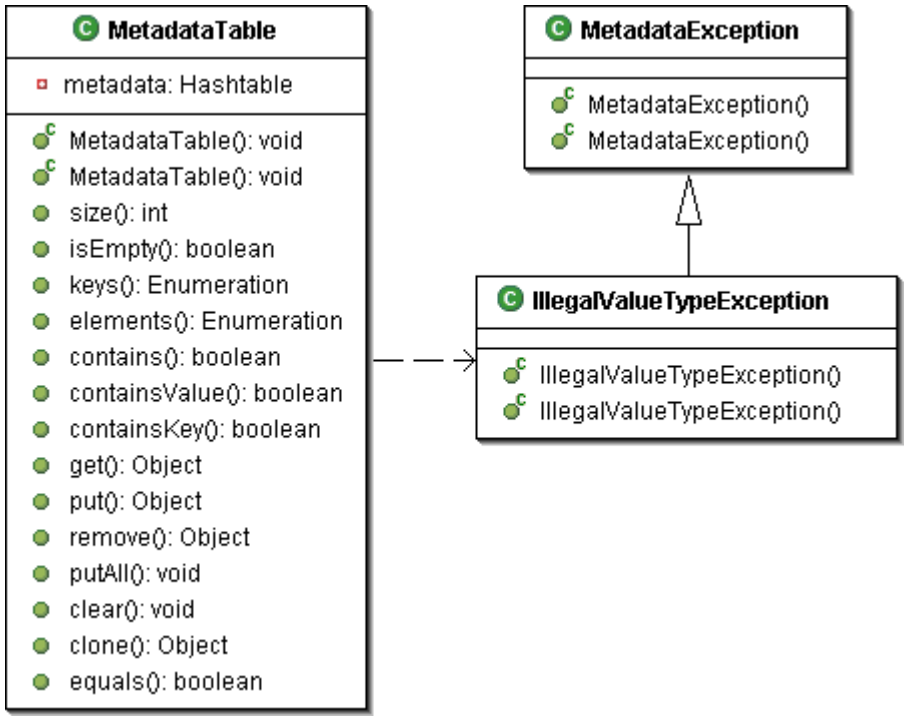

Abbildung 5.4: Die Klasse MetadataTable erzeugt bei der Angabe von ungültigen Metadatentypen Ausnahmefehler (Exceptions)

Eine abgekürzte Speichermöglichkeit für mehrere Metadaten auf einmal bietet die Methode putAll, die als Parameter eine Datenstruktur entsprechend der Java-Klasse *Map* erwartet und alle in dieser Struktur enthaltenen Metadaten in die Tabelle überträgt, sofern deren jeweiliger Datentyp zulässig ist. Mit *remove* können Elemente einzeln wieder aus der Metadatentabelle entfernt werden, während der Aufruf der Methode *clear* die komplette Tabelle leert.

Die Klasse *MetadataTable* wird durch zwei weitere Methoden vervollständigt. Eine von ihnen, *equals*, dient dazu, eine andere *MetadataTable* auf Gleichheit mit der aktuell betrachteten zu überprüfen, was anhand der Gleichheit aller in beiden Tabellen enthaltenen Metadaten festgestellt wird. Die letzte Methode, *clone*, erzeugt eine vollständige Kopie der Tabelle und aller darin enthaltenen Metadaten, d. h. auch die in den einzelnen Metadatenfeldern enthaltenen Objekte werden mit allen ihren Inhalten kopiert und nicht nur als Referenz weitergegeben. Dies ist beispielsweise dann von Bedeutung, wenn ein neues Lernobjekt sämtliche Metadaten eines bereits bestehenden Objektes zugewiesen bekommen soll. Im Fall von anschließenden Änderungen der Metadaten eines der beiden Objekte würden diese Änderungen sich anderenfalls auf die Metadatensätze beider Objekte auswirken, was zu unerwünschten Nebenwirkungen führen würde.

Im Folgenden werden die Datentypen beschrieben, die für das Einspeichern als Werte in die Metadatentabelle im Rahmen der E-Learning-Plattform zulässig sind. Dabei werden zunächst zwei spezifische Datentypen definiert, die für die Speicherung mehrsprachiger Strings sowie zur Verwaltung von textuellen Metadaten, für die gleichzeitig mehrere Werte angegeben werden können oder müssen, geschaffen wurden. Im Anschluß daran wird kurz darauf eingegangen, welche anderen Datentypen außerdem noch als Werte für Metadatenfelder erlaubt sind.

5.3.3 Mehrsprachige Metadaten: Die Klasse MultiLanguageString

Für den Einsatz der E-Learning-Plattform in einem internationalen Umfeld ist nicht nur eine Speicherung der Inhalte von Lernobjekten in verschiedenen Sprachen unerläßlich, sondern auch Metadaten müssen für das gleiche Objekt in unterschiedlichen Sprachen abgelegt werden können. Zu diesem Zweck wurde die Klasse *MultiLanguageString* entworfen, deren Objekte mehrsprachige Texte darstellen. Die verschiedenen Versionen des Textes werden intern in einer privaten Hashtable verwaltet. Jeder Eintrag in dieser Hashtable verfügt dabei über einen Schlüssel, der den Namen der Sprache repräsentiert, in der der enthaltene String geschrieben wurde.

Die verschiedenen Sprachen werden dabei durch abgekürzte Bezeichnungen benannt, die dem Internet-Standard zur Identifikation von Sprachen folgen, der in RFC 3066 definiert wurde.[278] Die nach diesem Standard definierten Bezeichnungen für Sprachen werden von der Internet Assigned Numbers Authority (IANA)[279] verwaltet. Demnach steht beispielsweise die Abkürzung "de" für einen deutschen Text, während "en" einen englischen Text bezeichnet. Entsprechend kennzeichnen "es", "it", "fr" und "ru" Texte in spanischer, italienischer, französischer und russischer Sprache usw. Die Anzahl der im Rahmen der E-Learning-Plattform verwendbaren Sprachen ist nicht begrenzt.

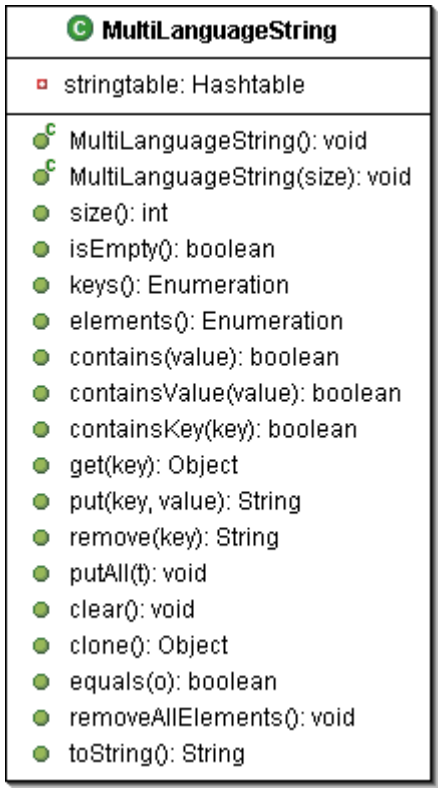

Abbildung 5.5: Klassendiagramm der Klasse MultiLanguageString (mit Parametern und Rückgabetypen)

Die Klasse *MultiLanguageString* verfügt über zwei verschiedene Konstruktoren, die beide jeweils einen leeren mehrsprachigen String zurückliefern und sich nur dadurch unterscheiden, daß im einen Fall eine Größe für die zu erzeugende Hashtable angegeben wird, während im anderen Fall eine initiale Größe von 4 Elementen voreingestellt wird. Dieser Wert wurde gewählt, weil in E-Learning-Systemen wie z. B. Monist, das im Wesentlichen zweisprachig (deutsch / englisch) ist, die Erstellung von Objekten in noch mehr Sprachen nur selten vorkommt. Wenn dennoch eine größere Anzahl von Sprachen verwaltet werden muß, kann dies jederzeit berücksichtigt werden, da die Tabelle sich selbst bei Bedarf automatisch vergrößert.

[278] Dieser "Internet Best Current Practices" Quasi-Standard kann eingesehen werden unter der Adresse: http://www.ietf.org/rfc/rfc3066.txt.
[279] Homepage: http://www.iana.org.

Die Methoden der Klasse *MultiLanguageString* entsprechen im Großen und Ganzen denen einer typischen Hashtable, mit der Einschränkung, daß sowohl als Schlüssel als auch als Werte von Einträgen in der Tabelle ausschließlich Strings zulässig sind. Die Methode *size* liefert die augenblickliche Anzahl der verschiedensprachlichen Einträge in dem *MultiLanguageString* zurück, während mit *isEmpty* festgestellt werden kann, ob die Sprachentabelle leer ist.

Die Methode *keys* liefert eine Aufzählung der Schlüssel, also der Namen aller in diesem *MultiLanguageString* aktuell verwendeten Sprachen, zurück, während *elements* eine Aufzählung der Inhalte der für diese Sprachen gespeicherten Datenfelder ausgibt. Mit *containsKey* kann festgestellt werden, ob ein bestimmter String als Schlüssel in der Tabelle enthalten ist, was gleichbedeutend damit ist, daß ein Text für die entsprechende Sprache vorhanden ist. Mit *containsValue* bzw. kürzer mit *contains* kann in gleicher Weise überprüft werden, ob ein bestimmter String als Version des Textes für eine der verwalteten Sprachen in der Tabelle vorhanden ist.

Die Methode *get* liefert für einen bestimmten Schlüssel, also für den Bezeichner einer Sprache, den für die entsprechende Sprache in der Hashtable gespeicherten Text zurück. Umgekehrt kann mit der Methode *put* für einen anzugebenden Sprachbezeichner ein String in der Tabelle gespeichert werden. Eine abgekürzte Speichermöglichkeit für Texte in mehrere Sprachen auf einmal bietet die Methode putAll, die als Parameter eine Datenstruktur entsprechend der Java-Klasse *Map* erwartet und alle in dieser Struktur enthaltenen Texte in die Tabelle überträgt. Mit *remove* können die Einträge für einzelne Sprachen wieder aus dem *MultiLanguageString* entfernt werden, während beim Aufruf der Methoden *clear* bzw. *removeAllElements* die komplette Datenstruktur geleert wird und damit die Einträge für alle verwendeten Sprachen gelöscht werden.

Die Klasse *MultiLanguageString* wird vervollständigt durch drei weitere Methoden. Die erste von ihnen, *equals*, dient dazu, einen anderen *MultiLanguageString* auf Gleichheit mit dem aktuell betrachteten zu überprüfen, was anhand der Gleichheit der in beiden Objekten enthaltenen Hashtables mitsamt den darin enthaltenen, mehrsprachigen Texten festgestellt wird. Die Methode *toString* liefert in Abhängigkeit von der vom aktuell eingeloggten Benutzer eingestellten Sprache den für diese Sprache in der *MetadataTable* gespeicherten String zurück. Falls für diese Sprache kein Text gespeichert ist, wird statt dessen als Voreinstellung der englischsprachige String ausgegeben, falls ein solcher in der Tabelle vorhanden ist.

Die Methode *clone* erzeugt eine vollständige Kopie des mehrsprachigen Strings und aller darin enthaltenen Texte für die einzelnen Sprachen, d. h. auch die in den einzelnen Datenfeldern der Hashtable enthaltenen Strings werden jeweils kopiert und nicht nur als Referenz weitergegeben. Dies ist beispielsweise dann von Bedeutung, wenn beim Kopieren eines Objektes mitsamt seiner Metadaten ein *MultiLanguageString* in der *MetadataTable* des neuen Objektes sämtliche textuellen Inhalte eines mehrsprachigen Strings zugewiesen bekommen soll, der bereits das ursprüngliche Objekt beschreibt. Würden bei diesem Vorgang die einzelnen Strings nicht kopiert, sondern statt dessen nur Referenzen an den neu erzeugten *MultiLanguageString* übergeben, dann würden sich im Fall von anschließenden Änderungen des betreffenden Metadatums in einem der beiden Objekte diese Änderungen auf die Metadatensätze beider Objekte auswirken, was zu unerwünschten Nebenwirkungen führen würde. Dies könnte sogar zum Verlust von Inhalten führen, sofern ein Speichermechanismus verwendet wird, der das Datenfeld *content* der Klasse *VirtualLearningObject* benutzt, bei dem es sich ebenfalls um einen MultiLanguageString handelt.

Typische Anwendungsfälle für die Klasse *MultiLanguageString* sind alle potentiell mehrsprachigen, textuellen Metadaten, beispielsweise die Elemente Titel (*DC.TITLE*), Beschreibung (*DC.DESCRIPTION*) oder Rechtliche Bestimmungen (*DC.RIGHTS*) aus dem Dublin Core Metadatensatz. Man beachte, daß für die Länge der Strings für die einzelnen Sprachen keine Begrenzung vorgegeben ist. Für eine effiziente Verwaltung der Metadaten empfiehlt es sich allerdings in jedem Fall, möglichst kurze textuelle Beschreibungen zu verwenden, weil dadurch tendenziell eine schnellere Suche in den in der Datenbank enthaltenen Metadaten möglich ist.

Im Rahmen der E-Learning-Plattform werden mehrsprachige Strings derzeit auch für die Speicherung von Informationen über die für die verschiedenen Sprachen in das System hochgeladenen Dateien verwendet. Die Dateigröße sowie der MIME Type als technisches Datenformat (*DC.FORMAT*) werden in den Metadaten der einzelnen VLOs in *MultiLanguageStrings* gespeichert, um diese Informationen in Abhängigkeit von der jeweils eingestellten Sprache des Benutzers anzeigen und zur Verwaltung und Anzeige der entsprechenden Inhalte verwenden zu können. Es ist nicht ganz auszuschließen, daß zukünftig noch zu entwickelnde, andere Speichermechanismen eventuell eine andere Verwaltungsmethode für diese Informationen vorsehen könnten. Der modulare Aufbau des Systems ermöglicht dies prinzipiell, da die Speicherung der Metadaten lediglich als den Anwendungen zur Verfügung gestellter Dienst konzipiert ist und die Verwendung anderer Möglichkeiten offen läßt.

5.3.4 Mehrwertige Metadaten: Die Klasse MultiValueString

Für die Verwaltung einiger Typen von Metadaten ist es erforderlich, zu einem Eintrag in der Metadatentabelle gleichzeitig mehrere, gleichwertige Strings angeben zu können. Dies ist beispielsweise bei Stichwörtern (keywords) der Fall. Auch für die Angabe von an der Erstellung eines Objektes beteiligten Personen (*DC.CONTRIBUTORS*) werden oftmals mehrere gleichberechtigte Einträge benötigt, wobei in der E-Learning-Plattform in diesem Fall die einzelnen Werte dieses Metadatums jeweils die String-Repräsentation eines *UniversalObjectIdentifiers* (UOI) darstellen, der auf einen bestimmten Benutzer verweist. Ebenso wird auch die Verwaltung der Zugriffsrechte für das Schreiben und Lesen von Objekten in der Klasse *VirtualLearningObject* gehandhabt.[280]

Für die Speicherung derartiger mehrwertiger Strings wurde die Klasse *MultiValueString* entworfen. Die Objekte dieser Klasse verwalten eine beliebige Menge von Strings, die so betrachtet werden, als ob sie alle zur gleichen Kategorie gehören und unabhängig voneinander auf die gleiche Weise behandelt werden sollen. Zu diesem Zweck offeriert die Klasse *MultiValueString* verschiedene Möglichkeiten für Iterationen über den in ihren Instanzen gespeicherten Strings sowie zahlreiche Methoden zum Hinzufügen, Bearbeiten und Löschen von einzelnen Werten.

Die Liste der in einem *MultiValueString* enthaltenen Elemente wird intern durch einen privaten Vektor repräsentiert, auf den gemäß der objektorientierten Datenkapselung nur über entsprechende Methoden zugegriffen werden kann. Der z. B. in der Programmiersprache Java vorhandene Datentyp Vektor kann als verkettete Liste beliebiger Objekten betrachtet werden. Seine Zugriffsmethoden geben ihm jedoch auch die Eigenschaften eines Arrays variabler Größe, auf dessen Felder sehr effizient zugegriffen werden kann. Ein *MultiValueString* stellt praktisch einen typisierten Vektor von einzelnen Strings dar.

[280] Siehe Abschnitt 5.3.1.

Die Klasse *MultiValueString* verfügt über drei verschiedene Konstruktoren. Die ersten beiden erzeugen je einen leeren *MultiValueString* und unterscheiden sich nur dadurch, daß im einen Fall eine initiale Größe für den zu erzeugenden Vektor angegeben wird, während im anderen Fall ohne eine entsprechende Mengenangabe eine Ausgangsgröße von 10 Elementen voreingestellt wird. Dieser Wert wurde gewählt, weil in den oben beschriebenen Fällen, in denen ein Objekt der Klasse *MultiValueString* eingesetzt wird, erfahrungsgemäß nur selten mehr als 10 Elemente benötigt werden. Andererseits ist die Vergrößerung eines Vektors über seine initiale Kapazität hinaus stets eine relativ aufwendige Operation, so daß die voreingestellte Größe von 10 Elementen einen guten Kompromiß für diejenigen Fälle darstellt, in denen eine initiale Größe nicht explizit angegeben wird.

Der dritte Konstruktor hingegen erzeugt einen neuen *MultiValueString* aus einem einzelnen, vorgegebenen String. Dieser wird als vorerst einziges Element des neuen *MultiValueStrings* betrachtet, dem somit eine initiale Größe von einem Element zugewiesen wird, wenngleich er natürlich später um weitere Strings erweitert werden kann. Dieser Konstruktor stellt für den Fall, daß beim Erstellen des *MultiValueStrings* genau ein einzelnes Element bekannt ist, eine Abkürzung gegenüber dem Erstellen und dem anschließenden Hinzufügen des einen Strings dar. Außerdem wird zunächst kein zusätzlicher Speicherplatz für weitere Elemente belegt, so daß der neue, einelementige *MultiValueString* nur einen minimalen Raum im Arbeitsspeicher benötigt. Nach Möglichkeit sollte in derartigen Fällen stets ausschließlich dieser Konstruktor verwendet werden.

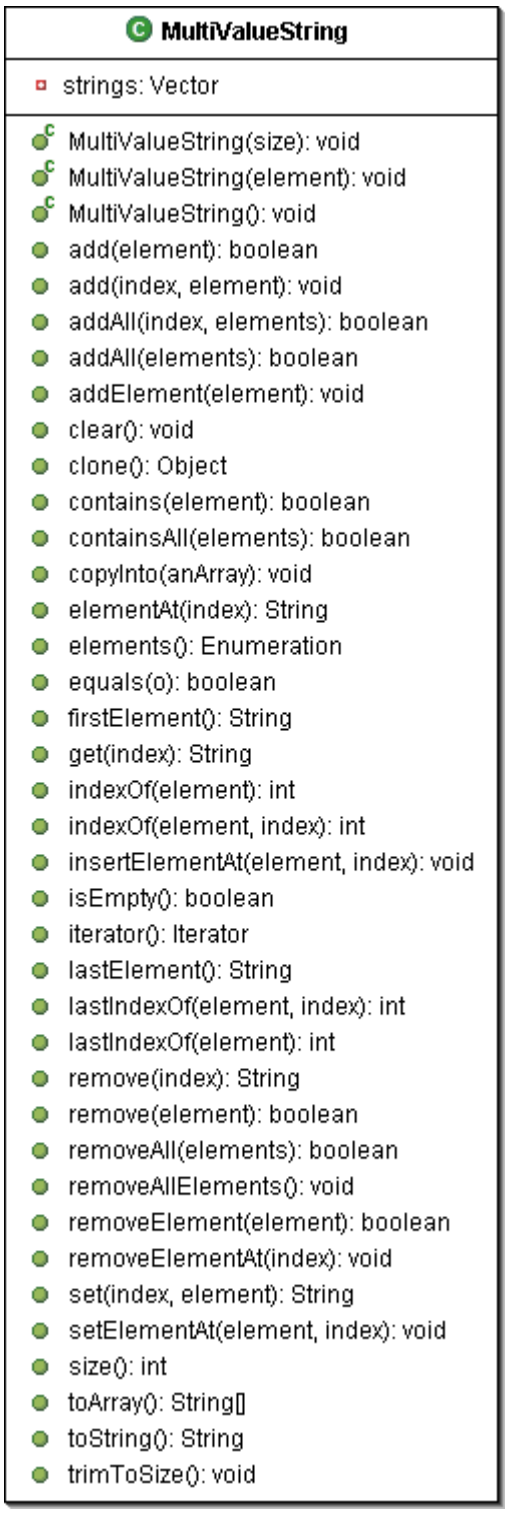

Abbildung 5.6: Klassendiagramm der Klasse MultiValueString (mit Methoden-Parametern und Rückgabetypen)

Objekte der Klasse *MultiValueString* können im Prinzip ähnlich wie Objekte der Klasse Vektor behandelt werden, da sie quasi über die gleichen Methoden verfügen, mit dem einzigen Unterschied, daß die Methoden der Klasse *MultiValueString* überall dort ausschließlich auf Strings operieren, wo die entsprechenden Methoden eines Vektors beliebige Objekte verarbeiten können. Auf diese Weise wird sichergestellt, daß keine unzulässigen Datentypen anstelle der Strings in einem *MultiValueString* gespeichert werden können, was anderenfalls zu erheblichen Problemen bei der Verarbeitung der enthaltenen Texte führen würde. Abbildung 5.6 zeigt ein Diagramm der Klasse *MultiValueString* mit allen von ihr zur Verfügung gestellten Methoden, deren Parametern und Rückgabetypen.

Die Klasse *MultiValueString* stellt insgesamt 35 Methoden zur Verwaltung der darin enthaltenen Strings zur Verfügung. Es gibt Methoden zum Hinzufügen von Strings am Ende (*add* bzw. *addElement* sowie *addAll* für eine Liste von mehreren Strings) oder an einer beliebigen Position (*add* bzw. *insertElementAt* sowie *addAll* für mehrere Strings) des enthaltenen Vektors sowie zum Löschen von einzelnen (*remove, removeElement* und *removeElementAt*), mehreren (*removeAll*) oder allen (*removeAllElements* bzw. *clear*) Strings. Mit *set* bzw. *setElementAt* kann der String an einer bestimmten Position durch einen neuen ersetzt werden. Die Methode *trimToSize* reduziert die Kapazität eines *MultiValueStrings* auf eine Größe, die der Anzahl seiner Elemente entspricht. Diese kann durch einen Aufruf der Methode *size* ermittelt werden. Ob der MultiValueString leer ist, kann zusätzlich mit der Methode *isEmpty* festgestellt werden.

Der Zugriff auf einzelne Elemente kann mit verschiedenen Methoden erfolgen. Die Methode *firstElement* liefert das erste, *lastElement* das letzte Element zurück. Mit *get* bzw. *elementAt* kann ein Element an einer bestimmten Indexposition gelesen werden. Ob ein bestimmter String im *MultiValueString* enthalten ist, kann mit *contains* ermittelt werden, für mehrere Strings steht entsprechend die Methode *containsAll* zur Verfügung. Mit den Methoden *indexOf* bzw. *lastIndexOf* kann die Position des ersten und letzten Vorkommens eines Strings im gesamten *MultiValueString* oder nach bzw. vor einer bestimmten Position festgestellt werden. Die Methode *iterator* liefert einen Iterator, mit dem über alle Elemente des MultiValueStrings iteriert werden kann, während *elements* eine Aufzählung (enumeration) der Elemente zurückliefert. Darüber hinaus können die enthaltenen Strings mit *toArray* in ein eindimensionales Feld (array) ausgegeben oder mit *copyInto* in ein bereits existierendes Feld hineingeschrieben werden.

Die Klasse *MultiValueString* wird durch drei weitere Methoden vervollständigt. Die erste von ihnen, *equals*, dient dazu, einen anderen *MultiValueString* auf Gleichheit mit dem aktuell betrachteten zu überprüfen, was anhand der Gleichheit der in beiden Objekten enthaltenen Vektoren mitsamt den darin enthaltenen Strings festgestellt wird. Die Methode *clone* erzeugt eine vollständige Kopie des mehrwertigen Strings und aller darin enthaltenen Elemente, d. h. auch die in den einzelnen Datenfeldern des Vektors enthaltenen Strings werden jeweils kopiert und nicht nur als Referenz weitergegeben. Die Methode *toString* liefert eine Repräsentation des gesamten *MultiValueString* in Form eines einzelnen Strings. Mit diesen Methoden sind für jeden Zweck hinreichende Möglichkeiten zum Zugriff auf die einzelnen Elemente des *MultiValueStrings* gegeben.

5.3.5 Weitere zulässige Datentypen für Metadaten

Neben den in den beiden vorangegangenen Abschnitten vorgestellten Datentypen für mehrsprachige sowie mehrwertige Strings, die speziell für die E-Learning-Plattform entworfen wurden, ist noch eine Reihe von weiteren Datentypen als Werte für die Inhalte von Metadatenfeldern zulässig. Dies betrifft zum einen normale Strings, die für textuelle Metadaten verwendet werden können, für die weder mehrere gleichberechtigte Werte gespeichert werden müssen noch eine mehrsprachige Verwaltung erforderlich ist. Beispiele hierfür sind der Status, der zwar in der Benutzeroberfläche in verschiedenen Sprachen angezeigt werden kann, aber nicht notwendigerweise mehrsprachig verwaltet werden muß, sowie der Objekttyp (*DC.TYPE*), der gleichermaßen sprachunabhängig zur Unterscheidung und Klassifizierung von Objekten verwendet werden kann. Auch das Datum (*DC.DATE*) wird im Rahmen der E-Learning-Plattform in einem einsprachigen und nicht mehrwertigen String gespeichert, der allerdings nie manuell eingegeben, sondern stets automatisch konform zum Datumsstandard ISO 8601 erzeugt wird.

Neben den Strings sind außerdem zum einen boolesche Werte (*True*, *False*) und zum anderen diverse numerische Datentypen als Inhalte von Metadatenfeldern zulässig. Die Menge der verwendbaren numerischen Typen entspricht dabei den Subtypen des abstrakten Datentyps *Number* in der Programmiersprache Java. Als unterschiedliche Ganzzahltypen sind *Byte*, *Short*, *Integer* und *Long* sowie die Klasse *BigInteger*, die ganze Zahlen beliebiger Länge speichern kann, erlaubt. Für Fließkommazahlen stehen die Klassen *Float* und *Double* sowie die zur Speicherung von beliebigen Werten befähigte Klasse *BigDecimal* zur Verfügung. Abbildung 5.7 gibt einen Überblick über alle Datentypen, die als Werte in der *MetadataTable* eines Virtuellen Lernobjektes verwendet werden können.

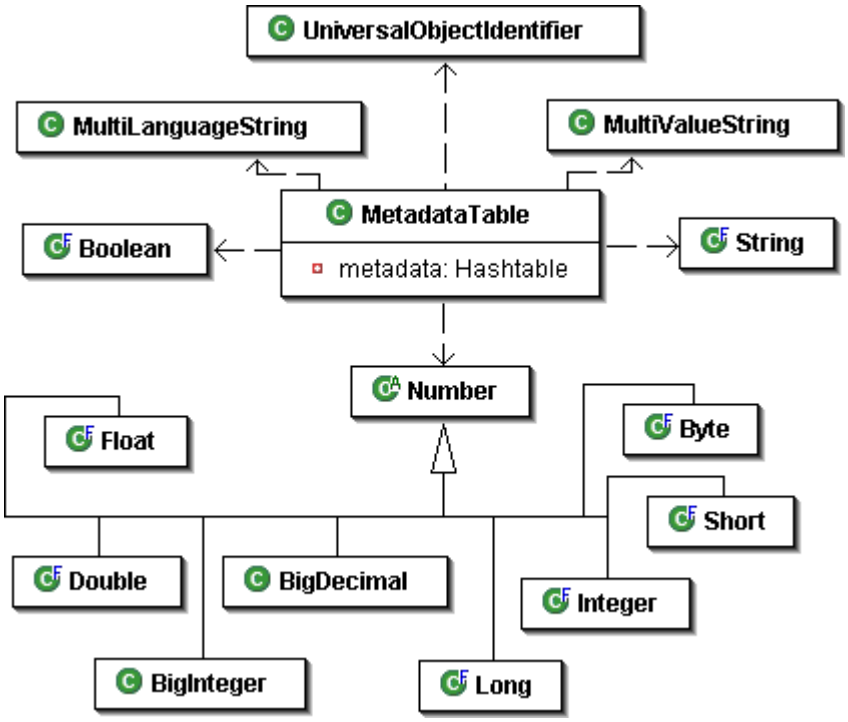

Abbildung 5.7: Zulässige Datentypen für Elemente in der MetadataTable

5.4 Flexible Ablagestruktur durch Virtuelle Verzeichnisse

Die im vorangehenden Abschnitt vorgestellten Virtuellen Lernobjekte repräsentieren im Rahmen der E-Learning-Plattform einzelne Lernobjekte und dienen zur Verwaltung von deren Metadaten und indirekt auch deren Inhalten. Damit bilden sie jedoch nur einen Teil der gesamten Datenstruktur, die zur Laufzeit des Programmes benötigt wird und im Sinne der in Kapitel 4 vorgestellten Fassadenobjekte den einzelnen Anwendungen zur Verfügung gestellt wird. Die einzelnen Lernobjekte stellen in der Regel keine isolierten, unabhängigen Objekte dar, sondern müssen stets in ihrem Kontext betrachtet werden. Die einzelnen Mediendateien gehören beispielsweise zu bestimmten Lehreinheiten, die wiederum in einem oder mehreren übergeordneten Kursen verwendet werden können. Diese können ihrerseits in unterschiedliche Fachbereiche eingeordnet sein sowie nach verschiedenen Kriterien in Verzeichnishierarchien organisiert werden. Die technische Grundlage für alle diese Elemente einer strukturierten Ablage von Lernobjekten bilden in der E-Learning-Plattform die sogenannten Virtuellen Verzeichnisse.

Der Begriff "virtuelle Verzeichnisse" bedeutet dabei, daß sich jedes Objekt gleichzeitig in mehreren verschiedenen Verzeichnissen befinden kann, vollkommen unabhängig von der physikalischen Speicherstruktur. Tatsächlich existieren die Virtuellen Verzeichnisse der E-Learning-Plattform überhaupt nicht als physikalische Verzeichnisse im Sinne des jeweils verwendeten Betriebssystems. Sie sind vielmehr eigenständige Lernobjekte, die in der Datenbank abgelegt werden können und zur Laufzeit des Systems durch Objekte der Klasse *VirtualDirectory* repräsentiert werden.

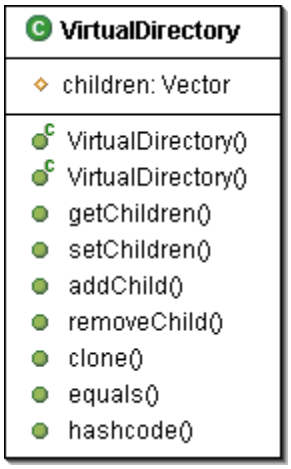

Abbildung 5.8: Klassendiagramm der Klasse VirtualDirectory (ohne Parameter und Rückgabetypen)

Diese Klasse definiert ein privates Datenfeld namens *children* zur Verwaltung von Virtuellen Lernobjekten, die im jeweiligen Verzeichnis enthalten sind. Diese Kindobjekte sind genau wie die Elternobjekte eines Virtuellen Lernobjektes in einem Vektor gespeichert. Dieser enthält für jedes Kindobjekt dessen UOI, so daß zur Laufzeit alle Objekte, die in das Virtuelle Verzeichnis gehören, dort eingeordnet werden können. Auf diese Weise ist eine flexible Anzeige der Verzeichnisinhalte möglich, die es beispielsweise erlaubt, eingeloggten Benutzern dynamisch nur diejenigen Kindobjekte eines Verzeichnisses anzuzeigen, die er anhand der für ihn geltenden Zugriffsrechte für das Lesen der betreffenden Objekte auch tatsächlich sehen darf.

Auf dieses Datenfeld kann über eine Reihe von verschiedenen Methoden zugegriffen werden. Kindobjekte können dem Vektor mit *addChild* einzeln hinzugefügt oder auch mit *removeChild* einzeln daraus entfernt werden. Die Methode *getChildren* liefert den enthaltenen Vektor zurück, während mit *setChildren* ein neuer Vektor von Kindobjekten angegeben werden kann.

In den Vektor der Kindobjekte eines Virtuellen Verzeichnisses können jedoch nicht nur Virtuelle Lernobjekte, die einzelne Mediendateien darstellen, eingeordnet werden. Zum Aufbau einer hierarchischen Verzeichnisstruktur ist es vielmehr erforderlich, daß auch die Virtuellen Verzeichnisse selbst ihrerseits in andere Virtuelle Verzeichnisse eingefügt werden können. Natürlich bedeutet dies umgekehrt ebenfalls, daß auch Virtuelle Verzeichnisse in gleicher Weise wie Virtuelle Lernobjekte andere Virtuelle Verzeichnisse als Elternobjekte haben können. Da für Virtuelle Verzeichnisse ebenfalls ein eindeutiger Identifizierungsschlüssel, eigene Metadaten sowie Zugriffsrechte zum Schreiben und Lesen verwaltet werden müssen, ist es eine naheliegende Entscheidung, sie als Unterklasse der Virtuellen Lernobjekte zu definieren, die als zusätzliche Eigenschaft Kindobjekte speichern kann. Daher wurde in der E-Learning-Plattform die Klasse *VirtualDirectory* als Unterklasse der Klasse *VirtualLearningObject* konstruiert, wie die nebenstehende Abbildung 5.9 verdeutlicht.

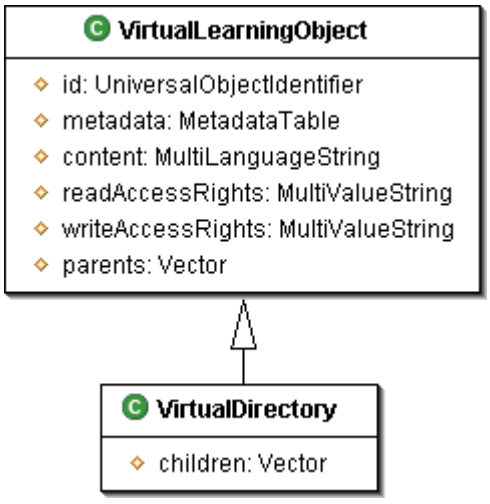

Abbildung 5.9: Die Klasse VirtualDirectory ist eine Subklasse der Klasse VirtualLearningObject

Virtuelle Verzeichnisse sind also ein spezieller Typ von Virtuellen Lernobjekten. Durch sie können in der E-Learning-Plattform für verschiedene Zwecke flexible Ablageformen für Lernobjekte geschaffen werden, die im Folgenden ausführlich betrachtet werden.

5.4.1 Eltern-Kind-Beziehungen zwischen Lernobjekten

Die erste Aufgabe von Virtuellen Verzeichnissen ist die Abbildung von Eltern-Kind-Beziehungen zwischen Lernobjekten, die einer herkömmlichen Verzeichnisstruktur entsprechen. Wie bereits dargelegt wurde, verfügen Virtuelle Verzeichnisse über eine Liste von Kindobjekten, während in allen Virtuellen Lernobjekten generell eine Liste von Elternobjekten gespeichert wird. Unabhängig voneinander kann also sowohl aus den im Kindobjekt als auch aus den im Elternobjekt gespeicherten Informationen eine Eltern-Kind-Beziehung abgeleitet werden. Diese Beziehung ist also nicht als einseitige, gerichtete Abhängigkeit zu verstehen, sondern als beiderseitige Zusammengehörigkeit der Objekte, die von beiden Seiten aus zugänglich ist und bei einer Navigation in der virtuellen Verzeichnisstruktur in beide Richtungen verfolgt werden kann.

Dabei werden jedoch auch die Zugriffsrechte zum Lesen von Objekten berücksichtigt, die für den jeweils eingeloggten Benutzer gelten und die in den dafür vorgesehenen Datenfeldern der Elternobjekte und der Kindobjekte gespeichert sind. Hierbei kommt die Tatsache zum Tragen, daß Virtuelle Verzeichnisse ebenfalls Virtuelle Lernobjekte sind und somit auch über die entsprechenden Datenfelder für Zugriffsrechte verfügen. Ein Lernobjekt, egal ob es sich dabei um ein Verzeichnis handelt oder nicht, ist stets nur für diejenigen Benutzer sichtbar, denen das Leserecht auf diesem Objekt eingeräumt wurde. Ist dies nicht der Fall, so wird das Objekt weder in einem ihm übergeordneten Verzeichnis als darin enthalten angezeigt, noch ist es als Verzeichnis von einem in ihm selbst enthaltenen Objekt aus über eine Navigation entlang einer Eltern-Kind-Beziehung zugänglich, da es bei der Betrachtung jenes Objektes unberechtigten Benutzern nicht als Elternobjekt angezeigt wird.

5.4.2 Klassifikation von Objekten und Einordnung in Objekthierarchien

Virtuelle Verzeichnisse können darüber hinaus zur thematischen Klassifikation von Objekten und zu ihrer fachlichen Einordnung gemäß unterschiedlicher Taxonomien dienen. Die virtuelle Verzeichnisstruktur ermöglicht die Abbildung hierarchischer Strukturen innerhalb eines Fachgebietes, beispielsweise im Rahmen einer Aufgliederung in dessen verschiedene Fachrichtungen, deren untergeordnete Teilbereiche, innerhalb dieser in die ihnen zugehörigen Spezialgebiete bis hin zu einzelnen Forschungsgegenständen. Jedes einzelne Lernobjekt kann gemäß seiner Position innerhalb dieser Hierarchie an einer entsprechenden Stelle eingeordnet werden, wobei die einzelnen Hierarchieebenen durch passend benannte Virtuelle Verzeichnisse dargestellt werden.

Eine Einordnung nach diesem Schema bedeutet jedoch keine starre Festlegung der Zugehörigkeit eines Objektes zu einem bestimmten Bereich, die seine Zugehörigkeit zu einem anderen Thema zur gleichen Zeit ausschließen würde. Die besondere Eigenschaft der virtuellen Verzeichnisstruktur, daß jedes Objekt zur gleichen Zeit mehrere Eltern haben kann, ermöglicht vielmehr eine flexible Einsortierung von Objekten an allen erforderlichen Stellen in der Hierarchie sowie ihre gleichzeitige Klassifikation nach mehreren Ordnungsschemata.

Eine bestimmte Lehreinheit z. B. aus dem Bereich Neuroinformatik kann auf diese Weise einerseits über den Fachbereich Informatik zugänglich sein, in den der Bereich Neuroinformatik als Teildisziplin der Informatik in Form eines Virtuellen Verzeichnisses eingeordnet ist. Andererseits könnte man die gleiche Lehreinheit ggf. auch über das Fach Biologie finden, wenn im Gebiet Neurobiologie die Disziplin Neuroinformatik als

Forschungsbereich im System enthalten ist und damit an einer weiteren Stelle in der virtuellen Verzeichnisstruktur auftaucht. Gleichzeitig könnten aber auch noch andere Ordnungsprinzipien im System verwendet werden. Beispielsweise könnten in einem weiterer Zweig der Verzeichnisstruktur die Inhalte des Systems nach den Standorten der dafür verantwortlichen Einrichtungen sortiert werden. Über die zuständige Universität, nachfolgend über die entsprechende Fakultät, den Lehrstuhl bzw. die Arbeitsgruppe und schließlich die Veranstaltung, für welche die Lehreinheit erstellt wurde, kann diese dann ebenfalls zugänglich gemacht werden. Falls das gleiche Objekt auch für eine andere Lehrveranstaltung (möglicherweise an einer anderen Universität) benutzt wurde, findet man es auch in dem virtuellen Verzeichnis, das dieser Veranstaltung entspricht.

Bezieht sich der Inhalt derselben Lehreinheit außerdem auf die Eigenschaften eines bestimmten Organismus, kann sie bei Bedarf zusätzlich in ein biologisches System zur Klassifikation der Arten eingeordnet werden, ohne daß sich dies auf die Sortierung nach den zuvor genannten anderen Schemata auswirken würde. Jedes Lernobjekt kann somit an genau den Stellen abgelegt werden, an denen es jeweils benötigt wird bzw. gemäß verschiedenen Verfahren einsortiert werden muß. Objekte können sich also unabhängig voneinander in verschiedenen Verzeichnissen befinden. Da Eltern-Kind-Beziehungen als Listen verwaltet werden, die jeweils vollkommen gleichberechtigte Verweise auf die *UniversalObjectIdentifier* der anderen Objekte enthalten, können Lernobjekte genauso, wie sie separat verschiedenen Virtuellen Verzeichnissen zugeordnet werden können, auch wieder einzeln aus diesen entfernt werden, ohne daß sich eine derartige Operation auf die anderen Verzeichnisse, in denen sie enthalten sind, auswirkt.

Abbildung 5.10 zeigt, wie ein Lernobjekt gleichzeitig in verschiedenen Zweigen einer virtuellen Verzeichnisstruktur eingeordnet sein kann. Von beiden Wurzelverzeichnissen aus ist unabhängig voneinander eine Navigation zu diesem Objekt möglich.

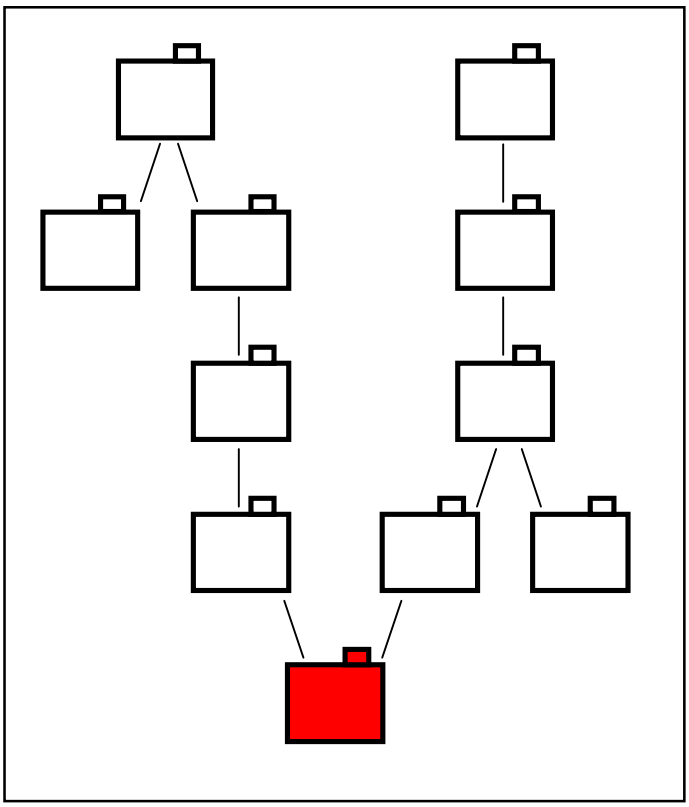

Abbildung 5.10: Einordnung eines Virtuellen Lernobjektes in verschiedenen Zweigen der virtuellen Verzeichnisstruktur

Durch diese Möglichkeit zur mehrfachen Einordnung von Objekten in Verzeichnisse ist auch eine Wiederverwertung von Lernobjekten in anderem Kontext ohne redundante Speicherung möglich. Wird beispielsweise eine Lehrsimulation innerhalb eines Kurses erstellt, kann sie zu einem späteren Zeitpunkt für eine andere Lehrveranstaltung erneut verwendet werden, indem sie in das Virtuelle Verzeichnis, das für diese Veranstaltung angelegt wird, oder eines seiner Unterverzeichnisse einsortiert wird. Das entsprechende Virtuelle Lernobjekt wird dabei nicht physikalisch kopiert, sondern das Verzeichnis der neuen Veranstaltung wird lediglich seiner Elternliste hinzugefügt, während das Objekt seinerseits der Liste der Kindobjekte des Verzeichnisses hinzugefügt wird.

5.4.3 Konfigurationen für Lehrsimulationen als Virtuelle Verzeichnisse

Eine weitere wichtige Verwendungsmöglichkeit für Virtuelle Verzeichnisse liegt darin begründet, daß sie zum Zusammenstellen von Dateien unterschiedlicher Typen benutzt werden können, wie dies für die Gestaltung multimedialer Lehreinheiten erforderlich ist. Ein Virtuelles Verzeichnis kann Virtuelle Lernobjekte aufnehmen, deren Inhalte den verschiedensten Datentypen angehören. Somit können die erforderlichen Bestandteile von Lehreinheiten bereits ohne zusätzlichen Aufwand in einem Verzeichnis gesammelt werden. Darüber hinaus ist lediglich noch eine Möglichkeit erforderlich, diese einzelnen Teile zu kompletten Lehreinheiten zusammenzusetzen.

In Kapitel 2.3 wurde die Anforderung aufgestellt, daß die einzelnen Lehrsimulationen als Konfigurationen für bestehende Simulationsprogramme erstellt werden sollen. Dies erfordert die Speicherung der entsprechenden Konfiguration (beispielsweise in einem XML-Format) zusammen mit den einzelnen Medien, die für die konkrete Simulation benötigt werden. An dieser Stelle wird die Tatsache ausgenutzt, daß es sich bei Virtuellen Verzeichnissen ebenfalls um Virtuelle Lernobjekte handelt. Als solche können sie nämlich ebenfalls über einen eigenen Inhalt (content) verfügen, der genau wie der Inhalt aller anderen VLOs in eigenen Dateien gespeichert werden kann.

Konfigurierbare Lehrsimulationen können somit als Virtuelle Verzeichnisse dargestellt werden, die ihre Konfigurationsdatei als Inhalt und die einzelnen benötigten Medien als Kindobjekte enthalten. Diese Unterscheidung ist von essentieller Bedeutung, denn der in der Datei abgelegte Inhalt (content) darf nicht mit den Objekten verwechselt werden, die dem Virtuellen Verzeichnis durch Eltern-Kind-Beziehungen zugeordnet werden. An dieser Stelle ist gewissermaßen eine konzeptionelle Abkehr vom Prinzip des traditionellen Verzeichnisses erforderlich, denn die Virtuellen Verzeichnisse müssen eher wie die Knoten eines Baumes betrachtet werden, in dem Inhalte nicht nur an den Blättern, sondern auch an den Knoten selbst gespeichert werden können. Als echte Verzeichnisse sind sie nur insofern zu betrachten, als daß ihnen zur Laufzeit dynamisch die zu ihnen gehörenden Objekte zugeordnet werden. Dies ist ein weiterer Grund für die Bezeichung als "virtuelle" Verzeichnisse im Rahmen der E-Learning-Plattform.

Dies ermöglicht die Schaffung einer einheitlichen Aktion zum Anzeigen von Objekten, die im System abgelegt sind. Soll auf Anweisung des Benutzers ein Lernobjekt zum Anzeigen geöffnet werden, so kann anhand seines Dateityps (MIME Type) festgestellt werden, mit welcher Anwendung es geöffnet werden kann oder welche im System vorgesehene Aktion zum Öffnen dieser Datei verwendet wird. Für konfigurierte Simulationen, die als Virtuelle Verzeichnisse vorliegen, gilt das gleiche. In diesem Fall wird nicht etwa eine Liste der Kindobjekte des Verzeichnisses angezeigt, sondern das zugehörige Simulationsprogramm wird unter Verwendung der Konfigurationsdatei geöffnet.

5.5 Benutzer und Benutzergruppen als Lernobjekte

Auch Benutzer werden in der E-Learning-Plattform als Lernobjekte betrachtet und als solche zur Laufzeit in Form von Virtuellen Lernobjekten verwaltet. Sie werden ebenfalls in der Datenbank abgelegt. Benutzer unterscheiden sich von anderen Lernobjekten lediglich durch einen anderen Metadatensatz, der in ihrer Metadatentabelle verwaltet wird. Anstelle von Stichwörtern, Angaben zum Urheberrecht, Datenformat oder Dateigröße werden für Benutzer der jeweilige Vorname und Nachname, die e-Mail-Adresse sowie bei Bedarf auch weitere personenbezogene Daten verwaltet, die im System zur Verwaltung benötigt werden.[281]

In gleicher Weise werden Virtuelle Verzeichnisse zur Verwaltung von Benutzergruppen verwendet. Die einzelnen Benutzer, die einer Gruppe zugeordnet werden, bilden die Liste der Kindobjekte des die Gruppe repräsentierenden Virtuellen Verzeichnisses, und dieses wird seinerseits in die Elternliste der Benutzerobjekte eingetragen. Außerdem ist es möglich, Gruppen in andere Gruppen einzusortieren und damit eine Hierarchie von Benutzergruppen zu erstellen. Beispielsweise könnte eine Gruppe, die Studenten eines Lehrstuhles beinhaltet, eine Untergruppe enthalten, in die nur die Teilnehmer einer ganz bestimmten Veranstaltung eingeordnet werden.

Im Rahmen der virtuellen Verzeichnisstruktur können Benutzer gleichzeitig Mitglieder in mehreren verschiedenen Gruppen sein. Dies ist beispielsweise dann von Bedeutung, wenn der lesende Zugriff auf die Inhalte bestimmter Kurse auf angemeldete Teilnehmer beschränkt werden soll, was durch die Zugehörigkeit zu einer Benutzergruppe realisiert werden kann, der die entsprechenden Zugriffsrechte eingeräumt werden. Durch die Möglichkeit zur Einsortierung in verschiedene Gruppen kann ein Benutzer gleichzeitig als Teilnehmer mehrerer derartiger Kurse gekennzeichnet werden und gehört zusätzlich immer noch anderen Gruppen an, denen er allgemein zugeordnet wurde, beispielsweise einer größeren Gruppe, die alle Studenten seiner Fakultät enthält. Fallen die Voraussetzungen für die Zugehörigkeit zu einer Gruppe weg, kann jeder Benutzer separat aus jeder beliebigen Gruppe entfernt werden, der er angehört, ohne daß dies Auswirkungen auf seine Zugehörigkeit zu anderen Gruppen hat.

In der E-Learning-Plattform sind zwei Benutzergruppen und zwei Benutzer vordefiniert. Bei der einen dieser beiden Gruppen handelt es sich um das Wurzelverzeichnis der Gruppenhierarchie, also sozusagen die "Wurzelgruppe". Alle anderen Gruppen in der Hierarchie sind direkte oder indirekte Untergruppen dieser Gruppe, und nur in diesem Teilbereich der Datenstruktur des Systems können neue Gruppen und Benutzer angelegt werden. Dadurch wird vermieden, daß Lernobjekte der sie repräsentierenden Typen unkontrolliert irgendwo im System angelegt werden, wo sie möglicherweise nur schwer wieder aufzufinden wären.

Außerdem wird durch einen vordefinierten Speicherort für die Gruppenhierarchie ein automatisches Update der im System vorhandenen Gruppen und Benutzer ermöglicht. Dieser Vorgang wird stets als erstes ausgeführt, sobald die Instanz der E-Learning-Plattform auf einem Computer sich über das Internet mit einer Installation des Systems auf einem anderen Computer verbindet. Die Wurzelgruppe trägt stets den vorgegebenen *UniversalObjectIdentifier* 0:1 und ist ihrerseits als Kindobjekt in das Wurzelverzeichnis des Systems eingeordnet, das über den ebenfalls systemweit unveränderlich festgelegten *UniversalObjectIdentifier* 0:0 angesprochen werden kann.

[281] Natürlich sind an dieser Stelle stets die geltenden datenschutzrechtlichen Bestimmungen zu beachten.

Die zweite im System fest vorgegebene Gruppe ist die Administratorengruppe, die den *UniversalObjectIdentifier* 0:2 trägt. Ihr gehören zwei ebenfalls voreingestellte Benutzer an, die aufgrund dieser Gruppenzugehörigkeit über Administratorenrechte verfügen, die ein Lese- und Schreibrecht auf alle im System vorhandene Objekte sowie die Kontrolle über die Benutzerverwaltung mit einschließen. Nur Administratoren dürfen Benutzer im System anlegen und ihre Gruppenzugehörigkeiten festlegen und verändern. Für normale User ist die Benutzerverwaltung nicht zugänglich, so daß sie weder selbst Änderungen vornehmen können (außer an dafür explizit freigegebenen, nur sie selbst betreffenden Daten wie z. B. ihrer eigenen Telefonnummer) noch unkontrolliert Einsicht in die Daten anderer Benutzer nehmen können. Die Anzeige von Benutzernamen, die beispielsweise im Rahmen von Metadatenfeldern verwendet werden, ist dennoch gewährleistet, da das Leserecht für die Benutzerobjekte den Usern nicht generell entzogen wurde, sondern lediglich der Zugriff auf diese Daten über die Benutzerverwaltung den Administratoren vorbehalten ist.

Die zwei vordefinierten Administratoren erfüllen unterschiedliche Aufgaben. Der erste wird vom System selbst bei Aktionen verwendet, die eine Authentifizierung erfordern, sofern die dabei vorzunehmenden Operationen auf den gespeicherten Daten über die Berechtigungen des eingeloggten Benutzers hinausgehen. Dies ist beispielsweise immer dann der Fall, wenn ein Benutzer eine Lehreinheit vom Server auf seinen eigenen Computer herunterlädt, die aus mindestens einem Virtuellen Verzeichnis besteht. Für solche Verzeichnisse verfügt normalerweise nur der Autor über Schreibrechte, um unbefugte Änderungen von Lehrmaterialien zu verhindern. Beim Download müssen jedoch die Objekte, die auf dem Server einem Virtuellen Verzeichnis als Kindobjekte zugeordnet sind, auch auf dem Zielrechner in dieses eingeordnet werden.

Zu diesem Zweck loggt sich das System sozusagen bei sich selbst als Administrator ein, der mit entsprechenden Rechten ausgestattet ist und die erforderlichen Operationen auf den Daten ausführen kann, die der gleichzeitig eingeloggte, menschlichen Benutzer nicht selbst ausführen darf. Ein weiteres Beispiel ist das weiter oben bereits erwähnte, automatische Update der Benutzerverwaltung beim Verbinden mit anderen Computern, das ebenfalls unter Verwendung der Rechte des systeminternen "Administrators" durchgeführt wird. Eine mißbräuchliche Verwendung dieser Möglichkeit ist ausgeschlossen, da während der Durchführung der Aktionen unter Nutzung der zusätzlichen Zugriffsberechtigungen keine anderen Operationen auf den Daten ausgeführt werden können. Außerdem ist ein manuelles Einloggen mit den internen Zugangsdaten des automatischen "Administrators" nicht möglich. Der Systembenutzer trägt den Benutzernamen *system* und wird durch den *UniversalObjectIdentifier* 0:3 eindeutig identifiziert.

Der zweite vordefinierte Administrator mit dem *UniversalObjectIdentifier* 0:4 und dem Benutzernamen *admin* dient hingegen zum Einloggen als menschlicher Administrator. Nach der ersten Installation der E-Learning-Plattform wird dieser Administratoraccount benötigt, um weitere Benutzer anzulegen und ggf. die ersten Virtuellen Verzeichnisse und Inhalte einzufügen. Letztere Aktionen können natürlich auch später von Autoren vorgenommen werden, die jedoch ebenfalls erst vom Administrator angelegt werden müssen. Auch das Anlegen weiterer Administratoraccounts ist möglich. Der Benutzer *admin* tritt somit als Ersteller für weitere Benutzer in Erscheinung, in deren Metadaten dies vermerkt wird. Als Erzeuger der im System vorgegebenen Systemobjekte (also des Wurzelverzeichnisses sowie der genannten zwei Gruppen und zwei Administratoren) wird dagegen der automatische Benutzer *system* eingetragen. Damit ist gleichzeitig auch sichergestellt, daß jedes Objekt in der E-Learning-Plattform einen Ersteller hat, womit Probleme bei diesbezüglichen automatischen Auswertungen vermieden werden.

5.6 Datenbankschnittstelle für die Speicherung der Metadaten

Die in diesem Kapitel bisher vorgestellten Datentypen dienen der Repräsentation von Lernobjekten (durch Virtuelle Lernobjekte) und den diesen übergeordneten Strukturen und Hierarchien (durch Virtuelle Verzeichnisse) zur Laufzeit der E-Learning-Plattform. Diese Objekte bedürfen jedoch auch einer persistenten Speicherung in einer Datenbank, um ihre dauerhafte Verwendbarkeit sicherzustellen.

Gemäß den zu Beginn in den Abschnitten 2.6 und 2.9 aufgestellten Anforderungen nach einer möglichst weitgehenden technischen Unabhängigkeit der E-Learning-Plattform von Fremdprodukten darf diese jedoch nicht ausschließlich von einem einzigen System zur Speicherung von Daten abhängig sein. Aus diesem Grund wurde die Entscheidung getroffen, keine Festlegung auf eine bestimmte Datenbank zu treffen, sondern statt dessen eine abstrakte Datenbankschnittstelle zu schaffen, für die dann für beliebige Datenbanken eine Implementation erfolgen kann.

Dieser sogenannte Datenbank-Adapter stellt abstrakte Methoden zur Verfügung, die es ermöglichen, Virtuelle Lernobjekte zu speichern und wiederzufinden, ohne dabei technische Details der Datenbank zu berücksichtigen. Insbesondere konstruiert die jeweilige Implementierung des Datenbank-Adapters eigenständig sämtliche zur Kommunikation mit der betreffenden Datenbank benötigten Aufrufe (z. B. SQL-Anfragen für relationale Datenbanken). Im Sinne der in Kapitel 4 dargestellten, vierschichtigen Systemarchitektur stellt der Adapter die der Datenbank vorgelagerte, interne Datenmodellschicht dar.

Durch seine Einführung müssen die zum Laden und Speichern von Objekten benötigten Befehle nicht in der darüberliegenden Anwendungsschicht zusammengesetzt werden, so daß diese bei einem Wechsel der Datenbank auch nicht neu programmiert werden muß. Lediglich ein für die neue Datenbank angepaßter Datenbank-Adapter ist in diesem Fall erforderlich. Alle Methoden in den darüberliegenden Schichten des Systems können hingegen unverändert weiter benutzt werden.

Im Folgenden wird zunächst die Spezifikation des Datenbank-Adapters mit allen darin definierten Methoden vorgestellt, mit denen Virtuelle Lernobjekte aus der Datenbank gelesen, in ihr gespeichert oder gelöscht werden können. Daneben stehen verschiedene Methoden zum Einsortieren und Bewegen von Objekten innerhalb der in der Datenbank abgelegten, virtuellen Verzeichnisstruktur zur Verfügung, die hier ebenfalls beschrieben werden. Außerdem wird der Mechanismus dargestellt, mit dem anhand von entsprechenden Einträgen in der Systemkonfiguration der E-Learning-Plattform automatisch erkannt wird, welche Datenbank verwendet wird, um daraufhin eine Instanz des zugehörigen Datenbank-Adapters zu erzeugen, sofern dieser zur Verfügung steht. Danach folgt eine kurze Betrachtung der Referenz-Implementation des Datenbank-Adapters, die für die frei erhältliche Datenbank MySQL vorgenommen wurde.

5.6.1 Spezifikation des Datenbank-Adapters

Der Datenbank-Adapter wird durch die abstrakte Klasse *DatabaseAdapter* dargestellt. Diese Klasse enthält eine Reihe von abstrakten Methoden, die Datenbankoperationen definieren. Konkrete Unterklassen dieser abstrakten Klasse müssen diese Methoden für die jeweils zugehörige Datenbank implementieren. Daneben verfügt die Klasse über zwei statische Methoden (also Klassenmethoden, die nicht auf den einzelnen Objekten,

sondern auf der Klasse selbst aufgerufen werden). Die erste dieser beiden Methoden, *getDatabaseAdapter*, wird beim Start der E-Learning-Plattform aufgerufen und dient dazu, festzustellen, welche konkrete Unterklasse des Adapters instanziiert werden muß. Zu diesem Zweck wird ein Systemparameter namens *DriverClassName* eingelesen, der in der Konfigurationsdatei *parameters.xml* angegeben werden muß und von der Klasse *ParameterManager*, die unter anderem für die Verwaltung derartiger Systemparameter zuständig ist, aus dieser Datei eingelesen und im System zur Verfügung gestellt wird.[282]

Der Wert des Parameters *DriverClassName* wird von der Methode *getDatabaseAdapter* als Klassenname der zu instanziierenden, konkreten Adapterklasse interpretiert, von der anschließend ein Objekt erzeugt wird, das zur Laufzeit der E-Learning-Plattform für die Zugriffe auf die jeweilige Datenbank verwendet wird. Zur Erzeugung dieses Objektes dient die statische Methode *getInstance*, die in der abstrakten Klasse *DatabaseAdapter* definiert ist. Die konkreten Unterklassen müssen diese Methode jeweils in geeigneter Weise überschreiben, um ein Adapterobjekt zur Verfügung zu stellen, da die Methode in der Klasse *DatabaseAdapter* den Wert `null` zurückliefert.[283] Nach der einmaligen Erzeugung des Adapterobjektes liefert *getInstance* immer dieses Objekt zurück, weitere Objekte seiner Klasse können nicht erzeugt werden. Das Adapterobjekt stellt also ein Singleton-Objekt dar und verwaltet sämtliche Zugriffe auf die Datenbank.

Abbildung 5.11 zeigt ein Klassendiagramm der abstrakten Klasse DatabaseAdapter.

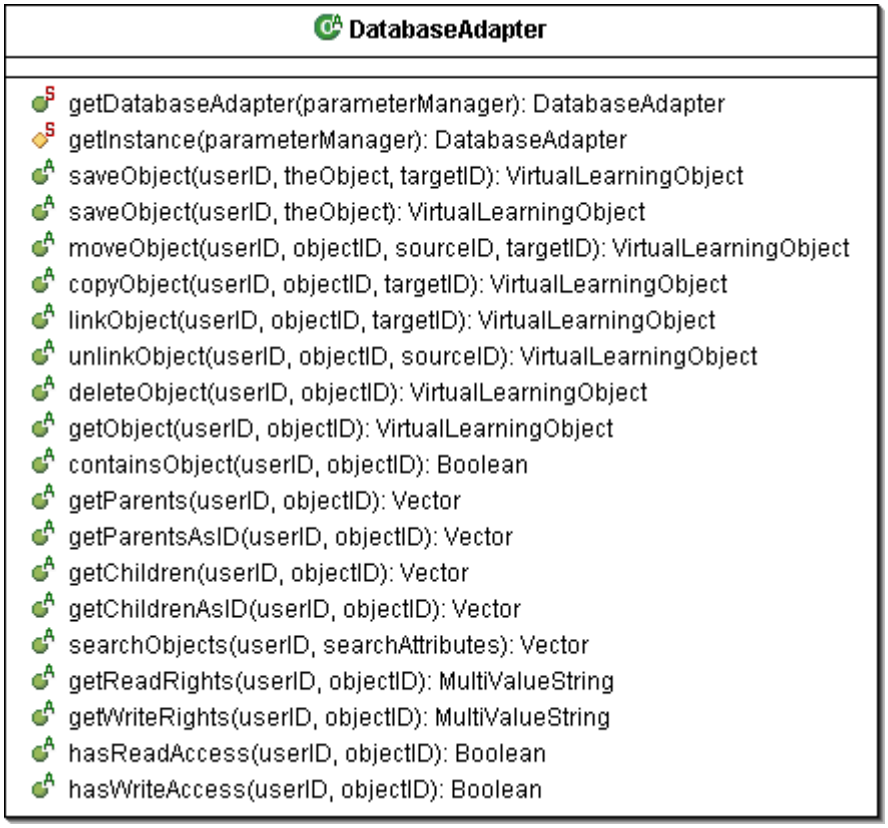

Abbildung 5.11: Klassendiagramm der abstrakten Klasse DatabaseAdapter (mit Methoden-Parametern und Rückgabetypen)

[282] Siehe Kapitel 8.6.

[283] Eigentlich sollte diese Methode besser als abstrakt deklariert werden. Der Grund dafür, daß dies in der Praxis nicht der Fall ist, liegt darin, daß in der Programmiersprache Java, in der die E-Learning-Plattform implementiert wurde, statische Methoden nicht als abstrakt deklariert werden können. Die Dokumentation zur Programmiersprache Java liefert leider keine Begründung für diese Einschränkung.

Die abstrakte Klasse *DatabaseAdapter* definiert die folgenden abstrakten Methoden:

5.6.1.1 saveObject (für neue Objekte)

Diese Methode speichert ein angegebenes neues Virtuelles Lernobjekt in ein bestimmtes Virtuelles Verzeichnis. Sie kann nur für Objekte benutzt werden, die zuvor noch nicht in der Datenbank abgelegt wurden. In jedem anderen Fall erzeugt diese Methode einen Ausnahmefehler (Exception). Der Grund für dieses Verhalten liegt darin, daß Virtuelle Lernobjekte gleichzeitig in vielen verschiedenen Virtuellen Verzeichnissen enthalten sein können, so daß es für bereits bestehende Objekte keinen Sinn macht, diese nur in einem dieser Verzeichnisse zu speichern, da es sich bei ihrem Vorkommen in anderen Verzeichnissen nicht um Kopien, sondern um das selbe Objekt handelt. Zum Kopieren von Virtuellen Lernobjekten sowie zum Verlinken in zusätzliche Verzeichnisse stehen andere Methoden zur Verfügung.

Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`theObject:` das Virtuelle Lernobjekt, das gespeichert werden soll.
`targetID:` ein String, der den *UniversalObjectIdentifier* des Virtuellen Verzeichnisses enthält, in das das Virtuelle Lernobjekt gespeichert werden soll.

Der Rückgabewert der Methode ist das gespeicherte *VirtualLearningObject*, das zur Sicherstellung seiner Aktualität neu aus der Datenbank gelesen wird.

Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectAlreadyExistsInDatabaseException:` wenn das Objekt bereits in der Datenbank enthalten ist.
- `TargetDirectoryNotFoundException:` wenn das Zielverzeichnis nicht in der Datenbank enthalten ist.
- `WritingNotPermittedException:` wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt. Diese Exception ist eine Unterklasse der Klasse `InsufficientRightsException`.
- `DatabaseOperationException:` wenn ein Datenbankfehler auftritt.

5.6.1.2 saveObject (für bestehende Objekte)

Diese Methode speichert ein angegebenes Virtuelles Lernobjekt in die Datenbank. Das Objekt muß bereits in der Datenbank vorhanden sein. Wenn es noch nicht existiert, wird ein Ausnahmefehler (Exception) erzeugt. Diese Variante der Methode *saveObject* stellt eine Update-Operation dar: Ein Objekt, das vorher bereits aus der Datenbank geladen und zwischendurch verändert wurde, wird anschließend wieder in die Datenbank zurück gespeichert. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`theObject:` das Virtuelle Lernobjekt, das gespeichert werden soll.

Der Rückgabewert der Methode ist das gespeicherte *VirtualLearningObject*, das zur Sicherstellung seiner Aktualität neu aus der Datenbank gelesen wird.

Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt noch nicht in der Datenbank enthalten ist.
- `TargetDirectoryNotFoundException`: wenn das Zielverzeichnis nicht in der Datenbank enthalten ist.
- `WritingNotPermittedException`: wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt. Diese Exception ist eine Unterklasse der Klasse `InsufficientRightsException`.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.3 moveObject

Diese Methode verschiebt ein angegebenes Virtuelles Lernobjekt aus einem bestimmten Quellverzeichnis in ein angegebenes Zielverzeichnis. Nach dieser Operation ist das Objekt in dem ursprünglichen Verzeichnis nicht mehr vorhanden. Andere virtuelle Verzeichnisse sind nicht betroffen. Wenn das Virtuelle Lernobjekt vor dem Verschieben in weiteren Verzeichnissen enthalten war, ist es dort hinterher ebenfalls noch vorhanden. Wenn das Objekt im Zielverzeichnis bereits vorhanden ist, wird es dort nicht dupliziert, sondern es wird lediglich aus dem Quellverzeichnis entfernt. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:`	ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:`	ein String, der den *UniversalObjectIdentifier* des zu verschiebenden Virtuellen Lernobjektes repräsentiert.
`sourceID:`	ein String mit dem *UniversalObjectIdentifier* des Quellverzeichnisses.
`targetID:`	ein String mit dem *UniversalObjectIdentifier* des Zielverzeichnisses.

Der Rückgabewert der Methode ist das verschobene *VirtualLearningObject*, das zur Sicherstellung seiner Aktualität neu aus der Datenbank gelesen wird.

Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt nicht in der Datenbank enthalten ist.
- `SourceDirectoryNotFoundException`: wenn das Quellverzeichnis nicht in der Datenbank enthalten ist.
- `TargetDirectoryNotFoundException`: wenn das Zielverzeichnis nicht in der Datenbank enthalten ist.
- `ObjectNotContainedInSourceDirectoryException`: wenn das Objekt nicht im Quellverzeichnis enthalten ist.
- `ReadingNotPermittedException`: wenn der eingeloggte Benutzer nicht über die erforderlichen Leserechte zum Durchführen dieser Aktion verfügt.
- `WritingNotPermittedException`: wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.4 copyObject

Diese Methode kopiert ein angegebenes Virtuelles Lernobjekt in ein bestimmtes Virtuelles Verzeichnis. Nach dieser Aktion existiert eine physikalische Kopie des Objektes in dem neuen Verzeichnis. Änderungen am neuen Objekt wirken sich nicht auf das alte Objekt aus, umgekehrt ist dies ebenfalls nicht der Fall. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String, der den *UniversalObjectIdentifier* des zu kopierenden Virtuellen Lernobjektes repräsentiert.
`targetID:` ein String mit dem *UniversalObjectIdentifier* des Zielverzeichnisses.

Der Rückgabewert der Methode ist das kopierte (neue) *VirtualLearningObject*, das zur Sicherstellung seiner Aktualität neu aus der Datenbank gelesen wird.

Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException:` wenn das Objekt nicht in der Datenbank enthalten ist.
- `TargetDirectoryNotFoundException:` wenn das Zielverzeichnis nicht in der Datenbank enthalten ist.
- `ReadingNotPermittedException:` wenn der eingeloggte Benutzer nicht über die erforderlichen Leserechte zum Durchführen dieser Aktion verfügt.
- `WritingNotPermittedException:` wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt.
- `DatabaseOperationException:` wenn ein Datenbankfehler auftritt.

5.6.1.5 linkObject

Diese Methode fügt ein angegebenes Virtuelles Lernobjekt zu einem bestimmten Virtuellen Verzeichnis hinzu. Bei dieser Operation werden keine neuen Objekte in der Datenbank erzeugt. Das Objekt wird lediglich der Liste der Kindobjekte eines weiteren Verzeichnisses hinzugefügt, und das Verzeichnis wird seinerseits in die Liste der Eltern des Objektes aufgenommen. Wenn das Objekt über das neue Elternverzeichnis geändert wird, sind diese Änderungen in jedem Verzeichnis sichtbar, in dem sich das Objekt befindet. Das Objekt muß vor der Operation bereits in der Datenbank gespeichert sein.

Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String, der den *UniversalObjectIdentifier* des zu verknüpfenden Virtuellen Lernobjektes repräsentiert.
`targetID:` ein String mit dem *UniversalObjectIdentifier* des Zielverzeichnisses.

Der Rückgabewert der Methode ist das durch die Modifikation innerhalb der Liste der Elternobjekte geänderte *VirtualLearningObject*, das zur Sicherstellung seiner Aktualität neu aus der Datenbank gelesen wird.

Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt nicht in der Datenbank enthalten ist.
- `TargetDirectoryNotFoundException`: wenn das Zielverzeichnis nicht in der Datenbank enthalten ist.
- `ReadingNotPermittedException`: wenn der eingeloggte Benutzer nicht über die erforderlichen Leserechte zum Durchführen dieser Aktion verfügt.
- `WritingNotPermittedException`: wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.6 unlinkObject

Diese Methode entfernt ein angegebenes Virtuelles Lernobjekt aus einem bestimmten Virtuellen Verzeichnis. Nach dieser Operation kann das Objekt noch in der Datenbank enthalten sein. Es wird lediglich aus der Liste der Kindobjekte des betreffenden Verzeichnisses entfernt, während dieses Verzeichnis im Gegenzug aus der Liste der Elternobjekte des VLO gelöscht wird. Wenn das Objekt nach dieser Unlink-Operation noch über mindestens ein weiteres Elternobjekt verfügt, wird es nicht aus der Datenbank entfernt. Wenn andererseits kein weiteres Elternobjekt vorhanden ist, wird das Objekt gelöscht. In diesem Fall entspricht das Ergebnis dieser Operation dem eines Aufrufes der Methode *deleteObject*.

Um zu vermeiden, daß durch die Löschung ihres einzigen Elternobjektes unerreichbare Objekte in der Datenbank zurückbleiben, werden zusätzlich die Kindobjekte des gelöschten Objektes überprüft. Gibt es Kindobjekte, die über keine weiteren Elternobjekte verfügen, werden diese ebenfalls gelöscht. Diese Vorschrift wird rekursiv angewendet, wobei allerdings durch eine entsprechende Überprüfung sichergestellt wird, daß dabei nicht versehentlich das Wurzelverzeichnis eines Servers gelöscht wird, da anderenfalls der Zugang zu den anderen gespeicherten Daten erheblich erschwert oder sogar völlig unmöglich gemacht würde.

Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID`: ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.

`objectID`: ein String, der den *UniversalObjectIdentifier* des zu entfernenden Virtuellen Lernobjektes repräsentiert.

`sourceID`: ein String mit dem *UniversalObjectIdentifier* des virtuellen Verzeichnisses, aus dem das Objekt entfernt werden soll.

Der Rückgabewert der Methode ist das durch die Modifikation innerhalb der Liste der Elternobjekte geänderte *VirtualLearningObject*, das zur Sicherstellung seiner Aktualität neu aus der Datenbank gelesen wird. Dies ist jedoch natürlich nur dann möglich, wenn sich das Objekt nach der Ausführung der Unlink-Operation weiterhin in der Datenbank befindet. Wenn dies jedoch nicht mehr der Fall ist, gibt die Methode statt dessen den Wert `null` zurück.

Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt nicht in der Datenbank enthalten ist.
- `SourceDirectoryNotFoundException`: wenn das Quellverzeichnis nicht in der Datenbank enthalten ist.
- `ObjectNotContainedInSourceDirectoryException`: wenn das Objekt nicht im Quellverzeichnis enthalten ist.
- `ReadingNotPermittedException`: wenn der eingeloggte Benutzer nicht über die erforderlichen Leserechte zum Durchführen dieser Aktion verfügt.
- `WritingNotPermittedException`: wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.7 deleteObject

Diese Methode löscht ein angegebenes Virtuelles Lernobjekt aus der Datenbank. Im Gegensatz zur Methode *unlinkObject* ist das Objekt nach dieser Operation in keinem Fall mehr in der Datenbank enthalten, unabhängig davon, in wie vielen Verzeichnissen es zuvor enthalten war. Das Objekt wird aus den Listen der Kindobjekte aller seiner Elternverzeichnisse gelöscht, und diese werden im Gegenzug aus seiner Elternliste entfernt. Anschließend wird das Objekt aus der Datenbank entfernt.

Um zu vermeiden, daß durch die Löschung ihres einzigen Elternobjektes unerreichbare Objekte in der Datenbank zurückbleiben, werden zusätzlich die Kindobjekte des gelöschten Objektes überprüft. Gibt es Kindobjekte, die über keine weiteren Elternobjekte verfügen, werden diese ebenfalls gelöscht. Diese Vorschrift wird rekursiv angewendet, wobei allerdings durch eine entsprechende Überprüfung sichergestellt wird, daß dabei nicht versehentlich das Wurzelverzeichnis eines Servers gelöscht wird, da anderenfalls der Zugang zu den anderen gespeicherten Daten erheblich erschwert oder sogar völlig unmöglich gemacht würde.

Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID`: ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.

`objectID`: ein String, der den *UniversalObjectIdentifier* des zu löschenden Virtuellen Lernobjektes repräsentiert.

Der Rückgabewert der Methode ist immer `null`. Sie kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt nicht in der Datenbank enthalten ist.
- `ReadingNotPermittedException`: wenn der eingeloggte Benutzer nicht über die erforderlichen Leserechte zum Durchführen dieser Aktion verfügt.
- `WritingNotPermittedException`: wenn der eingeloggte Benutzer über kein Schreibrecht im Zielverzeichnis verfügt.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.8 getObject

Diese Methode gibt zu einem angegebenen *UniversalObjectIdentifier* das entsprechende Objekt aus der Datenbank zurück. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String, der den *UniversalObjectIdentifier* des aus der Datenbank zu lesenden Virtuellen Lernobjektes repräsentiert.

Der Rückgabewert der Methode ist *VirtualLearningObject*, das durch den angegebenen *UniversalObjectIdentifier* identifiziert wird, sofern ein solches Objekt in der Datenbank enthalten ist und der anfragende Benutzer die erforderlichen Zugriffsrechte zum Lesen dieses Objektes besitzt. In jedem anderen Fall gibt die Methode den Wert `null` zurück. Man beachte, daß auf diese Weise ein unbefugter Zugriffsversuch zum selben Ergebnis führt, als wenn das betreffende Objekt gar nicht existieren würde. Die Methode erzeugt einen Ausnahmefehler (Exception) vom Typ `DatabaseOperationException`, wenn ein Datenbankfehler auftritt.

5.6.1.9 containsObject

Diese Methode überprüft, ob sich zu einem angegebenen *UniversalObjectIdentifier* ein Objekt in der Datenbank befindet. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, dessen Vorhandensein in der Datenbank überprüft werden soll.

Der Rückgabewert dieser Methode ist `true` (als Objekt der Klasse *Boolean*), wenn die Datenbank ein Objekt mit dem angegebenen UniversalObjectIdentifier enthält und der anfragende Benutzer über die erforderlichen Zugriffsrechte zum Lesen dieses Objektes verfügt. In jedem anderen Fall gibt die Methode den Wert `false` (ebenfalls in Form eines Objektes der Klasse *Boolean*) zurück. Man beachte, daß auf diese Weise ein unbefugter Zugriffsversuch zum selben Ergebnis führt, als wenn das betreffende Objekt gar nicht existieren würde. Die Methode erzeugt einen Ausnahmefehler (Exception) vom Typ `DatabaseOperationException`, wenn ein Datenbankfehler auftritt.

5.6.1.10 getParents

Diese Methode gibt einen Vektor mit den Elternobjekten eines angegebenen Virtuellen Lernobjektes zurück. Wenn das Objekt keine Eltern hat, ist der Vektor leer. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, dessen Eltern aus der Datenbank gelesen werden sollen.

Der Rückgabewert dieser Methode ist ein (eventuell leerer) Vektor, der diejenigen Elternobjekte des angegebenen Objektes enthält, für die der anfragende Benutzer über die erforderlichen Zugriffsrechte zum Lesen verfügt. Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt nicht in der Datenbank enthalten ist.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.11 getParentsAsID

Diese Methode verhält sich genau so wie *getParents*, mit dem einzigen Unterschied, daß der zurückgelieferte Vektor keine Virtuellen Lernobjekte, sondern lediglich deren *UniversalObjectIdentifier* enthält.

5.6.1.12 getChildren

Diese Methode gibt einen Vektor mit den Kindobjekten eines angegebenen Virtuellen Lernobjektes zurück. Wenn das Objekt keine Kinder hat, ist der Vektor leer. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID`: ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID`: ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, dessen Kinder aus der Datenbank gelesen werden sollen.

Der Rückgabewert dieser Methode ist ein (eventuell leerer) Vektor, der diejenigen Kindobjekte des angegebenen Objektes enthält, für die der anfragende Benutzer über die erforderlichen Zugriffsrechte zum Lesen verfügt. Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException`: wenn das Objekt nicht in der Datenbank enthalten ist.
- `DatabaseOperationException`: wenn ein Datenbankfehler auftritt.

5.6.1.13 getChildrenAsID

Diese Methode verhält sich genau so wie *getChildren*, mit dem einzigen Unterschied, daß der zurückgelieferte Vektor keine Virtuellen Lernobjekte, sondern lediglich deren *UniversalObjectIdentifier* enthält.

5.6.1.14 getReadRights

Diese Methode liefert zu einem Virtuellen Lernobjekt die *UniversalObjectIdentifier* der Benutzer und Gruppen zurück, die über Zugriffsrechte zum Lesen des Objektes verfügen. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID`: ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.

`objectID:` ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, dessen zugriffsberechtigte Benutzer ermittelt werden sollen.

Der Rückgabewert dieser Methode ist ein *MultiValueString*, der für alle Benutzer mit Leserechten auf das angegebene Objekt die *UniversalObjectIdentifier* enthält, die sie identifizieren. Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException:` wenn das Objekt nicht in der Datenbank enthalten ist.
- `ReadingNotPermittedException:` wenn der eingeloggte Benutzer nicht über das Leserecht auf dem angegebenen Objekt verfügt.
- `DatabaseOperationException:` wenn ein Datenbankfehler auftritt.

5.6.1.15 getWriteRights

Diese Methode liefert zu einem Virtuellen Lernobjekt die *UniversalObjectIdentifier* der Benutzer und Gruppen zurück, die über Zugriffsrechte zum Schreiben des Objektes verfügen. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, dessen zugriffsberechtigte Benutzer ermittelt werden sollen.

Der Rückgabewert dieser Methode ist ein *MultiValueString*, der für alle Benutzer mit Schreibrechten auf das angegebene Objekt die *UniversalObjectIdentifier* enthält, die sie identifizieren. Die Methode kann die folgenden Ausnahmefehler (Exceptions) erzeugen:

- `ObjectNotFoundInDatabaseException:` wenn das Objekt nicht in der Datenbank enthalten ist.
- `ReadingNotPermittedException:` wenn der eingeloggte Benutzer nicht über das Leserecht auf dem angegebenen Objekt verfügt.
- `DatabaseOperationException:` wenn ein Datenbankfehler auftritt.

5.6.1.16 hasReadAccess

Diese Methode prüft, ob ein Benutzer die Berechtigung zum Lesen eines bestimmten Objektes hat. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, für das die Leseberechtigung des Benutzers überprüft werden soll.

Der Rückgabewert dieser Methode ist `true` (als Objekt der Klasse *Boolean*), wenn der Benutzer über Leserechte auf das angegebene Objekt verfügt, anderenfalls gibt die Methode den Wert `false` (ebenfalls in Form eines Objektes der Klasse *Boolean*) zurück. Sie erzeugt einen Ausnahmefehler (Exception) vom Typ `DatabaseOperationException`, wenn ein Datenbankfehler auftritt.

5.6.1.17 hasWriteAccess

Diese Methode prüft, ob ein Benutzer die über die erforderliche Berechtigung zum Schreiben eines bestimmten Objektes hat. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`objectID:` ein String mit dem *UniversalObjectIdentifier* des Virtuellen Lernobjektes, für das die Schreibberechtigung des Benutzers geprüft werden soll.

Der Rückgabewert dieser Methode ist `true` (als Objekt der Klasse *Boolean*), wenn der Benutzer über Schreibrechte auf das angegebene Objekt verfügt, anderenfalls gibt die Methode den Wert `false` (ebenso als Objekt der Klasse *Boolean*) zurück. Sie erzeugt einen Ausnahmefehler (Exception) vom Typ `DatabaseOperationException`, wenn ein Datenbankfehler auftritt.

5.6.1.18 searchObjects

Diese Methode führt eine Suchoperation auf den Metadaten in der Datenbank gespeicherten Virtuellen Lernobjekte durch. Der Aufruf dieser Methode erfordert die Angabe der folgenden Parameter:

`userID:` ein String, der den *UniversalObjectIdentifier* des gerade eingeloggten Benutzers enthält, der die Methode aufgerufen hat.
`searchAttributes:` ein Satz von Metadaten, mit denen die Attribute der von der Suche zurückgelieferten Objekte übereinstimmen sollen.

Der Rückgabewert dieser Methode ist ein Vektor von *VirtualLearningObjects*, deren Metadaten alle in der Anfrage angegebenen Bedingungen erfüllen. Darin sind allerdings nur solche Objekte enthalten, für die der anfragende Benutzer über die erforderlichen Berechtigungen zum Lesen verfügt. Werden keine der Suchanfrage entsprechenden Objekte gefunden, die der Benutzer lesen darf, ist der Vektor leer. Die Methode erzeugt einen Ausnahmefehler (Exception) vom Typ `DatabaseOperationException`, wenn ein Datenbankfehler auftritt.

Der String `searchAttributes`, der die bei der Suche zu erfüllenden Bedingungen für die Metadaten enthält, muß unter Verwendung eines eigens für die Suche definierten XML-Formates konstruiert werden. Dieses Format sieht zunächst ein äußerstes XML-Element namens `<search>` vor. Innerhalb dieses Elementes kann entweder ein isoliertes `<expression>`-Element stehen, das ein einzelnes, zu überprüfendes Metadatum darstellt, oder mehrere solcher Elemente können unter Verwendung von `<and>`, `<or>` und `<not>` in beliebiger Weise zu logischen Ausdrücken kombiniert werden.

Ein `<expression>`-Element enthält seinerseits einen `<key>` für den Namen des zu durchsuchenden Metadatums sowie einen `<value>` für den für dieses Metadatum geforderten Wert. Dabei kann das `<expression>`-Element mit einem Attribut namens `comparator` versehen werden, das die Art des vorzunehmenden Vergleiches zwischen dem gesuchten Wert und den in der Datenbank enthaltenen Einträgen spezifiziert und einen der sieben Werte `lt` (less than), `le` (less or equal), `eq` (equal), `ne` (not

equal), `ge` (greater or equal), `gt` (greater than) oder `like` annehmen kann. Wenn kein `comparator` explizit angegeben wird, dann wird automatisch der einfache Vergleich (`eq`) als Voreinstellung gewählt.

Das zu jeder `<expression>` gehörende Element `<value>` kann außerdem mit einem Attribut namens `language` versehen werden, das für mehrsprachige Metadaten angegeben werden kann, um anzuzeigen, mit dem für welche Sprache gespeicherten Eintrag der Text verglichen werden soll, der als zu suchender Wert angegeben wurde. Die Bezeichnung der Sprachen ist dabei zwar aus technischer Sicht nicht formal festgelegt, soll aber dem in RFC 3066 definierten Internet-Standard zur Identifikation von Sprachen folgen. Damit ergibt sich für das als Parameter für die Suchfunktion zu verwendende XML-Format die folgende Document Type Definition (DTD):

```
<?xml version="1.0"?>

<!ELEMENT search (and | or | not | expression)>
<!ELEMENT and ((and | or | not | expression),
               (and | or | not | expression))>
<!ELEMENT or ((and | or | not | expression),
              (and | or | not | expression)>
<!ELEMENT not (and | or | not | expression)>

<!ELEMENT expression (key, value)>
<!ATTLIST expression comparator
    (lt | le | eq | ne | ge | gt | like) "eq">

<!ELEMENT key (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ATTLIST value language CDATA #IMPLIED>
```

Ein einfaches Beispiel für eine gemäß dieser DTD zusammengesetzten Suche ist folgender XML-String, der eine Suchanfrage nach Objekten darstellt, die unter Verwendung des im System voreingestellten Administrator-Accounts erstellt wurden (Creator = 0:4) und die den englischen Titel "E-Learning" tragen:

```
<search>
   <and>
      <expression comparator="eq">
         <key>Creator</key>
         <value>0:4</value>
      </expression>
      <expression>
         <key>Title</key>
         <value language="en">E-Learning</value>
      </expression>
   </and>
</search>
```

5.6.1.19 Übersicht über die vom Datenbank-Adapter erzeugten Exceptions

Zum Abschluß der Beschreibung der abstrakten Spezifikation des Datenbank-Adapters soll an dieser Stelle noch kurz auf die von den einzelnen Methoden generierten Ausnahmefehlern (Exceptions) eingegangen werden. Alle diese Exceptions gehören verschiedenen Unterklassen der Klasse *DatabaseException* an. Neben einer Reihe von Exceptions, die sich auf unerwartete Zustände der virtuellen Verzeichnisstruktur wie zum Beispiel fehlende Verzeichnisse und Objekte beziehen und die direkt von der Klasse *DatabaseException* abgeleitet sind, gibt es die Klasse *InsufficientRightsException*, die unzureichende Zugriffsrechte des aktuell eingeloggten Benutzers anzeigt und durch zwei Unterklassen, die fehlende Leserechte bzw. Schreibrechte signalisieren, genauer aufgegliedert wird. Die Menge der möglichen Exceptions wird vervollständigt durch die Klasse *DatabaseOperationException*, die auf Fehler während der Datenbankoperationen hinweist. Ihre verschiedenen Unterklassen bezeichnen unterschiedliche Ausprägungen derartiger Fehler. Abbildung 5.12 zeigt die Zusammenhänge zwischen den einzelnen Exceptions.

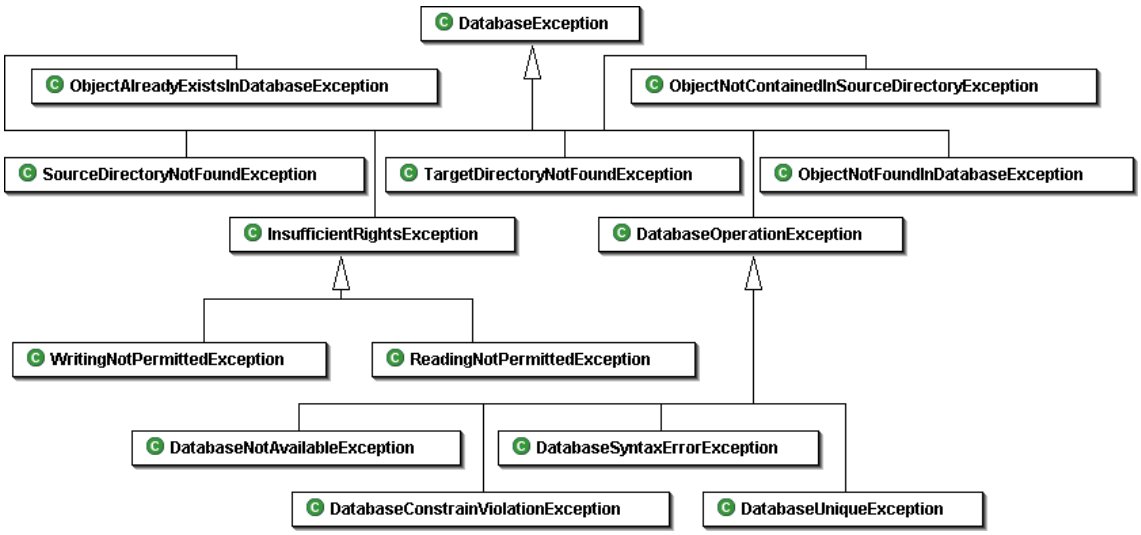

Abbildung 5.12: Vererbungshierarchie der Exceptions des Datenbank-Adapters

5.6.2 Referenz-Implementation des Adapters für die Datenbank MySQL

Im Rahmen des Projektes Monist wurde eine Referenz-Implementation des Datenbank-Adapters für die kostenlos erhältliche, relationale Datenbank MySQL erstellt.[284] Diese Maßnahme erfolgte unter Verwendung von JDBC sowie des ebenfalls frei erhältlichen Datenbanktreibers mm.mysql (mittlerweile umbenannt in MySQL Connector/J)[285]. Die Implementation wurde gemäß der hier vorgestellten Spezifikation von Herrn Holger Ebert von der Firma Media Engineering[286] vorgenommen. Der Vollständigkeit halber sollen an dieser Stelle kurz die verschiedenen Tabellen beschrieben werden, die in der Datenbank zur Speicherung der Virtuellen Lernobjekte und Virtuellen Verzeichnisse sowie der ihnen zugehörigen Metadaten angelegt wurden.

---

[284] Die entsprechenden Klassen befinden sich im Paket de.monist.database.mysql.

[285] Ehemalige Projekt-Homepage: http://mmmysql.sourceforge.net. Nachdem mm.mysql zum offiziellen JDBC-Treiber für MySQL erklärt wurde, steht der Treiber jetzt zum Download zur Verfügung unter der Adresse: http://www.mysql.com/downloads/api-jdbc.html.

[286] Homepage: http://www.media-engineering.de.

Die Tabelle ***data*** enthält für jedes Virtuelle Lernobjekt dessen *UniversalObjectIdentifier* und ordnet diesem neben dem Datum der letzten Modifikation des Objektes Verweise auf diejenigen Einträge in anderen Tabellen hin, die die Werte seiner Datenfelder für den content und die Lese- und Schreibrechte enthalten. Dabei stellt die Zahl in der Spalte *content* einen Eintrag in der Tabelle ***multilanguage*** dar, während Einträge in den Spalten *readaccess* und *writeaccess* auf die Tabelle ***multivalue*** verweisen. Die Spalte *type* speichert zu Verwaltungszwecken die Information, ob es sich bei dem Objekt um ein Verzeichnis handelt oder nicht.

Die Tabelle ***metadata*** ordnet jedem *UniversalObjectIdentifier* die für das dazugehörige Objekt gespeicherten Metadaten zu. Dabei bezeichnet die Spalte *mdtype* den Datentyp des jeweiligen Metadatums in Form eines Verweises in die Tabelle ***metadatatyp***, in der allen zulässigen Datentypen zur Verwaltung in der Datenbank numerische Schlüssel zugeordnet werden (z. B. 1=*String*, 6= *Integer*, 10=*MultiLanguageString* usw.). Die Spalte *mdkey* enthält den Namen des Metadatums, die Spalte *mdvalue* seinen Wert. Für Strings und numerische Datentypen steht an dieser Stelle direkt der Inhalt des Metadatums, während für *MultiLanguageStrings* und *MultiValueStrings* die Spalte Verweise in die Tabellen ***multilanguage*** und ***multivalue*** beinhaltet.

In diesen beiden Tabellen bezieht sich der numerische Wert in der Spalte *id* jeweils auf diese Zahlen, die Verweise aus anderen Tabellen darstellen. Zu jeder *id* kann es dabei mehrere Einträge in der Spalte *value* geben, die zusammen einen *MultiLanguageString* bzw. einen *MultiValueString* beschreiben. Während die in der Tabelle ***multivalue*** gespeicherten Einträge zu einer *id* jeweils gleichwertige Elemente des *MultiValueStrings* sind, kommt in der Tabelle ***multilanguage*** zusätzlich noch die Spalte *language* hinzu, in der die gemäß RFC 3066 angegebene Kurzbezeichnung für die jeweilige Sprache gespeichert wird.

Die virtuelle Verzeichnisstruktur wird durch die Tabelle ***datatree*** abgebildet, deren Zeilen jeweils genau eine Eltern-Kind-Beziehung zwischen zwei Lernobjekten darstellen. Dabei enthält jeweils die Spalte *parent* den *UniversalObjectIdentifier* des Elternobjektes und die Spalte *child* den des Kindobjektes. Zusätzlich wurde dieser Tabelle eine Spalte namens *number* hinzugefügt, mittels derer eine Reihenfolge der in einem Virtuellen Verzeichnis befindlichen Objekte hergestellt werden kann, indem jedem Objekt eine Positionsnummer zugewiesen wird. Zu den drei Methoden *saveObject*, *moveObject* und *linkObject* des Datenbank-Adapters gibt es jeweils eine Variante mit einem zusätzlichen, numerischen Parameter, mit dem diese Positionsangabe gesetzt werden kann.

Daneben gibt es noch die Tabelle ***rowid***, die ausschließlich Verwaltungszwecken dient und zu den anderen Tabellen jeweils den höchsten bisher vergebenen Wert der Spalte *id* speichert. Unter Verwendung von geeigneten Tools können Einträge von Objekten in der Datenbank bei Bedarf auch manuell durchgeführt werden. Beispielsweise wurden die beschriebenen, vordefinierten Objekte (das Wurzelverzeichnis, die beiden Gruppen für Benutzer und Administratoren sowie die beiden Administrator-Benutzer) auf diese Weise erstellt, da vorher ohne entsprechende Accounts nicht auf das System zugegriffen werden konnte. Natürlich ist für einen solchen direkten Zugriff auf die Datenbank eine entsprechende Zugangsberechtigung erforderlich.

Mit der Beschreibung der verwendeten Datenbank ist die Darstellung der Speicherung von Metadaten für Lernobjekte unter Verwendung von Virtuellen Lernobjekten und Virtuellen Verzeichnisse abgeschlossen. Zum Schluß dieses Kapitels folgt jetzt noch die Darstellung, wie die eigentlichen Inhalte der Lernobjekte im System abgelegt werden.

5.7 Speicherung der Inhalte auf einem HTTP-Server

Entsprechend dem beim Entwurf der E-Learning-Plattform verfolgten Prinzip einer strikten Trennung von Daten und Metadaten sind die Mechanismen zur Speicherung der Virtuellen Lernobjekte und der dazugehörigen Inhalte vollkommen voneinander unabhängig. Bei Bedarf kann jedes beliebige System zur Speicherung der Inhalte verwendet werden, ohne dabei auf die Benutzung der in diesem Kapitel vorgestellten Virtuellen Lernobjekte und Virtuellen Verzeichnisse verzichten zu müssen. Beispielsweise kann, wie bereits in Abschnitt 5.3.1 erwähnt wurde, das in der Klasse *VirtualLearningObject* definierte Datenfeld *content* verwendet werden, um darin die Speicherorte der zu einem VLO gehörenden Inhalte zu vermerken.

Bei der Implementation der E-Learning-Plattform wurde jedoch ein anderer Ansatz gewählt. Die Speicherung der Inhalte erfolgt über einen integrierten HTTP-Server, und die URL, die den Speicherort für eine bestimmte Datei bezeichnet, ergibt sich direkt aus dem *UniversalObjectIdentifier* des Objektes, seinem Dateityp und der Sprache, für die der Inhalt abgefragt wird. Der HTTP-Server[287] wird über einen Port angesprochen, der durch eine entsprechende Einstellung in der Systemkonfiguration frei gewählt werden kann. Zu diesem Zweck wird ein Systemparameter namens *httpPort* verwendet, der in der Konfigurationsdatei *parameters.xml* angegeben werden muß und von der Klasse *ParameterManager*, die unter anderem für die Verwaltung derartiger Parameter zuständig ist, aus dieser Datei eingelesen und im System zur Verfügung gestellt wird.[288]

Die URL für einen bestimmten Inhalt setzt sich aus folgenden Bestandteilen zusammen: Zunächst wird mit "http://" das Protokoll angegeben. Danach folgt der Name oder die IP-Adresse des anzusprechenden Computers. Diese ist stets bekannt, wenn über den im folgenden Kapitel beschriebenen XML-RPC-Kommunikationsmechanismus von diesem Computer Virtuelle Lernobjekte gelesen werden. Man beachte, daß auch der Zugriff auf Lernobjekte auf dem lokalen Computer über diesen Weg erfolgt. Nach der Adresse wird, wie in URLs üblich mit einem Doppelpunkt getrennt, der Port angegeben. Darauf folgt eine Pfadangabe wie in einer herkömmlichen URL, deren Teile allerdings nach einem besonderen Schema interpretiert werden müssen.

Der erste Teil des Pfades, der einer Verzeichnisangabe entspricht, enthält statt dieser in verschlüsselter Form den Benutzernamen und das Paßwort des anfragenden Benutzers. In der derzeitigen Version der E-Learning-Plattform werden diese beiden Werte, durch einen senkrechten Strich (|) getrennt, lediglich nach dem Base64-Schema in einen String kodiert. Natürlich steht es Systemen, die künftig auf der Basis der E-Learning-Plattform entwickelt werden, frei, statt dessen ein beliebiges Verschlüsselungsverfahren für die zu übermittelnden Daten einzusetzen. Der Rest der Pfadangabe beginnt mit einer String-Repräsentation des *UniversalObjectIdentifiers* des Objektes, bei der allerdings die beiden Teile des Identifiers nicht wie üblich durch einen Doppelpunkt, sondern durch einen Unterstrich (_) getrennt sind, da Doppelpunkte in einer URL nicht zulässig sind. Nach einem weiteren Unterstrich folgt die Sprache, für die der gesuchte Inhalt ausgegeben werden soll, in Form ihrer Kurzbezeichnung gemäß RFC 3066. Dahinter folgt nach einem Punkt die übliche Endung für den jeweiligen Dateityp, der aus den Metadaten des Virtuellen Lernobjektes ermittelt wird.

---

[287] Dabei handelt es sich um ein Objekt der Klasse de.monist.communication.httpServer. Da dieser Server nur eine minimale Funktionalität zur Verfügung stellen muß, erwies es sich als effizienter, einen nur für diesen Zweck maßgeschneiderten HTTP-Server zu verwenden, anstatt ein Fremdprodukt zu verwenden, das mit umfangreichen Funktionalitäten ausgestattet ist, die hier überhaupt nicht benötigt werden.

[288] Zum Mechanismus für die Angabe und das Einlesen von Systemparametern siehe Kapitel 8.6.

Ein Beispiel für eine nach diesem Schema zusammengesetzte URL ist:

http://www.monist.de:2003/QsfW5E1hRT3z/12345_67890_en.html

Über die vorstehende URL wird auf dem Server www.monist.de über den Port 2003 auf den englischen Inhalt eines Objektes mit dem UniversalObjectIdentifier 12345:67890 zugegriffen, für das in den Metadaten die Information gespeichert ist, daß es sich dabei um eine HTML-Datei handelt. Der String "QsfW5E1hRT3z" entspricht der kodierten Kombination aus dem Namen und dem Paßwort eines Benutzers. Das Zusammensetzen von derartigen URLs wird durch die Klasse MonistURL[289] erleichtert, die unter Angabe entweder der einzelnen, eben genannten Bestandteile oder einer nach dem angegebenen Schema aufgebauten URL instanziiert werden kann und über Methoden verfügt, die sowohl die URL als auch die einzelnen Bestandteile ausgeben können.

Zum Lesen, Schreiben und Löschen von Inhalten muß nur ein entsprechender HTTP-Request abgeschickt werden. Die im Header dieses Requests gesetzte HTTP-Methode entscheidet darüber, welche der drei Aktionen ausgeführt wird. Ein GET-Request liefert den Inhalt des betreffenden Objektes zurück. Mit einem PUT-Request kann eine neue Datei auf den HTTP-Server geschrieben werden, während ein DELETE-Request zum Löschen des durch die URL angegebenen Inhaltes dient. Der HTTP-Server überprüft beim Eintreffen einer dieser Anfragen zunächst die Zugriffsrechte des Benutzers, der den Request abgesetzt hat[290], und führt dann gegebenenfalls die zugehörige Aktion aus.

Auftretende Fehler werden durch die entsprechenden HTTP Response Codes angezeigt, mögliche Rückgaben sind dabei die folgenden Fehler:

- *400 Bad Request* - wenn eine andere Methode als GET, PUT oder DELETE für eine Anfrage an den HTTP-Server verwendet wurde.
- *401 Unauthorized* - wenn der Benutzer nicht über die benötigten Zugriffsrechte für die durchzuführende Aktion verfügt.
- *404 Not Found* - wenn eine angeforderte Datei nicht existiert.
- *500 Internal Server Error* - wenn während der Bearbeitung einer Anfrage durch den HTTP-Server Fehler beim Schreiben von Daten ins Dateisystem aufgetreten sind.

Das erfolgreiche Schreiben bzw. Löschen von Inhalten zeigen die folgenden Codes an:

- *200 OK* - nach dem erfolgreichen Löschen einer Datei.
- *201 Created* - nach dem erfolgreichen Schreiben einer Datei.
- *202 Accepted* - wenn die Anfrage zum Löschen einer Datei erfolgreich vom System entgegengenommen wurde, die Löschung allerdings noch nicht sofort erfolgt ist, sondern erst zu einem späteren Zeitpunkt vorgenommen wird.

Im Falle eines erfolgreichen Lesevorganges enthält die Antwort auf den HTTP-Request den angeforderten Inhalt. Der hier vorgestellte Speichermechanismus wird auch von dem in Kapitel 7 vorgestellten Navigator verwendet, der den Benutzern eine Sicht auf die gespeicherten Daten und Metadaten zur Verfügung stellt und das Einstellen von Lernobjekten und deren Inhalten in die E-Learning-Plattform ermöglicht.

---

[289] Der vollständige Klassenname lautet: de.monist.communication.MonistURL. Diese Benennung resultiert aus der Projektgeschichte des Projektes Monist, die Funktionalitäten der Klasse sind jedoch generell in der E-Learning-Plattform verwendbar.

[290] Diese Überprüfung erfolgt unter Zugriff auf die Datenbank über die zuvor beschriebene Schnittstelle.

6 Client-Server-Kommunikation

Wie bereits beim Entwurf der Systemarchitektur der E-Learning-Plattform in Kapitel 4 dargelegt wurde, wird die Kommunikation zwischen Client und Server (bzw. zwischen einzelnen Instanzen der E-Learning-Plattform) ebenso wie auch zwischen den verschiedenen Teilapplikationen innerhalb einer lokalen Installation des Programmes über eine gemeinsame Kommunikationsschnittstelle abgewickelt. Zu diesem Zweck wird ein einheitliches Kommunikationsprotokoll auf Basis von XML verwendet. In diesem Kapitel wird dieses Protokoll zunächst allgemein beschrieben[291], bevor auf die Implementation des Mechanismus und dessen Erweiterungen für die Übertragung der in Kapitel 5 für die E-Learning-Plattform definierten, speziellen Datentypen eingegangen wird.

6.1 XML-RPC als Kommunikationsprotokoll

Die Abkürzung "RPC" steht für "remote procedure call". Dabei handelt es sich um eine Technik zum Aufruf von ausführbarem Programmcode auf anderen Computern, welche, wie der Name schon sagt, ursprünglich aus der Welt der prozeduralen Progammiersprachen wie Pascal oder C stammt. Nachdem dieses Verfahren mit der allgemeinen Hinwendung zur objektorientierten Softwareentwicklung ein wenig in Vergessenheit geraten war, stellte die Entwicklung von XML und dessen Verwendung als Format für die Datenübertragung einen Wendepunkt dar. Heutzutage wird XML-RPC in vielen Client-Server-Applikationen eingesetzt, in denen Wert darauf gelegt wird, daß die zu übertragenden Aufrufe in einem möglichst einfachen und dazu plattformunabhängigen Format generiert werden können.[292]

Im Gegensatz zu anderen für die Kommunikation zwischen Objekten auf verschiedenen Computern entwickelten Technologien wie z. B. CORBA oder RMI kann unter Verwendung von XML-RPC jedes beliebige Objekt in leichter Weise für Aufrufe seiner Methoden über ein Netzwerk zugänglich gemacht werden. Dabei müssen keine Klassen zur Repräsentation des Objektes auf dem aufrufenden Client entwickelt werden, und das Hinzufügen von über das Netzwerk ansprechbaren Objekten auf dem Server oder das Verändern von Methoden dieser Objekte erfordert weder auf dem Server noch auf dem Client eine Änderung und Neukompilation von bestehendem Programmcode für die Übertragung der erforderlichen Aufrufe.

Um ein Objekt über einen XML-RPC-Kommunikationsmechanismus zugänglich zu machen, muß dieses lediglich unter einem bestimmten Namen beim XML-RPC-Server angemeldet werden. Bei Aufrufen über das Netzwerk muß dann lediglich dieser Name, der Name der aufzurufenden Methode sowie eine Liste der für diesen Aufruf benötigten Parameter angegeben werden. Ein für das Objekt beim Anmelden automatisch generierter sogenannter Handler prüft die Korrektheit der angegebenen Argumente, ruft die gewünschte Methode des Objektes auf und gibt den Rückgabewert der Methode über das Netzwerk an den Aufrufer zurück. Wenn eine Klasse eine bestimmte Schnittstelle implementiert, können ihre Objekte auch selbst die Rolle des Handlers übernehmen und erhalten damit die volle Kontrolle über die erforderlichen Methodenaufrufe und über die Generierung des Rückgabewertes.

---

[291] Die Beschreibung von XML-RPC und des verwendeten Software-Paketes folgt der Darstellung in: Brett McLaughlin, Java™ and XML, O' Reilly, Sebastopol (CA), 2000, Kapitel 10, Seiten 277-316.

[292] Weitere Informationen zu XML-RPC findet man auf der Internet-Homepage: http://www.xmlrpc.com.

Die Übertragung der Aufrufe geschieht dabei in Form eines HTTP-Requests, der die Anfragen in Form eines Textes enthält, der gemäß einem speziellen XML-Format für die RPC-Aufrufe zusammengesetzt wurde. Ein einfaches Beispiel für eine solche, in XML kodierte Anfrage ist der folgende XML-Code, der den Aufruf der Methode `multiply` eines unter dem Namen `calculator` registrierten Objektes mit den beiden ganzzahligen Parametern `2` und `3` darstellt:

```
<?xml version="1.0"?>
<methodCall>
   <methodName>calculator.multiply</methodName>
   <params>
      <param>
         <value><int>2</int></value>
      </param>
      <param>
         <value><int>3</int></value>
      </param>
   </params>
</methodCall>
```

Die Antwort auf diese Anfrage sieht folgendermaßen aus:

```
<?xml version="1.0"?>
<methodResponse>
   <params>
      <param>
         <value><int>6</int></value>
      </param>
   </params>
</methodResponse>
```

Die Anfragen und die darauf folgenden Antworten können natürlich bei Bedarf auch erheblich komplexer sein. Gemeinsam ist ihnen jedoch in jedem Fall, daß sie in ein sehr einfaches XML-Datenformat kodiert werden, das als Inhalt eines HTTP-Requests bzw. einer HTTP-Response über das Internet übertragen werden können.

Wenn ein Objekt beim Server registriert wird und seine Methoden und deren benötigte Parameter beim Erstellen des Clients bekannt sind, können die erforderlichen Anfragen über den XML-RPC-Kommunikationsmechanismus in ähnlicher Weise programmiert werden wie bei der Verwendung einer ganz normalen Programmierschnittstelle (API). Der Name der aufzurufenden Methode sowie die Parameter müssen angegeben werden, und als Ergebnis wird über die Kommunikationsschnittstelle ein bestimmtes Objekt zurückgegeben oder ggf. auch ein Ausnahmefehler (Exception) erzeugt, genauso, als ob ein einfacher Methodenaufruf erfolgt wäre. Die konkrete Implementation bleibt dem aufrufenden Objekt ebenso verborgen wie die gesamte Handhabung des Aufrufes über das Netzwerk inklusive des Aufbaus der dafür benötigten Netzwerkverbindung.

Neben ihrer Einfachheit verfügt die XML-RPC-Kommunikation noch über einen weiteren großen Vorteil gegenüber anderen Strategien. Durch die Verwendung des international anerkannten, plattformunabhängigen Standards XML als Datenformat ist der Mechanismus vollkommen unabhängig vom verwendeten Betriebssystem und auch von der Programmiersprache, in der Programme erstellt wurden. Damit können vollkommen beliebige Anwendungen auf diesem Weg miteinander kommunizieren.

6.2 Implementation des XML-RPC-Mechanismus

Zur Realisierung des XML-RPC-Kommunikationsmechanismus wurde im Rahmen der E-Learning-Plattform eine XML-RPC-Bibliothek verwendet, die ursprünglich von dem österreichischen Informatiker Hannes Wallnöfer entwickelt wurde und mittlerweile ein Open-Source-Projekt unter dem Dach der Apache Software Foundation geworden ist.[293]

Die wichtigsten Elemente dieses Softwarepaketes sollen hier kurz beschrieben werden. Die Klasse *XmlRpc* stellt zentrale Funktionalitäten wie die Umwandlung von Anfragen in XML-RPC-Requests und die Kodierung von Objekten gemäß dem dafür verwendeten XML-RPC-Datenformat sowie die Dekodierung dieses Formates mit einem XML-Parser unter Rekonstruktion der Objekte zur Verfügung. Diese Dienste werden von den Klassen *XmlRpcServer* und *XmlRpcClient* verwendet. Letztere dient zum Aufruf von Methoden über das Netzwerk, während der *XmlRpcServer* derartige Anfragen annimmt, an die entsprechenden Handler und über diese an die angemeldeten Objekte weiterleitet und die Rückgabewerte als XML-RPC-Response an den Client zurückschickt.

Beliebige Objekte können beim *XmlRpcServer* zur Laufzeit unter Aufruf der Methode *addHandler* registriert und durch *removeHandler* wieder entfernt werden. Sofern es sich um normale Objekte handelt, die nicht selbst als Handler agieren können, wird für diese ein Handler erzeugt, der die entsprechenden Methodenaufrufe an die Objekte weitergibt und somit als Mittler zwischen dem *XmlRpcServer* und den Objekten fungiert. Dieser Schritt entfällt, wenn die zu registrierenden Objekte einer Klasse angehören, die die Schnittstelle *XmlRpcHandler* implementiert.

Diese Schnittstelle definiert eine Methode *execute*, an die der Methodenname und die übermittelten Parameter übergeben werden und die die Kontrolle über die eigentlichen Methodenaufrufe ausübt. Dabei können zum einen die eingehenden Parameter auf Korrektheit geprüft werden, zum anderen kann auch eine beliebig komplexe Abfolge von tatsächlichen Methodenaufrufen hinter einem scheinbar simplen, einzigen Aufruf versteckt werden. Dies ermöglicht es, für derartige komplexe Objekte bei Bedarf eine stark vereinfachte Programmierschnittstelle zur Verfügung zu stellen, die dem in Kapitel 4 vorgestellten Entwurfsmuster einer Fassade entspricht. Die Schnittstelle kann dabei auch einer ganz anderen Klasse angehören als das eigentliche Objekt und die Aufrufe lediglich an dieses weiterleiten, wobei nur sie selbst als *XmlRpcHandler* registriert ist.

Alternativ dazu können Klassen, deren Objekte beim *XmlRpcServer* registriert werden sollen, auch die Schnittstelle *AuthenticatedXmlRpcHandler* implementieren. Diese definiert ebenfalls eine Methode *execute*, an die jedoch als zusätzliche Parameter der Name und das Paßwort des aufrufenden Benutzers übergeben werden. Damit kann beispielsweise vor dem Aufruf der eigentlichen Methoden eines Objektes zusätzlich geprüft werden, ob der anfragende Benutzer über die dafür benötigten Zugriffsrechte verfügt. Die Zugangsdaten können aber auch an die einzelnen Methoden weitergegeben werden, wenn diese ihrerseits selbst den Benutzernamen oder das Paßwort benötigen. Name und Paßwort des Benutzers werden unter Verwendung des in RFC 2617 beschriebenen *HTTP Basic Authentication* Mechanismus innerhalb des HTTP Headers des jeweiligen RPC-Requests übertragen.[294]

---

[293] Homepage der Apache Software Foundation: http://www.apache.org. Das XML-RPC-Projekt ist ein Teil des *Apache Web Services Project* mit einer eigenen Homepage unter: http://ws.apache.org/xmlrpc. Die von Hannes Wallnöfer entwickelte XML-RPC-Softwarebibliothek für die Programmiersprache Java steht dort kostenlos zum Download zur Verfügung.

[294] Siehe die Spezifikation der HTTP Basic Authentication unter http://www.ietf.org/rfc/rfc2617.txt.

Ein Beispiel für die Verwendung eines *AuthenticatedXmlRpcHandlers* innerhalb der E-Learning-Plattform stellt die Klasse *DatabaseApplication* dar, die gleichermaßen als Handler und Fassade für das Singleton-Objekt des in Kapitel 5.6.1 spezifizierten Datenbank-Adapters dient. Dieses Objekt, über das alle Anfragen an die Datenbank laufen, wird nicht selbst beim *XmlRpcServer* angemeldet. Statt dessen wird ein Objekt der Klasse *DatabaseApplication* als Handler eingetragen, die ihrerseits die Schnittstelle *AuthenticatedXmlRpcHandler* implementiert. In ihrer Methode *execute* wird zunächst die Korrektheit der übermittelten Benutzerinformationen ermittelt, indem überprüft wird, ob das gesendete Paßwort zu dem angegebenen Benutzer paßt.

Ist dies der Fall, so wird die gewünschte Methode aufgerufen und ihr Rückgabewert als Antwort per XML-RPC an den Client übermittelt. Man beachte, daß sämtliche in Kapitel 5.6.1 definierten abstrakten Methoden der Klasse *DatabaseAdapter* als ersten Parameter eine String-Repräsentation des *UniversalObjectIdentifiers* des aufrufenden Benutzers erwarten. Dieser Parameter wird von der *DatabaseApplication* automatisch aus den über die Basic Authentication übermittelten Daten des Benutzers ermittelt und muß daher nicht bei jedem Aufruf vom Client explizit als Parameter angegeben werden. Dadurch wird die Programmierschnittstelle für die Nutzung der Datenbank vereinfacht. Die DatabaseApplication trägt den Charakter einer Fassade, da sie die tatsächlichen Methodenaufrufe nach außen hin kapselt und einen externen Aufruf auf mehrere interne Aktionen abbildet, nämlich auf die Authentifizierung des Benutzers mit dem dafür erforderlichen Datenbankzugriff sowie die eigentliche Datenbankoperation.

Der folgende XML-Code entspricht einem Aufruf der in Kapitel 5.6.1.8 spezifizierten Methode *getObject*, wobei das gesuchte Objekt das Wurzelverzeichnis des Systems mit dem vordefinierten *UniversalObjectIdentifier* 0:0 ist. Die *DatabaseApplication* wurde in diesem Beispiel zuvor als *AuthenticatedXmlRpcHandler* unter dem Namen `database` registriert. Dieses Beispiel zeigt, daß der eingeloggte Benutzer im Unterschied zur Definition der Methode im *DatabaseAdapter* hier nicht als Parameter in Erscheinung tritt, da diese Information bereits durch die HTTP Basic Authentication übertragen wird:

```
<methodCall>
   <methodName>database.getObject</methodName>
   <params>
      <param>
         <value><id>0:0</id></value>
      </param>
   </params>
</methodCall>
```

Die Verwaltung von Handlern und das Versenden von XML-RPC-Requests wird im Rahmen der E-Learning-Plattform von der Klasse *Transmitter*[295] übernommen. Beim Systemstart wird ein (Singleton-) Objekt dieser Klasse erzeugt, das seinerseits das benötigte Objekt der Klasse *XmlRpcServer* erzeugt und verwaltet.[296] Teilapplikationen, die als Handler für Anfragen über den XML-RPC-Kommunikationsmechanismus agieren sollen, können über die Methode *addRpcApplication* unter einem anzugebenden Namen angemeldet werden.

---

[295] Der vollständige Klassenname lautet: de.monist.communication.Transmitter. Der Name des Paketes resultiert aus der Projektgeschichte des Projektes Monist, die Funktionalitäten der Klasse sind jedoch generell in der E-Learning-Plattform verwendbar.

[296] Dies gilt auch für die *XmlRpcClients* für die Kommunikation mit anderen Instanzen des Programmes.

Hierzu ist es allerdings erforderlich, daß die Applikationen neben einer der beiden Schnittstellen *XmlRpcHandler* oder *AuthenticatedXmlRpcHandler* auch die Schnittstelle *RPCApplication* implementieren. Diese Schnittstelle definiert Methoden, mit denen den Applikationen das *Transmitter*-Objekt, welches für XML-RPC-Aufrufe benötigt wird, sowie der *ParameterManager*, der für die Verwaltung von Systemparametern zuständig ist, übergeben werden können. Applikationen können über eigene Einträge in der Datei *parameters.xml* konfiguriert werden, die vom *ParameterManager* eingelesen wird.[297] Außerdem wird zur Implementation der Schnittstelle *RPCApplication* noch die Methode *needsUserObject* benötigt, die darüber Auskunft gibt, ob zur Verwendung des Handlers ein Benutzer im System eingeloggt sein muß oder nicht. Die entsprechende Methode der Klasse *DatabaseApplication* liefert beispielsweise `true` zurück, da für Operationen auf der Datenbank stets die Daten eines eingeloggten Benutzers benötigt werden.

Beim Systemstart werden automatisch alle diejenigen Applikationen beim Transmitter angemeldet, die in der Konfigurationsdatei *applications.ini* angegeben wurden. Dabei ist jeweils in einer eigenen Zeile zunächst der Name anzugeben, unter dem ein Objekt der Applikationsklasse registriert werden soll, gefolgt vom vollständigen Namen der betreffenden Klasse. Sämtliche Zeilen, die mehr als diese beiden Einträge (Tokens) enthalten, werden ignoriert, so daß zum Abschalten bestimmter Applikationen diese nicht notwendigerweise aus der Konfigurationsdatei entfernt werden müssen, sondern beispielsweise durch Hinzufügen von `[unused]` am Ende der Zeile deaktiviert werden können. Das folgende Beispiel zeigt einen Ausschnitt aus der Konfigurationsdatei *applications.ini* des Monist-Systems. Die Teilanwendungen für die Datenbank und den Navigator sind aktiviert, während der MUD Server abgeschaltet ist:

```
database  de.monist.database.DatabaseApplication
navigator de.monist.navigator.Navigator
mudserver de.monist.mud.MudServerApplication [unused]
```

Beim Systemstart wird jeweils ein Objekt der betreffenden Applikationsklassen erzeugt und unter dem angegebenen Namen als Handler angemeldet. Einschränkend muß dabei gesagt werden, daß dies nur für Klassen möglich ist, die über einen parameterlosen Konstruktor verfügen. Da allerdings die Handlerklassen in den meisten Fällen ohnehin Fassaden sind, die eingehende XML-RPC-Requests auf Methodenaufrufe der dahinter verborgenen, eigentlichen Anwendungsobjekte abbilden, bedeutet dies in der Praxis keine Einschränkung der in die E-Learning-Plattform integrierbaren Applikationen. Im Zweifelsfall muß nur eine entsprechend angepaßte Handler-Klasse als Fassade für eine neue Anwendung erstellt werden.

Zum Ausführen von XML-RPC-Requests wird die Methode *sendRpcRequest* des beim Systemstart generierten Transmitters aufgerufen. Von dieser Methode existieren zwei unterschiedliche Varianten mit vier bzw. sechs Parametern. Beim Aufruf angegeben werden müssen jeweils die IP-Adresse des Computers, an den der Request zu richten ist, der Name, unter dem die anzusprechende Applikation registriert wurde, der Name der Methode, die aufgerufen werden soll, und ein Vektor, der die für diesen Aufruf benötigten Parameter enthält. Als fünfter und sechster Parameter können eine String-Repräsentation des *UniversalObjectIdentifiers* eines Benutzers sowie sein Paßwort angegeben werden. Geschieht dies nicht, werden automatisch der Identifier und das Paßwort des aktuell eingeloggten Benutzers für die Basic Authentication verwendet.

[297] Siehe Kapitel 8.6.

6.3 Übertragung der spezifischen Datentypen für Lernobjekte

Die Liste der als Parameter für den Aufruf von Methoden verwendbaren Objekttypen ist im in der E-Learning-Plattform verwendeten XML-RPC-Kommunikationsmechanismus der Apache Software Foundation sehr beschränkt. In der zum Download erhältlichen Version können lediglich Objekte der Klassen *Integer*, *Double*, *Boolean*, *String* und *Date* sowie Vektoren und Hashtables, die ihrerseits nur Objekte der genannten Klassen enthalten dürfen, verwendet werden. Daneben sind lediglich noch Byte-Arrays zulässig.

Für die Übertragung der im Rahmen der E-Learning-Plattform verwendeten Datentypen, insbesondere von Virtuellen Lernobjekten und ihren Bestandteilen, reichen diese Übertragungsmöglichkeiten jedoch nicht aus. Es wäre zwar sicher möglich, für den Transfer dieser Objekte ein Format zu finden, in dem die speziellen Datenstrukturen auf Kombinationen von Vektoren und Hashtables abgebildet werden. In diesem Fall müßte sich jedoch entweder jede einzelne Anwendung selbst darum kümmern, die Lernobjekte in dieses Datenformat umzuwandeln, oder zwischen den Aufruf eines RPC-Requests durch eine Applikation und das tatsächliche Abschicken dieses Requests müßte ebenso ein zusätzlicher Schritt eingebaut werden wie zwischen das Empfangen des Requests durch den *XmlRpcServer* und seine Weiterleitung an den zuständigen Handler.

In beiden Fällen wäre eine zusätzliche Transformation von Virtuellen Lernobjekten in andere Objekte und zurück erforderlich. Für jede einzelne XML-RPC-Kommunikation wären somit insgesamt vier aufwendige Transformationsschritte erforderlich, bis das Ergebnis des Aufrufs beim anfragenden Client angekommen ist. Dadurch würde die Kommunikation zwischen verschiedenen Installationen der E-Learning-Plattform, aber auch innerhalb der selben Instanz, erheblich verlangsamt. Außerdem würde auf diese Weise noch ein weiteres zu verwaltendes Format für die gleichen Daten geschaffen. Um diesen Aufwand sowie eine unnötige Verkomplizierung des Systems zu vermeiden, wurde statt dessen der vorgegebene XML-RPC-Kommunikationsmechanismus um die Möglichkeit zur Übertragung der benötigten Datentypen erweitert.

Zunächst wurden die zusätzlichen numerischen Datentypen, die in der Metadatentabelle eines Virtuellen Lernobjektes zulässig sind, hinzugefügt. Dabei handelt es sich um die Klassen *Byte*, *Short*, *Long*, *Float*, *BigInteger* und *BigDecimal*. Zum Hinzufügen dieser Klassen waren nur geringfügige Änderungen erforderlich, da es für das Datenformat keinen großen Unterschied macht, ob beispielsweise Objekte der Klassen *Integer* und *Double* als Werte von Methodenparametern durch die Zeilen

```
<value><int>2</int></value> bzw.
<value><double>3.14159265</double></value>
```

oder Objekte der Klassen *Byte* und *Float* durch die Zeilen

```
<value><byte>2</byte></value> bzw.
<value><float>3.14159265</float></value>
```

übertragen werden.

Für die Übertragung von Objekten der Klasse *UniversalObjectIdentifier* wurde das bereits im vorigen Abschnitt verwendete Format definiert:

```
<value><id>12345:67890</id></value>
```

Für die Übertragung von Virtuellen Lernobjekten und Virtuellen Verzeichnissen wurde eine spezielle XML-Struktur geschaffen, die jeweils durch ein äußerstes XML-Element namens `<object>` bzw. `<directory>` begrenzt wird. Innerhalb dieser Elemente werden die einzelnen Datenfelder durch entsprechende Unterelemente dargestellt, und zwar nach folgender Zuordnung:

id	->	`<objectID>`
parents	->	`<parents>`
children	->	`<children>`
metadata	->	`<metadata>`
content	->	`<content>`
readAccessRights	->	`<readaccess>`
writeAccessRights	->	`<writeaccess>`

Man beachte, daß innerhalb dieser Elemente jeweils ein `<value>`-Element auftritt, das den eigentlichen Inhalt des betreffenden Datenfeldes enthält, so daß sich beispielsweise folgender Ausschnitt des XML-Codes für die Übertragung des Datenfeldes *id* ergibt:

```
<objectID><value><id>12345:67890</id></value></objectID>
```

Die Metadatentabelle als Objekt der Klasse *MetadataTable* wird durch das Element `<metadatatable>` dargestellt. Für jeden Eintrag in der Tabelle wird innerhalb dieses Elementes ein `<attribute>`-Element in den XML-String zum Übertragen des Lernobjektes geschrieben. Dieses Element enthält seinerseits ein `<key>`-Element für den Namen sowie ein `<value>`-Element für den Inhalt des betreffenden Metadatums. Das folgende Beispiel zeigt den XML-Code für die Übertragung einer Metadatentabelle mit einem einzigen Eintrag, der den Namen "Title" und den Inhalt "E-Learning" besitzt:

```
<metadatatable>
   <attribute>
      <key>Title</key>
      <value><string>E-Learning</string></value>
   </attribute>
</metadatatable>
```

Ein *MultiValueString* wird durch das Element `<multivaluestring>` dargestellt. Dieses enthält ein Element namens `<items>`, in dem für jeden Eintrag innerhalb des *MultiValueStrings* ein `<value>`-Element enthalten ist. Das folgende Beispiel zeigt den XML-Code für die Übertragung eines MultiValueStrings, der die Einträge "Assam", "Ceylon" und "Darjeeling" enthält:

```
<multivaluestring>
   <items>
      <value><string>Assam</string></value>
      <value><string>Ceylon</string></value>
      <value><string>Darjeeling</string></value>
   </items>
</multivaluestring>
```

Ein *MultiLanguageString* wird durch das Element `<multilanguagestring>` dargestellt. Dieses enthält für jeden der verschiedensprachlichen Einträge ein Element namens `<entry>`. Dieses Element enthält seinerseits ein `<language>`-Element für die jeweilige Sprache sowie ein `<value>`-Element für den Inhalt des für diese Sprache gespeicherten Eintrages. Das folgende Beispiel zeigt den XML-Code für die Übertragung eines *MultiLanguageStrings* mit dem englischen Inhalt "class" und dem deutschen Inhalt "Klasse":

```
<multilanguagestring>
   <entry>
      <language>en</language>
      <value><string>class</string></value>
   </entry>
   <entry>
      <language>de</language>
      <value><string>Klasse</string></value>
   </entry>
</multilanguagestring>
```

Man beachte, daß durch die unterschiedlichen Verwendungen des Elementes `<value>` das Format der zu übertragenden XML-Datei nicht allein durch eine Document Type Definition (DTD) spezifiziert werden kann, da die zulässige Schachtelung der Elemente vom jeweiligen Kontext abhängig ist. Beispielsweise darf innerhalb eines `<value>`-Elementes, das in der XML-Repräsentation eines *MultiValueStrings* auftritt, kein `<metadatatable>`-Element vorkommen, obwohl dieses an anderer Stelle innerhalb eines `<value>`-Elementes stehen muß. Diese Situation ist allerdings nicht erst durch die Erweiterungen entstanden, die für die speziellen Datentypen innerhalb der E-Learning-Plattform vorgenommen wurden, sondern bestand bereits zuvor im unveränderten Mechanismus bei der Übertragung von Vektoren und Hashtables. Die notwendigen Fallunterscheidungen werden jedoch beim Parsen der auf diese Weise erzeugten XML-Dokumente vom verwendeten SAX-Parser in Abhängigkeit vom jeweiligen Kontext getroffen, so daß keine Mehrdeutigkeiten auf der Seite des Empfängers einer solchen XML-Botschaft entstehen können.

Der Vollständigkeit halber sei an dieser Stelle noch erwähnt, daß Vektoren durch das Element `<array>` dargestellt werden, das jeweils genau ein `<data>`-Element enthält, innerhalb dessen die einzelnen im Vektor enthaltenen Objekte in `<value>`-Elementen abgebildet werden. Während die Umsetzung von Vektoren in das XML-Format damit sehr der Vorgehensweise für *MultiValueStrings* ähnelt, entspricht die Abbildung von Hashtables derjenigen von *MultiLanguageStrings*, wie dies auch anhand der internen Datenstrukturen dieser beiden für die E-Learning-Plattform entwickelten Klassen erwartet werden darf. Hashtables werden durch ein Element namens `<struct>` repräsentiert, das für jeden Eintrag in der Tabelle ein `<member>`-Element enthält. Dieses Element enthält seinerseits ein `<name>`-Element für den jeweiligen Schlüssel sowie ein `<value>`-Element für den Inhalt des für diesen Schlüssel gespeicherten Eintrages.

Das folgende Beispiel zeigt den XML-Code für ein virtuelles Verzeichnis mit der ID 12345:67890, das vom Administrator 0:4 im Wurzelverzeichnis 0:0 angelegt wurde und einen zweisprachigen Titel trägt. Es hat keine Kinder und ist für alle Benutzer lesbar, kann aber nur von den beiden vordefinierten Administratoren beschrieben werden. Der Inhalt des Datenfeldes *content* ist `null`, d. h. ihm wurde kein Wert zugewiesen.

```
<directory>
   <objectID><value><id>12345:67890</id></value></objectID>
   <parents>
      <value>
         <array>
            <data>
               <value><string>0:0</string></value>
            </data>
         </array>
      </value>
   </parents>
   <children>
      <value><array><data></data></array></value>
   </children>
   <metadata>
      <value>
         <metadatatable>
            <attribute>
               <key>Author</key>
               <value><id>0:4</id></value>
            </attribute>
            <attribute>
               <key>Title</key>
               <value>
                  <multilanguagestring>
                     <entry>
                        <language>en</language>
                        <value>
                           <string>test directory</string>
                        </value>
                     </entry>
                     <entry>
                        <language>de</language>
                        <value>
                           <string>Testverzeichnis</string>
                        </value>
                     </entry>
                  </multilanguagestring>
               </value>
            </attribute>
         </metadatatable>
      </value>
   </metadata>
   <content><value><null>null</null></value></content>
   <readaccess>
      <value>
         <multivaluestring>
            <items>
               <value><string>0:1</string></value>
            </items>
         </multivaluestring>
      </value>
   </readaccess>
   <writeaccess>
      <value>
         <multivaluestring>
            <items>
               <value><string>0:3</string></value>
               <value><string>0:4</string></value>
            </items>
         </multivaluestring>
   </writeaccess>
</directory>
```

7 Sicht auf die (Meta-) Daten: Der Navigator

Unter Verwendung der in den beiden vorangehenden Kapiteln vorgestellten Techniken können Daten und Metadaten zur Laufzeit des Systems verwaltet, darüber hinaus in der Datenbank bzw. über den integrierten HTTP-Server persistent gespeichert sowie von einem Computer zum anderen übertragen werden. Dadurch ist aber noch keine Möglichkeit für die Benutzer gegeben, auf die im System gespeicherten Inhalte zuzugreifen und Lernobjekte auszuwählen, um diese zu betrachten oder zu editieren sowie mit den dafür vorgesehenen Konfigurationsdateien die in der E-Learning-Plattform integrierten Simulationsprogramme zu starten. Diesen Zugang zu den Inhalten bietet der Navigator, der durch seine Benutzeroberfläche eine Sicht auf die im System abgelegten Virtuellen Lernobjekte mitsamt den gespeicherten Metadaten bietet und auch den Zugriff auf die dazugehörigen Inhalte ermöglicht.[298]

7.1 Navigation in den lokal vorhandenen Datenbeständen

Der Navigator ist genau wie die Datenbankschnittstelle der E-Learning-Plattform eine RPC-Applikation und wird als solche beim Transmitter und über diesen beim XML-RPC-Kommunikationsmechanismus angemeldet. Im Gegensatz zum Datenbank-Adapter, der ausschließlich Anfragen von anderen Bestandteilen des Systems beantwortet und keine eigenen RPC-Requests abschickt, ist der Navigator aufgrund seiner Interaktion mit dem Benutzer eine aktive Systemkomponente, die selbst Requests abschickt und auf diese Weise die Dienste anderer Applikationen, in erster Linie der Datenbank, in Anspruch nimmt.

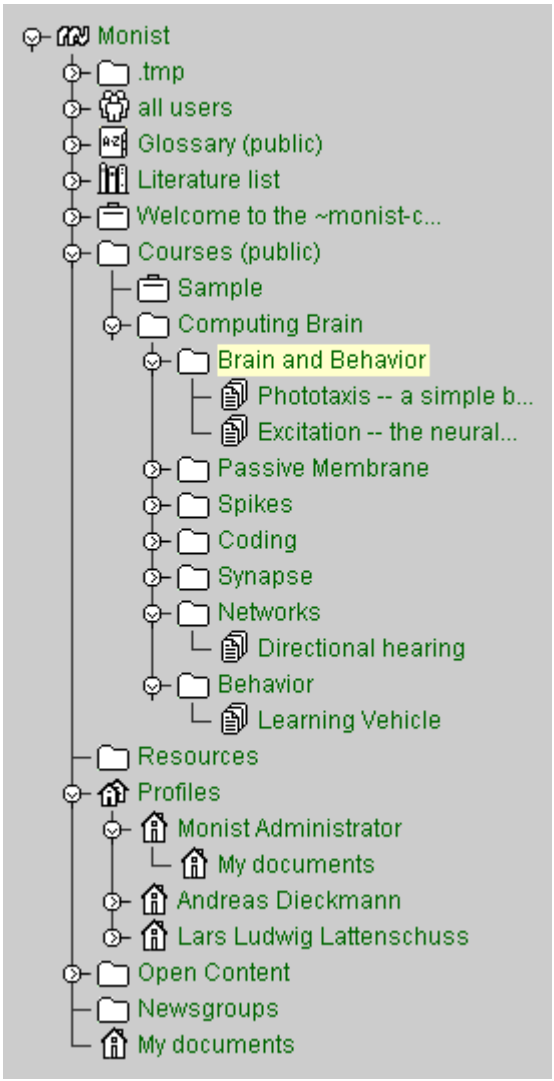

Abbildung 7.1: Screenshot des Verzeichnisbaumes im Navigator

Zentraler Bestandteil des Navigators ist die Darstellung der im System gespeicherten Virtuellen Lernobjekte und Virtuellen Verzeichnisse in Form einer Baumstruktur, die die virtuelle Verzeichnisstruktur wiedergibt. Abbildung 7.1 zeigt einen Screenshot dieses Verzeichnisbaumes aus dem Monist-System. Diese Ansicht enthält ein Beispiel sowohl für die mehrfache Einordnung von Objekten als auch für die Personalisierung des Systems für den eingeloggten Benutzer. Das in der letzten Zeile angezeigte Objekt "My documents" ist ein persönliches Verzeichnis, das für jeden Benutzer existiert. Angezeigt wird jedoch nur das Verzeichnis

[298] An der Entwicklung der Oberfläche des Navigators waren Herr Dipl.-Inform. Markus Oesker und die studentischen Hilfskräfte Martin Lümkemann, Elmar Bomberg, Timo Krause und Jan Sanders beteiligt.

für den aktuell eingeloggten Benutzer, in diesem Fall für den Monist-Administrator. Das selbe Objekt ist in der Abbildung auch fünf Zeilen weiter oben im Verzeichnis dieses Benutzers unter "Profiles" enthalten.

In der angezeigten Verzeichnisstruktur können die Benutzer nach Belieben navigieren, Verzeichnisse öffnen und schließen sowie Objekte auswählen, wobei die erforderlichen Aktionen sich nicht von der in anderen Programmen üblichen Handhabung von Baumstrukturen unterscheiden. Dabei werden stets nur diejenigen Objekte angezeigt, für die der eingeloggte Benutzer über die notwendige Berechtigung zum Lesen verfügt. Dies wird bereits durch die via RPC aufgerufenen Datenbankoperationen sichergestellt. Wie bereits in der Spezifikation des Datenbank-Adapters in Kapitel 5.6.1 dargestellt wurde, berücksichtigen diese stets die Zugriffsrechte des aktuellen Benutzers.

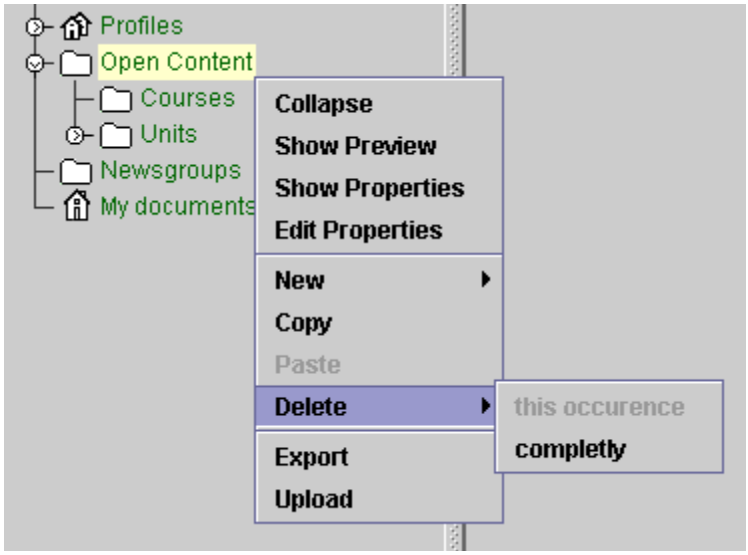

Abbildung 7.2: Screenshot des Kontextmenüs

Die Operationen zum Lesen der Virtuellen Lernobjekte sowie der Listen ihrer Kind- und Elternobjekte aus der Datenbank sind jedoch nicht die einzigen Methoden des Datenbank-Adapters, die im Navigator zugänglich sind. Auch andere Aktionen wie z. B. das Kopieren oder das Löschen von Lernobjekten können ausgeführt werden, sofern der Benutzer über die dafür erforderlichen Rechte verfügt. Zu einem im Baum ausgewählten Objekt stehen alle verfügbaren Methoden über ein Kontext-Menü zur Auswahl, wie der in Abbildung 7.2 gezeigte Screenshot verdeutlicht. Dieses Menü kann über die rechte Maustaste aufgerufen werden. Hier enthält es beispielsweise Einträge für die Methoden zum Kopieren und Löschen des aktuellen Objektes. Beim Löschen kann ausgewählt werden, ob das Objekt nur selektiv aus dem gerade im Baum angezeigten Verzeichnis entfernt oder komplett aus dem System gelöscht werden soll. Die erste der beiden Optionen ist hier deaktiviert, da das betrachtete Objekt nur in einem einzigen Verzeichnis enthalten ist.

Der Navigator präsentiert dem Benutzer darüber hinaus stets eine Vorschauansicht des im Baum gerade ausgewählten Objektes. Für die meisten Objekte werden dabei zwei ausgewählte Metadaten angezeigt, nämlich der Titel (*Title*)[299] und die Beschreibung (*Description*). Dabei wird die eingestellte Sprache des angemeldeten Benutzers berücksichtigt. Die Vorschauansicht enthält außerdem eine Liste der Elternverzeichnisse des ausgewählten Objektes, zu denen über einen Mausklick direkt navigiert werden kann. Über einen Button kann eine Ansicht des gesamten Metadatensatzes geöffnet werden[300], im Edit-Modus steht ein weiterer Button zum Öffnen des Formulars zum Bearbeiten der Metadaten zur Verfügung. Abbildung 7.3 zeigt einen Screenshot des Navigators mit der Vorschauansicht für eine Lehreinheit. Für Grafikdateien enthält die Vorschau zusätzlich eine Ansicht des Inhaltes, wie der entsprechende Screenshot in Abbildung 7.4 zeigt.

[299] Bei Benutzern und Gruppen erscheint an dieser Stelle der Benutzername bzw. der Name der Gruppe.
[300] Zu den Möglichkeiten zum Anzeigen und Editieren von Metadaten siehe den folgenden Abschnitt 7.2.

Abbildung 7.3: Screenshot des Navigators mit Baumstruktur und Vorschauansicht

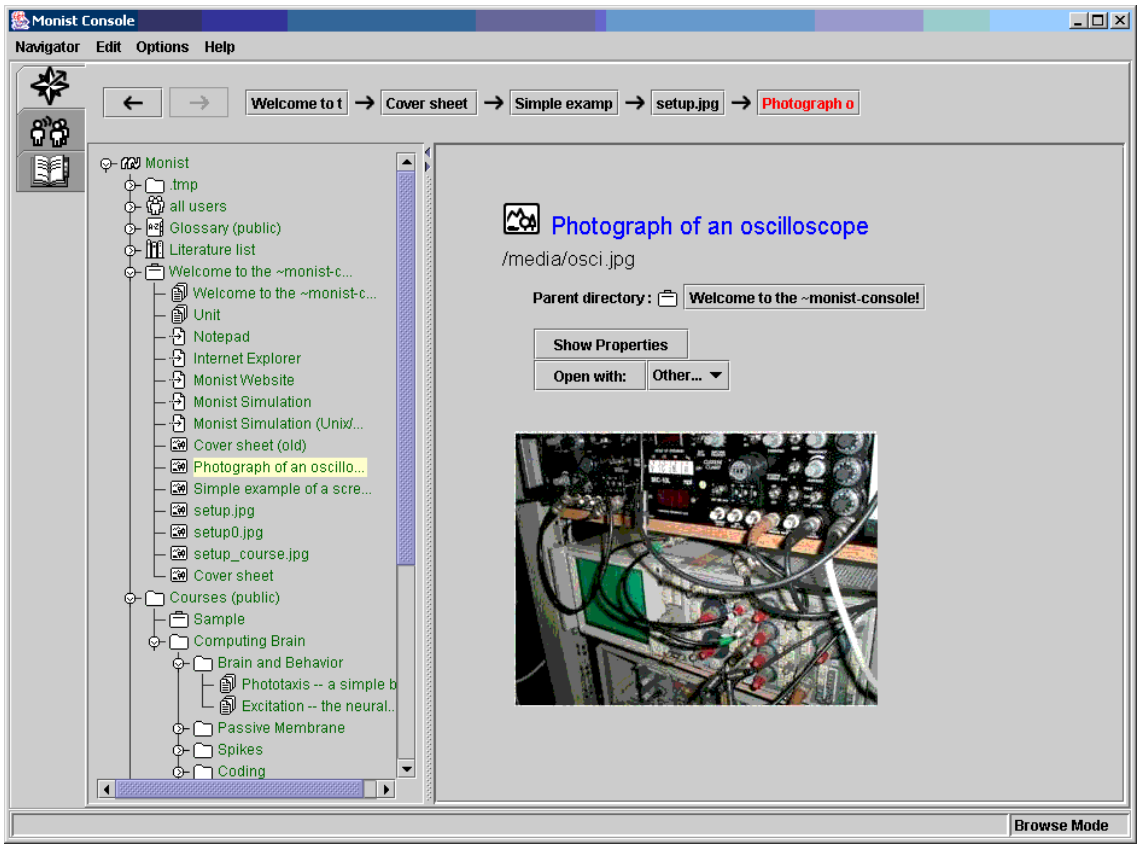

Abbildung 7.4: Screenshot des Navigators mit Vorschauansicht für eine Grafikdatei

Der Navigator dient jedoch nicht nur zur Anzeige von Metadaten und zur Navigation in der virtuellen Verzeichnisstruktur, sondern verwaltet auch Informationen darüber, mit welcher Anwendung die zu den jeweiligen Lernobjekten gespeicherten Inhalte geöffnet werden können. Dabei kann es sich sowohl um externe Applikationen handeln (z. B. Adobe Acrobat Reader zum Betrachten von PDF-Dateien) als auch um die jeweils in die E-Learning-Plattform integrierten Simulationsanwendungen.

Externe Anwendungen werden durch Programmaufrufe nach Maßgabe des jeweiligen Betriebssystems gestartet, wobei der Benutzer die Möglichkeit hat, diese Aktionen nach seinen jeweiligen Bedürfnissen frei zu konfigurieren. Interne Aufrufe werden dagegen in der Regel durch einen XML-RPC-Request behandelt, der vom Navigator an die zuständige Anwendung geschickt wird. Die für die verschiedenen Dateitypen erforderlichen Aufrufe werden in der Konfigurationsdatei *settings.xml* gespeichert.

7.2 Anlegen und Verändern von Objekten und Metadaten

Zu den Funktionalitäten des Navigators zählen auch Möglichkeiten zum Anlegen von neuen Virtuellen Lernobjekten und Virtuellen Verzeichnissen, zum Betrachten und zum Editieren von Metadaten sowie zum Upload von Dateien in das System, um die Objekte mit Inhalten zu versehen. Alle Funktionalitäten, die Änderungen von Objekten in der Datenbank hervorrufen, können nur dann angewählt werden, wenn der eingeloggte Benutzer über die dafür erforderlichen Schreibrechte verfügt. Außerdem muß der Benutzer zum Editieren von Objekten über einen entsprechenden Eintrag im Hauptmenü in den sogenannten Edit-Modus wechseln. Dies ermöglicht unterschiedliche Ansichten für die gleichen Lernobjekte in Abhängigkeit davon, ob diese gerade betrachtet oder editiert werden sollen.

Das Anlegen von neuen Objekten geschieht durch den Aufruf der Funktion *new*, die in Abbildung 7.2 im Screenshot des Kontextmenüs zu sehen ist. Ein Untermenü eröffnet dabei die Möglichkeit zur Auswahl des Typs des anzulegenden Objektes. Welche Typen angezeigt werden, ist abhängig von der Art des Verzeichnisses, in der ein Objekt erstellt werden soll. Beispielsweise können Benutzer nur in Benutzergruppen angelegt werden, während dort keine einfachen Lernobjekte oder Verzeichnisse erstellt werden können.[301]

Nach Auswahl des Typs des anzulegenden Objektes erscheint anstelle der Vorschau ein Formular zur Eingabe der Metadaten des neuen Lernobjektes. Die darin enthaltenen Elemente sind vom jeweiligen Typ abhängig.[302] Das gleiche Formular wird auch zum Verändern der Metadaten von bereits bestehenden Objekten verwendet. Diese Funktion kann im Edit-Modus entweder aus dem Kontextmenü oder über einen entsprechenden Button in der Vorschauansicht aufgerufen werden.

Für die verschiedenen Datentypen, die als Einträge in der Metadatentabelle zulässig sind, existieren jeweils vordefinierte Eingabemöglichkeiten. Beispielsweise gibt es für *MultiLanguageStrings* ein spezielles Oberflächenelement, das in seinem Grundzustand ein Eingabefeld sowie eine Drop-Down-Liste zur Auswahl der Sprache für die Eingabe enthält. Der Button *expand* ermöglicht die Erweiterung des Eingabeelementes um Textfelder für alle verwendeten Sprachen und wird dabei durch den Button *collapse* ersetzt, über den diese Veränderung des Formulars rückgängig gemacht werden kann.

[301] Zu den erforderlichen Schritten zur Definition unterschiedlicher Objekttypen siehe Kapitel 8.2.
[302] Die Festlegung der Metadatensätze für verschiedene Objekttypen wird ebenso in Kapitel 8.2 erläutert.

Abbildung 7.5 zeigt einen Screenshot des Eingabeformulars für die Metadaten, der aus dem Monist-System stammt. Darin sind unter anderem Eingabefelder für mehrere Elemente enthalten, die mehrsprachige Texte enthalten. Bei den Elementen *Description* und *Rights* ist dabei der Grundzustand des Oberflächenelementes für die Eingabe von *MultiLanguageStrings* sichtbar, während beim *Title* dieses Element aufgeklappt ist und getrennte Eingabefelder für mehrere Sprachen (hier nur Englisch und Deutsch) enthält. Daneben zeigt der Screenshot Formularelemente für einsprachige Strings (*Publisher*, *Source*) sowie für *MultiValueStrings*, deren Eingabe hier für *Author*, *Contributors* und *Keywords* in Form von Auswahllisten gestaltet ist. Die Auswahlliste für die *Keywords* ist im Screenshot als eigenes Fenster zu erkennen. Im Monist-System sind Schlüsselwörter eigene Objekte, die als Einträge im Glossar in Erscheinung treten, und werden somit genau wie Benutzer durch eigene *UniversalObjectIdentifier* repräsentiert.

Abbildung 7.5: Screenshot des Eingabeformulars für die Metadaten

Das Metadatenformular wird ebenfalls für den Upload von Inhalten für die Lernobjekte verwendet. Hierzu dient der Button *upload* neben dem Metadatum *File size*. Für jede Sprache kann eine Datei hochgeladen werden, deren Größe dann in der mehrsprachigen *File size* eingetragen wird. Das Eingabeelement für den Upload ähnelt dabei dem für *MultiLanguageStrings*, auch hier kann das Formular mit Hilfe der Buttons *expand* und *collapse* auf die mehrsprachige Variante erweitert und auch wieder auf die ursprüngliche Form reduziert werden. Beim Upload der ersten Datei für eine beliebige Sprache wird automatisch deren MIME Type ermittelt und als Wert des Metadatums *File format* eingetragen. Um Typsicherheit für Lernobjekte herzustellen, können weitere Dateien für dieses Objekt nur dann hochgeladen werden, wenn sie den selben MIME Type haben. Die Dateien können auch wieder gelöscht werden. Nach dem Löschen der Datei für die letzte Sprache wird das Metadatum *File type* wieder zurückgesetzt, so daß anschließend wieder ein Upload von Dateien mit einem anderen MIME Type möglich ist.

Natürlich ist nicht nur das Editieren, sondern auch die Anzeige von Metadaten für ein Virtuelles Lernobjekt möglich. Hierzu kann entweder der Eintrag "*Show properties*" im Kontextmenü (siehe Abbildung 7.2) oder der Button in der Vorschauansicht verwendet werden. Abbildung 7.6 zeigt einen Screenshot der Metadatenanzeige. Mehrsprachige Einträge werden dabei jeweils in der vom aktuell eingeloggten Benutzer eingestellten Sprache angezeigt. Bei *MultiValueStrings* werden alle Elemente angezeigt. Handelt es sich dabei wie im Beispiel der *Contributors* um *UniversalObjectIdentifier*, so können mehrere Attribute der zugehörigen Objekte gezeigt werden. Die Konfiguration der Anzeige von Metadaten für verschiedene Objekttypen wird in Kapitel 8.2 erläutert.

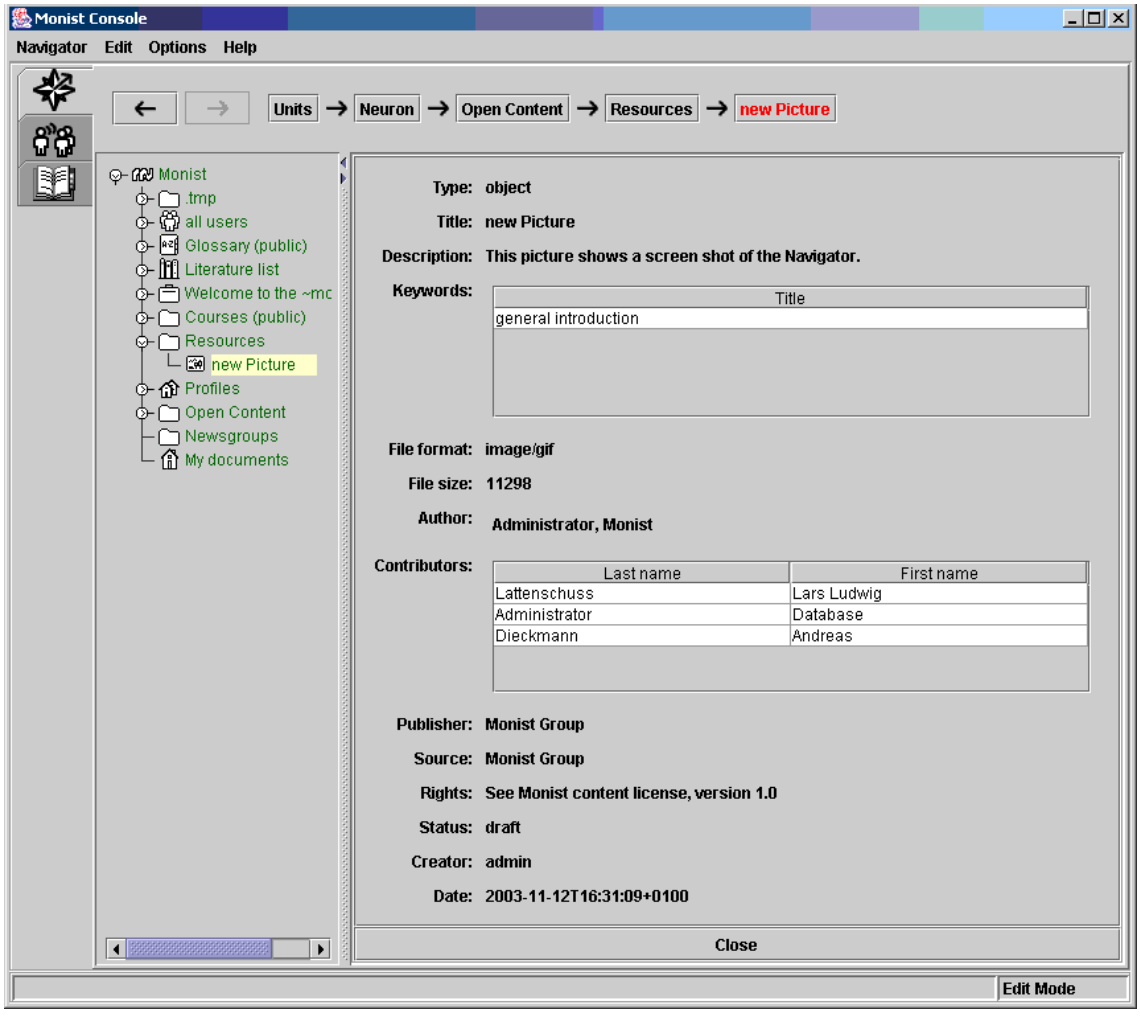

Abbildung 7.6: Screenshot der Metadatenanzeige

7.3 Suche in den lokal vorhandenen Metadaten

Lernobjekte können im Navigator nicht nur durch Navigation in der durch den Baum dargestellten virtuellen Verzeichnisstruktur gefunden werden. Die Benutzeroberfläche stellt auch eine umfangreiche Suchfunktionalität zur Verfügung. Durch den Aufruf des entsprechenden Eintrages im Menü *Navigator* erhält der Benutzer ein Suchformular, das eine kombinierte Suche unter Verwendung der für Lernobjekte definierten Metadaten ermöglicht. Abbildung 7.7 zeigt einen Screenshot des Suchformulars aus dem Monist-System. Die Suchergebnisse werden in einer baumähnlichen Struktur dargestellt, die in der Abbildung links anstelle des üblichen Baumes zu sehen ist, zu dem jedoch über einen Klick auf den entsprechenden Reiter "*Navigator*" zurück gewechselt werden kann.

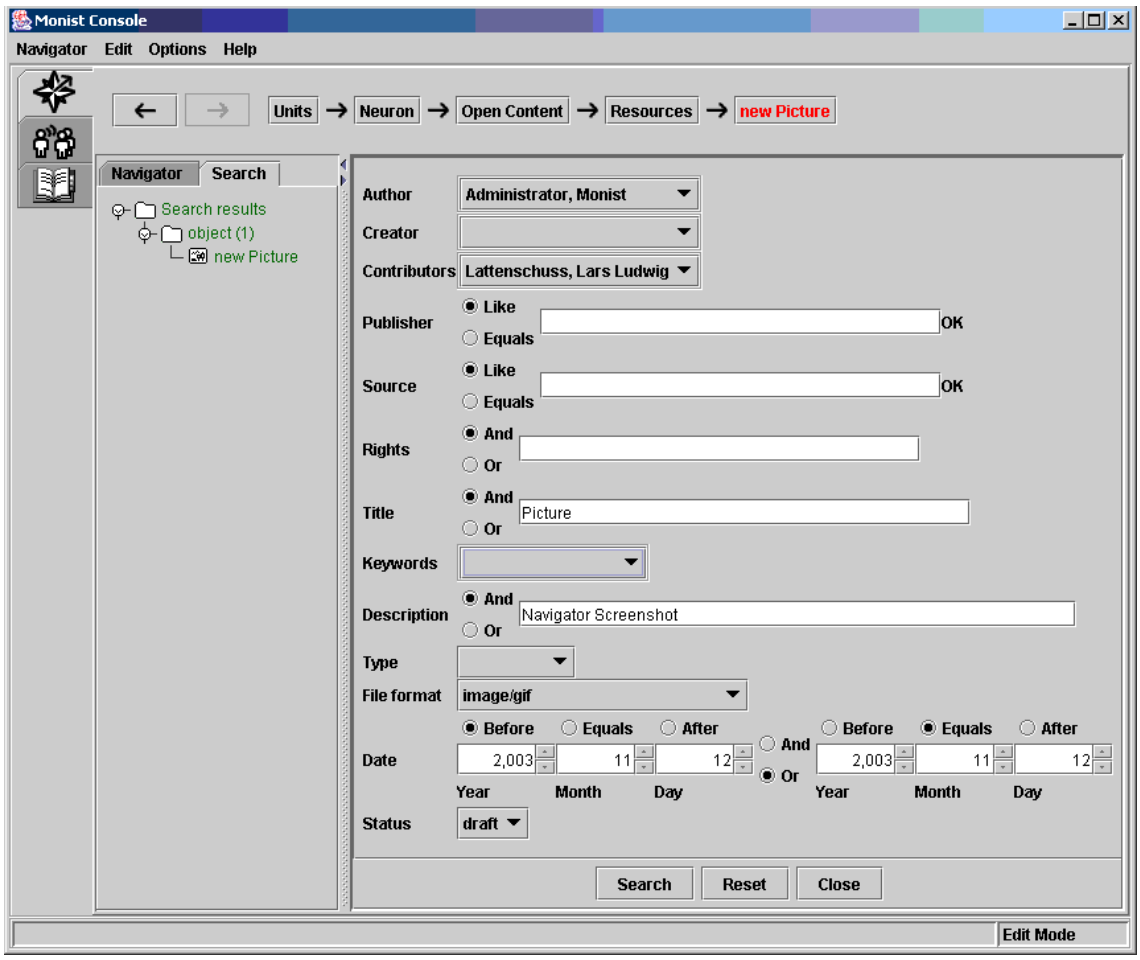

Abbildung 7.7: Screenshot des Suchformulars und der Darstellung der Suchergebnisse

Für die verschiedenen Typen von Metadaten, die als Einträge in der Metadatentabelle eines Virtuellen Lernobjektes zulässig sind, gibt es jeweils vordefinierte Eingabeelemente. Nach textuellen Metadaten, egal ob einsprachig oder mehrsprachig, kann man beispielsweise über Textfelder suchen, in die auch mehrere Wörter geschrieben werden können. Dabei kann ausgewählt werden, ob die Wörter durch einen der beiden logischen Operatoren AND oder OR verknüpft werden sollen, ob also alle abgegebenen Wörter in dem betreffenden Metadatum enthalten sein müssen oder nur eines davon. Für Metadaten, die *UniversalObjectIdentifier* von bestimmten Objekten, beispielsweise Benutzern oder Glossareinträgen, enthalten, kann aus Listen der im System enthaltenen Objekte vom betreffenden Typ ausgewählt werden. Außerdem können Objekte gesucht werden, die vor, an oder nach bestimmten Daten zuletzt bearbeitet wurden.

Welche Eingabeelemente im einzelnen im Suchformular zur Verfügung stehen, ergibt sich aus einer Konfigurationsdatei, die nach Bedarf angepaßt werden kann.[303] Jedes dieser Felder wird intern durch eine entsprechende Datenstruktur für den jeweiligen Datentyp repräsentiert. Diese Objekte generieren beim Abschicken der Suche wiederum eine Ausgabe im korrekten XML-Datenformat, das für den erfolgenden Aufruf der in Abschnitt 5.6.1.18 beschriebenen Suchmethode des Datenbank-Adapters über den XML-RPC-Kommunikationsmechanismus benötigt wird. Auf diese Weise erfolgt auch bei der Suche die Anbindung zwischen der Benutzeroberfläche und den in der Datenbank gespeicherten Metadaten der Virtuellen Lernobjekte.

7.4 Rechnerübergreifende Navigation und Suche

Die Funktionalitäten des Navigators können nicht nur zur Navigation, Bearbeitung und Suche von Lernobjekten innerhalb der Installation der E-Learning-Plattform auf einem einzelnen Computer genutzt werden. Benutzer können sich im Navigator über die Funktion *Connect to Server* auch mit einer Instanz des Programmes auf einem anderen Computer verbinden und dort über das Netzwerk die gleichen Aktionen wie auf ihrem lokalen Rechner vornehmen sowie Dateien zwischen den Computern hin und her kopieren. Die dafür zur Verfügung stehenden Computer werden in der Konfigurationsdatei *servers.ini* angegeben, wobei jede Zeile einen Namen eines Computers einer bestimmten IP-Adresse zuordnet. Zur Auswahl des Rechners steht das in Abbildung 7.8 gezeigte Auswahlmenü mit einer Drop-Down-Liste zur Verfügung.

Abbildung 7.8: Screenshot des Server-Auswahlmenüs

7.5 Mehrsprachigkeit der Benutzeroberfläche

Nicht nur die ins System eingestellten Daten und Metadaten können mehrsprachig sein, sondern auch die Benutzeroberfläche inklusive aller Menüs und Hinweistexte kann für verschiedene Sprachen angepaßt werden. Hierzu werden die in der Programmiersprache Java integrierten Konzepte zur Mehrsprachigkeit verwendet. In allen Klassen, die Texte in der Benutzeroberfläche anzeigen, werden anstelle der eigentlichen Strings Platzhalter verwendet, die durch entsprechende Einträge in Property-Dateien, die für jede gewünschte Sprache separat angelegt werden, in beliebige Sprachen übersetzt werden können.[304] Das Hauptmenü des Navigators enthält unter dem Menüpunkt *Options* unter anderem eine Funktion zum Wechseln der Sprache, die für den eingeloggten Benutzer eingestellt ist. Abbildung 7.9 zeigt das Menü *Options*, das daneben unter anderem auch Einträge zum Ändern des Paßwortes und zum Editieren der Zuordnungen von externen Programmen zu bestimmten Dateitypen enthält.

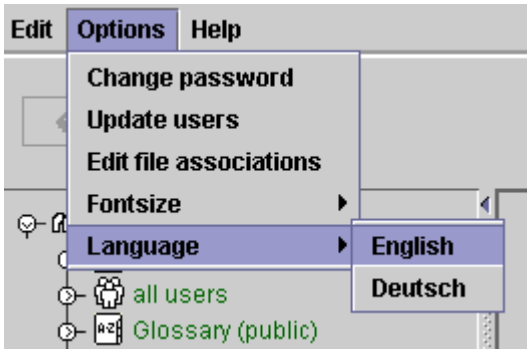

Abbildung 7.9: Screenshot des Options-Menüs

[303] Die Methode zur Angabe der Einträge im Suchformular und ihrer Darstellungsweise für den Benutzer in Form verschiedener Eingabeelemente wird in Kapitel 8.3 erläutert.
[304] Diese Dateien tragen die Endung *.properties* und sind im Rahmen des Monist-Systems zusammen mit allen weiteren Konfigurationsdateien im Verzeichnis *de/monist/config* zu finden. Siehe Kapitel 8.5.

8 Einbindung neuer Simulationsprogramme

In diesem Kapitel wird dargestellt, welche Schritte unternommen werden müssen, wenn weitere Komponenten, in erster Linie Simulationsprogramme für andere Fachbereiche, analog zu den Rahmen des Projektes Monist entwickelten Anwendungen als neue Teilapplikationen in die E-Learning-Plattform eingebunden werden sollen. Zu diesem Zweck ist es einerseits notwendig, bestimmte Schnittstellen zu implementieren, um eine neue Anwendung am XML-RPC-Kommunikationsmechanismus teilhaben zu lassen und sie bei Bedarf in die vorhandene Benutzeroberfläche zu integrieren. Andererseits gibt es eine Reihe von verschiedenen Konfigurationsdateien, in denen beispielsweise die vom System akzeptierten MIME Types, die verwendeten Sprachen für Metadaten und Benutzeroberfläche oder die Einträge des Suchformulars angepaßt werden können. Der jeweilige Aufbau und Zweck dieser Konfigurationsdateien werden in den folgenden Abschnitten ebenfalls beschrieben.

8.1 Implementation der XML-RPC-Schnittstelle und Anmeldung

Um über den XML-RPC-Kommunikationsmechanismus ansprechbar zu sein, sollte eine in die E-Learning-Plattform zu integrierende Anwendung über eine Klasse verfügen, die eine der beiden Schnittstellen *XmlRpcHandler* oder *AuthenticatedXmlRpcHandler* implementiert. Wie in Kapitel 6.2 dargestellt wurde, kann zwar im Prinzip ein Objekt jeder beliebigen Klasse beim XML-RPC-Server angemeldet werden, woraufhin dann für dieses Objekt automatisch ein entsprechender Handler generiert wird. Klassen, die selbst als Handler agieren können, verfügen jedoch über ein erheblich größeres Maß an Kontrolle über die eingehenden Methodenaufrufe. Außerdem ergibt sich auf diese Weise die Möglichkeit, die Handler-Klasse nach dem objektorientierten Entwurfsmuster einer Fassade[305] zu gestalten. Dadurch ist es außerdem möglich, einzubauende Klassen über XML-RPC zugänglich zu machen, ohne diese selbst verändern zu müssen. Es muß lediglich eine entsprechende Fassadenklasse hinzugefügt werden.

Zur Anmeldung beim *Transmitter*, der seinerseits die Registrierung der angemeldeten Klassen beim XML-RPC-Server durchführt, muß die neue Klasse außerdem die Schnittstelle *RPCApplication* implementieren. Wie ebenfalls bereits in Kapitel 6.2 dargelegt wurde, muß sie dafür die Methoden *setTransmitter* und *getTransmitter* zum Setzen und Zurückgeben des *Transmitters* sowie entsprechende Methoden *setParameterManager* und *getParameterManager* für den zur Verwaltung der Systemparameter verwendeten *ParameterManager*[306] definieren. Dementsprechend sollten diese beiden Objekte intern in Datenfeldern gespeichert werden. Der *Transmitter* wird für jeden abzuschickenden RPC-Request benötigt, während die Anwendung den *ParameterManager* zum Einlesen eigener Parameter verwenden kann.

Das Interface *RPCApplication* ist von dem übergeordneten Interface *Application* abgeleitet, das ebenfalls von jeder in das System einzubringenden Anwendung implementiert werden muß. Dies gilt auch für Applikationen, die keinen Gebrauch vom XML-RPC-Kommunikationsmechanismus machen. Die Methoden der Schnittstelle *Application* werden dazu verwendet, beim Systemstart die benötigten Anwendungen zu starten, indem Objekte ihrer Startklasse erzeugt und initialisiert werden.

---

[305] Siehe die Erklärung dieses Entwurfsmusters bei der Darstellung der Systemarchitektur in Kapitel 4.
[306] Das Vorgehen zum Setzen und Einlesen von Systemparametern wird in Kapitel 8.6 beschrieben.

Zu diesem Zweck definiert das Interface die Methode *init* zum Initialisieren einer Anwendung vor ihrem eigentlichen Start, der durch den später erfolgenden Aufruf der Methode *start* erfolgt. Bei der Initialisierung können beispielsweise Systemparameter eingelesen oder Variablen vorbelegt werden, die für den korrekten Ablauf des Startes der Applikation erforderlich sind. Umgekehrt dient die Methode *stop* zum Beenden der Anwendung, während die Methode *destroy* beim Verlassen des Programmes aufgerufen wird und so eine letzte Gelegenheit bietet, notwendige Aufräumarbeiten zu erledigen. Bei einem Aufruf von *queryStop* soll die Anwendung darüber Auskunft geben, ob sie zum Beenden bereit ist. Darüber hinaus enthält die Schnittstelle *Application* Methoden *setName* und *getName*, um ihr einen Namen zu geben und diesen wieder auszulesen, sowie *getVersion* und *getBuildDate*, die zu Verwaltungszwecken dienen und mit denen bei Bedarf festgestellt werden kann, ob die auf dem Computer eines Benutzers installierte Version einer Applikation eventuell veraltet ist.

Welche Klassen beim Systemstart beim Transmitter angemeldet, instanziiert und durch Aufruf der Methoden init und start auf den so erzeugten Objekten initialisiert und gestartet werden sollen, muß in der Konfigurationsdatei *applications.ini* angegeben werden. In jeder Zeile dieser Datei wird ein Name angegeben, der einen Klassennamen zugeordnet wird. Das beim Start neu erzeugte Objekt dieser Klasse wird unter der in der Datei angegebenen Bezeichnung registriert und kann anschließend unter diesem Namen über den XML-RPC-Kommunikationsmechanismus angesprochen werden.[307]

8.2 Neue Datentypen und Definition von Metadatensätzen

In Kapitel 2.2 wurde eine Möglichkeit zur Angabe unterschiedlicher Metadatensätze für verschiedene Datentypen gefordert. Dies ist erforderlich, um für Objekte entsprechend ihrer Rolle im System angepaßte Metadaten verwalten zu können. Zum Beispiel müssen für Benutzer oder Gruppen ganz andere Informationen verwaltet werden als für Kurse, Lehreinheiten oder Konfigurationen für Simulationsprogramme. Daneben können noch beliebig komplexe weitere Unterscheidungen erforderlich sein. Im Monist-System sind beispielsweise Glossareinträge und Literaturverweise eigenständige Lernobjekte, die speziellen Datentypen angehören, für die unterschiedliche Metadaten verwaltet werden.

Die im System verwalteten Datentypen und ihre jeweiligen Metadaten werden jeweils über eine eigene Konfigurationsdatei verwaltet, die in einem bestimmten XML-Format vorliegen muß. Diese Dateien sind in einem eigenen Verzeichnis namens *metadatatypes* gesammelt.[308] Sie tragen jeweils den Namen des von ihnen definierten Datentyps sowie die Endung *.xml* und werden beim Systemstart automatisch eingelesen. Die Definition der Metadaten des Objekttyps *user* sind beispielsweise in einer Datei namens *user.xml* enthalten. Die Menge der Datentypen kann beliebig erweitert werden, womit auch die Möglichkeit gegeben ist, für neu in das System zu integrierende Anwendungen eigene Typen zu definieren, die über einen von den neuen Applikationen verwendeten Satz von Metadaten verfügen.

---

[307] Im Monist-System wird über die Konfigurationsdatei applications.ini tatsächlich nur festgelegt, welche Klassen unter welchem Namen beim XML-RPC-Mechanismus angemeldet werden. Die zu startenden Applikationen werden hingegen in einer XML-Datei festgelegt, welche von der projektspezifischen Startklasse *de.Monist.MonistStarter* eingelesen wird. Für einen Einsatz der E-Learning-Plattform empfiehlt es sich, diese Startklasse durch eine eigene Klasse zum Aufrufen des Programmes zu ersetzen. Zumindest muß jedoch eine entsprechende, eigene Konfigurationsdatei anstelle der im Monist-System verwendeten XML-Starterdatei eingebunden werden.

[308] Im Monist-System handelt es sich dabei um das Verzeichnis *de/monist/config/metadatatypes*.

Das XML-Format der Metadaten-Konfigurationsdateien sieht zunächst ein äußerstes XML-Element namens `<metadata>` vor. Innerhalb dieses Elementes können beliebig viele `<attribute>`-Elemente enthalten sein, die jeweils ein einzelnes Metadatum des Objekttyps definieren. Jedes dieser Elemente enthält die folgenden drei Attribute: `key` bezeichnet den Namen des Metadatums, `class` den vollständigen Java-Klassennamen, durch den der Datentyp des Feldes festgelegt wird, und `style` die Art und Weise, in der dieses Feld dem Benutzer in Formularen zum Eingeben oder Ändern von Metadaten präsentiert wird. Nur Strings, numerische Typen (Subklassen von *java.lang.Number*, beispielsweise *Integer*), boolesche Werte (*java.lang.Boolean*), *MultiLanguageStrings*, *MultiValueStrings* und *UniversalObjectIdentifier* können als Inhalte von Medatatenfeldern verwendet werden. Ihre Namen sind beliebig. Die Metadaten werden stets in der gleichen Reihenfolge angezeigt, in der sie in der Konfigurationsdatei definiert wurden. Das Attribut `style` kann einen der folgenden fünf Werte annehmen:

1.) `style="input"`: Das normale Eingabefeld für den angegebenen Datentyp. Es wird kein voreingestellter Wert angegeben. Ein Beispiel aus der Konfigurationsdatei für Lernobjekte im Monist-System ist die folgende Definition des Eingabeelementes für das Metadatum *Title*:

```
<attribute key="Title"
           class="de.monist.datamanager.MultiLanguageString"
           style="input">
</attribute>
```

2.) `style="default"`: Dem Metadatenfeld wird ein voreingestellter Wert zugewiesen, der durch die Formulare zum Eingeben und Ändern von Metadaten nicht verändert werden darf. Der vorgegebene Wert muß in einem `<default>`-Element innerhalb des umschließenden `<attribute>`-Elementes angegeben werden, wie das folgende Beispiel aus dem Monist-System zeigt:

```
<attribute key="Status"
           class="java.lang.String"
           style="default">
   <default>draft</default>
</attribute>
```

3.) `style="select"`: Der Wert des Metadatums muß hier aus einer vorgegebenen Menge von Werten ausgewählt werden. Diese werden dem Benutzer als Einträge in einer Auswahlliste präsentiert. Die Liste kann die tatsächlichen Werte jedoch wie in HTML-Formularen durch aussagekräftigere Benennungen maskieren. Das Element `<attribute>` muß hier ein `<select>`-Element enthalten, das über ein Attribute namens `preset` verfügen muß, das einen zu Beginn vorgewählten Wert für die Auswahlliste enthält. Innerhalb des `<select>`-Elements muß für jeden auswählbaren Wert des Metadatums ein `<option>`-Element angegeben werden. Jedes dieser Elemente muß ein Attribut namens `value` aufweisen, das den möglichen Wert des Feldes beinhaltet, und kann daneben auch ein Attribute `name` enthalten, das einen aussagekräftigeren Namen für diesen Eintrag enthält, der dem Benutzer im Formular angezeigt werden kann. Fehlt dieses Attribut, wird statt dessen der tatsächliche Wert des Listeneintrages angezeigt. Das folgende Beispiel aus dem Monist-System zeigt die Angabe des Metadatums *Difficulty*, für das Werte aus einer Liste ausgewählt werden, die intern durch Zahlen repräsentiert werden:

```
<attribute key="Difficulty"
           class="java.lang.Byte"
           style="select">
   <select preset="3">
      <option name="very easy" value="1"/>
      <option name="easy" value="2"/>
      <option name="medium" value="3"/>
      <option name="difficult" value="4"/>
      <option name="very difficult" value="5"/>
   </select>
</attribute>
```

4.) `style="autodetect":` Einige Metadatenfelder können Werte enthalten, die nicht vom Benutzer eingegeben oder verändert werden sollen, sondern statt dessen vom System automatisch festgestellt werden müssen. Der zu ermittelnde Wert muß in einem Element namens `<autodetect>` innerhalb des umfassenden Elementes `<attribute>` angegeben werden. Die folgenden Einträge können vom System automatisch ermittelt werden:

- `system.date:` Das aktuelle Datum inklusive der Uhrzeit in dem im Rahmen der E-Learning-Plattform verwendeten Format, das dem als ISO 8601 festgelegten Standard für Datumsangaben entspricht.

- `currentuser.id:` Der *UniversalObjectIdentifier* des aktuell eingeloggten Benutzers, der die Metadaten unter Verwendung des Formulars bearbeitet.

- `file.size:` Die Größe der Datei (in Bytes), die sich gerade im Fokus des Upload-Formulars befindet. Dieser Wert ändert sich automatisch, wenn eine andere Datei ausgewählt wird.

- `file.format:` Der MIME Type der Datei, die sich gerade im Fokus des Upload-Formulars befindet. Dieser Wert ändert sich ebenfalls automatisch, wenn eine andere Datei ausgewählt wird.

- `form.purpose:` Der Typ des Lernobjektes, das gerade durch das Upload-Formular erzeugt wird bzw. dessen Metadaten bearbeitet werden. Diese Angabe entspricht dem Namen der jeweils gerade verwendeten Konfigurationsdatei für das Metadatenformular.

Das folgende Beispiel zeigt den aus dem Monist-System entnommenen XML-Code für die Definition des Metadatenfeldes *Date*, für das automatisch aus der Systemzeit das aktuelle Datum und die Uhrzeit ermittelt werden:

```
<attribute key="Date"
           class="java.lang.String"
           style="autodetect">
   <autodetect>system.date</autodetect>
</attribute>
```

5.) `style="list":` Der Wert des Metadatums muß hier aus einer Auswahlliste gewählt werden, die alle in der Datenbank enthaltenen Objekte eines bestimmten Typs enthält, der durch den jeweiligen Wert des Metadatums *Type* ermittelt wird. Zusätzliche Informationen müssen in einem Element namens `<list>` hinzugefügt werden, das innerhalb des umschließenden `<attribute>`-Elementes enthalten ist. Der betreffende Wert des Metadatums *Type* muß innerhalb des Elementes `<list>` in einem einzelnen `<type>`-Element enthalten sein. Das tatsächliche Erscheinungsbild der Objekte in der Auswahlliste kann folgendermaßen konfiguriert werden: Die Objekte können durch die Anzeige beliebiger Metadaten repräsentiert werden, die in einem oder mehreren `<display>`-Elementen spezifiziert werden können. Wenn mehrere dieser Elemente vorhanden sind, werden die betreffenden Metadaten in der gleichen Reihenfolge angezeigt, in der sie in der XML-Konfigurationsdatei genannt sind. Wenn kein anzuzeigendes Metadatum auf diese Weise angegeben wird, dann wird statt dessen der Wert des Metadatums *Title* angezeigt, sofern dieses vorhanden ist. Anderenfalls enthält die Auswahlliste automatisch den *UniversalObjectIdentifier* des Objektes. Der tatsächliche Wert, der in das betreffende Feld in der Metadatentabelle eingetragen wird, ist in jedem Fall der *UniversalObjectIdentifier* des vom Benutzer ausgewählten Objektes. Das folgende Beispiel zeigt die Definition des Eingabeelementes für die *Contributors* für Lernobjekte im Monist-System, wobei in der Liste Benutzer (*Type* = *user*) zur Auswahl stehen, für die jeweils die Metadaten für den Nachnamen (Last name) und Vornamen (*First name*) angezeigt werden:

```
<attribute key="Contributors"
           class="de.monist.datamanager.MultiValueString"
           style="list">
   <list>
      <type>user</type>
      <display>Last name</display>
      <display>First name</display>
   </list>
</attribute>
```

Unabhängig von der Definition der innerhalb der E-Learning-Plattform verwendeten Typen von Objekten kann außerdem angegeben werden, welche Dateiformate vom System verwaltet werden sollen. Hierzu muß in der Konfigurationsdatei *mimetypes.ini* eine Liste der zu verwendenden MIME Types angegeben werden, wobei in jeder Zeile der Datei jeweils zuerst der Name des betreffenden MIME Types stehen muß, gefolgt von einer oder mehreren Dateinamen-Endungen, die diesem Typ zugeordnet werden sollen. Die erste genannte Endung wird dabei als Standard-Endung für den betreffenden MIME Type interpretiert. Das Format der Angaben wird durch den folgenden Auszug aus der Datei *mimetypes.ini* aus dem Monist-System verdeutlicht:

```
application/octet-stream exe com
application/pdf pdf
image/gif gif
image/jpeg jpg jpeg jpe jfif
image/png png
image/tiff tif tiff
text/html html htm htx htmls
text/plain txt
text/java java
text/xml xml
```

Die in der Konfigurationsdatei angegebenen MIME Types werden beim Systemstart durch den *ParameterManager* eingelesen und an die Klasse *Mime* übergeben, die zur Laufzeit des Systems Informationen über die dem System bekannten Dateitypen liefert. Außerdem verwaltet diese Klasse für die einzelnen Dateitypen auch Icons, die beispielsweise in der im Navigator verwendeten Baumstruktur angezeigt werden können, aber ohne weiteres auch von anderen Teilen des Systems verwendet werden können. Zu den angegebenen MIME Types werden beim Systemstart im Verzeichnis *resources/icons* die entsprechenden Icons gesucht und im Erfolgsfall in Objekte der Klasse *ImageIcon* geladen, die von der Klasse *Mime* anderen Klassen zur Verfügung gestellt werden.

Die einzelnen Dateien müssen dabei vom Typ *.png* sein und als Dateinamen den Namen des zugehörigen MIME Types tragen, wobei die Gruppenbezeichnungen der MIME Types wie z. B. `image/` oder `text/` durch Unterverzeichnisse repräsentiert werden. Das Icon für den MIME Type `text/plain` wird also beispielsweise vom System im Verzeichnis *resources/icons/text/* in der Datei *plain.png* gesucht. Neben den normalen Icons, die üblicherweise eine Größe von 32 mal 32 Bildpunkten aufweisen, werden für eine verkleinerte Darstellung wie im Verzeichnisbaum des Navigators zusätzlich auch kleinere Icons geladen, in deren Dateinamen vor der Endung *.png* noch *.small* eingefügt werden muß. Für den MIME Type text/plain wird das verkleinerte Icon beispielsweise in der Datei resources/icons/text/plain.small.png erwartet.

Wenn neue MIME Types in der Konfigurationsdatei *mimetypes.ini* hinzugefügt werden, empfiehlt es sich, entsprechende Icon-Dateien in die dafür vorgesehenen Verzeichnisse einzustellen. Anderenfalls werden Dateien mit einem der neuen MIME Types durch voreingestellte Icons dargestellt, die einen unbekannten Typ repräsentieren. Wenn die in der E-Learning-Plattform verwendeten Objekttypen im Navigator ebenfalls durch spezielle Icons repräsentiert werden sollen, kann dies dadurch geschehen, daß man ihnen eigene MIME Types zuweist. Im Monist-System wurde von dieser Möglichkeit für verschiedene Typen der von der Simulationssoftware verwendeten Lernobjekte Gebrauch gemacht, wie folgender Auszug aus der Datei *mimetypes.ini* zeigt:

```
application/x-monist-directory mvd
application/x-monist-object mob
application/x-monist-group mgr
application/x-monist-user mdu
application/x-monist-glossary mgl
application/x-monist-keyword mkw
application/x-monist-literature mli
application/x-monist-reference mrf
application/x-monist-hyperlink mhl
application/x-monist-course mco
application/x-monist-chapter mch
application/x-monist-unit mse mun
```

Die Klasse Mime stellt eine Reihe von statischen Methoden zur Verfügung, mit der auf die beim Systemstart eingelesenen MIME Types und Icons zugegriffen werden kann. Die Methode *getKnownMimeTypes* liefert einen Vektor mit den Namen sämtlicher dem System bekannter MIME Types zurück. Zum gegenseitigen Abbilden der Dateinamen-Endungen und MIME Types dienen die beiden Methoden *getExtensionForMIME* und *getMIMEforExtension*, wobei ebenfalls nur die beim Start der E-Learning-Plattform eingelesenen Typen und Endungen berücksichtigt werden.

Die Methoden *getIconForMIME* und *getIconForExtension* liefern zu einem angegebenen Typ bzw. dessen Dateinamen-Endung das entsprechende *ImageIcon* in der normalen Größe zurück (oder an seiner Stelle das normale Icon für unbekannte Typen), während die Methoden *getSmallIconForMIME* sowie *getSmallIconForExtension* analog das verkleinerte *ImageIcon* (oder statt dessen die verkleinerte Version des Icons für unbekannte Typen) ausgeben.

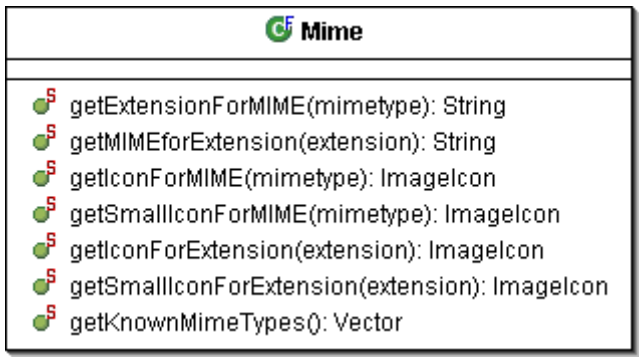

Abbildung 8.1: Klassendiagramm der Klasse Mime (mit Methoden-Parametern und Rückgabetypen)

8.3 Anpassung der Metadatensuche

Das Formular für die Suche in den Metadaten der im System enthaltenen Lernobjekte kann bei Bedarf ebenfalls angepaßt werden. In ähnlicher Weise wie die Formulare zum Eingeben der Metadaten für die unterschiedlichen Objekttypen wird auch der Aufbau der Suche durch eine Konfigurationsdatei gesteuert, die den Namen *search.xml* trägt. In dieser Datei werden die Einträge des Suchformulars in einem speziellen XML-Format spezifiziert, das in seinem Aufbau dem der Metadaten-Konfigurationsdateien ähnelt.

Dieses XML-Format sieht zunächst ein äußerstes XML-Element namens `<search>` vor. Innerhalb dieses Elementes können beliebig viele Elemente vom Typ `<element>` enthalten sein, die jeweils einen einzelnen Eintrag des Suchformulars für ein zu durchsuchendes Metadatum definieren. Jedes dieser Elemente enthält die folgenden drei Attribute: `key` bezeichnet den Namen des Metadatums, `class` den vollständigen Java-Klassennamen, durch den der Datentyp des Feldes festgelegt wird, und `style` die Art und Weise, in der dieses Feld dem Benutzer im Suchformular präsentiert wird.

Nur Strings, numerische Typen (Subklassen von *java.lang.Number*, z. B. *Integer*), boolesche Werte (*java.lang.Boolean*), *MultiLanguageStrings*, *MultiValueStrings* und *UniversalObjectIdentifier* können als Inhalte von Medatatenfeldern verwendet werden, wodurch die zu suchenden Werte entsprechend eingegrenzt werden. Die Einträge für die verschiedenen Metadaten werden im Suchformular stets in der gleichen Reihenfolge angezeigt, in der sie in der Konfigurationsdatei definiert wurden. Das Attribut `style` kann einen der folgenden sechs Werte annehmen:

1.) `style="input"`: Erzeugt ein einfaches Eingabefeld für den angegebenen Datentyp. Außerdem kann eine Reihe von Komparatoren angegeben werden, zwischen denen der Benutzer auswählen kann, um unterschiedliche Vergleichsoperationen für die Suche zur Anwendung zu bringen. Um diese Funktionalität zu nutzen, muß innerhalb des umfassenden Elementes `<element>` ein `<comparators>`-Element angegeben werden. Dieses muß über ein Attribut `default` verfügen, das einen voreingestellten Komparator angibt. Für jeden Komparator, der auswählbar sein soll, muß im `<comparators>`-Element ein Element `<comparator>` angegeben werden, der ein Attribute `name` enthält, das den Namen des Komparators definiert.

Wird kein Komparator angegeben, dann wird als Voreinstellung der Komparator `eq` (equals) benutzt. Als Vergleichsmöglichkeiten können alle in Kapitel 5.6.1.18 angegebenen Komparatoren angegeben werden, die die Art des vorzunehmenden Vergleiches zwischen dem gesuchten Wert und den in der Datenbank enthaltenen Einträgen spezifizieren: `lt` (less than), `le` (less or equal), `eq` (equal), `ne` (not equal), `ge` (greater or equal), `gt` (greater than) oder `like`. Das folgende Beispiel zeigt den aus dem Monist-System entnommenen XML-Code für die Definition eines Suchfeldes für das Metadatum *Learning Time*, das die zum Bearbeiten eines Lernobjektes erforderliche Zeit angibt. Der Benutzer kann dabei angeben, ob die gesuchte Dauer kleiner, gleich oder größer als der im Formular eingegebene Wert sein soll:

```
<element key="Learning time"
         class="java.lang.Integer"
         style="input">
   <comparators default="eq">
      <comparator name="lt"/>
      <comparator name="eq"/>
      <comparator name="gt"/>
   </comparators>
</element>
```

2.) `style="select":` Der Wert für das zu durchsuchende Metadatum muß hier aus einer Liste von vorgegebenen Werten gewählt werden, die dem Benutzer in einer Auswahlliste angezeigt werden. Die Liste kann die tatsächlichen Werte jedoch wie in HTML-Formularen durch aussagekräftigere Benennungen maskieren. Das Element `<element>` muß ein `<select>`-Element enthalten. Dabei wird jedoch kein Wert als vorgewählt deklariert. Innerhalb des `<select>`-Elements muß für jeden auswählbaren Wert des Metadatums ein `<option>`-Element angegeben werden. Jedes dieser Elemente muß ein Attribut namens `name` aufweisen, das den möglichen Wert des Metadatums beinhaltet, der dem Benutzer in der Auswahlliste angezeigt wird. Daneben kann auch ein Attribut `value` angegeben werden, das für den tatsächlichen Wert hinter dem angezeigten Listeneintrag steht. Existiert dieses Attribut, dann wird sein Wert für die Suche in der Datenbank verwendet, ansonsten der Wert des Attributes `name`. Außerdem können in der gleichen Weise wie im Fall `style="input"` auswählbare Komparatoren angegeben werden.

Das folgende Beispiel zeigt den aus dem Monist-System entnommenen XML-Code für die Definition eines Suchfeldes für das Metadatum *Difficulty*, das den Schwierigkeitsgrad eines Lernobjektes angibt. Da in diesem Fall der direkte Vergleich entscheidend ist, wird hier kein Komparator angegeben:

```
<element key="Difficulty"
         class="java.lang.Byte"
         style="select">
   <select>
      <option name="very easy" value="1"/>
      <option name="easy" value="2"/>
      <option name="medium" value="3"/>
      <option name="difficult" value="4"/>
      <option name="very difficult" value="5"/>
   </select>
</element>
```

3.) `style="mime":` Der Wert des Suchfeldes muß aus einer Liste ausgewählt werden, die alle im System bekannten MIME Types enthält. In diesem Fall enthält das Element `<element>` keine weiteren Unterelemente:

```
<element key="File format"
         class="java.lang.String"
         style="mime"/>
```

4.) `style="list":` Der Wert des Suchfeldes muß hier aus einer Liste ausgewählt werden, die alle in der Datenbank enthaltenen Objekte eines bestimmten Typs enthält, der durch den jeweiligen Wert des Metadatums *Type* ermittelt wird. Zusätzliche Informationen müssen in einem Element namens `<list>` hinzugefügt werden, das innerhalb des umschließenden `<element>`-Elementes enthalten ist. Der betreffende Wert des Metadatums *Type* muß innerhalb des Elementes `<list>` in einem einzelnen `<type>`-Element enthalten sein. Das tatsächliche Erscheinungsbild der Objekte in der Auswahlliste kann folgendermaßen konfiguriert werden: Die Objekte können durch die Anzeige beliebiger Metadaten repräsentiert werden, die in einem oder mehreren `<display>`-Elementen spezifiziert werden können.

Wenn mehrere dieser Elemente vorhanden sind, werden die betreffenden Metadaten in der gleichen Reihenfolge angezeigt, in der sie in der XML-Konfigurationsdatei genannt sind. Wenn kein anzuzeigendes Metadatum auf diese Weise angegeben wird, dann wird statt dessen der Wert des Metadatums *Title* angezeigt, sofern dieses vorhanden ist. Anderenfalls enthält die Auswahlliste den *UniversalObjectIdentifier* des Objektes. Die Liste wird dem Benutzer immer in aufsteigender alphabetischer Ordnung angezeigt. Der tatsächliche Wert, der für die Suche in den in der Datenbank gespeicherten Metadaten verwendet wird, ist in jedem Fall der *UniversalObjectIdentifier* des vom Benutzer ausgewählten Objektes.

Das folgende Beispiel zeigt den XML-Code für die Definition eines Suchfeldes für das Metadatum *Author*, wobei in der Liste Benutzer (*Type* = *user*) zur Auswahl stehen, für die jeweils die Metadaten für den Nachnamen (Last name) und Vornamen (*First name*) angezeigt werden:

```
<element key="Author"
         class="de.monist.util.UniversalObjectIdentifier"
         style="list">
   <list>
      <type>user</type>
      <display>Last name</display>
      <display>First name</display>
   </list>
</element>
```

5.) `style="text":` Erzeugt ein Eingabefeld für die Suche nach einem oder mehreren Strings in einem Metadatenfeld vom Typ *MultiValueString*. Jedes der Wörter, die der Benutzer eingibt, wird in der Datenbank in dem betreffenden Metadatum in allen Sprachen gesucht, die dem System bekannt sind. Die einzelnen Wörter können dabei wahlweise mit einem der beiden Operatoren AND oder OR logisch verknüpft werden, was bedeutet, daß der Wert des Metadatums bei einem Objekt, das bei der Suchanfrage ausgegeben wird, entweder alle oder nur mindestens eines der angegebenen Wörter enthalten muß.

Die Reihenfolge der beiden Operatoren kann dabei ebenso angegeben werden wie der beim Aufruf des Suchformulars voreingestellte Operator, und die Operatoren können auch weggelassen werden. Das umschließende Element `<element>` kann ein einzelnes `<operators>`-Element enthalten, innerhalb dessen einer der beiden folgenden XML-Ausdrücke stehen kann, oder auch beide:

```
<operator name="and"/>
<operator name="or"/>
```

Die Reihenfolge, in der die beiden Operatoren im Suchformular angezeigt werden, entspricht der Abfolge der Elemente in der XML-Datei. Der zuerst angegebene Operator dient dabei stets als Voreinstellung. Wenn nur ein Operator eingetragen ist, wird nur dieser im Suchformular angezeigt, und der andere kann nicht ausgewählt werden. Ungültige `<operator>`-Elemente werden komplett ignoriert. Wenn kein gültiges `<operator>`-Element angegeben ist oder das Element `<operators>` komplett fehlt, tritt die folgende Voreinstellung in Kraft: Beide Operatoren werden angezeigt, wobei der Operator AND an erster Position steht und als vorgewählter Operator im Suchformular erscheint.

Das folgende Beispiel zeigt die Definition des Suchfeldes für das Metadatum *Title*. Dabei sind beide Operatoren zur Auswahl angegeben, und der Operator AND dient als Voreinstellung:

```
<element key="Title"
         class="de.monist.datamanager.MultiLanguageString"
         style="text">
   <operators>
      <operator name="and"/>
      <operator name="or"/>
   </operators>
</element>
```

6.) `style="date"`: Diese Angabe erzeugt ein Eingabeelement, mit dem ein oder zwei Daten angegeben werden können, um entweder einen bestimmten Zeitpunkt oder eine Zeitspanne zu spezifizieren. Dieser Fall wird üblicherweise für zeitliche Metadatenfelder wie das Datum der Erstellung oder der letzten Modifikation von Lernobjekten angewendet. Jedes der beiden Daten wird im Suchformular durch separate Eingabeelemente für Jahr, Monat und Tag dargestellt. Außerdem kann der Benutzer zwischen den folgenden drei Komparatoren wählen: `lt` meint in diesem Fall "bevor". Der Komparator `like` steht hier für das exakte Datum, wird aber anstelle von `eq` verwendet, da nur der Tag und nicht die exakte Uhrzeit verglichen wird. Als dritte Möglichkeit dient `gt`, das in diesem Fall "nach" bedeutet.

Außerdem kann der Benutzer angeben, ob die beiden Datumsangaben miteinander über einen der beiden logischen Operatoren AND und OR verknüpft werden sollen. Voreinstellung ist dabei AND. Durch die Kombination der beiden Datumsangaben kann ein beliebiger Zeitraum für die zeitliche Suche angegeben werden. Folgender XML-Code aus dem Monist-System erzeugt ein Eingabeelement für eine temporale Suche im Metadatenfeld *Date*:

```
<element key="Date" class="java.lang.String" style="date"/>
```

8.4 Integration in die Benutzeroberfläche

Applikationen wie z. B. neue Programme zur Ausführung oder Gestaltung von Simulationen können grundsätzlich eine komplette eigene Benutzeroberfläche mit sich bringen. Für die Anbindung an die von der E-Learning-Plattform zur Verfügung gestellten Dienste wie z. B. die Verwaltung von Daten und Metadaten, aber auch zur Herstellung der Aufrufbarkeit der Anwendungen aus dem Navigator heraus, ist lediglich die bereits in Kapitel 8.1 dargelegte Implementation der XML-RPC-Kommunikationsschnittstelle sowie der Schnittstelle *Application* für die Anmeldung beim Transmitter erforderlich. Durch die in der letzteren definierten Methoden *init* und start, die vom Transmitter automatisch aufgerufen werden, können die Anwendungen in geeigneter Weise zum Laufen gebracht werden und dabei die von ihnen verwendete Oberfläche erzeugen.

Darüber hinaus können Anwendungen auch in die für das Monist-System geschaffene Benutzeroberfläche integriert werden. Hierzu ist die Implementation der zusätzlichen, ebenso wie *RPCApplication* vom übergeordneten Interface *Application* abgeleiteten Schnittstelle *GUIApplication* erforderlich. Diese definiert zusätzlich die beiden Methoden *setWindowManager* und *getWindowManager* zum Setzen und Ausgeben eines Objektes der Klasse *WindowManager*[309], das im Monist-System zur Verwaltung der in der Benutzeroberfläche enthaltenen Fenster dient. Während der Anmeldung von Applikationen beim Systemstart wird die Methode *setWindowManager* noch vor der Methode *init* aufgerufen, damit die Initialisierung unter Verwendung der vom *WindowManager* zur Verfügung gestellten Funktionalitäten erfolgen kann und sich die Anwendungen so von Anfang an in die Benutzeroberfläche integrieren können.[310]

Außerdem definiert die Schnittstelle *GUIApplication* die Methode *getApplicationPanel*, mittels derer jede implementierende Klasse ein Panel[311] zurückliefern muß, das für ihre Anzeige in dem vom *WindowManager* verwalteten Fenster verwendet wird, sowie die Methode *getMenuBar*, über welche die Klasse ein Menü[312] zur Verfügung stellen kann, das automatisch in das Menü des Hauptfensters integriert wird. Zur Gestaltung der Menüstruktur dient außerdem die Konfigurationsdatei *menu.xml*. Für die Integration in die im Monist-System verwendete Benutzeroberfläche stehen im Paket *de.monist.gui* diverse Schnittstellen zur Verfügung, deren Implementierung es Anwendungen erlaubt, in Bezug auf die Fenstertechnik in gleicher Weise gehandhabt zu werden wie die bereits in Monist integrierten Applikationen.

Die Implementation der Benutzeroberfläche des Monist-Systems ist allerdings kein Teil der vorliegenden Arbeit.[313] Daher sei an dieser Stelle für weitere Informationen über den Aufbau und die Erweiterung der Oberfläche auf die entsprechenden Kapitel im Monist-Handbuch verwiesen, das voraussichtlich im April 2004 erscheinen wird.

---

[309] Vollständiger Klassenname: *de.monist.gui.WindowManager*.

[310] Um Komplikationen zwischen den Applikationen von vornherein zu vermeiden, empfiehlt es sich, den Navigator, der ebenfalls die Schnittstelle *GUIApplication* implementiert, vor allen anderen Klassen, die die gemeinsame Benutzeroberfläche verwenden, anzumelden, was sich aus der Reihenfolge der Einträge in der Konfigurationsdatei applications.ini ergibt. Vor dem Navigator muß allerdings die Applikation für den Datenbank-Adapter angemeldet werden, da er den Zugriff auf diese bereits beim Starten benötigt.

[311] Dieses Panel muß eine Instanz der Klasse *de.monist.gui.ApplicationPanel* sein. Zur Integration in die Monist-Oberfläche müssen die speziell für diese geschaffenen GUI-Klassen verwendet werden.

[312] Dabei muß es sich um eine Instanz der Klasse *de.monist.gui.menus.MergeableJMenuBar* handeln.

[313] An der Entwicklung der Benutzeroberfläche des Monist-Systems waren als wissenschaftliche Mitarbeiter Herr Dipl.-Inform. Markus Oesker und Herr Dipl.-Inform. Bernd Küppers sowie als studentische Hilfskräfte Martin Lümkemann, Elmar Bomberg, Timo Krause und Jan Sanders beteiligt, letzterer später ebenfalls als wissenschaftlicher Mitarbeiter.

8.5 Nutzung der bestehenden Konzepte zur Mehrsprachigkeit

Die in der E-Learning-Plattform verwendeten Sprachen sind in der Konfigurationsdatei *languages.ini* festgelegt. Diese Liste kann bei Bedarf um beliebige weitere Sprachen ergänzt werden. Dabei ist jeweils in einer eigenen Zeile zunächst eine Abkürzung für die Sprache gemäß dem in RFC 3066 definierten Internet-Standard zur Identifikation von Sprachen[314] anzugeben, gefolgt von dem Namen der betreffenden Sprache in der Form, wie er im System angezeigt werden soll. Sämtliche Zeilen, die mehr als diese beiden Einträge (Tokens) enthalten, werden ignoriert, so daß zum Abschalten bestimmter Sprachen diese nicht notwendigerweise aus der Konfigurationsdatei entfernt werden müssen, sondern beispielsweise durch Hinzufügen von `[unused]` am Ende der Zeile deaktiviert werden können. Das folgende Beispiel zeigt den Inhalt der Konfigurationsdatei *languages.ini* des Monist-Systems:

```
en English
de Deutsch
fr Français
es Español
it Italiano [unused]
```

Diese Angaben werden auch vom Navigator verwendet, um die Formulare zum Upload von Objekten und zum Editieren ihrer Metadaten aufzubauen. Alle Sprachen, die in der Konfigurationsdatei languages.ini angegeben werden, erscheinen in diesen Formularen zur Auswahl in allen Eingabeelementen, durch die mehrsprachige Metadaten eingegeben werden können.[315] In Systemen mit durchgängig mehrsprachigen Inhalten und Metadaten empfiehlt es sich, beim Erzeugen von neuen *MultiLanguageStrings* stets eine hinreichende Größe zur Aufnahme der textuellen Einträge für sämtliche verwendete Sprachen zu wählen.[316]

Wie bereits in Kapitel 7.5 dargestellt wurde, können nicht nur die in der E-Learning-Plattform eingestellten Daten und Metadaten mehrsprachig gespeichert werden, sondern auch die Benutzeroberfläche kann für verschiedene Sprachen angepaßt werden. Hierzu wird der in der Programmiersprache Java integrierte Mechanismus zur Herstellung der Mehrsprachigkeit in Anwendungen eingesetzt. Dieser basiert auf der Benutzung von für jede Sprache separat angelegten Property-Dateien (*.properties*). Diese enthalten für alle Texte, die in der Benutzeroberfläche angezeigt werden, jeweils eine Übersetzung in die entsprechende Sprache. In Klassen, die Texte in der Oberfläche anzeigen, werden anstelle der Strings Platzhalter verwendet, die Verweise auf Einträge in diesen Dateien darstellen. In Abhängigkeit von der vom aktuell eingeloggten Benutzer eingestellten Sprache werden diese Platzhalter durch Texte in dieser Sprache ersetzt. Anwendungen, die neu in die E-Learning-Plattform eingefügt werden, können diesen Dateien die für ihre Benutzeroberfläche benötigten Platzhalter und Strings hinzufügen. Ebenso ist es jederzeit möglich, komplette neue Property-Dateien für weitere Sprachen anzulegen.

---

[314] Dieser "Internet Best Current Practices" Quasi-Standard kann eingesehen werden unter der Adresse: http://www.ietf.org/rfc/rfc3066.txt.

[315] Im Monist-System gilt an dieser Stelle allerdings die Einschränkung, daß nur diejenigen Sprachen zur Laufzeit berücksichtigt werden, für die eine eigene Version der Benutzeroberfläche durch die Definition entsprechender Property-Dateien zur Verfügung steht. Andere, zukünftig noch zu entwickelnde Systeme auf Basis der E-Learning-Plattform müssen diese Einschränkung jedoch keineswegs beibehalten.

[316] Siehe hierzu die Beschreibung der Klasse *MultiLanguageString* und insbesondere ihrer Konstruktoren in Kapitel 5.3.3

8.6 Hinzufügen von benötigten System-Parametern

Zur Verwaltung von Parametern, die von der E-Learning-Plattform insgesamt oder von einer der integrierten Applikationen benötigt werden, dient die Konfigurationsdatei *parameters.xml*, die beim Systemstart von der Klasse *ParameterManager*[317] eingelesen wird. Von dieser Klasse wird dabei genau ein Objekt erzeugt, während alle weiteren Versuche, einen *ParameterManager* zu erzeugen, das selbe Objekt zurückliefern. Es handelt sich also um ein Singleton-Objekt. Von dem Objekt, das die eingelesenen Parameter verwaltet, gibt es mithin stets nur eine einzige Instanz, um Mehrdeutigkeiten von vornherein auszuschließen.

Die Datei *parameters.xml* ist nach einem speziellen XML-Format aufgebaut. Dieses definiert zunächst ein äußerstes Element `<parameters>`, das beliebig viele Elemente namens `<parameter>` enthalten kann. Diese Elemente sind stets leer, müssen jedoch die folgenden drei Attribute enthalten: `name` spezifiziert den Namen des Parameters, `type` seinen Datentyp (in Form eines vollständigen Java-Klassennamens) und `value` den Wert, der für diesen Parameter gesetzt werden soll. Der Inhalt der Konfigurationsdatei *parameters.xml* muß also stets der folgenden Document Type Definition (DTD) entsprechen:

```
<?xml version="1.0"?>

<!ELEMENT parameters (parameter)*>
<!ELEMENT parameter EMPTY>
<!ATTLIST parameter name CDATA #REQUIRED
                    type CDATA #REQUIRED
                    value CDATA #REQUIRED>
```

Eine wichtige Einschränkung ist dabei, daß die Datei nur Parameter enthalten darf, die Klassen angehören, die über einen Konstruktor verfügen, der einen einzelnen String als Parameter akzeptiert. Auf diese Weise werden beim Systemstart entsprechende Objekte erzeugt, die vom *ParameterManager* verwaltet werden. Ein Aufruf der Methode *getParameter* dieser Klasse liefert zu einem angegebenen String den Parameter zurück, der den in diesem String enthaltenen Namen trägt. Das folgende Beispiel zeigt einige Zeilen aus der Konfigurationsdatei *parameters.xml* des Monist-Systems:

```
<parameters>
<parameter name="serverName"
           type="java.lang.String"
           value="natrix.biologie.uni-bielefeld.de"/>
<parameter name="startObject"
           type="de.monist.util.UniversalObjectIdentifier"
           value="0:0"/>
<parameter name="rpcPort"
           type="java.lang.Integer"
           value="5757"/>
<parameter name="httpPort"
           type="java.lang.Integer"
           value="5858"/>
</parameters>
```

[317] Vollständiger Klassenname: *de.monist.util.ParameterManager*.

Auch die in die E-Learning-Plattform integrierten Applikationen können auf diesem Wege eigene Parameter definieren, die beim Systemstart eingelesen werden müssen. Der Zweck dieser Maßnahme ist, Parameter, die einfache Änderungen des Verhaltens des Systems bewirken, auszulagern und durch unterschiedliche Konfigurationen auszudrücken, um die Notwendigkeit einer Neukompilation des gesamten Systems oder der betreffenden Anwendungen im Fall von Änderungen dieser Werte zu vermeiden. Die Konfiguration des Navigators enthält beispielsweise folgende Elemente, die sich auf die oberhalb des Baumes dargestellte Anzeige der zuletzt aufgerufenen Objekte beziehen:

```
<parameter name="navigator.history.stacksize"
           type="java.lang.Integer"
           value="100"/>
<parameter name="navigator.history.showsize"
           type="java.lang.Integer"
           value="5"/>
<parameter name="navigator.history.namelength"
           type="java.lang.Integer"
           value="12"/>
```

Der erste Parameter bezieht sich dabei auf die maximale Zahl der zu speichernden Schritte, der zweite auf die Anzahl der im Navigator anzuzeigenden, vom Benutzer betrachteten Objekte, während der dritte zur Begrenzung der Länge des im Fenster angezeigten Namens der Objekte dient, damit diese nicht den Rahmen der möglichen Anzeige sprengen. Andere Anwendungen können ebenfalls je nach Bedarf beliebige eigene Parameter definieren.[318]

Daneben ist der *ParameterManager* auch für das Einlesen der anderen Konfigurationsdateien zuständig. Dies betrifft die Liste der vom System akzeptierten MIME Types (*mimetypes.ini*, siehe Kapitel 8.2) ebenso wie die in den Formularen zur Eingabe von Metadaten verwendeten Sprachen (*languages.ini*, siehe Kapitel 8.5), die im Navigator auswählbaren Server (*servers.ini*, siehe Kapitel 7.4) und sogar die beim Systemstart anzumeldenden Applikationen (*applications.ini*, siehe Kapitel 8.1).

Vielleicht noch wichtiger ist die Tatsache, daß der *ParameterManager* ebenfalls den Zähler für die *objectID* verwaltet, die den innerhalb einer Installation der E-Learning-Plattform eindeutigen Objektschlüssel bezeichnet, welcher die eine Hälfte des *UniversalObjectIdentifiers* für jedes Objekt bildet. Die Methode *getNextObjectID* gibt stets die nächste, noch nicht verwendete *objectID* zurück und erhöht dabei den Zähler um 1, um mehrdeutige *UniversalObjectIdentifier* auszuschließen. Diese Funktionalität macht den *ParameterManager* zu einer der wichtigsten Klassen des Systems, und wenn er (beispielsweise aufgrund des Fehlens einer der benötigten Konfigurationsdateien) nicht korrekt initialisiert werden kann, kann die E-Learning-Plattform nicht gestartet werden. Es ist daher stets erforderlich, beim Starten des Systems als erstes ein Objekt der Klasse *ParameterManager* zu erzeugen, das von allen Applikationen benötigt wird.

Damit ist die Darstellung der erforderlichen Schritte zur Integration von Anwendungen in die E-Learning-Plattform und auch die Beschreibung des im Rahmen dieser Arbeit entworfenen Systems abgeschlossen. Im folgenden Kapitel werden darüber hinaus noch einige mögliche Erweiterungen des Systems beschrieben, die auf der Basis der bisher beschriebenen Systemeigenschaften zusätzliche nützliche Funktionalitäten für den Einsatz der E-Learning-Plattform in der universitären Lehre bieten können.

[318] Für die Datenbank werden im Monist-System beispielsweise 14 verschiedene Parameter verwendet.

9 Erweiterungen des Systems

Entsprechend der in Kapitel 2.9 aufgestellten Anforderung nach einer leichten Erweiterbarkeit wurde die E-Learning-Plattform als offenes System gestaltet, in das zusätzliche Anwendungen auf einfache Weise integriert werden können. Die dabei erforderlichen Schritte wurden in Kapitel 8 dargestellt. Darüber hinaus bietet die E-Learning-Plattform allerdings auch die Möglichkeit, bei Bedarf Anpassungen und Weiterentwicklungen vorzunehmen, zumal die Software im Rahmen einer Open-Source-Strategie zur Verfügung gestellt wird. Zwei mögliche Weiterentwicklungen, die im Rahmen des Projektes Monist zwar geplant, aber aus unterschiedlichen Gründen nicht bzw. nur teilweise umgesetzt wurden, werden in diesem Kapitel vorgestellt. Dabei handelt es sich in beiden Fällen um verschiedene Ausprägungen der in Kapitel 2.8 geforderten Funktionalitäten zur Kooperation und Kommunikation zwischen Benutzern der E-Learning-Plattform im Online-Betrieb.

9.1 Kommunikationssystem

In Kapitel 2.8 wurde der Anspruch an die E-Learning-Plattform formuliert, daß eine technische Möglichkeit für die Integration eines Kommunikationssystems gegeben sein muß. Die Software sollte gemäß dieser Anforderung in der Lage sein, die für einen Chat oder darüber hinausgehende synchrone Kommunikationsmöglichkeiten erforderlichen Nachrichten auszutauschen. Die dafür benötigten Funktionalitäten werden durch den XML-RPC-Kommunikationsmechanismus bereitgestellt. Folgender XML-Code könnte beispielsweise einen RPC-Aufruf darstellen, mit dem ein Benutzer in einem Chatraum eine für alle anderen darin befindlichen Benutzer sichtbare Nachricht schreibt:

```
<?xml version="1.0"?>
<methodCall>
   <methodName>chat.say</methodName>
   <params>
      <param>
         <value><string>Hello World!</string></value>
      </param>
   </params>
</methodCall>
```

Diese Nachricht kann bei Bedarf natürlich durch weitere Parameter ergänzt werden, beispielsweise um die Angabe eines von mehreren möglichen Chat-Räumen oder um die Namen oder *UniversalObjectIdentifier* von einzelnen Benutzern, die die Nachricht erhalten sollen. Außerdem ist die Definition beliebiger, komplexerer Funktionalitäten möglich, um ein Kommunikationssystem zu gestalten, das über einen einfachen Chat hinausgeht, oder um ein bereits vorhandenes derartiges System über eine entsprechende *RPCApplication*, die als Fassade für die tatsächlichen Anwendungsklassen verwendet wird, in die E-Learning-Plattform zu integrieren. Dabei kann es sich beispielsweise um einen sogenannten MUD handeln, in dem die Benutzer neben der reinen Kommunikation auch die Möglichkeit zur Generierung und Manipulation von virtuellen Objekten haben und sich in virtuellen Welten bewegen können. Durch die Verwendung der vom Datenbank-Adapter via XML-RPC zur Verfügung gestellten Methoden können auch die in der E-Learning-Plattform abgelegten Lernobjekte in einer solchen virtuellen Welt zur Verfügung gestellt und durch Benutzer verwendet oder manipuliert werden.

Im Monist-System wurde tatsächlich ein derartiger MUD namens *Key*[319] integriert. Dieser nutzt für die Client-Server-Kommunikation allerdings nicht den XML-RPC-Mechanismus. Dies liegt darin begründet, daß die Entscheidung über die Verwendung von *Key* bereits in einer sehr frühen Projektphase getroffen wurde, zu einem Zeitpunkt, als die Kommunikationsschnittstelle der E-Learning-Plattform noch nicht spezifiziert war. Der MUD wurde parallel zur übrigen Entwicklung durch umfangreiche Änderungen für die Zwecke des Monist-Systems angepaßt.[320]

Im späteren Projektverlauf stellte sich jedoch heraus, daß eine Umstellung des in *Key* vorhandenen Kommunikationsmechanismus auf XML-RPC nur mit einem unverhältnismäßig hohen Aufwand möglich gewesen wäre, weshalb die Entscheidung getroffen wurde, das in *Key* benutzte Verfahren zur Client-Server-Kommunikation beizubehalten. Nichtsdestotrotz fügt sich der MUD nahtlos in die übrigen Teilbereiche der E-Learning-Plattform ein und implementiert beispielsweise auch das Interface *Application*, er wird somit als Applikation beim Systemstart angemeldet und gestartet und ist vollständig in die Monist-Benutzeroberfläche integriert.

Bei *Key* handelt es sich allerdings um ein klassisches Client-Server-System, was dazu führt, daß sich in diesem Punkt im Monist-System Client und Server voneinander unterscheiden. Zwar wird dieser Unterschied allein dadurch realisiert, daß die E-Learning-Plattform mit geringfügig verschiedenen Konfigurationsdateien aufgerufen wird. Dies stellt jedoch eine Ausnahme von dem ansonsten verfolgten Prinzip dar, daß Client und Server einheitlich gestaltet sein sollen und gleichberechtigte Elemente eines Netzwerkes bilden, das auf diese Weise auch als Peer-to-Peer-System betrachtet werden kann.

Eine mögliche Erweiterung der E-Learning-Plattform besteht mithin darin, einen Chat oder MUD zu schaffen, der dezentral in jede Installation des Systems integriert ist. Auf diese Weise wäre es auch möglich, jederzeit ohne Beteiligung des Servers auf einem beliebigen Computer, auf dem die Software installiert ist, einen Chatraum zu eröffnen oder sogar eine eigene virtuelle Welt zu generieren. Ein so gestaltetes System bedeutet größere Freiheiten für die Benutzer bei einer erheblich verminderten Abhängigkeit von einem zentralen Server und bringt gleichzeitig eine deutlich größere Ausfallsicherheit mit sich. Außerdem wird der Server durch eine solche dezentrale Organisation von der Bearbeitung zahlreicher Aufrufe über das Netzwerk entlastet.

Auf diese Weise können im Rahmen der E-Learning-Plattform sogar für bestimmte Kurse vollkommen unabhängige Teilbereiche geschaffen werden. Wenn die Teilnehmer einer Veranstaltung sich gegenseitig die IP-Adressen ihrer Computer mitteilen[321], kann sowohl die Verteilung der Lehrmaterialien innerhalb dieser Gruppe arrangiert werden als auch die Organisation veranstaltungsspezifischer Chat-Sitzungen. Dies ermöglicht eine verstärkte Personalisierung und unterstützt zudem die Bildung einer Gemeinschaft (community), durch die kollaboratives Lernen gefördert wird.

---

[319] Es handelt sich dabei um einen in der Programmiersprache Java geschriebenen MUD, der im Rahmen einer Open-Source-Politik über die nach eigenen Angaben größte Internetseite für Open-Source-Software, SourceForge (http://sourceforge.net), erhältlich ist. Die Projekt-Homepage von *Key* findet man dort unter der Adresse http://key.sourceforge.net, dort steht *Key* auch kostenlos zum Download zur Verfügung.

[320] Die dafür erforderliche Programmierung wurde von Herrn Hubert Gorczytza als wissenschaftlichem Mitarbeiter sowie von den studentischen Hilfskräften Jens Springer und Timo Krause durchgeführt. Da der Monist-MUD somit kein Teil der vorliegenden Arbeit ist, sei für eine nähere Beschreibung an dieser Stelle auf das Monist-Handbuch verwiesen, das voraussichtlich im April 2004 erscheinen wird.

[321] Wie bereits in Kapitel 5.2 festgestellt wurde, können die IP-Adressen in DHCP-Netzwerken mit der Zeit auch wechseln. Für diesen Fall könnte ein entsprechender automatischer Update-Mechanismus als zusätzliche Erweiterung in die E-Learning-Plattform integriert werden.

9.2 Ablaufverfolgung und Teleteaching

Eine andere mögliche Erweiterung des Systems beruht ebenfalls auf der Verwendung des XML-RPC-Kommunikationsmechanismus. Bei der Verwendung von Simulationen, deren Aufbau durch unterschiedliche Konfigurationen der in die E-Learning-Plattform integrierten Simulationssoftware bestimmt wird, führt der Benutzer üblicherweise eine Abfolge von Aktivitäten aus, die sich auf den internen Zustand dieser Simulationen auswirken. Wenn man nun ein XML-basiertes Datenformat entwickelt, in dem derartige Aktionen abgebildet werden können, sowie die Möglichkeit schafft, daß Applikationen die vom Benutzer unternommenen Schritte in dieses Format exportieren, per XML-RPC an andere Anwendungen verschicken und ihre eigenen Aktionen selbst wieder einlesen und in Änderungen ihres internen Zustands übersetzen können, können damit mehrere nützliche Zusatzfunktionalitäten für die E-Learning-Plattform geschaffen werden.

Zum einen ist auf diese Weise eine Speicherung von erreichten Ergebnissen möglich, die weit über das bloße Archivieren des aktuellen Systemzustandes hinausgeht, indem auch die einzelnen Schritte, durch deren Ausführung dieser Zustand erreicht wurde, festgehalten werden. Die Aktionen des Benutzers können so während ihrer Ausführung automatisch protokolliert und zu einem späteren Zeitpunkt schrittweise nachverfolgt werden. Dies ermöglicht nicht nur den Lernenden, ihre eigenen Aktionen noch einmal nachzuvollziehen, sondern kann auch von Lehrenden zur Lernwegkontrolle verwendet werden. Außerdem bietet die Speicherung von Abläufen die Möglichkeit, dynamische und interaktive Übungsaufgaben zur Verfügung zu stellen, zu deren Bewertung nicht nur ein erreichter Endzustand einer Simulation herangezogen werden kann, sondern auch die auf dem Weg dorthin unternommenen Schritte. Für solche Aufgaben können auch schrittweise nachvollziehbare Musterlösungen zur Verfügung gestellt werden.

Eine weitere Anwendung für das über den XML-RPC-Kommunikationsmechanismus übertragbare Datenformat für Aktionen des Benutzers liegt in der Schaffung von Tele-Teaching-Funktionalitäten. In einem derartigen Szenario melden sich Simulationsprogramme, die gleichzeitig innerhalb einer oder mehrerer Instanzen der E-Learning-Plattform laufen, bei einer anderen Instanz sozusagen als Zuhörer an. Die Aktionen der Person, die auf diesem Computer eine bestimmte Simulation bedient, können dann via XML-RPC an die anderen Installationen übertragen und dort innerhalb des gleichen Simulationsprogrammes automatisch nachvollzogen werden. Auf diese Weise können Lehrende sowohl im Rahmen von Online-Veranstaltungen als auch in der Präsenzlehre in universitätseigenen Computerräumen von ihrem Computer aus die Benutzung einer Simulation demonstrieren. Umgekehrt können auch die Lernenden die von ihnen durchgeführten Aktionen zum Computer ihres Lehrers und zu ihren Kommilitonen übertragen, um ihre Ergebnisse vorzuführen oder auch Fragen zu stellen, wobei zu diesem Zweck die gleichzeitige Verwendung eines Chat- oder MUD-Systems hilfreich ist.

Der Mehrwert gegenüber klassischen Systemen für derartige Online-Präsentationen, bei denen üblicherweise nur der Bildschirminhalt übertragen wird, besteht darin, daß durch die Integration in die tatsächlich verwendete Simulationssoftware hier die einzelnen vorgeführten Aktionen nachvollziehbar sind, gespeichert werden können und von den Zuschauern zu einem späteren Zeitpunkt Schritt für Schritt wieder abgerufen werden können. Auch die Vorführung gespeicherter Abläufe ist möglich, so daß es nicht erforderlich ist, eine komplizierte Abfolge auswendig zu lernen, um sie "live" vorzuführen.

Im Projekt Monist war die Entwicklung eines Tele-Teaching-Systems zwar geplant, sie konnte jedoch im Rahmen der Projektlaufzeit nicht mehr in Angriff genommen werden.

10 Zusammenfassung

Mit der in der vorliegenden Arbeit entworfenen E-Learning-Plattform und ihrer parallel dazu erfolgten Implementierung in der Programmiersprache Java wurde eine technische Basis für die Verwendung interaktiver Lehrsimulationen geschaffen, die sowohl in der Präsenzlehre als auch im Selbststudium eingesetzt werden können. Die entwickelten Datenstrukturen und Speichermechanismen ermöglichen das Einbringen von beliebigen, multimedialen Inhalten sowie deren komfortable Verwaltung durch den Gebrauch von frei definierbaren Metadaten. Zusammengesetzte Lehreinheiten können in Form von verschiedenen Konfigurationen der benutzten Simulationssoftware, beispielsweise in einem XML-basierten Datenformat, vorliegen und als Inhalt (*content*) von Virtuellen Verzeichnissen gespeichert werden, die außerdem die einzelnen Medien, die für die Zusammensetzung der betreffenden Simulation anhand der Konfigurationen erforderlich sind, in Form von Virtuellen Lernobjekten (VLO) enthalten.

Durch die in jede Installation des Systems integrierte Datenbank und den verwendeten XML-RPC-Kommunikationsmechanismus ist die Benutzung der E-Learning-Plattform sowohl online als auch offline möglich. Im Online-Betrieb können dabei jederzeit neue Inhalte verbreitet bzw. heruntergeladen werden, und die Möglichkeit zur Integration von umfangreichen Kommunikationsfunktionalitäten ist ebenfalls gegeben. Die Verwaltung von Benutzern und Zugriffsrechten ermöglicht sowohl die Personalisierung des Systems als auch die Festlegung von Berechtigungen für das Lesen, Anlegen und Verändern von im System enthaltenen Objekten. Die Software ist plattformunabhängig implementiert und kann bei Bedarf beliebig erweitert werden. Zu diesem Zweck wird sie im Rahmen einer Open-Source-Politik zur Verfügung gestellt.[322] Damit erfüllt das System alle in Kapitel 2 aufgestellten Anforderungen.

Durch die in Kapitel 4 vorgestellte Vier-Schichten-Architektur entstand wie geplant ein sehr flexibles System, das modular aufgebaut ist und dessen einzelne Ebenen bei Bedarf auch leicht gegen andere Komponenten ausgetauscht werden können. Beispielsweise wurde die Referenzimplementation des Datenbank-Adapters gemäß der in Kapitel 5.6 entwickelten Schnittstelle von einer externen Firma unabhängig vom parallel dazu stattfindenden Projektablauf an der Universität vorgenommen. Der Datenbank-Adapter für MySQL weist keinerlei Abhängigkeiten zu anderen Teilen des Systems auf als zu der definierten Schnittstelle, die er implementiert.

Somit kann die vorhandene Datenbank leicht durch eine andere ersetzt werden, um beispielsweise die Performanz des Systems zu verbessern oder andere Datenbanken zu verwenden, für die in Projekten, in denen die E-Learning-Plattform künftig eingesetzt wird, ggf. bereits anderweitig eine Lizenz erworben wurde. Zu diesem Zweck müßte für die neue Datenbank lediglich ein eigener Adapter entwickelt werden, der seinerseits die vorgegebene Schnittstelle implementiert. Der Wechsel der Datenbank läßt sich dann durch einige simple Einträge in einer Konfigurationsdatei realisieren.[323] Auf diese Weise ist es sogar möglich, zwischen verschiedenen, auf einem Computer installierten Datenbanken umzuschalten.

---

[322] Siehe hierzu die Homepage des Projektes Monist, http://www.monist.de, auf der ein Download des Monist-Systems und der E-Learning-Plattform möglich ist. Als Kontaktadresse und für Fragen steht die E-Mail-Adresse info@monist.de zur Verfügung.

[323] Es handelt sich dabei um die Datei *parameters.xml*, die auch für andere Systemparameter verwendet wird. Siehe hierzu Kapitel 5.6.1, in dem die Datenbank-Parameter beschrieben werden, sowie Kapitel 8.6 für Informationen über die Verwendung von Parametern im Allgemeinen.

Ebenso kann man allein durch entsprechend angepaßte Konfigurationen auf Client und Server oder auch auf den einzelnen Clients unterschiedliche Datenbanken verwenden. Beispielsweise kann auf den Computern der einzelnen Benutzer ein frei erhältliches System wie MySQL benutzt werden, während auf dem Server (oder bei Bedarf auch auf mehreren, voneinander unabhängigen Servern) eine leistungsfähigere, eventuell auch lizenzpflichtige Datenbank wie z. B. Oracle zum Einsatz kommt. Daher könnte die Entwicklung verschiedener spezieller Adapter für marktübliche Datenbanken eine sinnvolle Ergänzung des Systems darstellen.[324]

Während auf diese Weise die unterste Schicht der Vier-Schichten-Architektur, die Datenbank, ausgetauscht werden kann, ohne daß dafür Veränderungen in den anderen Schichten vorgenommen werden müssen, gilt dies auch für die Benutzeroberfläche. Ein Beispiel dafür ist der Navigator, dessen Oberfläche im Laufe des Projektes Monist zahlreichen Veränderungen unterworfen war. Diese betrafen jedoch stets ausschließlich die Anwendungsschicht, während die darunterliegende Fassadenschicht, in welcher unter anderem die Virtuellen Lernobjekte und Virtuellen Verzeichnisse definiert wurden, ebensowenig verändert werden mußte wie die Datenmodellschicht, der Datenbank-Adapter oder der XML-RPC-Kommunikationsmechanismus.

Durch die modulare Struktur des Systems, in der das Zusammenspiel der Komponenten auf der Verwendung klar definierter Schnittstellen basiert, ist die Integration von neuen Anwendungen problemlos möglich, ohne dafür die bereits bestehenden Teile verändern zu müssen. So wurde beispielsweise der Navigator, der als erste für den Benutzer sichtbare Teilapplikation betrachtet werden kann, erst längere Zeit nach Fertigstellung der Fassadenschicht begonnen und in die bestehende Architektur eingepaßt. Die im Projekt Monist verwendeten Simulationsprogramme, die zum Teil bei Projektbeginn schon existierten, konnten ebenfalls problemlos integriert werden. Dies beweist, daß nicht nur maßgeschneiderte Komponenten als Teilapplikationen in der E-Learning-Plattform verwendet werden können, sondern auch bereits bestehende Programme.

Als scheinbares Gegenbeispiel soll hier der in Kapitel 9.1 beschriebene MUD erwähnt werden, der nicht vollständig integriert werden konnte. Der Grund liegt in diesem Fall jedoch darin, daß das verwendete System auf eigenen Mechanismen zur Client-Server-Kommunikation und Datenspeicherung basiert und dafür keine offenen Schnittstellen bietet. Der MUD stellt insofern einen Spezialfall dar, als daß es sich dabei eigentlich um eine für einen vollkommen selbständigen Einsatz geschaffene Plattform zur internetbasierten Kommunikation handelt und nicht um eine einzelne Anwendung, die wie die Simulationssoftware separat auf einem einzelnen Computer ausgeführt wird. Um den gemeinsamen XML-RPC-Kommunikationsmechanismus der E-Learning-Plattform zu benutzen und damit einen höheren Integrationsgrad der verschiedenen Anwendungen zu ermöglichen, empfiehlt es sich, das Kommunikationssystem nachträglich an diesen Mechanismus anzupassen. Dies ist jedoch kein Widerspruch zur anfangs aufgestellten Forderung nach der Integrierbarkeit von Applikationen, da die Kommunikation im Sinne der E-Learning-Plattform einen Basisdienst darstellt, an dessen Schnittstelle sich die zu integrierenden Anwendungen orientieren müssen.

[324] Insbesondere für große Datenmengen könnte dies ggf. auch aus Performanzgründen erforderlich sein. Im Projekt Monist erwiesen sich die verwendete Datenbank und ihr Adapter als bestimmender Faktor für die Performanzprobleme, die leider beim Auslesen und Anzeigen von Verzeichnissen, die viele Objekte enthalten, auftreten. Zur Beschleunigung derartiger Vorgänge wurde der vorhandene Adapter zwar bereits beispielsweise durch die nachträgliche Einführung eines Cache optimiert. Die Datenbankoperationen sind jedoch im Vergleich mit der Datenübermittlung über den XML-RPC-Kommunikationsmechanismus und den damit verbundenen Codierungs- und Decodierungsoperationen immer noch zeitlich aufwendiger und stellen den Hauptgrund für die im Navigator bemerkbare Verzögerung dar.

Der verwendete XML-RPC-Kommunikationsmechanismus stellt eine Besonderheit der E-Learning-Plattform sowohl gegenüber anderen E-Learning-Systemen (wie den in Kapitel 3 beschriebenen Programmen) als auch gegenüber den meisten vergleichbaren Client-Server-Systemen oder Peer-to-Peer-Systemen dar. Im Gegensatz zu anderen für die Kommunikation zwischen Objekten auf verschiedenen Computern entwickelten Technologien wie z. B. CORBA oder RMI, die heutzutage in vielen internetbasierten Systemen eingesetzt werden, ist die XML-RPC-Schnittstelle sowohl in Bezug auf das für die Übertragung verwendete Datenformat als auch in der Komplexität ihrer Implementierung eine sehr einfache Alternative und bietet dennoch die volle Funktionalität, um jedes beliebige Objekt für Aufrufe seiner Methoden von anderen Computern aus über ein Netzwerk zugänglich zu machen.

Durch die Verwendung von XML als Datenformat für die Kommunikation profitiert die E-Learning-Plattform davon, daß XML inzwischen als weltweiter Standard anerkannt ist und zur Verarbeitung dieses Formates diverse Werkzeuge zur Verfügung stehen. Die Kommunikationsschnittstelle benutzt beispielsweise einen SAX-Parser, und auch die verwendete XML-RPC-Softwarebibliothek steht im Internet bei der Apache Software Foundation[325] als Open-Source-Software zur Verfügung. Applikationen können bei der Erstellung syntaktisch korrekter XML-RPC-Requests oder bei der Beantwortung von auf diesem Wege eingehenden Anfragen ebenfalls auf vorhandene Tools zurückgreifen, was die Benutzung des Kommunikationsmechanismus erleichtert.

Die einfache Möglichkeit für Anwendungen, Schnittstellen bereitzustellen, die über den XML-RPC-Kommunikationsmechanismus angesprochen werden können, ermöglicht eine beliebig umfassende Kommunikation von Applikationen untereinander sowohl auf dem gleichen Computer als auch zwischen verschiedenen Rechnern über das Internet. Wie weit dies gehen kann, zeigt das Zusammenspiel zwischen dem Datenbank-Adapter und dem Navigator, der bei praktisch allen Operationen auf den in seiner Oberfläche angezeigten Daten via XML-RPC auf die von der Datenbankschnittstelle zur Verfügung gestellten Methoden zurückgreift. Auch der MUD verwendet XML-RPC, um mit den anderen Teilanwendungen der E-Learning-Plattform zu interagieren. Lediglich für die Kommunikation verschiedener Simulationsanwendungen untereinander gibt es bisher noch kein praktisches Beispiel. Derartige Fälle sind aber durchaus denkbar, etwa, wenn Simulationsprogramme aus dem gleichen Fachgebiet untereinander Daten austauschen müßten, was ebenfalls über XML-RPC ermöglicht werden könnte.

Das zweite Charakteristikum der E-Learning-Plattform liegt in den speziell für sie entwickelten Datentypen zur Speicherung von Metadaten in Virtuellen Lernobjekten sowie von flexiblen Eltern-Kind-Beziehungen zwischen diesen Objekten in der virtuellen Verzeichnisstruktur. Lernobjekte können mit beliebigen Metadaten versehen werden, deren Inhalte aus Texten und Zahlen, aber auch aus den in dieser Arbeit entworfenen, mehrsprachigen (*MultiLanguageString*) oder mehrwertigen Strings (*MultiValueString*) bestehen können. Auf diese Weise kann zur Klassifikation und Wiederauffindung von Objekten in der E-Learning-Plattform ein umfassender, selbstdefinierter Metadatensatz verwendet werden, der in seiner Zusammensetzung sowohl einem der unterschiedlichen Standards für Metadaten im E-Learning-Bereich (DC, LOM, AICC, SCORM etc.) folgen kann als auch eigene Metadaten enthalten kann. Im Projekt Monist wurde beispielsweise eine Kombination aus diesen beiden Möglichkeiten realisiert, indem die Metadaten des Dublin Core um eine Reihe von Elementen des LOM-Standards sowie einige für dieses Projekt spezifische Elemente ergänzt wurden.

[325] Homepage der Apache Software Foundation: http://www.apache.org. Das XML-RPC-Projekt ist ein Teil des *Apache Web Services Project* mit einer eigenen Homepage unter: http://ws.apache.org/xmlrpc.

Dabei können für unterschiedliche Datentypen verschiedene Metadatensätze angegeben werden.[326] Von dieser Möglichkeit wird im Monist-System in erheblichem Umfang Gebrauch gemacht. Beispielsweise sind Lehreinheiten (*unit*), Kurse (*course*) oder auch einzelne Simulationen (*simulation*) eigene Objekttypen, die mit unterschiedlichen Metadatensätzen versehen sind. Insgesamt gibt es in diesem System fünfzehn verschiedene Typen. Auch Links sind eigene Objekte, die über spezielle Metadaten zur Angabe von Startpunkt und Ziel des Links verfügen. Auf diese Weise können Links einem Autor zugeordnet und mit eigenen Zugriffsrechten versehen werden, woraufhin sie Benutzern, die nicht über die erforderlichen Leserechte verfügen, erst gar nicht angezeigt werden. Außerdem können sie angelegt und gelöscht werden, ohne daß dabei die Objekte selbst verändert werden müssen.

Literaturhinweise sowie Einträge im Glossar sind gleichermaßen Objekte von speziell für diese Zwecke spezifizierten Typen, für die ebenfalls eigene Metadatensätze definiert sind. Alle diese Anwendungsfälle werden ermöglicht durch die Nutzung der flexiblen Speichermöglichkeit für die Metadaten in den Virtuellen Lernobjekten. Die Metadaten, die für einen bestimmten Typ verwaltet werden und zu diesem Zweck vom Benutzer beim Erzeugen von neuen Objekten dieses Typs angegeben werden müssen, werden in Konfigurationsdateien festgehalten, die in einem eigenen, XML-basierten Datenformat vorliegen. Auch in diesem Fall ermöglicht die Verwendung von XML die Benutzung von Tools, die für die Verarbeitung von diesem Standard entsprechenden Formaten geschaffen wurden.[327] Für die Eingabe der Metadaten beim Erzeugen von neuen Objekten oder beim Ändern von bereits bestehenden Objekten werden gemäß der jeweiligen Konfiguration für jeden Objekttyp unterschiedliche Formulare erzeugt.

Auch Benutzer und Gruppen werden in der E-Learning-Plattform durch Virtuelle Lernobjekte und Virtuelle Verzeichnisse repräsentiert. Benutzer können dabei gleichzeitig auch in mehrere Gruppen eingeordnet werden, ebenso wie Lernobjekte in mehreren Verzeichnissen enthalten sein können. Für einzelne Medien und Simulationen, aber auch für ganze Lehreinheiten und Kurse ermöglicht diese besondere Eigenschaft der virtuellen Verzeichnisstruktur eine einfache Wiederverwendbarkeit in einem anderen Kontext ohne redundante Speicherung. Außerdem können auf diese Weise die gleichen Inhalte im System auf unterschiedlichen Wegen zugänglich gemacht werden, wodurch unterschiedliche Herangehensweisen an den enthaltenen Lernstoff ermöglicht werden.

Objekte können gleichzeitig an unterschiedlichen Stellen in eine nach einem beliebigen Muster aufgebaute, hierarchische Struktur eingeordnet werden, die durch Virtuelle Verzeichnisse abgebildet wird. Sie können auch parallel nach mehreren derartigen Ordnungsschemata klassifiziert werden, beispielsweise nach ihrer Zugehörigkeit zu bestimmten Fachgebieten und deren Untergliederungen, aber ebenso auch nach den damit durchgeführten Lehrveranstaltungen, den an ihrer Entwicklung beteiligten Instituten oder Autoren, nach Ort oder Zeitpunkt der Erstellung oder jedem beliebigen anderen benutzerdefinierten Schema, das sich in Form einer Verzeichnisstruktur darstellen läßt. Ein Objekt erscheint für den Benutzer an jeder Stelle in einer solchen Hierarchie, an der es darin eingeordnet wurde, es handelt sich dabei jedoch stets um das selbe Objekt.

Ein Beispiel aus dem Projekt Monist soll dazu dienen, diesen Sachverhalt zu verdeutlichen: Im Monist-System gibt es auf oberster Ebene unterschiedliche Verzeichnisse, die zur Aufnahme von einzelnen Lehreinheiten (*units*) einerseits und ganzen Kursen

[326] Siehe Kapitel 8.2.
[327] Dies gilt in gleicher Weise auch für die Konfigurationsdatei *parameters.xml*, die zur Verwaltung von allgemeinen Systemparametern dient. Siehe hierzu Kapitel 8.6.

(*courses*) andererseits dienen. Die in das System eingestellten Lehreinheiten erscheinen sowohl in Unterverzeichnissen des Verzeichnisses für die *units* als auch innerhalb von Kursen, die im Verzeichnis *courses* liegen, sind aber jeweils nur ein einziges Virtuelles Lernobjekt (bzw. ein Virtuelles Verzeichnis mit den darin enthaltenen Teilobjekten der betreffenden Lehreinheit).

Die globale Eindeutigkeit von Lernobjekten wird durch die ebenfalls im Rahmen der vorliegenden Arbeit spezifizierten *UniversalObjectIdentifier* sichergestellt. Auch nach der Übertragung auf einen anderen Computer behält ein Lernobjekt stets die selbe ID und ist über diese eindeutig identifizierbar und referenzierbar. Damit ist sichergestellt, daß Verweise sich immer auf das selbe Objekt beziehen. Für einen dauerhaften und verläßlichen Einsatz in der Lehre ist die auf diese Weise sichergestellte Eindeutigkeit von Lernobjekten unerläßlich, insbesondere in einem verteilten System, in dem die für die Benutzer zugänglichen Inhalte auf unterschiedlichen Wegen und ohne eine zentrale Kontrolle verbreitet werden können.

Die E-Learning-Plattform bietet mit den in dieser Arbeit beschriebenen Funktionalitäten die erforderlichen Basistechnologien für ein verteiltes E-Learning-System, ist jedoch für sich allein genommen noch kein vollständiges derartiges System. Ohne die Integration der Lernsoftware sowie der fachlichen Inhalte würde sie nur eine leere Hülle darstellen, deren Funktionalität über die reine Datenverwaltung nicht hinaus ginge. Im Projekt Monist, das als Referenzprojekt für die E-Learning-Plattform angesehen werden kann, wurde eine konfigurierbare Simulationssoftware geschaffen, ein spezieller Editor für derartige Konfigurationen entwickelt und unter Verwendung dieses Werkzeuges ein umfangreicher Grundstock an fachlichen Lerninhalten für das Gebiet der Neuro- und Kognitionswissenschaften angelegt.[328] Durch das Zusammenwirken mit diesen drei Komponenten entsteht erst ein System, das den vollen Umfang der Funktionalitäten der E-Learning-Plattform nutzen kann und den beabsichtigten Einsatz in der universitären Lehre ermöglicht.

Für eine weitergehende Nutzung der E-Learning-Plattform ist es daher erforderlich, für andere Fachbereiche die Entwicklung entsprechender Simulationsprogramme vorzunehmen, diese Anwendungen und eventuell bereits vorhandene Software in das System zu integrieren sowie eine kritische Masse an fachspezifischen Inhalten bereitzustellen. Damit ein auf diese Art und Weise entstehendes E-Learning-System von den Lernenden dauerhaft angenommen wird, empfiehlt sich darüber hinaus das regelmäßige Abhalten von Lehrveranstaltungen unter Verwendung der elektronischen Lehre sowie die fortgesetzte Erstellung neuer Inhalte, um das vermittelte Wissen stets auf dem aktuellen Stand der wissenschaftlichen Forschung zu halten. Dies erfordert zwar beides einen gewissen Aufwand an Zeit und Personal, wird aber auch beides durch die für genau diesen Zweck geschaffene E-Learning-Plattform unterstützt.

[328] Diese drei Aufgabenbereiche und ihre Bearbeitung im Rahmen des Projektes Monist werden in den zum Zeitpunkt der Abgabe der vorliegenden Arbeit noch unveröffentlichten Dissertationen von Sören Lorenz, Markus Oesker und Wolfram Horstmann beschrieben.

Anhang A: Verwendete Abkürzungen

a. a. O.	am angegebenen Ort
ADL	Advanced Distributed Learning
AG	Aktiengesellschaft
AG	Arbeitsgruppe
AICC	Aviation Industry CBT Committee
AIX	Advanced Interactive Executive
API	Application Programming Interface
BMBF	Bundesministerium für Bildung und Forschung
bzw.	beziehungsweise
CA	California, Kalifornien
CBT	Computer Based Training
CD	Compact Disk
CFW	CourseFactoryWeb
CL-HTTP	Common Lisp Hypermedia Server
CM	Content Management
CMG	Computer Management Group
CMS	Content Management System
CORBA	Common Object Request Broker Architecture
d. h.	das heißt
DB	Database, Datenbank
DC	Dublin Core
DCMI	Dublin Core Metadata Initiative
de	deutsch
DHCP	Dynamic Host Configuration Protocol
Dipl.-Biol.	Diplom-Biologe
Dipl.-Inform.	Diplom-Informatiker
DLS	DistanceLearningSystem
DMS	Document Management System, Dokumentenmanagementsystem
DOE	Department of Education
Dr.	Doktor
DTD	Document Type Definition
E-Learning	Electronic Learning, elektronisches Lernen
E-Mail	Electronic Mail, elektronische Post
ECMS	Enterprise Content Management System
Ed.	Editor
EDV	Elektronische Datenverarbeitung
EIMS	Enterprise Information Management Systemen
en	english, englisch
eq	equal
es	español, spanisch
etc.	et cetera
Fak.	Fakultät
FAQ	Frequently Asked Questions
FB	Fachbereich
FHG	Fraunhofer Gesellschaft
fr	français, französisch
ge	greater or equal
ggf.	gegebenenfalls
GH	Gesamthochschule

GIF	Graphic Interchange Format
GmbH	Gesellschaft mit beschränkter Haftung
GNU	GNU' s Not Unix
gt	greater than
GUI	Graphical User Interface
GWDG	Gesellschaft für Wissenschaftliche Datenverarbeitung Göttingen
HAUP	Hyperwave Academic User Program
HIS	Hochschulinformationssystem
HIS	Hyperwave Information Server
HP	Hewlett-Packard
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IBM	International Business Machines
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IICM	Institut für Informationsverarbeitung und Computergestützte neue Medien (TU Graz)
ILIAS	Integriertes Lern-, Informations- und Arbeitskooperationssystem
Inc.	Incorporated
int	Integer, Ganzzahl
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
IT	Information Technology, Informationstechnologie
it	italiano, italienisch
JDBC	Java Database Conntectivity
JPG, JPEG	Joint Photographics (Expert) Group
LCC	Library of Congress Classification
LCSH	Library of Congress Subject Headings
le	less or equal
LGPL	GNU Library General Public Licens
LOC	Library of Congress
LOM	Learning Object Metadata
LS	Lehrstuhl
lt	less than
LTSC	Learning Technology Standardization Committee
MESH	Medical Subject Headings
MILESS	Multimedialer Lehr- und Lernserver Essen
MIME	Multipurpose Internet Mail Extensions
MIT	Massachusetts Institute of Technology
MMC	Multimedia Catalogue
MONIST	Modellsimulation neuronaler und kognitiver Informationsverarbeitung -- Schule der Techniken
MS	Microsoft
MUD	Multi User Dungeon *oder* Multi User Dimension
NCSA	National Center for Supercomputing Applications
ne	not equal
NIH	National Institute of Health
NLM	National Library of Medicine
NPS	Network Productivity System

NT	New Technology
OCLC	Online Computer Library Center
OpenUSS	Open University Support System
OS	Operating System, Betriebssystem
PC	Personal Computer
PDF	Portable Document Format
PHP	PHP Hypertext Preprocessor
PIM	Personal Information Manager
PNG	Portable Network Graphics
Prof.	Professor
RAID	Redundant Arrays of Independent Disks
RFC	Request for Comments
RMI	Remote Method Invocation
RPC	Remote Prodedure Call
ru	russisch
SAP	Systeme, Anwendungen, Produkte
SAX	Simple API for XML
SCORM	Sharable Content Object Reference Model
SMS	Short Message Service
SPARC	Scalable Processor Architecture
SQL	Structured Query Languge
SuSE	Software und Systementwicklung
TM	Trademark, Warenzeichen
TU	Technische Universität
UMTS	Universal Mobile Telecommunications System
UNIX	entstanden aus: UNICS, Uniplexed Information and Computing Service
UOI	Universal Object Identifier
URL	Uniform Resource Locator
USA	United States of America
usw.	und so weiter
UTC	Universal Time Coordinates
VIP	Versatile Internet Platform
VirPa	Virtuelles Prüfungsamt
VLO	Virtuelles Lernobjekt
Vol.	Volume
VU	Virtuelle Universität
WAP	Wireless Access Protocol
WBT	Web Based Training
WS	Wintersemester
WWW	World Wide Web
XML	Extensible Markup Language
z. B.	zum Beispiel
€	Euro
©	Copyright
®	Registriertes Warenzeichen

Anhang B: Literaturverzeichnis

B.1 Gedruckte Quellen

B.1.1 Bücher

Katharina Angerhausen, Dietmar Weiß, Holger Mertens, Dokumenten-Management-Systeme im Vergleich, Oxygen Verlag, Feldkirchen, 2002

Helmut Balzert, Lehrbuch der Software-Technik, Band 1, Spektrum Akademischer Verlag, Heidelberg, Berlin, 2. Auflage, 2000

Hans J. Bullinger, Renate Mayer, Marktstudie Dokumentenmanagement, Dr. Th. Gabler Verlag, Wiesbaden, 1994

Andreas Dieckmann, Basisfunktionen für ein generisches virtuelles Labor, Diplomarbeit an der Technischen Fakultät der Universität Bielefeld, 1998

Christian Fuchs, Entwurf einer einheitlichen und formalisierbaren Beschreibung von Dokumenten-Management-Systemen (DMS) auf der Basis einer vergleichenden Untersuchung bestehender DMS, Diplomarbeit an der Fakultät für Informatik und Automatisierung der Universität Ilmenau, 2001

Erich Gamma, Entwurfsmuster: Elemente wiederverwendbarer objektorientierter Software, Addison-Wesley-Longman, Bonn, 1996

Cay S. Horstmann, Gary Cornell, Core Java 1.1, Volume I - Fundamentals, Sun Microsystems Press, Mountain View, California, 1997

Brett McLaughlin, Java™ and XML, O' Reilly, Sebastopol (CA), 2000

Cornelia Versteegen, Dokumentenmanagement, Verlag IT Research, Sauerlach, 2000

B.1.2 Artikel in Zeitschriften

Frank Kappe, Hermann Maurer, Nikolai Scherbakov, Hyper-G - a Universal Hypermedia System, in: Journal of Educational Multimedia and Hypermedia, Vol. 2, Nr. 1, 1993, S. 39-66

Keith Andrews, Frank Kappe, Hermann Maurer, Serving Information to the Web with Hyper-G, in: Computer Networks and ISDN Systems, 27 (6), 1995, S. 919-926; auch in: Proceedings of the Third International World Wide Web Conference, Darmstadt, 1995

Keith Andrews, Frank Kappe, Hermann Maurer, The Hyper-G Network Information System, in: Journal of Universal Computer Science, Vol. 1, Nr. 4 (Special issue: Proceedings of the Workshop on Distributed Multimedia Systems, held in Graz, Austria, Nov. 1994), April 1995, S. 206-220

Hermann Maurer, Hyperwave and Hyper-G, in: Encyclopedia of Computers and Computer History, Vol. One (Ed.: R. Rojas), Fitzroy Dearborn Publishers, Chicago, 2001, S. 383-384

B.2 Internet-Quellen

Anmerkung: Da auf den im Folgenden genannten Internet-Seiten nahezu durchgängig keine Informationen über die Autoren enthalten sind, können diese hier leider nicht angegeben werden. Das gleiche gilt für das Erstellungsdatum.

B.2.1 Projekt Monist

Homepage des Projektes Monist, http://www.monist.de

Projektskizze zur Einreichung bei dem Förderprogramm des Bundesministeriums für Bildung und Forschung: "Neue Medien in der Bildung", http://natrix.biologie.uni-bielefeld.de/monist/Archiv/monistskizze_oz.html

B.2.2 Fachmessen

Homepage der Messe CeBIT, http://www.cebit.de

Homepage der Messe Internet World, http://www.internetworld-messe.de

Homepage der Messe LEARNTEC, http://www.learntec.de

Homepage der Messe LEARNTEC 2003 (Archiv), http://feuerflitzer.de/lt2003website

Homepage der Messe Systems, http://www.systems-world.de

B.2.3 Content Management Systeme

Homepages der Firma Coextant, http://www.coextant.de (deutsch) *bzw.* http://www.coextant.com (englisch)

Hyper.Net Produkt Übersicht, Informationsbroschüre der Firma Coextant, http://www.coextant.de/Content/54B6995196A51982C1256D6000276972/71b92be9f54949b800256d6000279364/7c17cee002f949f900256d6000225481/Produkt%20Overview%20-%20deutsch%20final%20web.pdf

Beschreibung der Systemanforderungen von Coextant Hyper.Net, http://www.coextant.de/topic.aspx?DOC_UNID=EBDFAE81615A4f959D5F145CD54D4895

Homepages der Firma Documentum Inc., http://www.documentum.de (deutsch) *bzw.* http://www.documentum.com (englisch)

Produktliste der Firma Documentum Inc., http://www.documentum.de/products/glossary/product_glossary_home.htm

Informationen der Firma Documentum zum Lieferstart des neuen Produktes Documentum 5 Enterprise Content Management Platform, http://www.documentum.de/products/launch/documentum_5.htm

Produktbeschreibung der Documentum Enterprise Content Management Platform, http://www.documentum.de/products/platform/index.htm

Documentum Technical White Paper: Developing Web Services with Documentum, http://www.documentum.de/products/collateral/platform/wp_tech_web_svcs.pdf

Produkt-Datenblatt der Firma Documentum zu Documentum 5, http://www.documentum.de/products/collateral/platform/ds_documentum5.pdf

Kundenliste der Firma Documentum Inc., http://www.documentum.de/customer_success/customer_list.html

Homepage der Gauss Interprise AG, http://www.gauss.de

Produktbeschreibung zum Gauss VIP Content Manager, http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/content-manager_german.pdf

Technical White Paper: Web-Content-Management mit der VIP CM Suite, http://ham-hal.gauss.de/gaussvip_prod/gaussvip/Global/Downloads/Brochures/DE/Whitepaper_VIP_Content_Management_Suite_8_de.pdf

Homepage der Imperia Software Solutions GmbH, http://www.imperia.net

Produktdatenblatt zu Imperia 7, http://www.imperia.net/imperia/md/content/folder/Imperia_7_Datenblatt.pdf

Homepages der Firma Infopark AG, http://www.infopark.de (deutsch) *bzw.* http://www.infopark.com (englisch)

Produktdatenblatt der Infopark AG zum Content Management System NPS 5, http://www.infopark.de/de/products/nps/nps550_datasheet.pdf

NPS 5 - Technische Funktionsübersicht, Broschüre der Infopark AG, http://www.infopark.de/de/products/Funktionsuebersicht_NPS5.pdf

Produktdatenblatt der Infopark AG zum Content Management System NPS 5, http://www.infopark.de/de/products/nps/nps550_datasheet.pdf

Homepage der Pironet NDH AG, http://www.pironet-ndh.com

pirobase CMS 5.0.3 White Paper, http://www.pironet-ndh.com/servlet/PB/show/1001982/whitepaper5-0_v12_de.pdf

Pressemitteilung der Firma Pironet NDH zur Messe Systems 2002, http://www.pironet-ndh.com/servlet/PB/menu/1006180

B.2.4 Dokumentenmanagementsysteme

Homepage der Docuware AG, http://www.docuware.com

Homepage der BETA Systems Software AG, http://www.betasystems.com

Homepage der CEYONIQ Technology GmbH, http://www.ceyoniq.de

Unternehmensportrait der CEYONIQ Technology GmbH,
http://www.ceyoniq.de/pages/presse/files/pdf/Unternehmensportrait.PDF

CEYONIQ Server, Der strategische Baustein für zukunftssichere Archivierung,
Broschüre der CEYONIQ Technology GmbH,
http://www.ceyoniq.de/pages/produkte/files/pdf/CEYONIQ_Server.pdf

Homepages der FileNET Corporation, http://www.filenet.de (deutsch) *bzw.*
http://www.filenet.com (englisch).

FileNET Content Manager, Broschüre der FileNET Corporation,
http://www.filenet.com/deutsch/Produkte/datenblatt/030420005.pdf

Homepages der Firma Hummingbird Ltd., http://www.hummingbird.com (englisch)
bzw. http://www.hummingbird.com/international/germany/index.html (deutsch)

Übersicht über die Umbenennung und Neustrukturierung der Hummingbird-Produkte
(Product Rebranding Glossary),
http://www.hummingbird.com/services/productrebranding.html

Hummingbird DM Datenblatt,
http://mimage.hummingbird.com/alt_content/binary/pdf/collateral/ds/german/dmds-de.pdf

Homepages der Hyperwave AG, http://www.hyperwave.de *bzw.*
http://www.hyperwave.com (beide mehrsprachig)

Homepage des Instituts für Informationsverarbeitung und Computergestützte neue
Medien (IICM) an der Technischen Universität Graz, http://www.iicm.edu

Homepage der Technischen Universität Graz, http://www.tugraz.at

Hermann Maurer, Hyperwave and Hyper-G,
http://www.iicm.edu/iicm_papers/hyperwave.doc

Keith Andrews, Frank Kappe, Hermann Maurer, Serving Information to the Web with
Hyper-G,
http://www.igd.fhg.de/archive/1995_www95/proceedings/papers/105/hgw3.html

Datenblatt zum Hyperwave Information Server 6,
http://www.hyperwave.de/d/downloads/documents/is6_datasheet_de.pdf

Datenblatt zur Hyperwave eLearning Suite,
http://www.hyperwave.de/d/downloads/documents/els_datasheet_de.pdf
Homepage der Netscape Communications Corporation, http://www.netscape.com

Homepage des Hyperwave Academic User Program, http://www.haup.org

Homepages der OpenText Corporation, http://www.opentext.de (deutsch), http://www.opentext.com (englisch)

Produktinformationen der OpenText Corporation über Livelink, http://www.opentext.de/de/livelink/index.html

Livelink Produktübersicht, Version 9.0.0.1, http://www.opentext.de/livelink/download/Livelink_9_0_0_1_Produktübersicht.pdf

Basis: Das Wissen im Griff - überall und jederzeit, Produktbeschreibung der OpenText Corporation, http://www.opentext.de/de/livelink/download/BASIS_Flyer_(deutsch).pdf

B.2.5 Knowledge Management Systeme

Christoph Treude, Anforderungen an Wissensmanagement, Seminararbeit zum Proseminar Elemente des E-Learning an der Universität-GH Siegen, 2002, http://www.die.informatik.uni-siegen.de/lehre/ws_2002_03/proseminar_e-learning/Anforderungen_an_Wissensmanagement__Ausarbeitung.pdf

Homepage der USU AG, http://www.usu.de

Homepage der International Organization for Standardization, http://www.iso.org

ISO Standard 13250 "Topic Map", Version 1, 1999, http://www.y12.doe.gov/sgml/sc34/document/0129.pdf

ISO Standard 13250 "Topic Map", Version 1, 2002, http://www.y12.doe.gov/sgml/sc34/document/0322_files/iso13250-2nd-ed-v2.pdf

Grundlagen zum USU KnowledgeMiner 2.0, Informationsbroschüre der USU AG, 2001, http://www.wissensmanagement-competence-center.de/offerer.nsf/F2392111DCF2F8AFC1256BD500546A2C/$File/knowledgeminer_faq.pdf

Homepage der Comma Soft AG, http://www.comma-soft.com

Produkt-Homepage der Comma Soft AG zum Produkt infonea, http://www.infonea.com

FAQ infonea: Antworten auf häufig gestellte Fragen zu infonea, Broschüre der Comma Soft AG, http://www.infonea.com/download/pdf/infonea_3.0_FAQ.zip

Factsheet infonea 3.0, Broschüre der Comma Soft AG, http://www.infonea.com/download/pdf/infonea_3.0_Factsheet.zip

B.2.6 CampusSource

Homepage der Initiative CampusSource, http://www.campussource.de

Projektbeschreibung EdoWorkSpace, http://www.campussource.de/software/ews

Projektbeschreibung ILIAS, http://www.campussource.de/software/ilias

Projektbeschreibung Javanti, http://www.campussource.de/software/javanti

Javanti Benutzerhandbuch,
http://www.campussource.de/dl/pub/javanti/jtapBenutzerhandbuch.pdf

Projektbeschreibung litw³, http://www.campussource.de/software/litw

Projektbeschreibung MILESS, http://www.campussource.de/software/miless

Projektbeschreibung MMC, http://www.campussource.de/software/mmc

Projektbeschreibung OpenUSS, http://www.campussource.de/software/openuss

Projektbeschreibung StudIP, http://www.campussource.de/software/studip

Projektbeschreibung SuperX, http://www.campussource.de/software/superx

Projektbeschreibung Uni Open Platform, http://www.campussource.de/software/uop

Projektbeschreibung VirPa, http://www.campussource.de/software/virpa

Projektbeschreibung VU, http://www.campussource.de/software/vu

Projektbeschreibung WebAssign, http://www.campussource.de/software/webassign

B.2.7 E-Learning-Plattformen

Homepage der bureau42 GmbH, http://www.bureau42.de

Homepage der Fraunhofer Gesellschaft, http://www.fraunhofer.de

broker42 - Personalisierte Information Just-in-time und Just-enough, Produktbroschüre der bureau42 GmbH, http://www.bureau42.de/download/broker42_Produktblatt.pdf

trainer42 - Personalisiertes Learning on demand, Produktbroschüre der bureau42 GmbH, http://www.bureau42.de/download/trainer42_Produktblatt.pdf

author42 - Einfaches Erstellen wieder verwendbarer Lernobjekte, Produktbroschüre der bureau42 GmbH, http://www.bureau42.de/download/author42_Produktblatt.pdf

Homepage der EEDO Knowledgeware Corporation, http://www.eedo.com

FORCE TEN: Die schnelle und mächtige eLearning Plattform, Produktbroschüre der EEDO Knowledgeware Corporation,
http://www.eedo.com/deutsch/produkte/ForceTen_brochure_german.pdf

EEDO Produktübersicht, Broschüre der EEDO Knowledgeware Corporation,
http://www.eedo.com/deutsch/produkte/Overview_german.pdf

EEDO WebClass, Produktbroschüre der EEDO Knowledgeware Corporation,
http://www.eedo.com/deutsch/produkte/webclass_german.pdf

Homepage der ets e-learning corporation, http://www.ets-online.de

Produktbeschreibung CourseFactoryWeb,
http://www.ets-online.de/etso01q02/10000.html

Produktbeschreibung CentraOne, http://www.ets-online.de/etso01q02/11200.html

Homepages der Hyperwave AG, http://www.hyperwave.de *bzw.*
http://www.hyperwave.com (beide mehrsprachig)

Homepage der imc information multimedia communication AG, http://www.im-c.de

Produktbeschreibung CLIX Campus, http://www.im-c.de/homepage/clix_campus.htm

Produktbeschreibung Lecturnity,
http://www.im-c.de/homepage/lecturnity/funktionen_ueberblick.htm

Homepage der ORBIS AG, http://www.orbis.de

Produktbeschreibung NetCoach,
http://www.orbis.de/products/elearn_uebersicht.cfm?StaID=elearn2,
http://www.orbis.de/products/elearn_uebersicht.cfm?StaID=elearn3 und
http://www.orbis.de/products/elearn_technologie.cfm

Homepage der Firma WebCT Inc., http://www.webct.com

Informationsbroschüre WebCT VISTA,
http://www.webct.com/service/ViewContent?contentID=9102515

Informationsbroschüre WebCT Campus Edition,
http://www.webct.com/service/ViewContent?contentID=6205796

Transforming the Educational Experience, Broschüre zur WebCT Campus Edition,
http://www.webct.com/service/ViewContent?contentID=6205875

B.2.8 Metadaten und Standards

Homepage des Online Computer Library Center, http:// www.oclc.org

Homepage des National Center for Supercomputing Applications, http://www.ncsa.uiuc.edu

Homepage der Dublin Core Metadata Initiative, http://www.dublincore.org

Dublin Core Metadata Element Set, Version 1.1: Reference Description, http://www.dublincore.org/documents/dces

Metadata-Tags zur Erschließung von Internetquellen, Metadata-Elemente des Dublin Core, eingedeutscht von Diann Rusch-Feja, http://www.mpib-berlin.mpg.de/DOK/metatagd.htm

Homepage der U. S. Library of Congress, http://www.loc.gov

Homepage der U. S. National Library of Medicine, http://www.nlm.nih.gov

Homepage des U. S. National Institute of Health, http://www.nih.gov

RFC 3066: Tags for the Identification of Languages, http://www.ietf.org/rfc/rfc3066.txt

Homepage der Internet Assigned Numbers Authority, http://www.iana.org

Homepage der LOM-Arbeitsgruppe des IEEE, http://ltsc.ieee.org/wg12

Homepage des Learning Technology Standardization Committee des IEEE, http://ltsc.ieee.org

Homepage des Institute of Electrical and Electronics Engineers, http://www.ieee.org

LOM, Standard for Learning Object Metadata, http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf

Homepage des Aviation Industry CBT Committee, http://www.aicc.org

Homepage der Advanced Distributed Learning Initiative des US Department of Defense, http://www.adlnet.org

B.2.9 Software und Technologien

MySQL Homepage, http://www.mysql.com

Ehemalige Projekt-Homepage zum Datenbanktreiber mm.mysql, http://mmmysql.sourceforge.net

Download-Seite für MySQL Connector/J (ehemals mm.mysql), http://www.mysql.com/downloads/api-jdbc.html

Homepage der Firma Media Engineering, http://www.media-engineering.de

XML-RPC Home Page von UserLand Software, Inc., http://www.xmlrpc.com

Homepage der Apache Software Foundation, http://www.apache.org

Homepage des Apache XML-RPC Projekts, http://ws.apache.org/xmlrpc

Spezifikation der HTTP Basic Authentication, http://www.ietf.org/rfc/rfc2617.txt

SourceForge Homepage, http://sourceforge.net

Key Projekt-Homepage, http://key.sourceforge.net

Anhang C: Abbildungsverzeichnis

Anhang D: Erklärung

Hiermit versichere ich, daß die vorliegende Dissertation von mir selbständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt worden ist. Wörtlich oder sinngemäß übernommene Stellen sind als solche gekennzeichnet.

Diese Arbeit wurde angefertigt unter Verwendung des Textverarbeitungsprogrammes Microsoft® Word® 97 sowie des Omondo® UML Plugins für Eclipse® 2.1, mit dem die Klassendiagramme erstellt wurden. Die Implementierung der E-Learning-Plattform erfolgte in der Programmiersprache Java™ unter Verwendung des Java™ 2 Software Development Kit (J2SDK), Standard Edition, Version 1.4. Als Entwicklungsumgebung diente Eclipse® 2.1.

An der Programmierung der Benutzeroberfläche des Monist-Systems und damit auch an der Entwicklung der Oberfläche des Navigators waren außer mir als wissenschaftliche Mitarbeiter Herr Dipl.-Inform. Markus Oesker und Herr Dipl.-Inform. Bernd Küppers sowie als studentische Hilfskräfte Martin Lümkemann, Elmar Bomberg, Timo Krause und Jan Sanders beteiligt, letzterer später ebenfalls als wissenschaftlicher Mitarbeiter. Das Design des Monist-Systems stammt von Thilo Paul-Stüve.

Die Referenz-Implementation des Datenbank-Adapters für die Datenbank MySQL wurde gemäß der in Kapitel 5.6.1 vorgestellten Spezifikation von Herrn Holger Ebert von der Firma Media Engineering vorgenommen. Die übrigen, in den Kapiteln 4 bis 6 und 8 beschriebenen Basistechnologien der E-Learning-Plattform wurden von mir selbst implementiert. Sofern fremder Programmcode verwendet wurde, der im Rahmen einer Open-Source-Politik frei zur Verfügung steht, wurde dies angegeben.

Mit Abschluß des Projektes Monist werden neben der vorliegenden Arbeit drei weitere Dissertationen vorgelegt, die bisher noch nicht veröffentlicht sind. Die Arbeit von Herrn Dipl.-Biol. Wolfram Horstmann befaßt sich mit inhaltlich-didaktischen Konzepten, die Dissertation von Herrn Dipl.-Inform. Sören Lorenz beschreibt die im Projekt Monist und Vorgängerprojekten geschaffene Simulationssoftware, und die Arbeit von Herrn Dipl.-Inform. Markus Oesker hat die Entwicklung von Editoren für Simulationssoftware zum Inhalt. Die genannten Arbeiten weisen keinerlei inhaltliche Überschneidungen mit der vorliegenden Dissertation auf.

Ausschnitte aus dieser Arbeit werden in englischer Übersetzung im Monist-Handbuch erscheinen, das voraussichtlich im Monat April 2004 veröffentlicht wird. Weitere Informationen sind auf der Homepage des Projektes Monist, http://www.monist.de, erhältlich. Unter dieser Internet-Adresse wird die E-Learning-Plattform ebenso wie das Monist-System zum Download angeboten. Für Kontaktaufnahme und Fragen steht die E-Mail-Adresse info@monist.de zur Verfügung.

Anhang E: Danksagungen

Mein Dank gilt Herrn Professor Dr. Robert Giegerich als Betreuer und Gutachter dieser Arbeit und Herrn Professor Dr. Helge Ritter als zweitem Gutachter sowie für seine Beteiligung und die seiner Arbeitsgruppe am Projekt Monist. Ebenfalls danke ich Herrn Professor Dr. Martin Egelhaaf als federführendem Professor des Projektes Monist, an dessen Lehrstuhl ich während des Projektes als wissenschaftlicher Mitarbeiter tätig war und ohne den diese Arbeit nicht möglich gewesen wäre.

Für die gute und erfolgreiche Zusammenarbeit während der dreijährigen Laufzeit des Projektes Monist danke ich allen meinen Kollegen in diesem Projekt, namentlich den wissenschaftlichen Mitarbeitern Hubert Gorczytza, Wolfram Horstmann, Bernd Küppers, Sören Lorenz, Maik Lutterklaas und Markus Oesker, sowie den studentischen Hilfskräften Elmar Bomberg, Timo Krause, Martin Lümkemann, Thilo Paul-Stüve, Jan Sanders und Jens Springer.