Fakultät Wissenschaftszentrum Weihenstephan

für Ernährung, Landnutzung und Umwelt

Fachgebiet für Biometrie und Angewandte Informatik

Modellierung und clientseitige Verarbeitung von E-Learning-Inhalten im Bereich der Life Sciences mit XML-basierten Auszeichnungssprachen und JavaScript

Sebastian Paar

Vollständiger Abdruck der von der Fakultät Wissenschaftszentrum Weihenstephan für Ernährung, Landnutzung und Umwelt der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Forstwissenschaft

genehmigten Dissertation.

Vorsitzende(r): apl. Prof. Dr. M. Weber

Prüfer der Dissertation:

1. Univ.-Prof. Dr. H.-D. Quednau, i. R.
2. Univ.-Prof. Dr. A. Fischer
3. Univ.-Prof. Dr. Th. F. Knoke

Die Dissertation wurde am 23.02.2015 bei der Technischen Universität München eingereicht und durch die Fakultät Wissenschaftszentrum Weihenstephan für Ernährung, Landnutzung und Umwelt am 20.07.2015 angenommen.

Kurzfassung

Mit der bereits etablierten Extensible Markup Language (XML) steht auch im Bereich E-Learning ein reines Text-Datenformat für die Erfassung, den Austausch und die Speicherung von E-Learning-Inhalten zur Verfügung. Als Metasprache bietet XML die Möglichkeit eigene Sprachen über ein XML-Schema zu definieren (z. B. „Document Type Definition“ (DTD), „XML Schema Definition“ (W3C XSD)). Die an LMML (Learning Material Markup Language) der Universität Passau angelehnte Sprache WLMML (WELPE-LMML, Weihenstephaner E-Learning- ProjektE) stellt ein Beispiel dafür dar. Sie wurde im Rahmen dieser Arbeit als vereinfachte Version von LMML konzipiert und in Form einer einzelnen W3C XSD-Datei entwickelt. WLMML ist nach semantischen Gesichtspunkten gegliedert und ermöglicht eine Unterscheidung z. B. zwischen Beispielen, Definitionen, Vertiefungen oder Übungen. Sie bildet E-Learning-Inhalte in strukturierter Form nach didaktischen Gesichtspunkten ab. Unabhängig von der Darstellung werden die Inhalte in reiner Textform (XML-Format) abgespeichert. Es damit auch möglich, WLMML-Dateien nicht nur mit spezifischen XML-Programmen, sondern auch mit einfachen (reine Textdateien erzeugenden) Editoren zu erstellen.

Die komplette Funktionalität der E-Learning-Inhalte wird vom Client-Rechner sichergestellt. Zur clientseitigen Verarbeitung von WLMML (XML)-Dateien wird das im Rahmen dieser Arbeit entwickelte WLMML-Framework mit seinen XML-, XSLT-, HTML-, CSS- und JavaScript-Dateien herangezogen. Durch den konsequenten Einsatz dieser internationalen Spezifikationen beziehungsweise Standards soll ein möglichst hohes Maß an Plattformunabhängigkeit, Kompatibilität, Zukunftssicherheit, Austauschbarkeit und Nachhaltigkeit der E-Learning-Inhalte erreicht werden. Die im WLMML-Framework abgelegten E-Learning-Inhalte sind in allen Standard-Browsern wie z. B. „Microsoft Internet Explorer“, „Mozilla Firefox“, „Apple Safari“ oder „Google Chrome“ lauffähig. Es ist keine Installation und im Grundsatz auch keine Verbindung zu einem Web-Server (Internet oder Intranet) nötig - bei voller Funktionstauglichkeit z. B. der Mehrsprachigkeit oder der Volltext-Suchmaschine. Der Grund dafür ist, dass nach einem Download der E-Learning-Inhalte bei der Verarbeitung der WLMML-Dateien keine Datenübertragung zwischen einem Server und Client (Browser) stattfinden muss (clientseitige Verarbeitung). Die identischen Lehrinhalte können damit sowohl online über einen Web-Server (z. B. aus einer Lernplattform heraus oder direkt über die Angabe einer URL) oder offline (nach dem Download) im Browser betrachtet werden.

Abstract

XML (Extensible Markup Language), a commonly used markup language that provides a plain-text format for recording, exchanging and storing data, is also applicable for e-learning data. Since XML is a meta language, it offers the possibility to define one's own languages with the help of an XML scheme (e. g. Document Type Definitions (DTDs), W30 XSD (XML Scheme Definitions). The language WLMML (WELPE-LMML, Weihenstephaner E-Learning-ProiektE) is an example. It was the intention of this project to develop WLMML from the Learning Material Markup Language (LMML), used by the University of Passau, in a simplier version with only one W3C XSD file. WLMML is semantically structured and allows to make a distinction between examples, definitions, details or exercises. E-learning contents can thus be pedagogically structured. They are always saved in text format (XML format), irrespective of the way they are displayed. It is therefore also possible to make WLMML-files with the help of simple text editors and not just by using specific XML software.

The client computer is able to supervise the complete functionality of e-learning content. It is the WLMML framework that was developed during this project with its XML, XSLT, HTML, CSS and JavaScript files that allows to process WLMML (XML) files on the client computer. A consistent application of these international specifications (or standards, respectively) should ensure a high degree of platform independence, compatibility, interchangeability, durability and long-lasting future for e-learning content. E-learning data stored in WLMML framework are supported by all common browsers (e. g. Internet Explorer or Mozilla Firefox, Apple Safari, Google Chrome) without installation or a connection to an internet/íntranet-server (including e. g. the service of multi languages and searching of words). A data-transfer between server and client (browser) is thus not necessary when XML (WLMML) files are processed with the aid of XSLT and JavaScript (client-based processing) after loading e-learning content. For this very reason, it is possible to make use of the same contents online with the help of a server (e. g. on a learning management system or directly by typing a URL) or offline (after download) by means of a browser.

Dankesworte

Die vorliegende Arbeit wurde neben meiner beruflichen Tätigkeit erstellt. Viele Urlaubstage und Wochenenden waren aus diesem Grund dem Arbeiten am PC gewidmet. Ohne die Unterstützung und die guten Nerven meiner Frau wäre sie wohl nicht zum Abschluss gekommen.

Besonders herzlich will ich meinem Doktorvater Herrn Professor Dr. Quednau danken, der mir diese Arbeit ermöglichte, mir mit Geduld, Rat und Tat zur Seite stand und mich nie im Regen stehen ließ.

Ein herzlicher Dank gilt auch Herrn Olaf Strehl, der mir nicht nur bei unseren monatlichen Treffen immer wieder wertvolle Gedanken und Anregungen gab.

Zusätzlich möchte ich mich bei den Diplomanden Frau Katrin Geiger und Herrn Vlad A. Radulescu bedanken, die in Ihren Diplomarbeiten eine Vorgängerversion der XML-basierten Auszeichnungssprache WLMML einsetzten.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

Abkürzungsverzeichnis*

ADL	Advanced Distributed Learning
AICC	Aviation Industry Computer based Training Committee
Ajax	Asynchronous JavaScript and XML
ANSI	American National Standards Institute
API	Application Programming Interface
ARIADNE	Alliance of Remote Instructional Authoring and Distribution Networks for Europe
ASCII	American Standard Code for Information Interchange
ASTD	American Society for Training & Development
CBT	Computer Based Training
CEN	Comité Européen de Normalisation, Europäisches Komitee für Normung, European Committee for Standardization
CEN/ISSS WS-LT	Comité Européen de Normalisation/Information Society Standardization System Workshop on Learning Technologies
CMS	Content Management System
CSS	Cascading Style Sheets
DocBook	Semantische Auszeichnungssprache für technische Dokumentationen
DOM	Document Object Model
DTD	Document Type Definition
ECMA	European Computer Manufacturers Association
E-Learning	electronic learning („elektronisch unterstütztes Lernen“)
eLML	eLesson Markup Language
EML	Educational Markup Language
E4X	ECMAScript for XML
HTML	Hypertext Markup Language
http	Hypertext Transfer Protocol
IEC	International Engineering Consortium, Standardisierungskomitee für Elektroniksysteme
IEEE	Institute of Electrical and Electronics Engineers
IEEE LTSC	Institute of Electrical and Electronics Engineers Learning Technology Standards Committee

IMS GLC	Instructional Management System (Project) Global Learning Consortium
IMS LD	IMS Learning Design
IMS CP	IMS Content Packaging
IMS MD	IMS/LOM Meta-Data Specification
IMS QTI	IMS Question and Test Interoperability Specification
ISO/IEC	International Organization for Standardization/International Electrotechnical Commission
ISSS	Information Society Standardization System
LMS	Learning Management System
LCMS	Learning Content Management System
LMML	Learning Material Markup Language
LO	Learning Object (Lernobjekt)
(IEEE) LOM	Learning Object Metadata (IEEE 1484.12.1 - 2002)
MathML	Mathematical Markup Language
<ML>^{3}	Multidimensional Learning Objects and Modular Lectures Markup Language
OUNL-EML	Open University of the Netherlands- Educational Markup Language
PaKMaS	Passauer Knowledge Management System
PHP	Personal Home Page Tools, später: PHP: Hypertext Preprocessor
PLE	Personal Learning Environment
PTM	Passauer Teachware-Modell
Podcast	Kunstwort aus Markenbezeichnung iPod und Broadcasting, Audio- oder Videodatei
PROMETEUS	Promotion of Education & Training in European Society
RELOAD	Reusable eLearning Object Authoring & Delivery
SCORM	Sharable Content Object Reference Model
SGML	Standard Generalized Markup Language
SVG	Scalable Vector Graphics
UBL	Universal Business Language
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

UTF	Unicode Transformation Format
W3C	World Wide Web Consortium
WBT	Web Based Training
Weblog	Kunstwort aus Web (von World Wide Web) und Log (von Logbuch), Online-Tagebuch, auch Blog, Web-Log genannt
WELPE	Weihenstephaner E-Learning-Projekte
Wiki	hawaiianisch für „schnell“ (Webseiten, die von Benutzern online gelesen und bearbeitet werden können)
WLMML	WELPE LMML (Weihenstephaner E-Learning-Projekte Learning Material Markup Language)
XBRL	eXtensible Business Reporting Language
XHTML	Extensible HyperText Markup Language
XML	eXtensible Markup Language
XPath	XML Path Language
XSD	XML Schema Definition
XSL	eXtensible Stylesheet Language
XSLT	eXtensible Stylesheet Language Transformation

*Der Plural wird mit einem angehängten Kleinbuchstaben ausgedrückt (z. B. LMSs, LOs)

1 Einleitung

Das Motto „Lebenslanges Lernen“ spielt in unserer Gesellschaft eine immer größere Rolle. Das Bundesministerium für Bildung und Forschung hält im Jahr 2008 fest (BMBF 2008): „*Lernen hört nach Schule, Ausbildung oder Studium nicht auf, denn Lernen ist das wesentliche Werkzeug zum Erlangen von Bildung und damit für die Gestaltung individueller Lebens- und Arbeitschancen. Lebenslanges Lernen heißt das Schlüsselwort, wenn man auf dem Arbeitsmarkt mithalten, einen Berufs- oder Schulabschluss nachholen oder sich einfach nur weiterbilden will.*“ Die besondere Bedeutung des lebenslangen Lernens unterstreicht die deutsche Bundesregierung 2008 zusätzlich mit der Verabschiedung der Konzeption für das Lernen im Lebenslauf (BMBF 2010 a).

Auch die EU unterstützt das lebenslange Lernen. Sie definiert es wie folgt: “*All learning activity undertaken throughout life, with the aim of improving knowledge, skills and competence, within a personal civic social and/ or employment-related perspective*” (ESAE 2005). In einem eigenen europäischen Bildungsprogramm setzt sich die EU für das lebenslange Lernen ein. Das „*Programm für lebenslanges Lernen*“ (EU 2006) umfasst den lebensbegleitenden Bildungsweg von der Schule über die Hochschule und Berufsbildung bis zur Erwachsenenbildung. Es fasst verschiedene ältere Programme zusammen (LEONARDO DA VINCI, COMENIUS, ERASMUS, GRUNDTVIG) (EU 2011, BMBF 2010 b).

Die UNESCO setzt sich ebenfalls mit ihrem „Institute for Lifelong Learning“ in Hamburg international für ein lebenslanges Lernen ein (UNESCO 2011).

E-Learning spielt eine wichtige Rolle in dem international wie national unterstützten lebenslangen Lernen (LAUR-ERNST 2002, EUROPÄISCHE-KOMMISSION 2009, CEDEFOP 2002). Durch die steigende Verbreitung des Internets und der Personal Computer wird der Zugang zu Lerninhalten immer leichter möglich. Die E-Learning-Inhalte können online und offline zur Verfügung gestellt werden. Online werden sie z. B. über Learning Management Systeme (LMS) dem Lernenden angeboten. Offline müssen lokal installierte Software-Produkte für die Verarbeitung der Inhalte zur Verfügung stehen.

E-Learning wird wohl in Zukunft im Bereich des lebenslangen Lernens nicht mehr wegzudenken sein.

2 Motivation und Ziel

Nach einem Beschluss der europäischen Wissenschaftsminister (BOLOGNA 1999) werden die 8-10 semestrigen Studiengänge durch die zweistufigen Bachelor- und Master-Studiengänge abgelöst. Durch die Umstellung soll es den Studenten unter anderem möglich sein, nach einem Bachelor-Abschluss unter den vielfältigen europaweiten Master-Studiengängen auszuwählen. Studenten aus unterschiedlichen Ländern mit verschiedenen Bachelor-Abschlüssen können sich damit z. B. in denselben Master-Studiengang einschreiben. Dabei war allerdings – besonders bei nicht konsekutiven Studiengängen - zu beobachten, dass die Vorkenntnisse der Studienbewerber in den einzelnen Fächern sehr unterschiedlich waren, sodass die Dozenten nicht von einem einheitlichen Wissensstand ausgehen konnten (GEIGER 2005, S. 9). Dies führte zu der Überlegung, durch E-Learning-Kurse den Kenntnisstand der Studenten auf ein gewünschtes Niveau zu bringen.

Als Anforderungen an die in E-Learning-Kursen eingesetzten Inhalte werden oftmals Plattform-Unabhängigkeit, Kompatibilität, Zukunftssicherheit, Austauschbarkeit, Nachhaltigkeit und Mehrsprachigkeit genannt (siehe auch PONGRATZ 2002, MONTANDON 2004, S. 6 f, ADLNET-SCORM 2014, Synopsis, KOPER et al. 2003 a). Diesen Ansprüchen werden sie aber nicht immer gerecht.

Ziel der Arbeit ist es eine Möglichkeit aufzuzeigen E-Learning-Inhalte abbilden und darstellen zu können, die eben diese genannten Anforderungen erfüllen. Im Detail wird dieses Ziel mit seinen Teilaspekten in den folgenden Unterkapitel näher beleuchtet.

2.1 Standardisierung

Um die benannten Ansprüche erfüllen zu können, sollte bei der Erstellung und Darbietung von E-Learning-Inhalten u. a. besonderer Wert auf den Einsatz möglichst weitverbreiteter (bzw. standardisierter) und frei verfügbarer Auszeichnungs-/Programmiersprachen gelegt werden. Etablierte Sprachen z. B. des World Wide Web Consortiums (W3C 2011) stellen dafür Optionen zur Verfügung.

Als Ziel der Arbeit sollen konsequent auf internationalen Standards beruhende Auszeichnungs-/Programmiersprachen wie z. B. XML, XSLT, HTML, CSS oder JavaScript eingesetzt werden, um die Möglichkeit einer plattformunabhängigen, nachhaltigen Nutzung der E-Learning-Inhalte zu schaffen.

2.2 Software zur Erzeugung und Darstellung von E-Learning-Inhalten

Es erscheint sinnvoll, nicht nur die Erzeugung, sondern auch die Darstellung der E-Learning-Inhalte mit Standard-Software oder freier Software zu ermöglichen.

Ziel der Arbeit ist es mit weltweit zur Verfügung stehender Software diese Aspekte zu erfüllen. Es soll z. B. mit einem einfachen Text-Editor möglich sein E-Learning-Inhalte zu erstellen und sie mit den häufigsten Browsern anzeigen zu können.

2.3 Abbildung von E-Learning-Inhalten

Zur Abbildung der E-Learning-Inhalte sollte ein standardisiertes, frei verfügbares Datenformat eingesetzt werden, das die Beschreibung, den Austausch, die Darstellung und die Manipulation von strukturierten Daten erlaubt. Dabei bietet es sich an, Inhalt, Formatierung und Verarbeitung voneinander zu trennen.

Die E-Learning-Inhalte (insbesondere Inhalte der „Life Sciences“) sollten in einfacher Form modular abgebildet werden können. Ein nach semantischen Gesichtspunkten gegliederter Aufbau erschiene sinnvoll (W3C-SEMANTIC-WEB, siehe auch W3C-OWL).

Viele E-Learning-Inhalte liegen allerdings in proprietären Datenformaten vor.

Als Ziel der Arbeit sollen E-Learning-Inhalte nach semantischen Gesichtspunkten modelliert und über ein standardisiertes, weltweit verbreitetes Datenformat abgebildet werden können. Zusätzlich besteht die Anforderung, dass diese Inhalte in Module zerlegbar und zwischen verschiedenen Systemen (z. B. unterschiedliche Lernplattformen) austauschbar sind. Mit Hilfe dieser Module soll es z. B. möglich sein E-Learning-Inhalte mehrfach zu verwenden und mehrere Module zu größeren Einheiten (z. B. in einem Kurs) zusammenzusetzen.

2.4 Mehrsprachigkeit

Um Bachelor-Absolventen aus unterschiedlichen Ländern anzuwerben, stellen Hochschulen die Inhalte ihrer Masterkurse nicht nur in ihrer Landesprache, sondern häufig mindestens auch in englischer Sprache zur Verfügung (QUEDNAU et al. 2007).

Zusätzlich ließe sich ein Lernthema nicht nur in der Muttersprache der Studenten, sondern auch in einer anderen Wissenschaftssprache anbieten. Außerdem könnten sich einheimische Studenten mit Hilfe des Angebotes mehrsprachiger E-Learning-Inhalte auf einen möglichen späteren Aufenthalt im Ausland vorbereiten (QUEDNAU & PAAR 2007).

Auch der internationale Austausch von Lerneinheiten lässt es sinnvoll erscheinen, E-Learning-Inhalte in unterschiedlichen Sprachen anzubieten (siehe auch HODGINS 2005).

Ziel der Arbeit ist es E-Learning-Inhalte insbesondere bei Studiengängen mit verschiedensprachigen Studenten in einer oder mehreren unterschiedlichen Sprachen anbieten und ohne größeren Aufwand weitere Sprachen nachträglich einfügen zu können. Dabei soll es für den Betrachter möglich sein die Sprache während des Lernprozesses komplett zu wechseln oder sich bestimmte Inhalte in der jeweiligen anderen ausgewählten Sprache anzeigen zu lassen.

2.5 Client-/serverseitige Verarbeitung

In vielen Ländern der Erde steht den Einwohnern noch kein schneller Internetzugang flächendeckend zur Verfügung (siehe z. B. Zahlen zur weltweiten Internetnutzung unter INTERNET-WORLD-STATS 2011). Mitunter ist zwar z. B. eine Universität per Hochgeschwindigkeit an das Internet angebunden, kann aber noch keinen permanenten Internetzugang an allen Rechnern zur Verfügung stellen. In dieser Situation wäre es günstig, wenn sich E-Learning-Inhalte auf allen Rechnern (mit und ohne schnellen Internetzugang) voll funktionstauglich anbieten ließen.

Um die E-Learning-Inhalte allgemein vielseitig nutzen zu können, erscheint es deshalb vorteilhaft die identischen Lehrinhalte sowohl online über einen Web-Server (z. B. aus einer Lernplattform heraus oder direkt über die Angabe einer URL) oder offline (nach dem Download oder von einem Datenträger) im Browser betrachten zu können. Die Interaktivität der E-Learning-Inhalte sollte dabei nicht verloren gehen.

Häufig sind allerdings E-Learning-Inhalte nur im Zusammenspiel mit einem Server bzw. nur lokal auf einem Client-PC lauffähig.

Als Ziel der Arbeit läßt sich daraus ableiten, dass sich die zu erstellenden E-Learning-Inhalte mit gängigen Browsern ohne zusätzliche Software-Installation online und offline betrachten und auch in Lernplattformen wie z. B. MOODLE (2014) einsetzen lassen. Dabei soll es so gut wie keinen Unterschied ausmachen, ob die Inhalte offline über z. B. CD-ROM, USB-Stick, lokale Festplatte oder online über einen Web-Server angeboten werden. Auch für den Test der erstellten E-Learning-Inhalte besteht die Anforderung, dass nach Möglichkeit keine Server-Anbindung notwendig ist, sondern der Einsatz gängiger Browser ausreicht.

3 Ausgangssituation

Im Folgenden soll ein Einblick über verschiedene Möglichkeiten der Modellierung und clientseitigen Verarbeitung von E-Learning-Inhalten mit XML-basierten Auszeichnungssprachen und JavaScript gegeben werden. Das Hauptaugenmerk liegt dabei auf Technologien wie XML (inklusive XSLT), HTML, CSS, JavaScript (inkl. Ajax), IMS CP, IEEE LOM und SCORM. In diesem Zusammenhang werden die Begriffe Datei und Dokument synonym verwendet.

Die Darstellung wird mit einem Überblick über lerntheoretische Grundlagen, E-Learning-Themen, Lernplattformen und die Marktanteile der verschiedenen Browser in Deutschland und weltweit abgerundet.

3.1 Lerntheoretische Grundlagen

Zum Einstieg in lerntheoretische Grundlagen werden im Folgenden die Begriffe Lehren und Lernen beleuchtet.

Den Begriff Lehren definiert SCHRÖDER (2002, S. 59) wie folgt: „*Lehren ist ein Verhalten, das Erfahrung vermittelt mit der Absicht, Lernen zu bewirken.*“ LUKESCH (2007) greift ebenfalls den Begriff Erfahrung auf: „*‚Lehren‘ bedeutet, gezielt Erfahrungen für Lernende herbeizuführen, die wiederum solche Lernprozesse (Veränderungen) in den Lernenden auslösen sollen, deren Ergebnisse intendierten Lehrzielen entsprechen.*“ Lehren an sich dient keinem Selbstzweck, sondern will das Lernen gezielt beeinflussen (COENEN 2002, S. 17). POHL (1999, S. 36) weist darauf hin, dass Lehrtheorien die Frage behandeln „*wie ein gewünschtes Lernergebnis möglichst effizient erreicht werden kann.*“ Damit erhält das Lernziel, das über Lerninhalte und Wissensniveau klassifiziert werden kann (POHL 1999, S. 38), eine zentrale Bedeutung. Er beschreibt, dass im Gegensatz zu Lehrtheorien die Lerntheorien deskriptiv sind und keine Vorgaben machen, wie Wissen vermittelt werden soll.

Der Begriff des Lernens wird von SCHRÖDER (2002, S. 14) wie folgt definiert: „*Lernen bewirkt eine relativ dauerhafte Verhaltensänderung aufgrund von Erfahrung.*“ Der Lernende sammelt damit im Lernprozess neue Erfahrungen bzw. Wissen (siehe auch COENEN 2002, S. 18, POHL 1999, S. 36). Ähnlich definieren PLASSMANN & SCHMITT (2007) das Lernen (siehe auch HOLZINGER 2000, S. 106 f): „*Lernen beschreibt den Prozess zum Erwerb von Erlebens- und Verhaltensweisen, welche durch eine Interaktion mit der Umwelt zustande kommen. … Kurzum: Lernen*

bedeutet eine Veränderung des Erlebens und Verhaltens aufgrund von individuellen Erfahrungen in bzw. mit der Umwelt.“ BAUMGARTNER et al. (2002 a, S. 16) heben dagegen mehr auf das Aneignen definierter Wissenselemente und einen Qualifizierungsaspekt ab: „*Lernen wird oft als ein zielgerichteter Vorgang verstanden, bei dem es um die Aneignung definierter Wissenselemente im Sinne eines Qualifizierungsvorganges geht.*“ KERRES 2012 (S. 20) betont, dass Lernen ein aktiver Vorgang ist und nicht passiv „*erduldet*“ werden kann. Er stellt fest: „*Für Lernen muss ich mich entscheiden; ich muss bereit sein, zu lernen. Ich kann belehrt werden, aber nicht ‚gelernt werden‘.*“

COENEN (2002, S. 17) greift - wie in den meisten vorher aufgeführten Definitionen von Lehren und Lernen - den prozessorientierten Ansatz auf (siehe auch BAUMGARTNER et al. 2002 a, S. 19, MARESCH 2008, S. 17). Er hält fest, dass diese Prozesse nicht standardisiert festgelegt sind, sondern je nach den Vorlieben der Lehrenden bzw. Lernenden variieren. Trotzdem lassen sich nach seinen Aussagen einige (meist gültige) Eigenschaften und Abläufe festhalten. Abbildung 1 zeigt drei Hauptfunktionen des Lehrprozesses (Lehrplanung, Lehrdurchführung, Lehrkontrolle) und führt verschiedene Lernprozesse mit ihren Interdependenzen auf. Dabei weist COENEN (2002, S. 19) darauf hin, dass der Lehrprozess so umzusetzen ist, dass er auf alle Prozessschritte des Lernens einwirkt.

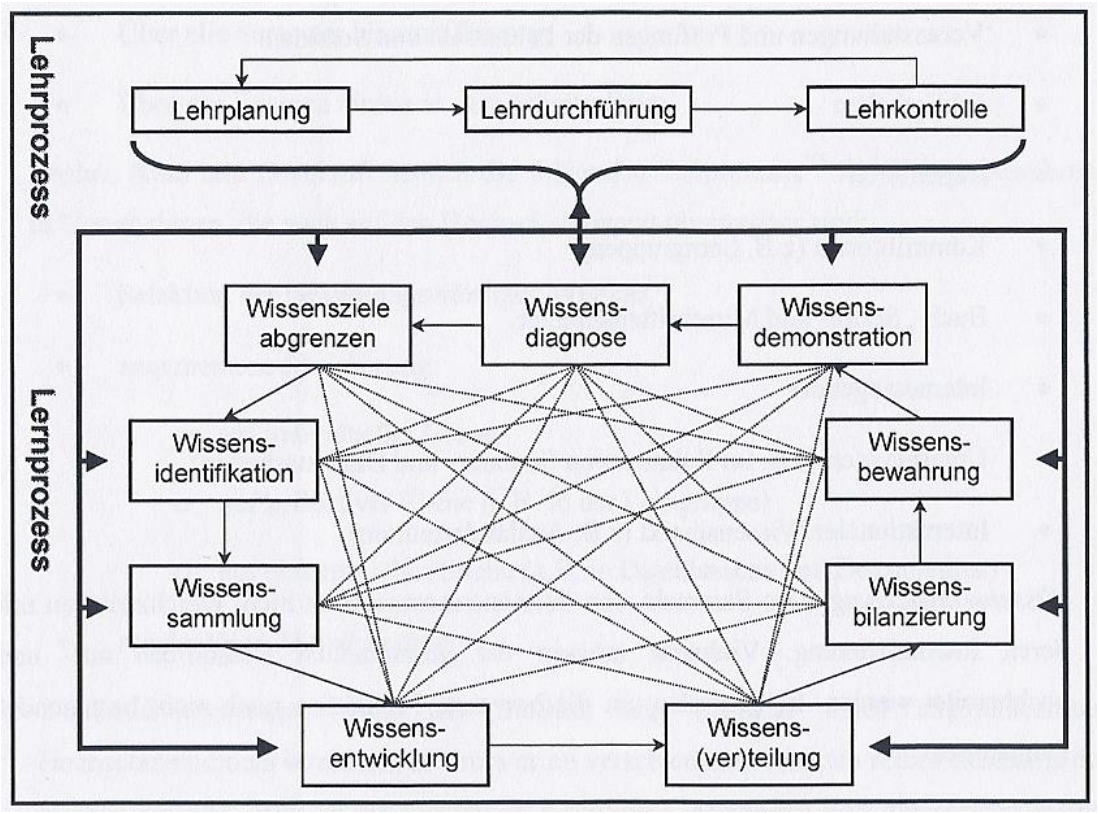


Abbildung 1: Lehr und Lernprozess (COENEN 2002, S. 19)

Wie Lernprozesse vermutlich zusammenhängen, soll im nächsten Kapitel Lerntheorien beleuchtet werden.

3.1.1 Lerntheorien

Eine Lerntheorie beschreibt nach POHL (1999, S. 31, siehe auch S. 36) vermutete Ursache-Wirkungs-Zusammenhänge von Lernprozessen. Auch ARNOLD et al. (2011, S. 101) greifen den Prozessgedanken auf: „*Als Lerntheorien werden hier bestimmte, zu Systemen zusammengefasste Auffassungen darüber verstanden, was Lernen und Wissen ist und wie der Prozess der Aneignung des Wissens verläuft*." Ähnlich beschreibt BLUMSTENGEL (1998, S. 107) Lerntheorien als den Versuch „*Kenntnisse bzw. Auffassungen über das Lernen in einem einheitlichen System zusammenzufassen*".

TÖNNSEN (2007, S. 8) hält zu dem Begriff Lerntheorie fest: „*Eine Lerntheorie stellt eine paradigmatische Auffassung darüber dar, wie der Wissenserwerb erfolgt.*" Er führt weiter aus, dass die Lehr- und Lernforschung im 20. Jahrhundert drei verschiedene Lerntheorien Behaviorismus, Kognitivismus und Konstruktivismus hervorgebracht hat, die weithin anerkannt und etabliert sind (siehe auch BAUMGARTNER & PAYR 1997, REY 2009, S. 32, COENEN 2002, S. 29, HOLZINGER 2000, S. 110 ff, MARESCH 2008, S. 17, ARNOLD et al. 2011, S.101). Diese drei Lerntheorien werden in den folgenden Kapiteln erläutert.

3.1.1.1 Behaviorismus

COENEN (2002, S. 30) beschreibt den Behaviorismus wie folgt: „*Der Behaviorismus (Verhaltenstheorie) versucht Ursache-, und Wirkungszusammenhänge von Lernprozessen ohne Einblicke in die internen Prozesse des Lerners zu erklären. Der Lernende ist eine ‚Black-Box', bei der nur der Input und Output beobachtet werden kann.*" KERRES (2012, S. 112) weist erweiternd darauf hin, dass „*das Verhalten nicht durch Vorgänge im Inneren der Person gesteuert wird, sondern durch die Konsequenzen, die auf das gezeigte Verhalten folgen*". Ein Verhalten wird damit häufiger gezeigt, wenn es eine positive Konsequenz der Umwelt auslöst.

Das zu lehrende Wissen wird nach MARESCH (2008, S. 18) „*in kleinen Lerneinheiten zergliedert und in einer für die Lehrenden optimal erscheinenden Reihenfolge verabreicht*". KERRES (2012, S. 118) hält fest, dass ein derartiges Lernangebot von Lernenden als sehr monoton empfunden und nicht immer akzeptiert wird. Allerdings weist er auf die aktuelle Diskussion über „*Microlearning*" (Lernen mit kleinen Lerneinheiten, siehe WIKIPEDIA-MIKROLERNEN 2011) und das

mobile Lernen (unterwegs lernen) hin, die gerade das Angebot (sehr) kleiner Lerneinheiten favorisieren. Auch GRÖHBIEL & SCHIEFNER (2007, S. 15) erwähnen den Gedanken der kleinen Lerneinheiten und halten fest: „*Das Informationsvermittlungsparadigma (Behaviorismus) geht davon aus, dass Mitarbeiter lernen, indem ihnen Lernstoff entweder vorgetragen oder in kleinen Einheiten vorgelegt wird.*“ Ziel sei es eine korrekte Antwort auf eine Frage oder Aufgabe zu liefern. Diese Reaktion würde dann „*positiv oder negativ verstärkt bzw. an den Mitarbeiter zurückgemeldet*“. Die benannten Vorstellungen lassen sich mit Hilfe der EDV gut umsetzen (z. B. in Multiple-Choice Aufgaben, siehe MARESCH 2008, S. 20). So verwundert es nicht, dass frühe Ansätze des Behaviorismus (in etwa Mitte des 20. Jahrhunderts) mit der Entwicklung der ersten computergesteuerten Lehrmaschinen einhergingen (KERRES 2012, S. 112).

BAUMGARTNER & PAYR (1999, S. 101 f) halten fest, dass der Behaviorismus „*heute stark in Mißkredit geraten*“ ist, er allerdings „*in einem kleinen, begrenzten Bereich große Erfolge erzielt hat*“. MARESCH (2008, S. 21) erwähnt als Beispiele das Trainieren „*von körperlichen Fertigkeiten, von Fingerübungen (z. B. beim Lernen von Maschinenschreiben, Klavierspielen, Jonglieren) und des Memorierens von Fakten (z. B. Vokabeln)*“ (siehe auch BAUMGARTNER & PAYR (1999, S. 102). Allerdings weisen BAUMGARTNER & PAYR (1999, S. 102) darauf hin, dass wohl bei den „*hirnlosen*“ drillartigen Übungsmethoden zur Erreichung spezialisierter Fähigkeiten diese Fähigkeiten „*nicht nur einen hohen Grad an kognitiver Tätigkeit voraussetzen, sondern überhaupt erst auf einer bestimmten, kognitiven Lernstufe sinnvoll geübt werden können*“.

Als Kritikpunkte am Behaviorismus führt MARESCH (2008, S. 21, siehe auch BLUMSTENGEL 1998, S. 110 f, BAUMGARTNER & PAYR 1999, S. 102 f) unter anderem an, dass das Lernen auf die „*Konditionierung des menschlichen Gehirns*“ („*Black Box*“) reduziert wird, die „*innerpsychische Verarbeitung der Informationen und Umweltreize*“ nicht berücksichtigt werden und die „*Möglichkeit der individuellen Schwerpunktsetzungen*“ kaum vorhanden ist.

Häufig genannte Vertreter des Behaviorismus sind Watson, Skinner, Thorndike und Pawlow (MARESCH 2008, S. 18, HOLZINGER 2000, S. 111).

3.1.1.2 Kognitivismus

Die Idee des Behaviorismus wurde durch die „*kognitive Wende*“ in den 60er-Jahren zurückgedrängt (PLASSMANN & SCHMITT 2007). Als führender Vertreter des

Kognitivismus wird Piaget (ARNOLD et al. 2011, S. 102, siehe auch HOLZINGER 2000, S. 111) genannt.

Beim Kognitivismus werden im Gegensatz zum Behaviorismus menschliches Erleben und Verhalten nicht durch Umweltbedingungen (oder einfache äußere Reize), sondern über kognitive Prozesse (z. B. Wahrnehmen, Denken, Urteilen, Verstehen, Erkennen) erklärt (PLASSMANN & SCHMITT 2007, siehe auch BLUMSTENGEL 1998 S. 111, GRÖHBIEL & SCHIEFNER 2007, S. 15). Ein Individuum wird als informationsverarbeitendes Wesen betrachtet. Das Grundprinzip der EDV (Eingabe, Verarbeitung, Ausgabe) wird dabei auch auf den Menschen angewandt (Abbildung 2).

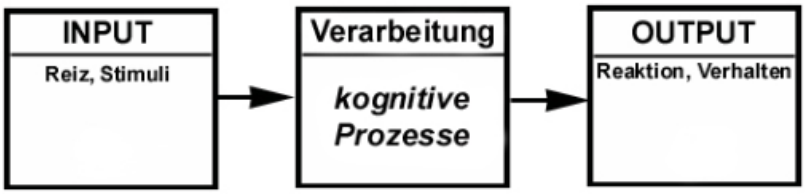


Abbildung 2: Mensch als Informationsverarbeitungseinheit (PLASSMANN, A. & SCHMITT, G. 2007)

Ähnlich beschreibt MARESCH (2008, S. 21) den Kognitivismus: „*Der Kognitivismus stellt im Gegensatz zum Behaviorismus die inneren Prozesse des menschlichen Gehirns in den Mittelpunkt seiner Untersuchungen.*“ Im Folgenden wird festgehalten, dass das Gehirn als „*Informationsverarbeitungsmaschine*“ betrachtet wird. Lernen wird damit als Informationsverarbeitungsprozess gesehen (ARNOLD et al. 2011, S. 102). Auch BAUMGARTNER & PAYR (1999, S. 103) sehen beim Kognitivismus das menschliche Gehirn nicht mehr wie beim Behaviorismus als „*Black Box*“, sondern als informationsverarbeitendes Gerät (siehe auch COENEN 2002, S. 32, PLASSMANN, A. & SCHMITT, G. 2007). Dieser computernahe Ansatz führt dazu, dass eine mitunter enge Verbindung zum Forschungsfeld der künstlichen Intelligenz entsteht (siehe MARESCH 2008, S. 22, ARNOLD et al. 2011, S. 102).

SCHRÖDER (2002, S. 41) betont in seiner Beschreibung kognitiver Lerntheorien, dass das Lernen bei diesem Ansatz in der „*Gewinnung von Einsicht*“ besteht. Er postuliert: „*Bei den kognitiven Lerntheorien besteht Lernen nicht in einer einfachen Reiz-Reaktions-Verbindung, sondern in der Gewinnung von Einsicht.*“ BAUMGARTNER & PAYR (1999, S. 105) gehen nicht explizit auf den Ansatz „*Gewinnung von Einsicht*“ ein, sondern sprechen als typisches Paradigma des Kognitivismus die Problemlösung an: „*Es geht nicht mehr darum, auf gewisse Stimuli die (einzig) richtige Antwort zu produzieren* [Behaviorismus, Anm. d. Verf.]*, sondern*

weit allgemeiner darum, richtige Methoden und Verfahren zur Problemlösung zu lernen, deren Anwendung dann erst die (eine oder mehrere) richtige Antworten ergeben.“

REY (2009, S. 33) stellt in einer abschließenden Beurteilung der kognitiven Lerntheorien fest, dass sie vor allem wegen der Vernachlässigung sozialer, motivationaler und emotionaler Aspekte kritisiert wird (siehe auch KERRES 2012, S. 123, BAUMGARTNER & PAYR 1999, S. 106). Er führt weiter aus, dass jedoch genau diese Aspekte im Lernprozess eine wichtige Rolle spielen.

3.1.1.3 Konstruktivismus

Seit etwa Mitte der 90er-Jahre wird die lerntheoretische Diskussion vor allem durch Aussagen über konstruktivistische Theorien und Konzepte belebt. COENEN (2002, S. 35) führt weiter aus, dass in dieser Theorie gegenüber dem Kognitivismus die Auffassung vertreten wird, dass es keine objektiv richtige Beschreibung der Realität gibt. Im Gegensatz dazu konstruiert der Lernende im Lernprozess seine individuelle subjektive Vorstellungswelt (siehe auch SIEBERT 2003, S.37 ff). Den Gedanken der individuellen subjektiven Vorstellungswelt greift auch BLUMSTENGEL (1998, S. 115) auf: „*Im Gegensatz zum Behaviorismus betont der Konstruktivismus die internen Verstehensprozesse.*“

Ähnlich beschreiben PLASSMANN & SCHMITT (2007) als Kernaussage des Konstruktivismus: „*Individuen reagieren nicht auf die objektive Welt, sondern bilden eine subjektive Realität ab, die auf individuellen Konstruktionen und Interpretationen von der Welt basiert.*“

Auch BAUMGARTNER & PAYR (1999, S. 107) halten fest, dass der Konstruktivismus „*die Gültigkeit einer sogenannten ‚objektiven‘ Beschreibung oder Erklärung der Realität*“ ablehnt. Sie führen weiter aus, dass jedoch nicht die Realität an sich abgelehnt wird, sondern nur die Möglichkeit ihrer objektiven Wahrnehmung. Realität wird in ihren Augen „*als eine interaktive Konzeption verstanden, in der Beobachter und Beobachtetes gegenseitig und strukturell miteinander gekoppelt sind*“. Sie betonen, dass im konstruktivistischen Ansatz Lernen als ein aktiver Prozess betrachtet wird. In diesem Prozess konstruieren Menschen in komplexen Lebenssituationen ihr Wissen in Beziehung zu ihren früheren Erfahrungen. Sie schreiben weiter, dass die praktischen Lebenssituationen allerdings einzigartig, nicht vorhersehbar und deren Probleme nicht offensichtlich sind. Auf diesen Vorgaben aufbauend stellen sie bezüglich des Konstruktivismus fest: „*Im Gegensatz zum*

Kognitivismus steht nicht das Lösen bereits präsentierter Probleme im Vordergrund, sondern das eigenständige Generieren von Problemen“ (siehe auch MARESCH 2008, S. 24).

MARESCH (2008, S. 24) beschreibt als zentrale Feststellung der konstruktivistischen Theorie die Ansicht, dass „*Wissen durch eine interne subjektive Konstruktion von Ideen und Konzepten entsteht*“. Er erläutert bezüglich des konstruktivistischen Ansatzes: „*Im Vordergrund steht also im Gegensatz zum Kognitivismus nicht das Lösen bereits bestehender Probleme, sondern vielmehr das Generieren von Problemstellungen.*“

Wie schon vorher BAUMGARTNER & PAYR (1999, S. 107) und MARESCH (2008, S. 24) heben ARNOLD et al. 2011 (S. 103) bei der Beschreibung des Konstruktivismus das eigenständige Entdecken von Problemen hervor: „*Lernen wird nicht – wie im Kognitivismus – als Informationsverarbeitung, sondern als Konstruktion eines aktiven, lernenden Individuums in einem konkreten sozialen Kontext verstanden. Betont wird deshalb auch das eigenständige Entdecken von Problemen.*“

Auch BLUMSTENGEL (1998, S. 115) geht auf den Aspekt des sozialen Kontextes ein: „*Der Sichtweise von Lernen als einem Informationsverarbeitungsprozeß wird die Vorstellung von Wissen als der individuellen Konstruktion eines aktiven Lerners in einem sozialen Kontext gegenübergestellt. Dabei ist das Vorwissen des Lernenden von entscheidender Bedeutung, da das neue Wissen immer im Bezug darauf konstruiert wird.*“ Sie vertritt die Ansicht, dass beim Lernen „*die Aktivierung von Vorkenntnissen, ihre Ordnung, Korrektur, Erweiterung, Ausdifferenzierung und Integration*“ eine zentrale Rolle spielen.

GRÖHBIEL & SCHIEFNER (2007, S. 15 f) greifen ebenfalls die Vorstellung einer individuellen Wissenskonstruktion sowie eines aktiven Lerners auf und beschreiben den Konstruktivismus unter Hinweis auf SCHÜSSLER (2003, S. 77) wie folgt: „*Der Konstruktivismus geht davon aus, dass Wissen bei jedem Mitarbeiter konstruiert wird. Es gibt kein objektives Wissen. Somit kann Wissen auch nicht vermittelt werden. Lernen kann also nur ermöglicht, nicht erzeugt werden (Arnold/Schüssler 2003)*“ (vgl. in der Didaktik den Begriff der „*Ermöglichungsdidaktik*“ im Gegensatz zur „*Erzeugungsdidaktik*“, SIEBERT 2003, S. 42). Sie weisen darauf hin, dass „*in konstruktivistischen Lernumgebungen vor allem die Selbsttätigkeit und Aktivierung*

des Lernenden“ propagiert werden und „*lebensnahe, inhaltlich komplexe, ganzheitliche Problemstellungen*“ wichtig sind.

Wenn die Rolle des Lernenden in diesem Maße betont wird, stellt sich die Frage nach den Aufgaben des Ausbilders. Die vordringliche Aufgabe des Lehrers im Konstruktivismus sieht BLUMSTENGEL (1998, S. 116) als die eines „*Coaches*“, der „*den individuellen Konstruktionsprozeß anregen und unterstützen, aber nicht wirklich steuern kann (und soll)*“. Er sei vielmehr verantwortlich für die „*Aktivierung der Lernenden, die Anregung des (natürlichen und individuellen) Lernprozesses sowie die Förderung von Metakognition und Toleranz für andere Perspektive*“. Der Lehrer sollte eine Lernumgebung zur Verfügung stellen, die den „*Lernenden dazu anregt, Probleme in Zusammenarbeit mit anderen zu lösen*“. Im Lernprozess steht damit der Lernende stärker im Vordergrund als der Lehrende. Der Vorteil dieses Vorgehens besteht darin, dass das Wissen nicht aufgezwungen wird, sondern vom Lernenden individuell entwickelt, verstanden und damit auch besser behalten wird (BLUMSTENGEL 1998, S. 116).

Als Kritikpunkt gegenüber dem Konstruktivismus führen BAUMGARTNER & PAYR (1999, S. 109 f) unter anderem ein „*Wissenschafts-Argument*“ an. Wenn es nur „*Beobachter-relative Referenzsysteme*“ gibt und kein universelles Referenzsystem existiert, setzt sich die Welt aus einer Vielzahl verschiedener Realitäten zusammen. Dann wäre allerdings eine wissenschaftliche Forschung nicht möglich, da jede Wahrnehmung subjektiv wäre und damit nur relativen Wert hätte. Als weitere Kritik am Konstruktivismus halten sie fest, dass diese Lerntheorie bisher große Probleme hat „*soziale Faktoren und Prozesse wie Sprache oder Gesellschaft befriedigend zu erklären*“. Kritisch äußert sich BLUMSTENGEL (1998, S. 125 f, siehe auch MARESCH 2008, S. 32) bezüglich des meist größeren Zeitaufwandes beim Lernen: „*Schüler und Studenten können nicht in kurzer Zeit ‚entdecken‘, wofür die gesamte Menschheit Jahrtausende gebraucht hat.*“

Als prominente Vertreter des Konstruktivismus werden z. B. Montessori und Comenius bezeichnet (MARESCH 2008, S. 24, BLUMSTENGEL 1998, S. 114).

3.1.1.4 Zusammenfassung

BLUMSTENGEL (1998, S. 108) (siehe auch Baumgartner & Payr 1999, S. 110, S.174) stellt in einer Tabelle Ansatzpunkte der verschiedenen Lerntheorien (inklusive Softwareaspekte) zusammen (Tabelle 1). Zur Erläuterung versteht sie dabei unter dem Kürzel ITS (letzte Zeile der Tabelle) ein intelligentes tutorielles System, das in

der Lage sein soll „*unterschiedliche Anforderungen der Lernenden an den Grad der Schwierigkeit und Unterstützung zu erfüllen*“.

Ähnlich fassen KERRES (2012, S. 128) und BAUMGARTNER et al. (2000, S. 1) in einer Tabelle Aussagen der drei bereits behandelten Lerntheorien zusammen (Tabelle 2, Tabelle 3). Zur Erläuterung versteht KERRES (2012, S. 305, S. 318, siehe auch GRINTSCH 2011) dabei unter „*Exposition*“ die Darbietung von Lernangeboten, bei denen die Präsentation mit ihren Text-, Audio-, Video-Inhalten im Vordergrund steht. Durch die vorgegebene Sachstruktur und Lernwege sei der Lernende in der Regel stärkeren Steuerungsmechanismen unterworfen. Im Gegensatz dazu würden bei „explorativem“ Lernen die Lernaktivitäten wesentlich stärker vom Lernenden gesteuert.

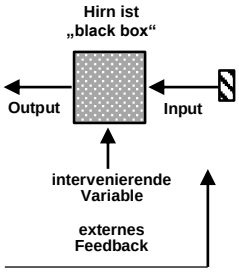
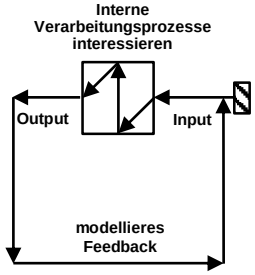
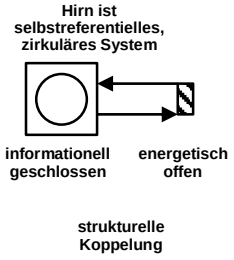
Tabelle 1: Lerntheorien und Softwaretypologie (BLUMSTENGEL 1998, S. 108)

Kategorie	**Behaviorismus**	**Kognitivismus**	**Konstruktivismus**
Hirn ist ein	Passiver Behälter	Informationsverarbeitendes "Gerät"	Informationell geschlossenes System
Wissen wird	Abgelagert	Verarbeitet	Konstruiert
Wissen ist	Eine korrekte Input-Output-Relation	Ein adäquater interner Verarbeitungsprozeß	Mit einer Situation operieren zu können
Lernziele	Richtige Antworten	Richtige Methoden zur Antwortfindung	Komplexe Situationen bewältigen
Paradigma	Stimulus-Response	Problemlösung	Konstruktion
Strategie	Lehren	Beobachten und helfen	Kooperieren
Lehrer ist	Autorität	Tutor	Coach, (Spieler)Trainer
Feedback	Extern vorgegeben	Extern modelliert	Intern modelliert
Interaktion	Starr vorgegeben	Dynamisch in Abhängigkeit des externen Lernmodells	Selbstreferentiell, zirkulär, strukturdeterminiert (autonom)
Programmerk-male	Starrer Ablauf, quantitative Zeit- und Antwortstatistik	Dynamisch gesteuerter Ablauf, vorgegebene Problemstellung, Antwortanalyse	Dynamisch, komplex vernetzte Systeme, keine vorgegebene Problemstellung
Software-Paradigma	Lernmaschine	Künstliche Intelligenz	Sozio-technische Umgebungen
"idealer" Softwaretypus	Tutorielle Systeme, Drill & Practice	Adaptive Systeme, ITS	Simulationen, Mikrowelten, Hypermedia

Tabelle 2: Übersicht über Aussagen der drei Lerntheorien Behaviorismus, Kognitivismus, Konstruktivismus (KERRES 2012, S. 128)

	Behaviorismus	**Kognitivismus**	**Konstruktivismus**
Lernen geschieht durch …	Reaktionen der Umwelt	Aufbau kognitiver Strukturen	(Re-)Konstruktion von Wissen, Partizipation an kultureller Praxis
Resultat des Lernens ist …	Reiz-Reaktions-Verbindung	abstraktes, möglichst generalisierbares Wissen (Schemata, Problemlöse-fertigkeiten etc.)	kontextualisiertes, in Situationen anwendbares („viables“) Wissen
Forderung an didaktisches Design …	Aufteilung der Lehrinhalte in kleinere Lerneinheiten	Anpassung des präsentierten Lernmaterials an Lernvoraussetzungen bzw. –fortschritt	Einbindung in Anwendungskontexte, Authentizität, Lernmaterial, „Situierung“
bevorzugte didaktische Methode …	sequentiell aufbereitete Exposition	Exposition und Exploration	Exploration, Projektmethode, Kooperation
Kontrolle des Lernweges	Fremdsteuerung	Fremd- und Selbststeuerung in Abhängigkeit vom Lernfortschritt	Selbststeuerung
Kontrolle des Lernerfolgs	regelmäßig mit jedem Lernschritt, zwingend für die Anpassung des Lernangebotes	regelmäßig nach einer sinnhaften Lerneinheit, möglichst eingebettet in Lernaufgaben	nur zur Eigendiagnose, anwendungsnahe Übungsaufgaben, vor allem wg. Transfersicherung
Rolle des Mediums	Steuerung und Regelung des Lernprozesses	Präsentation von Wissen, Interaktivität und Adaptivität	Angebote für (kollaborative) Konstruktionsaktivitäten

Tabelle 3: Aspekte von Lehr- und Lernparadigmen (BAUMGARTNER et al. 2000, vgl. auch HOLZINGER 2000, S. 111)

	Behaviorismus	**Kognitivismus**	**Konstruktivismus**
Schematisches Hirnmodell			
implizite Annahmen zur Struktur des Wissens	statisch, objektiv (deklarativ), Faktenwissen („know that“- Wissen, daß etwas der Fall ist)	dynamisch, bedingt objektiv (prozedural), Verfahren- und Bedingungswissen („know how“ - Wissen wie und wann etwas der Fall ist)	praktisch, intersubjektiv, Fertigkeiten, Können, soziale Praktiken
Bevorzugte Lehrhandlung	lehren, erklären; Monolog	beobachten, helfen, vorzeigen, beraten; Dialog	kooperieren, gemeinsam umsetzen; Interaktion
Lehrstrategien	vermitteln, einüben	unterstützen („scaffolding“), ausüben, anwenden, umsetzen	legitimierte periphere Partizipation, schrittweise Verantwortung übergeben, („fading“)
Implizites Lehrziel	erinnern, merken, wiedererkennen	Probleme lösen, Wissen nutzen	Situationen bewältigen, reflektierend Handeln
Erkenntnis-theoretische Position	Positivismus	nicht eindeutig; vom kritischen Rationalismus bis zum Skeptizismus	Skeptizismus, Neo-kantianismus
Symbolische Lehrsituation			

Als Resümee der drei Lerntheorien hält BLUMSTENGEL (1998, S. 128) fest, dass „*insgesamt eine stärkere Betonung kognitionstheoretischer und konstruktivistischer Lernkonzepte vor allem bei der praktischen Realisierung von Lernsystemen wünschenswert ist*“. Allerdings weist sie darauf hin, dass jede Lerntheorie in bestimmten Situationen ihre Berechtigung hat und betont, dass eine einseitige Festlegung auf eine Lerntheorie und vollständige Ablehnung anderer Vorstellungen kaum sinnvoll ist (siehe auch TÖNNSEN 2007, S. 13 f).

KERRES (2012, S. 112) weist darauf hin, dass die verschiedenen Lerntheorien „*keine grundlegend verschiedenen konkurrierenden Paradigmen*“ sind, sondern nur unterschiedliche Sichtweisen auf den Lernvorgang beinhalten. Diese unterschiedlichen Sichtweisen sind nicht falsch oder richtig (bzw. gut oder schlecht), sie schließen sich nicht aus, sondern sie können sich auch ergänzen KERRES (2012, S. 129). KERRES (2012, S. 112, S. 128 ff) verwendet in diesem Zusammenhang den Begriff „*Pragmatismus*“ (siehe auch ARNOLD 2011, S. 111, MARESCH 2008, S. 36 ff).

BLUMSTENGEL (1998, S. 128) hält abschließend fest, dass moderne Lerntheorien die Bedeutung sozialer Komponenten für das Lernen hervorheben. Sie betont: „*Eine stärkere Berücksichtigung des sozialen Kontextes ist in jedem Fall wünschenswert.*“ GRÖHBIEL & SCHIEFNER (2007, S. 16) erwähnen in diesem Zusammenhang den Begriff Konnektivismus. Als geistiger Vater des Begriffes beschreibt ihn SIEMENS (2004) wie folgt: „*Connectivism is the integration of principles explored by chaos, network, and complexity and self-organization theories.*“ GRÖHBIEL & SCHIEFNER (2007, S. 16) bringen diesen lerntheoretischen Ansatz in Zusammenhang mit Schlagwörtern wie Web 2.0, „*Social Software*“ und „*Lernen unterwegs*“ (bzw. an beliebigen Orten). Sie heben hervor, dass soziale Lernformen und Communities zum Wissenserwerb und -aufbau immer mehr an Bedeutung gewinnen. Weiter führen sie aus: „*Netzwerke und soziale Verbindungen werden als die Hauptquellen des Lernens angesehen.*“ EHLERS (2008, S. 11) bringt zum Ausdruck, dass der Konnektivismus in seinen Ideen über die Ansätze des Behaviorismus, Kognitivismus und des Konstruktivismus hinausgehe, indem er „*die zunehmende Tendenz des Lernenden hin zu informellem, vernetztem und elektronisch gestütztem Lernen*“ einbeziehe. Er führt weiter aus: „*Das Lernen wird dabei gesehen als zunehmend kontinuierlicher, lebenslanger Prozess, der in alltägliche Arbeits- und sogar Freizeitaktivitäten eindringt und sowohl den Einzelnen als auch die Organisation und deren Verbindungen untereinander beeinflusst.*“ Auch ARNOLD et al. (2011, S.101) weisen darauf hin, dass im Zusammenhang mit virtuellen Lehr- und Lernarrangements als

zugrunde liegende Lerntheorien (neben Behaviorismus, Kognitivismus, Konstruktivismus) seit 2005 auch der Konnektivismus genannt wird. Allerdings sei umstritten, ob es sich tatsächlich um eine Lerntheorie handle (siehe auch EHLERS 2008, S. 11).

3.1.2 Didaktik

Nach KLIMSA (1993, S. 242) bestimmen Lerntheorien einen "*allgemeinen Rahmen für didaktische Überlegungen*" (siehe auch COENEN 2012, S. 29). Diese Aussage führt zum dem Themenfeld Didaktik. Der Begriff wird in der Fachliteratur unterschiedlich beschrieben (siehe z. B. KLIMSA 1993, S. 13). KLIMSA (1993, S. 13) formuliert, dass sich der Begriff Didaktik „*als eine Theorie der Methoden des Lehrens und Lernens umschreiben läßt*“. Er führt weiter KLAFKI (1976, S. 77 ff, zitiert nach KLIMSA 1993, S. 14) mit einer Erklärung der Didaktik im weiteren Sinne an, wonach der Begriff Entscheidungen, Entscheidungsvoraussetzungen und Entscheidungsbegründungen über Ziele, Inhalte, Methoden und Medien des Unterrichts umfasst (Abbildung 3).

Didaktik im weiteren Sinne

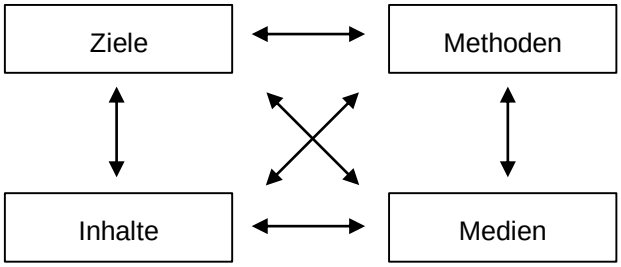


Abbildung 3: Didaktik im weiteren Sinne nach Klafki (1976, zitiert nach KLIMSA 1993, S. 15)

JAHNKE, T. (2006) definiert den Begriff Didaktik wie folgt: „*Die Didaktik umfasst die wissenschaftliche und praktische Beschäftigung mit dem Zusammenhang von Unterrichten und Lernen. Sie ist die wissenschaftliche Reflexion von organisierten Lehr- und Lernprozessen*“ (siehe auch STANGL 2014, WIKIPEDIA-DIDAKTIK 2012, TÖNNSEN 2007, S. 14 ff). STANGL (2014) führt zum Begriff Didaktik weiter aus: „*Didaktik ist also die Lehre des Unterrichts unter Berücksichtigung der pädagogischen Absichten, Mittel und Ziele.*“

SCHRÖDER (2002, S. 217) konstatiert: „*Didaktik ist als Theorie des Unterrichts die kritische Auseinandersetzung mit dem Unterricht, also mit seinen formalen Merkmalen (Organisation, Institutionalisierung und Interaktion) und seinen*

Strukturmerkmalen (Zielen, Inhalten, Methoden).“ Dabei definiert er Unterricht als „*organisierte Interaktion von Lehren und Lernen*“.

Wird die Bedeutung der Medien im Unterricht eingehender betrachtet, taucht der Ausdruck Mediendidaktik auf. Unter dem Ausdruck Medien versteht REY (2009, S. 16) „*verschiedene Objekte oder technische Geräte … mit denen sich Informationen speichern und/oder kommunizieren lassen*“. ISSING & KLIMSA (2002, S. 14) halten fest, dass „*die Mediennutzungsfrage als Aufgabe der Mediendidaktik – also einer speziellen Didaktik- verstanden wird*“. Weiter führen sie aus: „*Die Allgemeine Didaktik überließ die Beschäftigung mit Multimedia einer speziellen Didaktik, der Mediendidaktik, und diese ist wiederum vor allem auf Forschungsdesiderate der Allgemeinen Didaktik angewiesen.*“ KERRES (2012, S. 37) beschreibt die Mediendidaktik als eine Disziplin, die sich „*mit der Funktion und Bedeutung von Medien in Lehr- und Lernprozessen beschäftigt*“ (siehe auch KLIMSA 1993, S. 155 ff). Abbildung 4 zeigt die der Mediendidaktik übergeordnete Medienpädagogik (siehe auch BAUMGARTNER & PAYR 1999, S. 124 f, HOLZINGER 2000, S. 232) und deren Nachbardisziplinen.

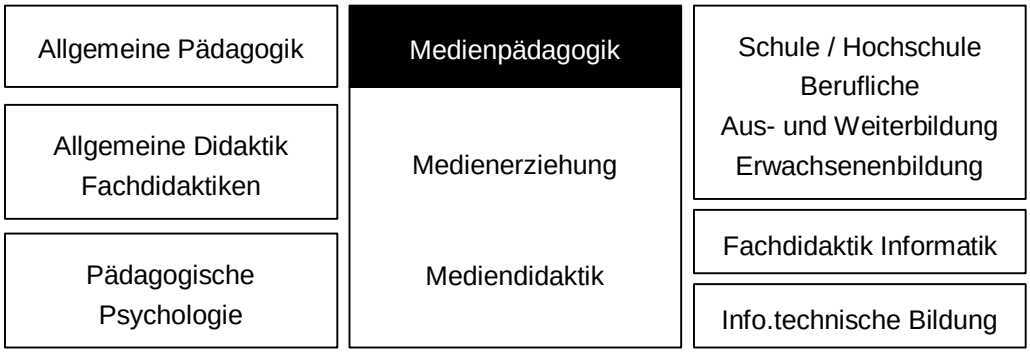


Abbildung 4: Medienpädagogik und Nachbardisziplinen (KERRES 2012, S. 37)

Die Diskussion über Didaktik (insbesondere Mediendidaktik) kann nach KERRES (2012 S. 45 ff) auf vier Ebenen stattfinden kann. Er benennt dabei als Modell-Ebenen unter anderem Planungsmodelle der Didaktik und didaktische Methoden. Unter Planungsmodellen können z. B. das Berliner Modell von Heimann, das Hamburger Modell von Schulz (Weiterentwicklung des Berliner Modells) oder das Modell der Didaktische Analyse von Klafki (siehe WIKIPEDIA BERLINER MODELL 2012) aufgefasst werden. Andere didaktische Modelle wie z. B. das hierarchische Lernmodell von Dreyfus/Dreyfus, das 5-Stufen-Modell von Salmon, die „*Cognitive Load Theorie*“ von Sweller (siehe MARESCH 2008 S. 39 ff) oder das heuristische Lernmodell von Baumgartner (siehe BAUMGARTNER & PAYR 1999 S. 95) versuchen Schlüsse für eine geeignete Didaktik im Lernprozess anzubieten.

Im Folgenden wird auf die Modell-Ebene der didaktischen Methoden (siehe oben) eingegangen. PLASSMANN & SCHMITT (2007) definieren Methode als „*planmäßiges Vorgehen bei der Lösung theoretischer und praktischer Aufgaben*“ unter besonderer Berücksichtigung der Art und Weise des Vorgehens. SCHRÖDER (2002, S. 85) führt aus: „*Methoden sind im erziehungswissenschaftlichen Sprachgebrauch Formen und Verfahrensweisen, mit denen Menschen unter pädagogischen Zielvorstellungen das Lernen anderer Menschen bewusst und planmäßig zu beeinflussen versuchen.*“ Lern-/Lehrmethoden beschäftigen sich damit „wie“ beim Lernen (aus der Sicht des Lernenden) bzw. Lehren (aus der Sicht des Lehrenden) vorgegangen werden kann (siehe auch TERHART 2005, S. 22 ff, SCHULZ 1965, S. 30 f). KERRES (2012, S. 297) hält fest: „*Didaktische Methoden beschreiben, wie aus Lerninhalten Lernangebote werden.*“ Er erläutert (KERRES 2012, S. 47), dass sich zwar didaktische Methoden schwer ordnen lassen, sich allerdings immer wiederkehrende Grundmuster zeigen. Als Grundmuster benennt er Exposition und Exploration sowie Problemorientierung und Kooperation (Tabelle 4).

Tabelle 4: Beschreibung verschiedener didaktischer Methoden (expositorische, explorative, problemorientierte und kooperative) (KERRES 2012, S. 301 f)

Expositorische Methoden
Prinzip: Präsentation im Vordergrund
Chance: Systematische Vermittlung von Fachwissen
Herausforderung: Aktivierung von Lernprozessen durch Beispiele und Übungen
Exploratives Lernen
Prinzip: Selbststeuerung im Vordergrund
Chance: Aktivierung von Interesse und Neugiermotiv
Herausforderung: Aufbereitung des Lernangebotes zur Anregung von Exploration
Problemorientierte Methoden
Prinzip: Lernen mit Fällen, Projekten, Simulationen und Spielwelten
Chance: Entwicklung von Kompetenzen im Umgang mit komplexen Problemen
Herausforderung: Verzahnung mit abstrakter Wissensbasis
Kooperative Methoden
Prinzip: Lernen in Interaktion mit anderen
Chance: Entwicklung von sozialen Kompetenzen
Herausforderung: Sicherung des Lernziels für alle

Grundsätzlich hält KERRES (2012, S. 297) fest, dass sich Lerninhalte nicht alleine durch ihre Präsentation vermitteln lassen. Vielmehr sind sie mit einer didaktischen Methode so vorzubereiten (siehe auch ISO-19796-1 2005), dass Lernprozesse zuverlässig aktiviert werden (Abbildung 5). Diesen Vorgang bezeichnet er als „*didaktische Transformation*“ (KERRES 2012, S. 300 ff). Zusätzlich hebt er die Fülle der didaktischen Ansätze hervor, mit denen Lerninhalte aufbereitet werden können (KERRES 2012, S. 301).

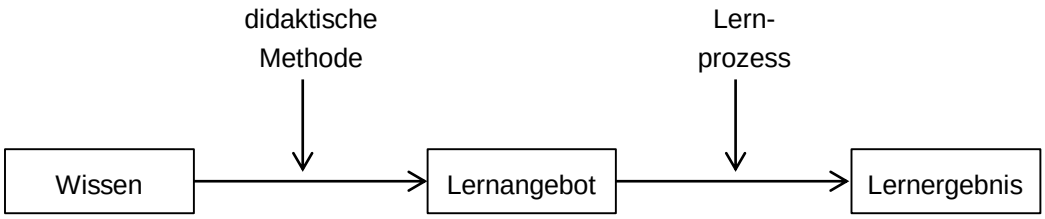


Abbildung 5: Methodische Aufbereitung von Wissen für Lernangebote (KERRES 2012, S. 301)

Außerdem weist KERRES (2012, S. 43) darauf hin, dass außerhalb des deutschsprachigen Raumes der Begriff Didaktik ungebräuchlich ist. Ausgehend von den USA habe sich hier der Begriff „*Instructional Design*“ etabliert. Er definiert den Begriff (S. 197) als „*die systematische Entwicklung von Lernangeboten auf der Grundlage empirischer Forschung zum Lehren und Lernen*“. Weiter führt er aus, dass es sich bei dem Begriff Design nicht um das Design von grafischen Benutzeroberflächen handle, sondern „*vielmehr um die pädagogischen Entscheidungen bei der der Konzeption eines Lernangebots*“.

Bezüglich der Definition des Begriffes (mediengestützter) Lernangebote legt sich KERRES (2012, S. 300) wie folgt fest: „*Mediengestützte Lernangebote verstehen wir als Teil einer gestalteten Umwelt, die immer zu bestimmten Aktivitäten auffordert. Durch die Gestaltung der Umwelt werden Affordanzen* [Angebote, Anm. d. Verf.] *geschaffen, die bestimmte Aktivitäten und Ergebnisse wahrscheinlich machen.*“ Für die didaktische Konzeption eines Lernangebotes hält er fest, dass sie wesentlich von dem angestrebten Lernziel beeinflusst wird. Deswegen fordert er, dass die mit dem Lernangebot verbundenen Ziele aufzuführen sind (KERRES 2012, S. 275, siehe auch S. 47). Allgemein gibt er zu bedenken: „*Ein konkretes Lernangebot, ein Kurs oder Lehrgang, vereint fast immer mehrere didaktische Methoden, auch weil mehrere unterschiedliche Lernziele verfolgt werden*“ (siehe auch HÄFELE & MAIER-HÄFELE 2008).

KERRES (2012, S. 339) hält weiter fest: „*Die meisten mediengestützten Lernangebote folgen, wie konventioneller Unterricht, expositorischen Methoden der Wissensvermittlung, teilweise erweitert um explorative Elemente.*“ Er führt aus, dass sich diese Methoden (expositorische, explorative) besonders sinnvoll bei der Vermittlung von deklarativem Wissen einsetzen lassen. Bei komplexeren Themen sind jedoch andere didaktische Methoden wie Problemorientierung und Kooperation einzusetzen, um die Lernziele zu erreichen (KERRES 2012, S. 339). Er weist darauf hin, dass der Einsatz dieser Methoden allerdings einen größeren Aufwand in der Planung und Umsetzung hervorruft und von den Lernenden und Lehrenden (Betreuenden) mehr Kompetenzen verlangt.

Gerade bei der Strukturierung eines interaktiven multimedialen Lernangebotes (z. B. Hypertext, siehe auch Kapitel 3.2.2 Übersicht Methoden und Medien) mit wahlfreiem Zugriff auf Lerninhalte bieten sich explorative Methoden der Wissensvermittlung an (KERRES 2012, S. 325, siehe auch BLUMSTENGEL 1998, S. 112 f). Er beschreibt, dass im Gegensatz dazu aus technischen Gründen bei linearen Medien (z. B. Büchern) aufgrund des sequentiellen Aufbaus expositorische Methoden eingesetzt werden. Ähnlich weist HOLZINGER (2000, S. 189) darauf hin, dass bei Hypertext „*die Kontrolle über die Sequenzierung von einzelnen Lerneinheiten ausschließlich bei den Lernenden liegt*“ und die „*Kontrolle über den Lernprozess*“ eine aus dem Konstruktivismus stammende Forderung ist, „*die in engem Zusammenhang mit Begriffen wie ‚entdeckendes Lernen‘, ‚exploratives Lernen‘ und ‚selbstbestimmendes Lernen‘ steht*“.

Für die didaktische Strukturierung multimedialer Lernsoftware aus der Sicht des Lehrenden empfiehlt ISSING (2002, S. 160 ff) bewährte Vorgehensweisen aus der allgemeinen Didaktik. Er führt in Abbildung 6 Arbeitsschritte des Unterrichtens als didaktische Strukturierungshilfe für die Entwicklung multimedialer Lernsoftware auf.

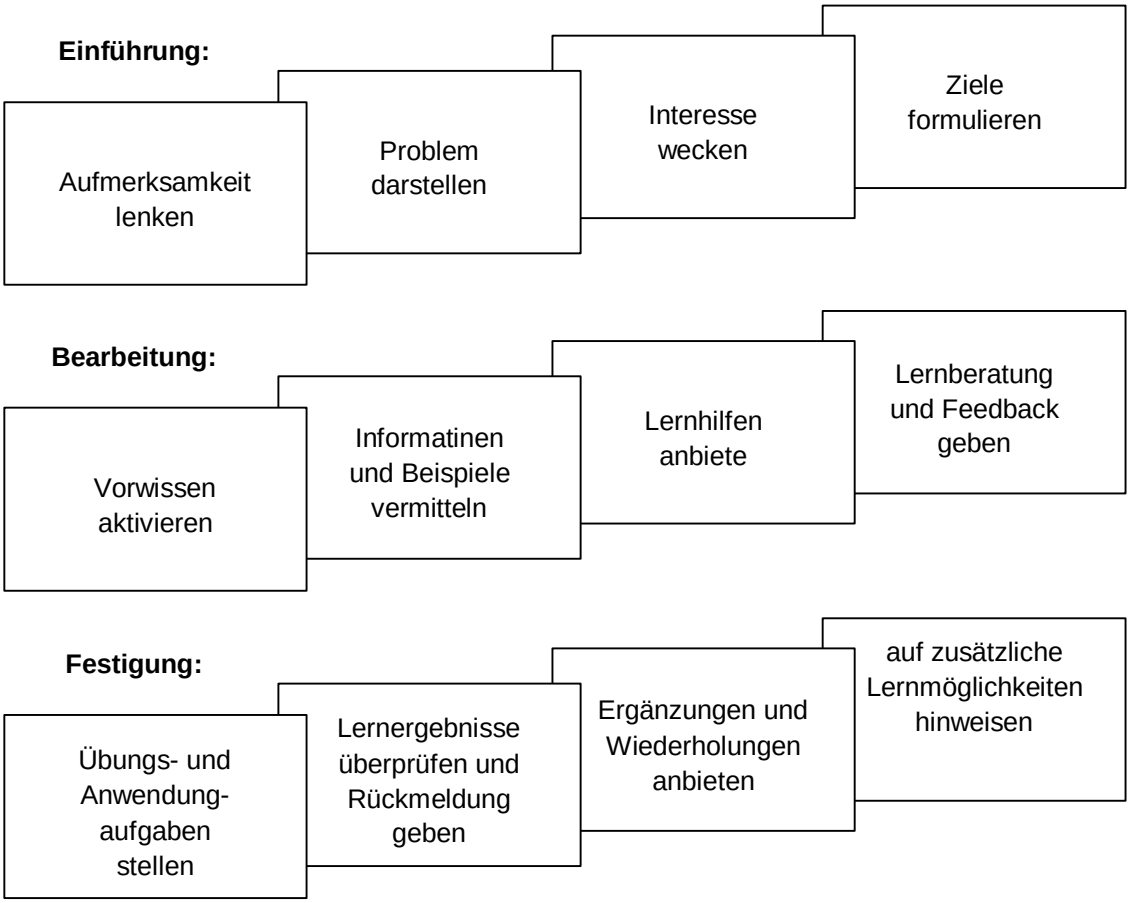


Abbildung 6: Arbeitsschritte des Unterrichtens als didaktische Strukturierungshilfe für die Entwicklung multimedialer Lernsoftware (ISSING 2002, S. 162)

Für die Abfolge der Lernschritte schlägt er ebenfalls vor auf altbewährte didaktische Regeln zurückzugreifen (ISSING 2002, S. 163): “

- *vom Allgemeinen zum Besonderen,*
- *vom Bekannten zum Unbekannten,*
- *vom Einfachen zum Komplexen,*
- *vom Leichten zum Schwierigen,*
- *vom Naheliegenden zum Entfernten,*
- *vom Interessanten zum weniger Interessanten,*
- *entsprechend der chronologischen Abfolge*
- *entsprechend der natürlichen Prozesse*
- *entsprechend der Fachsystematik*
- *entsprechend der Sachlogik usw.*“

Im Allgemeinen enthalten Lernangebote nach KERRES & JECHLE 2001, S. 5 meist „*verschiedene Formate von Lernmaterialien (z.B. Skripte, WBTs, Tests,*

Lernaufgaben) sowie lernförderliche Maßnahmen personeller und (infra-)struktureller Art (z. B. Betreuung, technische Hotline, Präsenzphase)“. In solchen Fällen erscheint es sinnvoll für den Lernenden die unterschiedlichen Formate und lernförderliche Maßnahmen in „*hybriden Lernarrangements*“ zu verbinden (ineinandergreifen zu lassen), um z. B. unterschiedliche Lernerfahrungen und Lernbedürfnisse zu unterstützen (KERRES & JECHLE 2001, S. 5). Tabelle 5 zeigt typische Bestandteile von Lernangeboten in hybriden Lernarrangements.

Tabelle 5: Typische Bestandteile von Lernangeboten in hybriden Lernarrangements (KERRES 2012, S. 390)

Bestandteil	***Traditionelle Varianten***	***Varianten mit digitalen Medien***
Vortrag mit Diskussion	Vortrag im Seminarraum, Hörsaal	Podcast, Video auf Abruf (Streaming), Videokonferenz
Selbstlernaktivität	Buch	Interaktives Lernprogramm im Internet, Multimedia (DVD)
Kooperatives Lernen	Partner- und Gruppenarbeit im Klassenraum	Videokonferenz, Groupware-basierte Kooperation
Tutoriell betreutes Lernen	Mentoren-Modelle (auch: Peer-Tutoren)	Online-Coaching, Tele-Tutoring
Kommunikatives Lernen	Gruppenansätze (Team-Building, Gruppenfeedback, Metakommunikation etc.	Soziale Netzwerke, Chat-Räume, Diskussionsforen
Beratung	Einzelgespräche, Informationsveranstaltungen	Beratung per E-Mail, FAQ-Liste, Community-basierte Ansätze (peer-to-peer)
Tests, Zertifizierung	Klausur, mündliche Prüfung	Computerbasiertes (adaptives) Testen

WILBERS (2011, S. 2 f) weist darauf hin, dass sich ein Lernprozess in einer Situation vollzieht. Das Lernen sei in dieser Situation nicht nur von einem Lehrenden, sondern von einer Fülle von Umweltfaktoren abhängig. Um dies zu betonen, würden heutzutage die Begriffe „*Lernarrangement*“ bzw. „*Lernumgebung*“ verwendet. In diesem Zusammenhang führt er zu diesen Begriffen weiter aus (WILBERS 2011, S. 3): „*Die didaktische Gestaltung von E-Learning in diesem Sinn ist das Entwerfen, Durchführen und Evaluieren von Lernumgebungen bzw. Lernarrangements bzw. Szenarien, die elektronische Medien und Dienste integrieren.*“ Auch ARNOLD et al. (2011, S. 101) verwenden den Begriff Lernarrangement ein und betonen, dass „*Kenntnisse über Lerntheorien für die Entwicklung von virtuellen Lehr- und Lernarrangements von großer Bedeutung*“ sind.

BAUMGARTNER (2006, S. 2) verwendet ebenfalls den Begriff „*Lernarrangement*“ und bringt ihn in Verbindung mit dem Thema „*Didaktisches Szenario*“: „*Ein Didaktisches Szenario ist demnach ein Skript für die Inszenierung eines bestimmten Lernarrangements und stellt die notwendigen Erfordernisse – Handlungen in der (Lern-)Zeit bzw. Ausstattung im (virtuellen) Raum – für die Umsetzung zusammen.*“ Er weist darauf hin, dass im E-Learning neben der Wiederverwendbarkeit von Lerninhalten auch die Wiederverwendbarkeit von didaktischen Ansätzen eine Rolle spielt. Didaktische Szenarien könnten dazu herangezogen werden.

Auch SCHULMEISTER (2006, S. 199 ff) erwähnt den Begriff und beschreibt ihn wie folgt: „*Bei den didaktischen Szenarien … handelt es sich um Beispiele für Lehren und Lernen, um Unterrichtssituationen und –modelle, die in ihren Komponenten, den Relationen der Komponenten untereinander und in den Prozessen, die davon ihren Ausgangspunkt nehmen, möglichst konkret und möglichst formal beschrieben werden.*“ Er weist bei einem Einsatz von didaktischen Szenarien auf Vorteile bei der didaktischen Planung, der Selektion von Kursen (z. B. Vergleich von Kursangeboten) und der Evaluation von Kursen hin.

Im Zusammenhang mit hybriden Lernarrangements greift KERRES (2012, S. 388) den Begriff „*Blended Learning*“ (auch „*integriertes Lernen*“ genannt, WIKIPEDIA-IL 2014) auf. Der Begriff habe etwa seit dem Jahr 2000 nach enttäuschenden Erfahrungen mit E-Learning (vor allem in der Wirtschaft) an Bedeutung gewonnen (siehe auch MONNE & LINKER 2011, S. 3). Er beschreibt ihn als „*Kombination von Präsenzelementen und medienbasierten Lernangeboten*“.

HÄFELE & MAIER-HÄFELE (2008, S. 15) erklärt den Begriff „Blended Learning“ wie folgt: „*Blended Learning bedeutet wörtlich ‚gemischtes Lernen‘ und bezeichnet die Verbindung von Online- und Präsenzelementen in Lernangeboten.*“ Sie weisen darauf hin, dass im deutschen Sprachraum meist der Begriff „*hybrides Lernen*“ benutzt wird, der sich gleichermaßen auf „*die Mischung aus mediengestütztem und Präsenzlernen*“ bezieht (siehe auch REY 2009, S. 18).

SCHULMEISTER (2006, S. 3) hält zu dem Begriff „Blended Learning“ fest, dass bei den aktuellen Diskussionen mehr die Integration des E-Learning in die Präsenzlehre betont wird als die Schaffung virtueller Universitäten. Er führt aus: „*In der Tat ist die Integration von eLearning in die Präsenzlehre, das sogenannte Blended Learning, sinnvoll.*“ Auch BAUMGARTNER et al. (2002 a, S. 13) weisen auf diesen Sachverhalt

hin: „*sondern der Mensch soll eine wesentliche Rolle spielen; also ist hybrides Lernen (voriges Jahr) – bzw. wie es dieses Jahr ganz modern heißt ‚blended learning' - angesagt.*" MONNET & LINKER (2011, S. 11) betonen auf der Grundlage der Untersuchungen von Hattie, dass „*Techniken ohne soziale Einbindung" wie „interaktive Video-Methoden, Computer-unterstützte Unterweisungen, … Web-basiertes Lernen*" alleine nicht ausreichen, um das Lernziel zu erreichen. Sie knüpfen daran an: „*Genau diese Erkenntnis gab ja den Anlass, sich verstärkt mit Blended Learning zu beschäftigen.*"

Die in den vorangegangenen Kapiteln beschriebenen Lerntheorien und didaktischen Ansätze können und werden im „Blended Learning" – d. h. in der Präsenzlehre und im E-Learning - eingesetzt (siehe auch z. B. MARESCH 2008, S. 17 ff, COENEN 2002, S. 28 ff, KERRES 2012, S. 8 ff, S. 388 ff, BLUMSTENGEL 1998, STANGL 2012, WILBERT 2011, S. 2 ff).

Im Folgenden wird der bereits erwähnte Begriff E-Learning genauer betrachtet.

3.2 E-Learning

Für den Begriff E-Learning werden unterschiedliche Definitionen und Schreibweisen angeboten. Im Folgenden wird die Schreibweise „E-Learning" verwendet (DUDEN 2012). Bezüglich der Definition von E-Learning hält z. B. KERRES 2012 (S. 18) Folgendes fest: „*Definition: E-Learning ist ein Oberbegriff für alle Varianten der Nutzung digitaler Medien zu Lehr- und Lernzwecken, sei es auf digitalen Datenträgern oder über das Internet, etwa um Wissen zu vermitteln, für den zwischenmenschlichen Austausch oder das gemeinsame Arbeiten an digitalen Artefakten*" (siehe auch WIKIPEDIA-E-LEARNING-DE 2012, KÖHLER 2010, ROTERING-STEINBERG & RICHTER 2011). REY (2009) greift ebenfalls den Ausdruck Medien auf und versteht auf Seite 15 unter E-Learning das *„… Lehren und Lernen mittels verschiedener elektronischer Medien*". Er führt aus, dass beim multimedialen Lernen verschiedene Aspekte berücksichtigt werden:

- Multimedialität (verschiedene Objekte oder technische Geräte zur Speicherung und Kommunikation, z. B. Buch, Audioplayer, Videoplayer, PC)
- Multicodalität (verschiedene Darbietungsarten der zu vermittelnden Informationen z. B. (Hyper-)Text, Bild, Animation)
- Multimodalität (verschiedene Sinnesmodalitäten zur Wahrnehmung und Verarbeitung, z. B. visuell, auditiv)

- Interaktivität (verschiedene Eingriffs- und Steuermöglichkeiten, z. B. Computersimulation, Feedback, Diskussion)

Auch CEDEFOP (2008, S. 69) erwähnt den Ausdruck Medien in seiner Definition von E-Learning: „*Bezeichnet Lernen, das durch Informations- und Kommunikationstechnologien (IKT) unterstützt ist.*“ Es wird zusätzlich angemerkt: „*eLernen beschränkt sich nicht nur auf ‚digitale Bildung‘ (Erwerb von IKT-Kompetenzen). Dazu gehören können auch unterschiedlichste Formate und hybride Methoden: der Einsatz von Software, Internet, CD-ROM, Online-Lernen oder anderer elektronischer bzw. interaktiver Medien; - eLernen kann sowohl in der Fernlehre als auch unterstützend im Präsenzunterricht eingesetzt werden*.“ Ähnlich schreibt WILBERS (2011, S. 2): *„‚E-Learning‘ ist eine Wortschöpfung aus ‚electronic‘ und ‚learning‘ und bezeichnet die Unterstützung von Lernprozessen durch elektronische Medien und Dienste. Dies verlangt Informationstechnik.*“

Vergleichbar mit den Aussagen zur Definition von E-Learning durch KERRES 2012 (S. 18) verstehen BAUMGARTNER et al. (2002 a, S. 14 f) unter E-Learning einen übergeordneten Begriff für softwareunterstütztes Lernen. Darunter fallen sowohl das Lernen mit lokal installierter Software (Lernprogramme, CD-ROM) als auch das Lernen über das Internet. BAUMGARTNER et al. (2002 a, S. 16) halten in diesem Zusammenhang weiter fest, dass nicht nur stationäre Computer, sondern auch andere Endgeräte wie z. B. Handy, Smartphone, Tablet-PC, eBook-Reader wichtige Funktionen im Lernprozess übernehmen können. Sie weisen noch zusätzlich auf Folgendes hin (2002 a, S. 16): „*Eigentlich ist e-Learning nur die eine Seite eines wechselseitigen Prozesses, der Lehren (e-Teaching) und Lernen (e-Learning) umfasst.*“ Solange allerdings der Begriff „*e-Education*“ noch nicht gebräuchlich ist, verwenden auch BAUMGARTNER et al. (2002 a, S. 16) diesen Begriff in der inhaltlich erweiterten Form inklusive „*e-Teaching*“. Im Folgenden wird sich dieser Definition angeschlossen.

Für den Ausdruck E-Learning oder auch E-Lernen werden synonym weitere Bezeichnungen wie z. B. CBT, Online-Lernen, Tele-Lernen, multimediales Lernen oder Distance-Learning verwendet (siehe REY 2009, S. 15, ZAWACKI-RICHTER et al. 2010, S. 3 f, WIKIPEDIA-E-LEARNING-DE 2012).

Im Folgenden werden Standardisierungsbemühungen im Themenbereich E-Learning beleuchtet.

3.2.1 Standardisierungsbemühungen

Viele Autorensysteme für E-Learning-Inhalte und Lernplattformen werden auf dem Software-Markt angeboten, weiterentwickelt und neu erstellt. Die Dynamik innerhalb dieses Metiers legt den Einsatz internationaler E-Learning-Standards nahe. ARNOLD et al. (2012, S. 325, siehe auch NIEGEMANN et al. 2004, S. 269) heben hervor: „*Standardisierungsinitiativen im E-Learning beschäftigen sich nicht nur mit der Frage, welchen Standards die technischen Anforderungen im E-Learning genügen müssen. Vielmehr weiten sich die Standardisierungsbemühungen auf den Bereich der didaktischen Gestaltung von virtuellen Lernszenarien aus. Dabei geht es sowohl um Standardisierungen in der Aufbereitung der Lerninhalte als auch in der Gestaltung der Lernprozesse und der Interaktion innerhalb der Lehr-Lern-Szenarien.*“ Sie halten weiter fest, dass bei der Definition von Standards durch JAKOBS (2000, S. 11) für den Bereich E-Learning die Sammlung von Eigenschaften und der Prozess der Standardisierung eine wichtige Rolle spielen.

3.2.1.1 Vorteile der Standardisierung

Standards finden sich häufig an Grenzstellen zwischen Systemen. Mit ihnen soll sichergestellt werden, dass ein Produkt sinnvoll in ein anderes System eingebunden werden kann. Die Möglichkeit E-Learning-Inhalte in andere Systeme funktionstauglich einzubinden schafft Unabhängigkeit von den verschiedenen - auch erst entstehenden - Lernplattformen oder Autorenwerkzeugen und spielt für die zukünftige Verwendung der erstellten E-Learning-Inhalte eine gewichtige Rolle. Erst bei einer Erstellung von standardisierten E-Learning-Inhalten wird der Wirtschaftlichkeit bei der Produktion dieser Inhalte Rechnung getragen werden können. Um Weiterentwicklungen nicht zu bremsen, sollte allerdings laut KNEBEL (2002) eine Standardisierung nicht bis ins kleinste Detail durchgeführt werden.

Nach SCHULMEISTER (2004) werden Einheitlichkeit oder Standards dort benötigt, wo es um den Austausch von Content oder den Transfer von Lerninhalten geht. Kompatibilität lässt sich „*mit der Einhaltung von Standards wie IMS oder ADL SCORM für den Export und Import von Inhalten*“ erreichen. Weiter führt er aus: „*Die großen Hersteller von Lernplattformen arbeiten fast alle zudem in der Open Knowledge Initiative (OKI) mit, die das MIT zusammen mit neun namhaften amerikanischen Universitäten ins Leben gerufen hat und in der APIs für derartige Zwecke und Referenzlösungen entwickelt werden.*"

Mit einer Standardisierung im E-Learning-Bereich könnten E-Learning-Inhalte vergleichbar, austauschbar, von verschiedenen Quellen/Systemen kombinierbar

(Interoperabilität), (auch in Zukunft) kompatibel und wieder verwertbar gehalten werden (siehe auch PONGRATZ 2002, ADLNET-SCORM 2014, Synopsis, KOPER et al. 2003 a). Ähnliche Begriffe werden auch z. B. bei MONTANDON (2004, S. 6 ff), SANCHEZ-ALONSO et al. (2011, S. 138) oder DEIBLER et al. (2008) als positive Aspekte einer Standardisierung aufgeführt:

- *„Accessibility" („Erreichbarkeit")*
- *„Adaptability" („Anpassbarkeit")*
- *„Affordability" („Wirtschaftlichkeit")*
- *„Durability" („Nachhaltigkeit")*
- *„Interoperability" ("Austauschbarkeit", „Kompatibilität", „System- und Plattformunabhängigkeit")*
- *„Reusability" („Wiederverwendbarkeit")*
- *„Discoverability" („Auffindbarkeit")*
- *„Extensibility" („Erweiterbarkeit")*
- *„Manageability" („Handhabbarkeit", „Einfachheit", „Flexibilität bei der Gestaltung")*.

3.2.1.2 Standardisierung de jure <-> de facto

Allgemein betrachtet können Spezifikationen von Standardisierungsgremien auf Länderebene oder von übergeordneten Standardisierungsgremien zu Standards erhoben werden. Dabei wird der Begriff Standard im englischsprachigen Bereich verwandt. In kontinentaleuropäischen Sprachen hat sich der Begriff Norm durchgesetzt. Peter SLOEP (2002) schreibt: "*Official certification bodies may be found at the national level (NEN, DIN, BSI, ANSI) and supranational level (CEN, IEEE, ISO). Strictly speaking, only after certification of a specification can one call it a true, i. e. de jure, standard. In some language - such as French, German, Dutch - de jure standards are called norms.*" So werden die Vorschläge (bzw. Spezifikationen) verschiedener Interessengruppen wie IMS und PROMETEUS bei nationalen und übergeordneten Standardisierungsgremien eingereicht und für die Standardisierung (Normierung) vorgeschlagen (Abbildung 7).

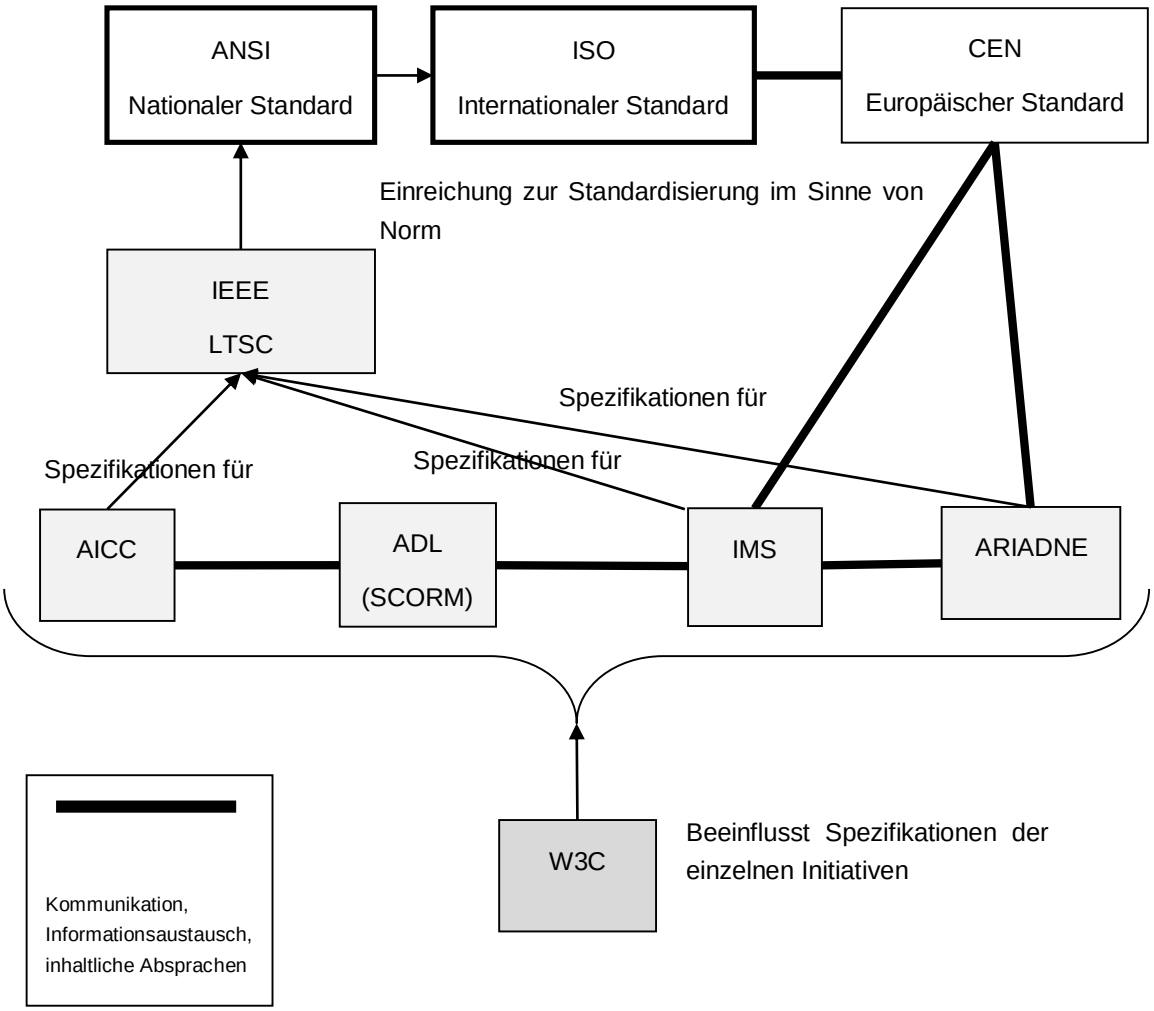


Abbildung 7: Verschiedene Interessengruppen und Standardisierungsgremien (verändert nach BAUMGARTNER et al. 2002, S. 279, siehe auch JUNGMANN 2012, S. 92, MONTANDON 2004, S. 10)

Im Folgenden sind einige Organisationen aufgeführt, die Standardisierungsbemühungen im Bereich E-Learning entwickeln (siehe auch HÄFELE 2002, BAUMGARTNER et al. 2002 b, S. 278 ff, NIEGEMMANN et al. 2004, S. 270, BÖR 2003, CEN/ISSS-WS/LT 2000, ELOQ 2014, MONTANDON 2004, S. 9 ff). Die Unterscheidung zwischen „*de jure*"- und „*de facto*"-Standards wurde von SLOEP (2002) angepasst übernommen (siehe auch KLEBL 2004, S. 5). Unter „*de jure*"-Standards werden Standardisierungsgremien aufgezählt, die auf einer rechtlichen Grundlage basierend Spezifikationen zu Standards erheben dürfen. Unter „*de facto*"-Standards werden Gremien aufgeführt, die Spezifikationen erstellen. Oft werden diese Vorschläge zur Standardisierung an die Standardisierungsgremien weitergereicht oder durch ihre weite Verbreitung selbst mitunter zu „*de facto*"-Standards (z. B. Betriebssysteme „Unix", „Microsoft Windows"). ARNOLD et al. (2012, S. 327) halten fest: „*Bei der Entwicklung eines ‚Standards' bedarf es, im Gegensatz zur ‚Norm', keiner staatlich anerkannten Institution.*"

Beispiele für Organisationen, die „*de facto*“-Standards schaffen:

- ISO „International Standards Organisation“ (http://www.iso.org/)
  (z. B. im Bereich E-Learning: ISO/IEC 16262 (siehe Kapitel 3.6 JavaScript), ISO/IEC 19796 (siehe Ende dieses Kapitels))
- ANSI „American National Standards Institute“ (http://www.ansi.org/)
- CEN/ISSS „Comité Européen de Normalisation/Information Society Standardization System“ (http://www.cenorm.be/isss/Workshop/lt/Default.htm)

Beispiele für Organisationen, die „*de facto*“-Standards schaffen (oder Spezifikationen an *„de jure“* Standardisierungsgremien weiterreichen):

- IEEE LTSC „Institute of Electrical and Electronics Engineers Learning Technology Standards Committee“ (http://ltsc.ieee.org/)
  (z. B. im Bereich E-Learning: LOM)
- AICC „Aviation Industry Computer based Training Committee “ (http://www.aicc.org/)
- ADL/SCORM „Advanced Distributed Learning/ Sharable Content Object Reference Model“ (http://www.adlnet.org/)
  (z. B. im Bereich E-Learning: SCORM)
- IMS „Instructional Management System“ (http://www.imsglobal.org/) (z. B. IMS LD, IMS CP, IMS MD)
- ARIADNE „Alliance of Remote Instructional Authoring and Distribution Networks for Europe“ (http://www.ariadne-eu.org/)
- ASTD „American Society for Training & Development“, „Certification Institute ECC E-Learning Courseware Certification“ (http://www.astd.org/)
- W3C „World Wide Web Consortium“ (http://www.w3.org/)
  (z. B. im Bereich E-Learning: XML, XSLT, XPath, XHTML, HTML, CSS)

Die verschiedensten Interessengruppen entwickeln und arbeiten im Bereich E-Learning. Viele Aktivitäten beschäftigen sich dabei mit ähnlichen Problemstellungen. Die in diesem Prozess entstehenden Spezifikationen werden bei Bedarf an die Standardisierungsgremien („*de jure*“-Standards) weitergereicht (siehe Abbildung 7, Abbildung 8). Andere (auch kommerzielle) Gruppen können auf vorhandene Standards aufbauen und Software für E-Learning-Plattformen, Darstellung von Lerneinheiten („Player“) oder Editoren für Lerneinheiten erstellen. Der Nachteil des

Standardisierungsprozesses liegt oftmals in seiner Langwierigkeit. Besonders im sich schnell wandelnden IT-Bereich kann sich das als problematisch erweisen.

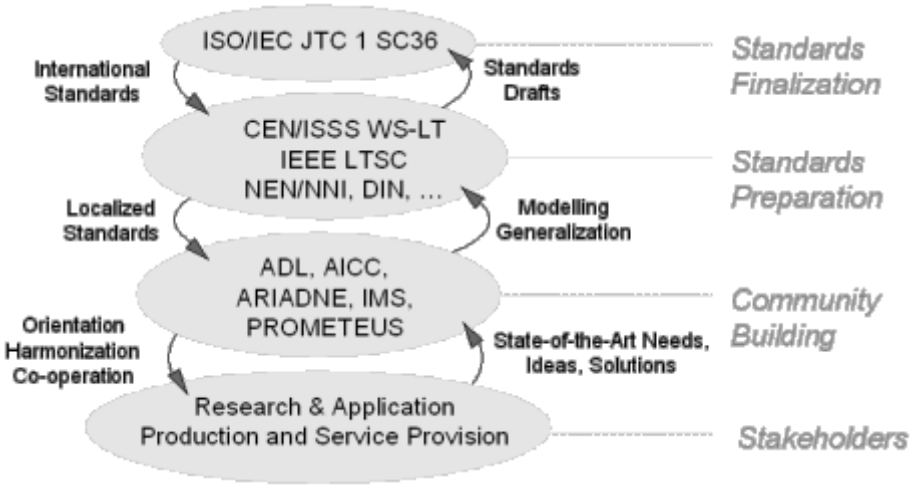


Abbildung 8: Ebenen der internationalen Zusammenarbeit im E-Learning Bereich (KNEBEL 2002)

Seit 2005 stellt die ISO für die Entwicklung und zur Qualitätssicherung von E-Learning eine eigene Norm zur Verfügung (darüber hinaus auch Norm ISO 29990:2010 „*Lerndienstleistungen für die Aus- und Weiterbildung - Grundlegende Anforderungen an Dienstleister*“). Der erste Teil dieser Norm ISO/IEC 19796-1:2005 (bzw. deutsche Fassung: DIN EN ISO/IEC 19796-1:2009-08 (D)) trägt die Bezeichnung „*Informationstechnik - Lernen, Ausbilden und Weiterbilden - Qualitätsmanagement, -sicherung und -metriken - Teil 1: Allgemeiner Ansatz*“. Sie bietet ein Rahmenmodell für das Qualitätsmanagement und die Qualitätssicherung im Bereich des E-Learning (ISO-19796-1 2005, KERRES 2012, S. 220 ff). In der ISO-Norm wird ihr Hauptaspekt angeführt: „*The main aspect is the Reference Framework for the Description of Quality Approaches (RFDQ).*“ EHLERS & PAWLOWSKI (2006, S. 68) halten fest: „*The standard is an instrument to develop quality in the field of e-learning.*“ KERRES (2012, S. 221 f) führt die Haupt-Punkte des allgemeinen Rahmenmodells Anforderungsermittlung, Analyse der Rahmenbedingungen, Konzeption, Entwicklung/Produktion, Einführung, Durchführung und Evaluation auf. Unter dem Punkt Konzeption werden z. B. Themen wie Lernziele, Inhalte, didaktisches Konzept und Methoden sowie Medieneinsatz aufgeführt (ISO-19796-1 2005, KERRES 2012, S. 221, siehe auch Kapitel 3.1 Lerntheoretische Grundlagen). Im nächsten Kapitel wird auf Methoden und Medien des E-Learning eingegangen.

3.2.2 Übersicht Methoden und Medien

DÖRING (2001) weist darauf hin, dass Methodenkenntnisse beim Lernen am PC eine wichtige Rolle spielen: „*Systematisches, organisiertes Lernen am PC macht nur*

dann Sinn, wenn es in ein Gesamtkonzept handlungsorientierten und ganzheitlichen Lernens eingebettet wird. Schüler wie Erwachsene müssen Methodenkenntnisse im Prozess der eigenen Lernpraxis erwerben können“ (siehe Kapitel 3.1.2 Didaktik). Zum Einsatz von Methoden betonen RÖPNACH & AUERT (2011, S. 8), dass eine durchdachte Kombination verschiedener Methoden bei der Vermittlung von Lerninhalten hilfreich ist, um insbesondere „*die in einer Lerngruppe vermutlich breit gefächerten individuellen Lernvorlieben aufzugreifen und zu bedienen*“.

Neben dem sinnvollen Einsatz verschiedener Methoden trägt auch der sinnvolle Einsatz verschiedener Medien zum Lernerfolg bei. Unter dem Ausdruck Medien versteht REY (2009, S. 16) „*verschiedene Objekte oder technische Geräte … mit denen sich Informationen speichern und/oder kommunizieren lassen*“ (siehe Kapitel 3.1.2 Didaktik). Auch SCHULZ (1965, S. 34) geht auf den Kommunikationsaspekt ein und beschreibt Medien wie folgt: „*Als Medien werden hier alle Unterrichtsmittel bezeichnet, deren sich Lehrende und Lernende bedienen, um sich über Intention, Themen und Verfahren des Unterrichts zu verständigen.*“ KERRES (2012, S. 131 ff) legt dar, dass der Medienbegriff erstaunlich „*vage*“ ist. Als eine mögliche Erklärung des Ausdruckes greift er u. a. neben dem Kommunikationsaspekt (vgl. auch SCHULMEISTER 2006, S. 135 ff), den Gedanken von CLARK (1983) auf, dass Medien als „*Transporter*“ von Lernangeboten betrachtet werden können.

Abbildung 9 zeigt lerntheoretische, medienorientierte und methodenorientierte Ansätze, die kombiniert im „Blended Learning“ (E-Learning und Präsenzlernen) eingesetzt werden können (vgl. zum Thema didaktische Methoden Kapitel 3.1.2 Didaktik). HÄFELE & MAIER-HÄFELE (2008) gehen auf viele weitere E-Learning-Methoden ein („101 e-Learning Seminarmethoden“).

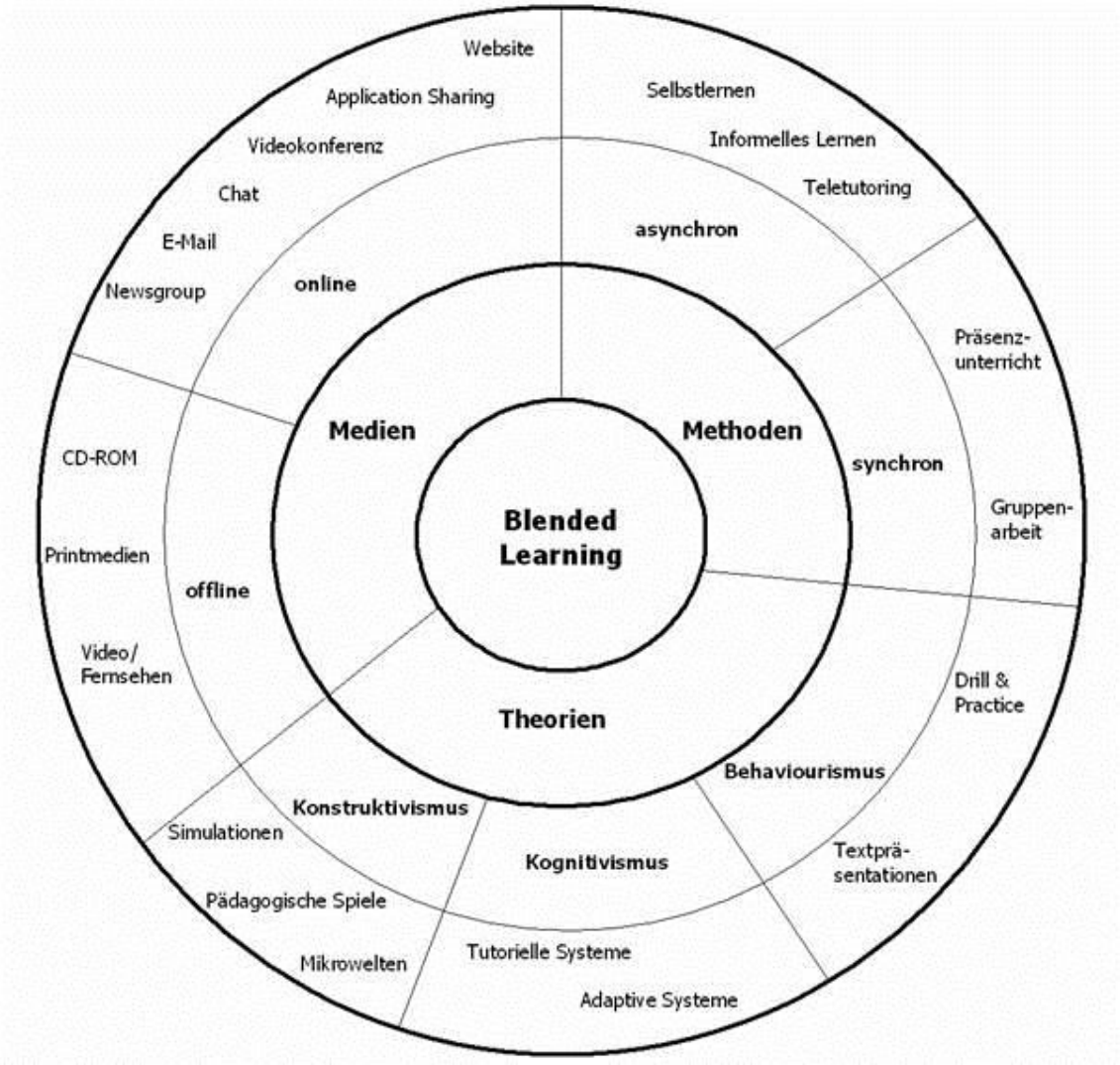


Abbildung 9: Medien, Methoden und Theorien im „Blended Learning“ (E-Learning und Präsenzlernen) (WIEPCKE 2006, S. 69)

GRÖHBIEL & SCHIEFNER (2006, S. 4) zeigen in Abbildung 10 einen Überblick über Methoden und Medien des E-Learning. Die sogenannte „*E-Learning-Landkarte*“ soll laut ihren Äußerungen einen Überblick über Lehr-/Lernmethoden sowie Medien geben und kann als „*Orientierungs-, Kommunikations-, Planungs- und Entscheidungsinstrument bei der Entwicklung eines E-Learning unterstützten Trainings dienen*“ (GRÖHBIEL & SCHIEFNER 2006, S. 1). Darüber hinaus werden „*Lernmethoden und -technologien in Bezug auf ihre Eignung zur Erreichung unterschiedlicher Lernziele, ihre Funktion im Lernprozess und ihre Unterstützung unterschiedlicher Interaktionsformen und Lerninhaltsstrukturen*“ abgebildet.

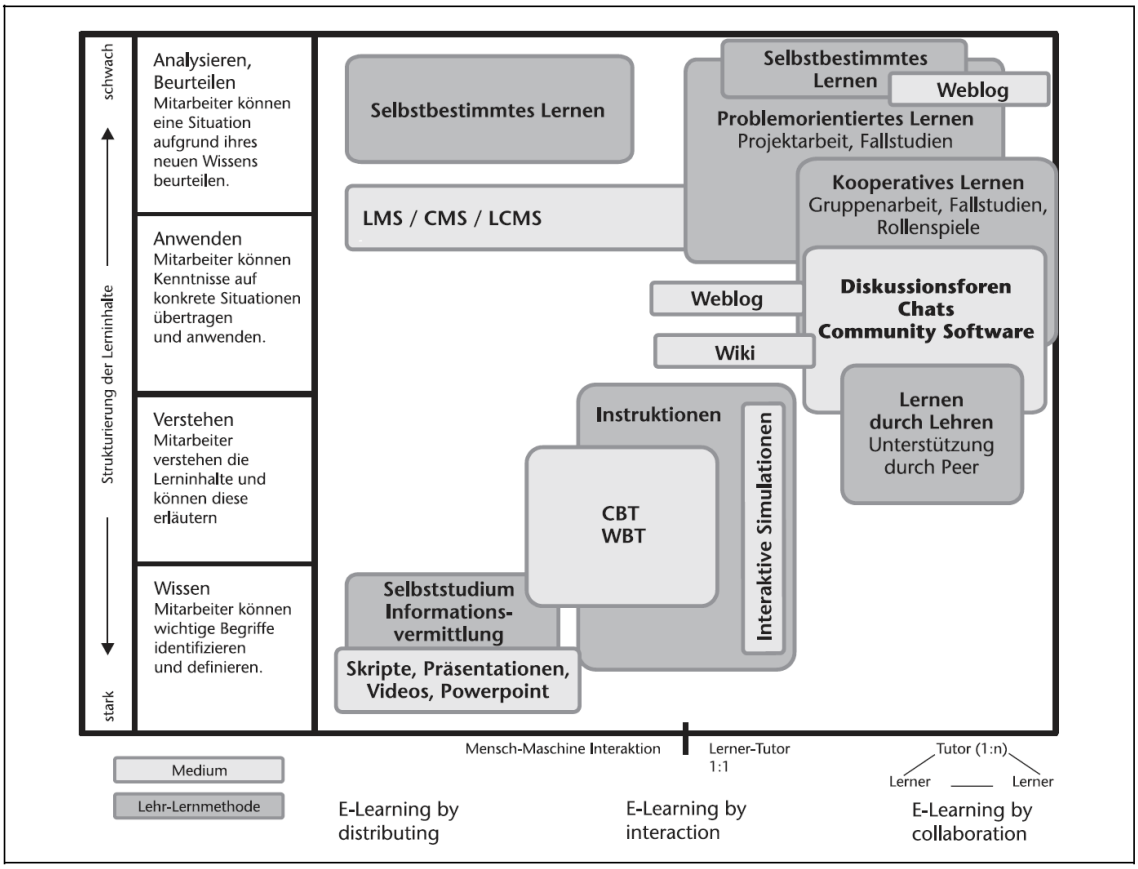


Abbildung 10: Die E-Learning-Landkarte mit Methoden und Medien des E-Learning (GRÖHBIEL & SCHIEFNER 2006, S. 19)

GRÖHBIEL & SCHIEFNER (2007, S. 1) gehen in ihrer E-Learning-Entscheidungsmatrix ebenfalls auf Medien ein. Sie halten zur Entscheidungsmatrix Folgendes fest: „*Die ‚E-Learning-Entscheidungsmatrix' veranschaulicht den Entscheidungsweg bei der E-Learning-Planung. Dabei werden eine wertschöpfungsorientierte Zielfindung und die Auseinandersetzung mit betrieblichen Rahmenbedingungen angeregt. In der Matrix werden verschiedene in einem Unternehmen grundsätzlich realisierbare Sozialformen, Lehrmethoden, Medien und Technologien übersichtlich dargestellt.*" Der Begriff Sozialformen des Lernens bezieht sich nach ihren Worten „*auf die pädagogische Interaktion (der Lernenden untereinander)*" (GRÖHBIEL & SCHIEFNER 2007, S. 16, SCHULZ 1965, S. 32 f, zu Lernformen siehe auch KLIMSA 1993, S. 277 f). Als mögliche Formen führen sie selbstbestimmtes oder selbstgesteuertes Lernen, Lernen durch Interaktion und Lernen durch Kooperation auf (vgl. SCHULZ 1965, S. 32, siehe auch den Begriff Lernform bei MEIER 2006, S. 21 ff). Die Sozialform des Lernens kann auch als ein Aspekt von Lehr-/Lernmethoden betrachtet werden (vgl. Abbildung 9, siehe WILBERS 2011, S. 6, siehe WIKIPEDIA-METHODIK-PÄDAGOGIK 2012). Abbildung 11 zeigt ein Beispiel einer E-Learning-Entscheidungsmatrix (mit möglichen

Sozialformen des Lernens und Medien), bei der die Methoden aus dem Zielsystem abgeleitet wurden.

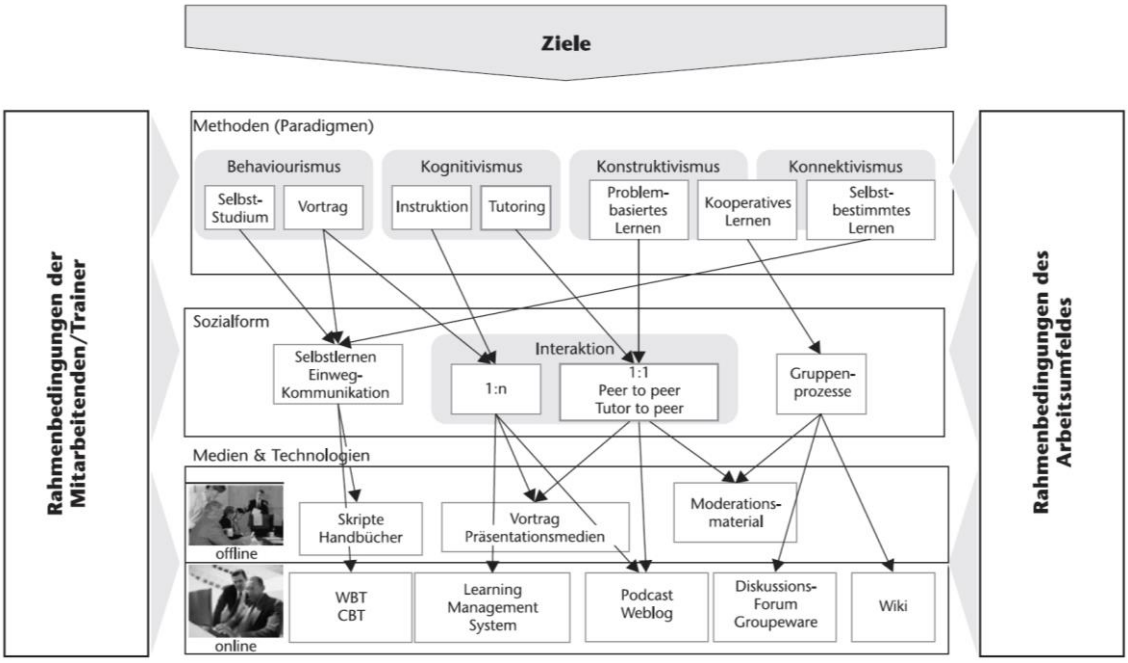


Abbildung 11: Beispiel einer E-Learning-Entscheidungsmatrix mit einsetzbaren Medien (GRÖHBIEL & SCHIEFNER 2007, S. 13)

Im Zusammenhang mit den Sozialformen des Lernens sollte auf die wachsende Bedeutung sozialer Netzwerke wie z. B. „Facebook“, „Google+“, „XING“, „Twitter“ oder anderer Plattformen hingewiesen werden. Nach KERRES (2012, S. 161 ff) kann der Austausch von Informationen über diese Medien als informelles Lernen bezeichnet werden, „*bei dem sich Menschen auch ohne betreuende oder lehrende Instanz organisieren*“.

Im Folgenden sind abschließend einige weitere Ausdrücke für Medien aufgeführt (siehe auch ARNOLD et al. 2011, S. 135 ff, HÄFELE & MAIER-HÄFELE 2008, S. 16, STANGL 2012): Instant-Messaging (Versand von Nachrichten zwischen zwei Rechnern, KERRES 2012, S. 16), virtuelles Klassenzimmer (Konferenzwerkzeug mit speziellen Funktionen für Kommunikation und Kooperation, KERRES 2012, S. 17), Webcast (broadcast = Ausstrahlung über das World Wide Web mit Möglichkeit der Interaktion, WIKIPEDIA-WEBCAST 2012), Podcast (auditives oder audiovisuelles Material per Internet zur Verfügung gestellt, ARNOLD et al. 2011, S. 181) oder Internet-Telefonie. HOLZINGER (2000, S. 187) erwähnt des Weiteren den Ausdruck Hypermedia und definiert ihn wie folgt: „*Hypermedia ist ein Oberbegriff für Hypertext und unterscheidet sich von Multimedia per definitionem lediglich durch die nicht lineare Verknüpfung der Informationsknoten.*“ Weiter gibt er für das Hypertext-

Konzept die beiden grundlegende Elemente Knoten und Links an. Er hält fest: „*Knoten sind die atomaren Informationseinheiten von Hypertexten, die über Links nicht linear miteinander verbunden (verlinkt) sind.*“ Diese Knoten eines Hyperdokumentes können Text und / oder Grafiken beinhalten. Wenn die Knoten daneben z. B. auch Audioelemente, Videos oder Animationen enthalten, „*so wird von Hypermedia gesprochen*“ (siehe auch KLIMSA 1993, S. 110). BLUMSTENGEL (1998, S. 128) sieht die wesentliche Bedeutung von Hypermedia darin, dass es „*weitgehend flexibel ist und an unterschiedliche Lernsituationen durch verschiedene Grade der Benutzerführung anpaßbar ist.*“ Sie hält weiter zu dem Ausdruck Hypermedia fest: „*Aufgrund seiner nichtlinearen Struktur und der Möglichkeit der Integration reichhaltiger Darstellungsformen ist es jedoch besonders für die Unterstützung selbstgesteuerten, problemorientierten Lernens geeignet.*“

HÄFELE & MAIER-HÄFELE (2008, S. 16) führen Medien unter dem Ausdruck „*Kommunikationswerkzeuge*“ und betonen damit den Kommunikationsaspekt des Medienbegriffes. ARNOLD et al. (2011, S. 135 ff) verwenden den Ausdruck „*Bildungsressourcen*“.

Im nächsten Kapitel wird detaillierter auf die Modellierung von E-Learning-Inhalten (mit ihren Medien) eingegangen.

3.2.3 Modellierung von E-Learning-Inhalten

Nach SÜSS (2004, S. 20) kann E-Learning „*aufgrund der Weitergabe von Wissen und des Lernens von Informationen als Teil des Themenbereiches Informationsmanagement aufgefasst werden*“. FREITAG (2002 b) stellt in Abbildung 12 in vereinfachter Form den Kreislauf von Wissen, Informationen und Daten zwischen Mensch und Maschine im Informationsmanagement dar.

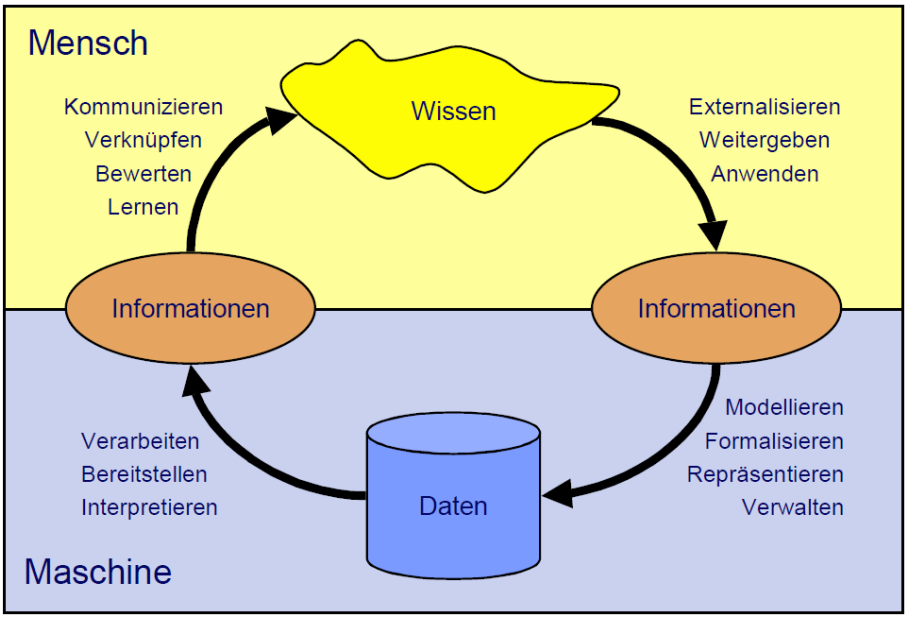


Abbildung 12: Informationsmanagement, Zyklus von Daten, Informationen und Wissen in schematisierter Form (FREITAG 2002 b)

Er hält weiter Folgendes fest: „*Informationsmanagement hat die Aufgabe, Informationen so zu modellieren, zu repräsentieren, zu organisieren, zu verwalten und verteilen, dass sie der richtigen Zielgruppe zur rechten Zeit im adäquaten Zusammenhang unter dem relevanten Blickwinkel und in der gewünschten Form zur Verfügung stehen und auf sie auf dem jeweils passenden Weg zugegriffen werden kann.*“ SÜSS (2004, S. 21) überträgt diese Anforderungen an das Informationsmanagement auf das Management von E-Learning-Inhalten. Er definiert E-Learning-Inhalte demnach wie folgt: „*Ein eLearning-Inhalt ist eine Menge von formalisierbaren, modellierbaren, maschinell repräsentierbaren und verwaltbaren Informationen, die im softwareunterstützten Lehren und Lernen verarbeitet, bereitgestellt und interpretiert werden, um das Wissen einer eLearning-Domäne zu lehren oder zu lernen.*“ Diese Anforderungen an das Management von E-Learning-Inhalten stellt er in einer „*internen Sicht*“ Abbildung 13 dar. Unter Domäne kann ein Einsatzbereich verstanden werden (WIKIPEDIA ANWENDUNGSDOMÄNE 2012).

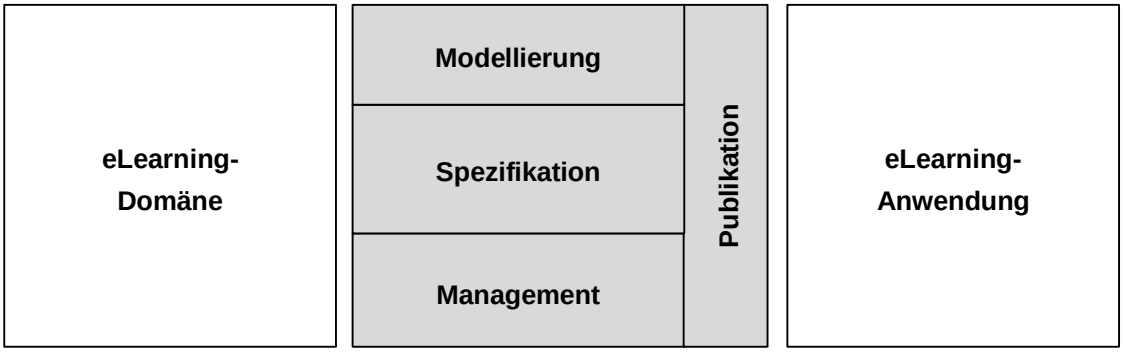


Abbildung 13: Externe (außen) und interne (innen, dunkel eingefärbt) Sichten auf das Management von E-Learning-Inhalten (SÜSS 2004, S. 21)

Nach seinen Ausführungen beschreibt die erste Ebene „*Modellierung*“ (der „*internen Sicht*“) die Strukturierung von E-Learning-Inhalten (vgl. z. B. auch ISO-13250 2003 oder GERSDORF 2003): „*XML Topic Maps*“ als Instrument, „*um Wissen in Form semantischer Netze zu modellieren*“). Der Begriff Strukturierung wird von JUNGMANN (2012, S. 43) unter Hinweis auf SUHR & SUHR (1993, S. 68) als „*Zerlegung einer Ganzheit in Sinneseinheiten unter Anwendung der Prinzipien Hierarchisierung und Modularisierung*“ erklärt. Die zweite Ebene „*Spezifikation*“ beschreibt die Repräsentation von E-Learning-Inhalten z. B. mittels XML-basierter Spezifikationen oder proprietärer Dateiformate, die dritte Ebene „*Management*“ die Verwaltung von E-Learning-Inhalten z. B. in Datenbanken. Der Bereich „*Publikation*“ stellt die Schnittstelle zur E-Learning-Anwendung dar und beschreibt die Nutzung der E-Learning-Inhalte durch z. B. Autoren oder Lernende (SÜSS 2004, S. 21).

Den Begriff Modellierung erläutern DELFMANN & KAROW (2008) als einen Vorgang „*bei dem eine Person auf Basis ihrer Wahrnehmung eine Repräsentation des Sachverhaltes (real, gedacht) konstruiert*“ (Modellbildung als Konstruktion). Er weist darauf hin, dass die Modellierung zweckorientiert ist und der Modellzweck durch den Modellnutzer definiert wird. Abschließend hält er fest: „*Modelle werden häufig in künstlicher, formalisierter Sprache dargestellt (Modellierungssprachen)*“. KECHER (2006, S. 13) hebt allgemein die Bedeutung der Modellierung hervor, indem er auf die „*Unumgänglichkeit der Modellierung großer Softwaresysteme*“ hinweist. Ähnlich stellt FROMMER (2010, S. 281) die Einbindung eines formalen Modells (Modellierung) als „*generelle Vorgehensweise in der Informatik*“ dar (siehe auch SUHR & SUHR 1993, S. 23 f). GEEB (2003, S. 27) beschreibt Datenmodellierung als den „*Entwurf von Dateneinheiten und ihren funktionalen Beziehungen ohne Rücksichtnahme auf bereits bestehendes Datenmaterial*“. Im Gegensatz dazu

bezeichnet er Datenstrukturierung als „*Tätigkeit, bei der aus bestehenden Material eine Struktur abgebildet (und diese in XML dargestellt) wird*“.

SÜSS (2004, S. 31) beschreibt in seinen weiteren Ausführungen den Weg von der E-Learning-Domäne zur Informationsmodellierung (Abbildung 13). Bei diesem Begriff unterscheidet LOBIN (2000, S. 9) zwischen der Ebene der konkreten Daten (z. B. multimediale Daten in Form von JPEG-Dateien; entspricht in XML einem Daten-Element) und der Ebene der abstrakten Einheiten (Gruppierung der Daten, Zuordnung von Funktionen, entspricht in XML einem Container-Element, das Daten-Elemente oder Container-Elemente enthalten kann). SÜSS (2004, S. 31) betont, dass bei letzterer eine modulare Strukturierung eine Grundvoraussetzung u. a. für die Wiederverwendung von E-Learning-Inhalten und Adaptation (Anpassbarkeit, siehe S. 19) ist. Er setzt dafür das „*abstrakte Datenmodell (ADM)*“ ein und definiert das ADM wie folgt (SÜSS, S. 38): „*Das Abstrakte Datenmodell (ADM) modelliert unabhängig von einem spezifischen Aspekt die modulare Fragmentierung von Informationen, Metadaten für Informationseinheiten, sowie Beziehungen zwischen Informationseinheiten, Beziehungen, Themen und Kontexten.*“ Abbildung 14 zeigt das ADM. Zur Visualisierung wird die grafische Sprache UML (WIKIPEDIA-UML 2012, KECHER 2006, S. 15, ISO-19505 2012) eingesetzt. Abbildung 14 stellt die Inhaltsfragmente in Form von UML-Klassen (Rechtecke) dar (SUESS 2004, S. 27 f). FORBRIG (2007, S. 18) definiert eine UML-Klasse wie folgt: „*Eine Klasse beschreibt ein Sammlung von Objekten mit gleichen Eigenschaften (Attributen), gemeinsamer Funktionalität (Methoden), gemeinsamen Beziehungen zu anderen Objekten und gemeinsamer Semantik*“ (vergleiche auch KECHER 2006, S. 32). Zu dem Begriff Objekt hält er u. a. fest: „*Ein Objekt ist allgemein ein Gegenstand des Interesses, insbesondere einer Beobachtung, Untersuchung oder Messung. Objekte können Dinge oder Begriffe sein.*“ Als abstrakte Klasse bezeichnet er eine Klasse zu der keine Objekte existieren können. Wenn Objekte zu einer Klasse definiert werden können wird sie als konkrete Klasse bezeichnet (FORBRIG 2007, S. 27, siehe SUESS 2004, S. 39 ff). Ein in der Abbildung 14 ausgefülltes kleines Viereck stellt eine „*Komposition*“ dar. FORBRIG (2007, S. 24) definiert eine UML-Komposition mit folgenden Worten: „*Eine Aggregation mit sehr starker Bindung wird als Komposition bezeichnet.*“ Als Aggregation bezeichnt er „*eine ‚Teil-Ganzes-‘ oder ‚Ist-Teil-von-‘ Verknüpfung*“. Ein nicht ausgefülltes Dreieck stellt in der Abbildung eine Vererbung dar. FORBRIG (2007, S. 26) definiert Vererbung wie folgt: „*Klassen werden in eine Hierarchie eingeordnet, die eine Weiterleitung von Informationen von oben nach unten ermöglicht. Eine Unterklasse verfügt dann über die Eigenschaften und das Verhalten der Oberklasse. Sie ‚erbt‘ diese Charakteristika. Entsprechend wird die*

Hierarchie auch als Vererbungsstruktur oder Klassenhierarchie genannt.“ Er hält weiter fest, dass eine Vererbung nur zwischen Klassen möglich ist.

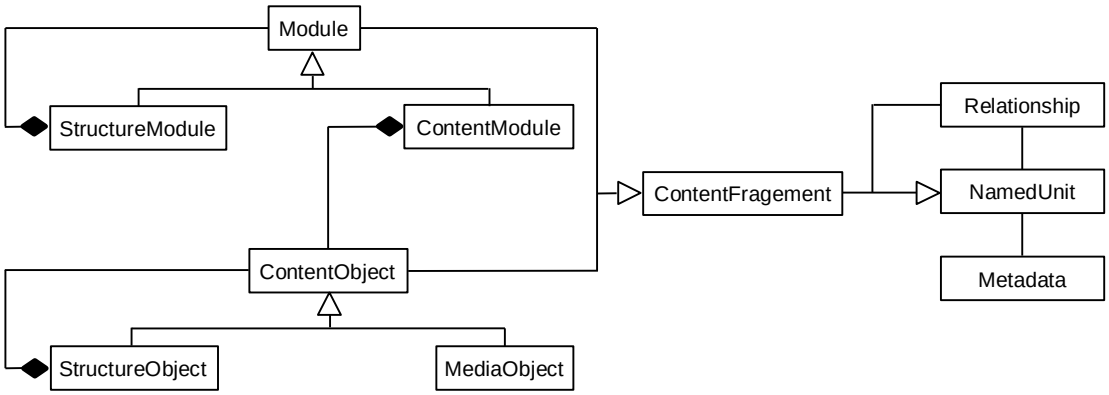


Abbildung 14: Abstraktes Datenmodell (ADM) (Darstellung einer modularen Strukturierung von E-Learning-Inhalten) nach SÜSS (2004, S. 32, 40)

Gemäß seiner Definition des ADM (siehe oben) und weiteren Ausführungen stellt für SÜSS (2004, S. 38 ff) das ADM den allgemeinen modularen Aufbau von E-Learning-Inhalten dar und „*beschreibt … die unterschiedlichen Strukturierungsmöglichkeiten ohne deren direkte Instantiierung festzulegen*“. Diese E-Learning-Inhalte setzen sich aus Modulen zusammen, die selbst wiederum Module beinhalten können. Inhaltsmodule (ContentModule) können inhaltlich über Strukturobjekte (StructureObject) strukturiert sein und Medienobjekte (MediaObject) enthalten (Abbildung 14). Ein Modul wird von SÜSS (2004, S. 39) wie folgt definiert: „*Ein Modul ist eine abstrakte Repräsentation einer für die Wiederverwendung benennbaren und mit Metadaten beschreibbaren Informationseinheit.*“

Die angesprochenen E-Learning-Inhalte vereinen in sich bei genauerer Betrachtung verschiedene Aspekte wie didaktikorientierte Anordnung, medienorientierte Präsentation oder sachlogikorientierte Hierarchie (SÜSS 2004, S. 33 ff). Alle diese Aspekte können modelliert werden. Die didaktikorientierte Anordnung läßt sich mit eigenen didaktischen Modellen beschreiben (siehe z. B. Kapitel 3.1.2 Didaktik, 3.3.7.3 IMS LD) und kann als „*Didaktikfolie*“ über die sachlogische Strukturierung gelegt werden (SÜSS 2004, S. 35 ff). Die medienorientiere Präsentation von E-Learning-Inhalten kennt ebenfalls verschiedene Modellierungsansätze (z. B. Strukturierung für einen hypermedialen Kurs (SÜSS 2004, S. 35) oder Strukturierung von gedruckten Unterlagen). Das Modell der „*modularen Grobstruktur*“ (SÜSS & FREITAG 2000 a, S. 115 ff, vgl. SÜSS et al. 1999, S. 6 ff) kann als Beispiel herangezogen werden. Der Aspekt der sachlogikorientierten Hierarchie wird über das Passauer Teachware-Modell (PTM) modelliert. Als eine Instanz des ADM

spezialisiert es „*für die sachlogische Informationsmodellierung von E-Learning-Inhalten die Inhaltsfragmente des ADM*“ (SÜSS 2004, S. 44, Abbildung 15, ADM-Bausteine als blaue Rechtecke, sachlogikorienterte Hierarchie-Bausteine als gelbe Rechtecke).

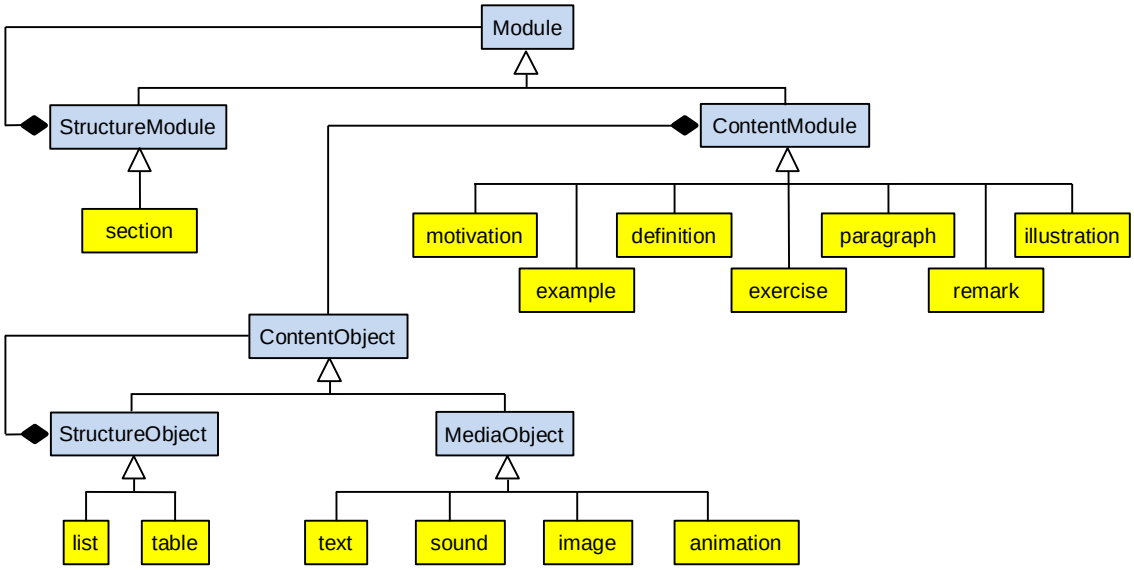


Abbildung 15: Sachlogische Fragmentierung von E-Learning-Inhalten im PTM (SÜSS 2004, S. 45)

Ähnlich den wieder verwendbaren Modulen von SÜSS (siehe oben, vgl. auch Lernmodul bei KERRES 2012, S. 11 f) beschreibt JUNGMANN (2012, S. 80) wieder verwendbare Lernobjekte als Ergebnis der Modularisierung. Dabei hält sie fest, dass die Festlegung der geeigneten Granularität eine der größten Herausforderungen bei der Modularisierung darstellt (vgl. auch JUNGMANN et al. 2004, S. 19 f). Ihrer Ansicht nach kann unter Hinweis auf SOUTH & MONSON (2000) auch ein ganzer Kurs ein Lernobjekt sein, da die Eigenschaft der Wiederverwendbarkeit entscheidend ist. Sie definiert Lernobjekte als „*digitale, wiederverwendbare Ressourcen, die eine didaktische Funktion im Lernprozess erfüllen*“ und aus Medienobjekten bestehen (z. B. Video, Animation, Grafik) (JUNGMANN 2012, S. 79). Ähnlich versteht KOPER (2003) unter einem Lernobjekt: „*any digital, reproducible and addressable resource used to perform learning activities or learning support activities, made available for others to use*“ (siehe auch WILEY 2000, S. 2 ff). Sehr viel allgemeiner ist die Definition HODGINS (2005, DEIBLER et al. 2008, S. 75 (11-3)) gefasst: „*Learning Objects are defined here as any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning.*“ BAUMGARTNER et. al. (2002 a, S. 42) definieren den Begriff „Learning Object“ wieder enger: „*Ein LO (Learning Object) ist die kleinste sinnvolle Lerneinheit, in die ein Online-Kurs zerlegt werden kann.*“ Ein „Learning Object“ kann somit nach ihren Angaben z. B. aus einem

einzelnen Bild, einem Text, einer kurzen Anweisung mit definiertem Lernziel oder einem Test zur Erfolgskontrolle bestehen. Weiter halten sie fest: „*Wenn diese LO's mit Metadaten versehen und zu größeren Online-Kurseinheiten kombiniert werden können, dann spricht man von RLO's (Reusable Learning Objects = wieder verwendbare Lernobjekte)*“. Auch SÜSS (2004, S. 55) erwähnt den Ausdruck „*Reusable Learning Objects*“ und hält die Ähnlichkeit eines von DEIBLER et al. (2008, S. 75 (11-3), HODGINS 2005) beschriebenen (reusable) „Learning Object“ mit einem ADM-Modul fest.

LÖSER et al. (2003, S. 5) unterscheiden bei „Learning Objects“ verschiedene Typen und weisen darauf hin, dass sie „*unterschiedliche Qualitäten auf verschiedenen Granularitätsebenen*“ aufweisen. In Anhalt an diese Unterscheidung werden nach STREHL et al. (2005, s. 2) die folgenden Lernobjekt-Typen in ihrer Granularität differenziert:

- „*Atomares Informationsobjekt*“
  Unter einem atomaren Informationsobjekt werden einzelne Medienbausteine wie z.B. Bilder oder Videos verstanden.
- „*Lernmodule*“
  Ein Lernmodul enthält ein abgeschlossenes Thema mit einem konkreten Lernziel. Es ist unabhängig von anderen Lernmodulen, kann aber auf andere Lernmodule vorbereiten, Inhalte vertiefen oder weiter ausführen. Konzeptionell entspricht ein Lernmodul einem RLO (zu RLO siehe BAUMGARTNER et. al. 2002 a (S. 42) und vorherige Ausführungen).
- „*Lerneinheit*“
  Eine Lerneinheit setzt sich aus Lernmodulen zusammen. Es bildet ein komplexeres und themenübergreifendes Lernobjekt.
- „*Kurs / Vorlesung*“
  Ein Kurs steht für eine komplette Lehrveranstaltung. Dieses Lernobjekt besteht aus Lerneinheiten und Lernmodulen und kann kommunikative oder kollaborative Elemente beinhalten.

Im Folgenden wird sich dieser Vorstellung der Unterscheidung von Lernobjekten angeschlossen. Weitere Hinweise zu „Learning Objects“ führen z. B. HAMEL & RYAN-JONES (2002) auf.

Zusammenfassend stellen FREITAG et al. (2002, S. 354) fest, dass die flexible Wiederverwendung von E-Learning-Inhalten eine modulare Strukturierung, “*die Beschreibung mit geeigneten Metadaten, sowie die strikte Trennung von Inhalt, Navigation, didaktischer Strukturierung, Curriculum-Design, Layout und*

Visualisierung“ voraussetzen. XML-basierte Spezifikationen bieten sich als geeignete Instrumente zur Umsetzung dieser Vorstellungen an (siehe z. B. Abbildung 14 zweite Ebene „*Spezifikation*“, LOBIN et al. 2003, S. 2 ff). LOBIN (2000, S. 3) weist in diesem Zusammenhang allgemein darauf hin, dass XML ein Instrument zur Modellierung von strukturierten Informationen ist (siehe auch ANDERSON et al. 2000, S. 47, S. 123 ff, siehe auch SKULSCHUS & WIEDERSTEIN 2004, S. 25 ff, S. 41 ff). Auch GEEB (2003, S. 12) beschreibt XML als ein „*abstraktes Werkzeug zur Modellierung und Strukturierung von Informationen und zur Haltung von Daten*“.

3.3 XML

Für die Modularisierung, die Wiederverwendbarkeit und den Austausch von Inhalten mediengestützter Bildungsprozesse ist eine Trennung von Lernsystem und Lehrinhalt sinnvoll (KLEBL 2004, S. 4). Damit die E-Learning-Inhalte selbst zwischen Institutionen ohne größere Reibungsverluste ausgetauscht und wieder verwendet werden können, bietet es sich an, sie in einem einheitlichen Datenformat zu erstellen. Die weltweit verbreitete Extensible Markup Language XML kann dies leisten (ADOLPHE 2012). Nach STÜHRENBERG (2004, S. 403) ist sie eine viel genutzte Grundlage zur strukturierten Speicherung von E-Learning-Inhalten. Als textbasierte Auszeichnungssprache erlaubt sie die Beschreibung, den Austausch, die Darstellung und die Manipulation von strukturierten Daten (siehe auch LIAM 2010 a). KORNELSEN et al. (2004, S. 31, siehe auch JUNGMANN et al. 2004, S. 4) stellen im Zusammenhang mit XML fest: "*Nicht nur textuelle Elemente können so strukturiert, semantisch ausgezeichnet und mit Metadaten annotiert werden, sondern auch dynamische bzw. interaktive Multimedia-Komponenten, Formeln u. v. m.*“

Einige Autoren weisen darauf hin, dass zur Repräsentation von E-Learning-Inhalten zunehmend XML eingesetzt wird (RINN et al. 2003, S. 26, STÜHRENBERG 2004, S. 403). VOIGT & TAVANGARIAN (2003, S. 9) gehen noch einen Schritt weiter und erklären: „*Die Nutzung von XML als Grundlage zur Auszeichnung von eLearning-Inhalten hat sich mittlerweile auf breiter Front durchgesetzt. Die Vorteile liegen klar auf der Hand: XML stellt ein plattformunabhängiges, textbasiertes, leicht lern- und lesbares Datenformat dar, welches zudem das Prinzip der Trennung des Inhalts vom Layout sinnvoll unterstützt sowie flexible Möglichkeiten zur Zwischen- und Weiterverarbeitung bietet.*“ Auch BUNSCHKOWSKI et al. (2004, S. 139) legen dar: „*Zur Erstellung digitaler Lehr- und Lernmaterialien hat sich XML als ein Beschreibungsmittel für Dokumenteninhalte durchgesetzt.*“

Für Office-Anwendungen wie Textverarbeitung, Tabellenkalkulation, Präsentationssoftware hat sich das Datenformat XML bei Microsoft, OpenOffice und anderen Organisationen durchgesetzt. Es wird z. B. in „Microsoft-Office“-Produkten wie „Word“ und „Excel“ als Standard-Speicherformat eingesetzt (ab Version 2007). Die Dateinamenserweiterung z. B. „.docx“ gibt mit dem Buchstaben „x“ den Hinweis auf XML.

SCHREYER hält bereits 2002 fest: „*XML gilt heute als die wichtigste Sprache für den Austausch, die Verwaltung und die Repräsentation von strukturierten Informationen. Sie wird derzeit von praktisch allen maßgeblichen Softwareentwicklern im Bereich des Internets als die zentrale Zukunftstechnologie angesehen.*“ Auch (BINSTOCK et al. S. 4) betont: „*XML has become the definitive – and perhaps the de facto – standard for transmitting data between distinct pieces of software – and computers.*“

XML ist eine Untermenge des ISO-Standards „*Standard Generalized Markup Language*“ SGML (ISO-8879 1986). XML wurde 1998 vom „W3C Konsortium“ verabschiedet und liegt derzeit als „*Recommendation*“ vom 26. November 2008 vor (BRAY et al. 2008). LIAM (2010 a) beschreibt XML wie folgt: „*The Extensible Markup Language (XML) is a simple text-based format for representing structured information: documents, data, configuration, books, transactions, invoices, and much more.*“ Ähnlich weist HAROLD (2004, S. 32) darauf hin, dass XML die „*Struktur und Bedeutung eines Dokuments*“ (und keine Formatierung) beschreibt. GOLDFARB & PRESCOD (2002, S. 32) halten den Zweck von XML wie folgt fest: „*XML is for the digital representation of documents.*“ GEEB (2003, S. 18) legt dar, dass XML - im Gegensatz zu z. B. HTML (siehe Kapitel 3.4 HTML) oder der verschiedenen XML-Anwendungen (siehe 3.3.2 XML-Anwendung) - kein Vokabular zur Auszeichnung von Dokumentinhalten aufweist. Vielmehr ist XML „*eine formale Sprache zur Erzeugung (Konstruktion, Definition) von Auszeichnungssprachen*“ (siehe auch DEITEL et al. 2001, S. 111). Nach GEEB (2003, S. 18) kann sie aus diesem Grund auch als Metasprache bezeichnet werden (siehe auch ROTHFUSS & RIED 2001, S. 132, HAROLD 2004, S. 31, HAROLD & MEANS 2001, S. 3), „*wobei die Objekte von XML die konkreten Auszeichnungssprachen sind*“.

Die folgende Aufzählung führt einige XML-Charakteristika auf (siehe z. B. ANDERSON et al. 2000, S. 36, GEEB 2003, S. 32, S. 198 ff, PHILLIPS 2000, S. 9 ff, HOLZNER 2001, S. 36 ff, LIAM 2010 a).

- W3C-Standard, firmen- und plattformunabhängig
- Datenaustauschformat (reiner Text)
- Metasprache (eigene Sprachen definierbar, z. B. für IEEE LOM, IMS CP, IMS LD, LMML, siehe auch 3.3.2 XML-Anwendung)
- Extensible (erweiterbar, eigene Elemente definierbar)
- Markup Language (Markups „<>“)
- Selbstbeschreibend („sprechende Elementnamen“)
- Strukturierte Daten (Baumstruktur, Wohlgeformtheit, Gültigkeit)
- Trennung von:
    - Inhalt (XML)
    - Grammatik (Schema- oder DTD-Datei)
    - Layout/Formatierung (XSL Formatting Objects, Cascading Style Sheets)
    - Transformation (XSLT)

Das W3-Consortium fasst unter dem Begriff „*XML-Technologies*“ u. a. XML selbst, XML Namespaces, XML Schema, XSLT, XPath, XQuery zusammen (W3C-XML-Technology 2013, siehe auch Ausdruck XML-Familie z. B. bei GE EB 2003, S. 17 oder ROTHFUSS & RIED 2001, S. 129).

3.3.1 Zeichensätze

Wird eine (XML-)Datei in den Arbeitsspeicher geladen, befinden sich darin für die verschiedenen Zeichen „*Byte-Werte*“ (SELFHTML-KODIERUNGEN 2007). Um daraus auf dem Bildschirm erkennbare Zeichen erstellen zu können, muss die jeweilige Bitfolge mit Hilfe einer Codetabelle eines Zeichensatzes in ein darstellbares Zeichen (z. B. einen Buchstaben) übertragen werden (SELFHTML-KODIERUNGEN 2007). Der ASCII (siehe WIKIPEDIA-ASCII 2013) gilt als der Zeichensatz mit der weitesten Verbreitung (HAROLD 2004, S. 200). Auf dem ASCII aufbauend hat die ISO weitere Zeichensätze wie z. B. den bei westeuropäischen Sprachen häufig eingesetzten „ISO 8859-1 (Latin-1)“ definiert (siehe WIKIPEDIA-ISO-8859 2013, HAROLD 2004, S. 202). Um aber mit XML nicht nur westeuropäische Sprachen, sondern möglichst viele unterschiedliche Sprachen mit demselben Zeichensatz abbilden zu können, wurde der Unicode (UNICODE 2013) entwickelt (DÜRST & FREYTAG 2013, 1. Introduction, HAROLD 2004, S. 200). Auch GEEB (2003, S. 21) weist darauf hin, dass bei XML „*von Beginn an eine Internationalisierung der XML-konformen Daten durch einen gemeinsamen Zeichensatz beabsichtigt*“ war. Er hält

weiter fest, dass - ohne entsprechende Angaben im XML-Dokument- der Unicode in der Kodierung „*UTF-8*“ als Standard eingesetzt wird. GEEB (2003, S. 21) schreibt weiter: „*UTF-8 ist eine Unicode-Variante, in der die ersten 128 Zeichen von ASCII (also das Grundalphabet) erkannt werden…Bei der Verwendung von Sonderzeichen wie ä wird in UTF-8 jedoch der ASCII-Bereich verlassen und eine Darstellung der Zeichen in mehreren Byte angewandt.*“ Er weist darauf hin, dass ASCII-Editoren dies nicht mehr leisten können und deshalb „*für die Nutzung von Zeichen über den lower-ASCII-Bereich hinaus ein unicodefähiger Editor erforderlich*“ ist. Mit Hilfe eines solchen Editors lassen sich dann im Unicode kodierte XML-Dokumente erstellen.

3.3.2 XML-Anwendung

Die Metasprache XML (Extensible Markup Language, erweiterbare Auszeichnungssprache) ermöglicht es, eigene Sprachen zur Beschreibung von Inhalten in Dokumenten z. B. mit Hilfe einer „*Document Type Definition*“ (DTD) (BRAY et al. 2008) oder eines Schema (W3C-SCHEMA 2013) zu definieren. Diese konkreten Sprachen werden als XML-Anwendungen bezeichnet (GEEB 2003, S. 17, HOLZNER 2001, S. 60). UBL, XBRL, XHTML, MathML, SVG, OUNL-EML, LMML, IMS LD, IMS CP und IEEE LOM sind Beispiele dafür (Abbildung 16). Im Folgenden wird der Ausdruck (XML-)Sprache und XML-Anwendung synonym verwendet.

Partl (2002) schreibt: „*Unter XML-Anwendung oder XML-Applikation versteht man die Festlegung (Normierung) von XML-Befehlen für eine Klasse von XML-Dokumenten gleicher Struktur, also für einen bestimmten Zweck. Das Format und die Struktur der XML-Files sowie die Eigenschaften und die Schachtelung der darin vorkommenden Elemente (XML-Befehle, ‚Tags‘, ‚Entities‘) werden für eine XML-Anwendung mit einer DTD oder einem Schema definiert*“. LOBIN (2000, S. 4) hält fest, dass eine XML-Anwendung angibt, „*was für Typen von abstrakten und konkreten Informationseinheiten es gibt, gibt ihnen Namen zur eindeutigen Identifizierung und spezifiziert ggfs. weitere Beschreibungsmerkmale*“. GEEB (2003, S. 17) weist darauf hin, dass „*XML-Anwendungen keine Benutzerschnittstelle haben, also keine Programme sind*“. Einen Einblick in die Vielfalt der vorhandenen XML-Anwendungen bietet z. B. OASIS (2001).

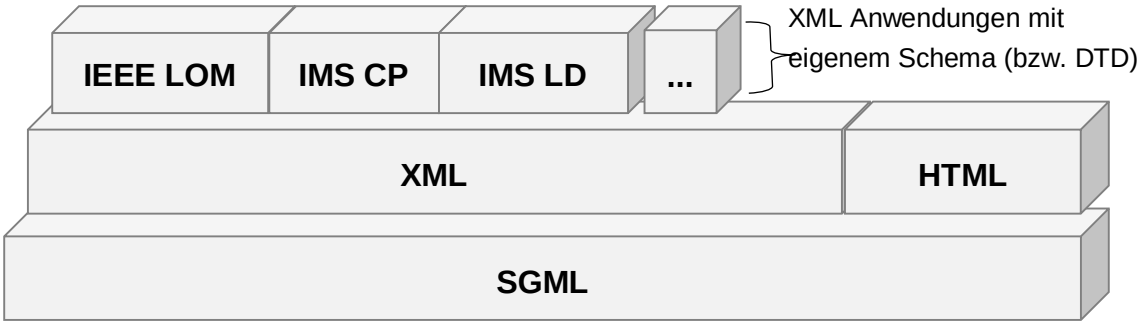


Abbildung 16: XML Anwendungen

Nach BRAY et al. (2008) weist jedes XML-Dokument eine physische und eine logische Struktur auf. Physisch betrachtet setzt sich ein Dokument aus Einheiten („*Entities*“) zusammen, wobei eine Einheit andere Einheiten einbinden kann. Logisch betrachtet besteht das Dokument aus Deklarationen, Elementen, Kommentaren, Zeichenreferenzen und Verarbeitungsanweisungen, die alle innerhalb des Dokumentes eindeutig durch Textauszeichnungen („*markup*“) gekennzeichnet werden (BRAY et al. 2008). Die Daten werden in einem XML-Dokument in einer Baumstruktur mit einem einzelnen Wurzelelement abgelegt (Abbildung 17, BRAY et al. 2008). XML-Dokumente müssen dabei wohlgeformt sein (BRAY et al. 2008). Das bedeutet, dass z. B. die Elemente korrekt verschachtelt sind, Attributwerte in Anführungszeichen gesetzt sind und Elemente ohne Inhalt auf „/>“ enden (LIAM 2010 a). Hält sich die Struktur eines XML-Dokumentes an die in einem XML-Schema-Definitions- oder DTD-Dokument vorgegebenen Syntax, wird es zusätzlich als gültig (valide) bezeichnet (BRAY et al. 2008, LIAM 2010 a).

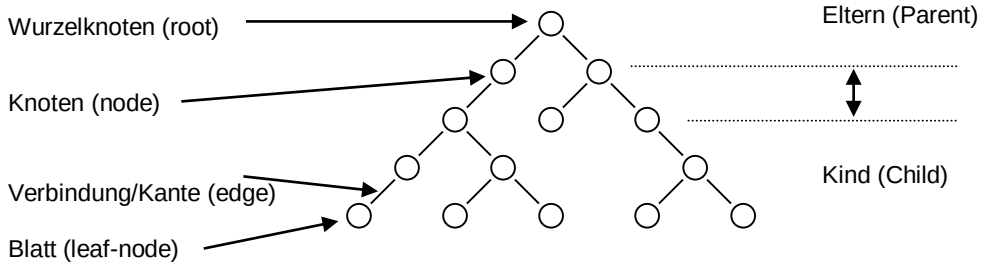


Abbildung 17: Struktur eines XML-Dokumentes. Der Wurzelknoten symbolisiert das gesamte Dokument (abgeändert nach BONGERS 2004, S. 44)

Listing 1 zeigt ein Beispiel eines wohlgeformten und gültigen XML-Dokumentes. Die Gültigkeit ist gegenüber dem „LMML-CS-Schema“ geprüft (XML-Anwendung LMML, Zeile 4). Der Themenbereich „XML Schema“ wird im nächsten Kapitel 3.3.3 DTD / W3C XML Schema Definition beleuchtet. Das in Zeile 2 und 3 aufgeführte Attribut „xmlns“ wird im Kapitel 3.3.4 XML-Namensräume beschrieben. Das Element

„section“ stellt einen Knoten aus Abbildung 17 dar und ist gleichzeitig Child-Element des übergeordneten Elementes „lmml“ („Parent“).

```
1 <?xml version="1.0" encoding="ISO-8859-1"?>
2 <lmml xmlns=http://www.lmml.de/LMML-CS
3 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4 xsi:schemaLocation="http://www.lmml.de/LMML-CS ../lmml/LMML-CS.xsd">
5  <section>
6    <paragraph>
7      <LMMLtext>LMML (Learning Material Markup Language)</LMMLtext>
8    </paragraph>
9 </section>
10 </lmml>
```

Listing 1: Beispiel eines wohlgeformten und gültigen XML-Dokumentes

XML-Dokumente können von derzeit gängigen Browsern wie z. B. „Microsoft Internet Explorer“, „Mozilla Firefox“, „Google Chrome“ mit Hilfe von XSLT- und JavaScript-Anweisungen verarbeitet und angezeigt werden.

3.3.3 DTD / W3C XML Schema Definition

LIAM (2010 b, siehe auch VLIST 2001, S. 4) beschreibt den Ausdruck „*XML Schema*“ wie folgt: „*An XML Schema is a language for expressing constraints about XML documents.*“ Als weit verbreitete Beispiele für Schema-Sprachen führt er DTD („*Document Type Definition*“, definiert in der „*XML Recommendation*“, siehe derzeit BRAY et al. 2008), Relax-NG, Schematron und „*W3C XSD*“ („*XML Schema Definition*“, W3C-SCHEMA 2013, GAO et al. 2012, FALLSIDE & WALMSLEY 2004) auf, wobei er darauf hinweist, dass DTD und „*W3C XSD*“ (im Folgenden XSD genannt) die wichtigsten vom W3C definierten Schema-Sprachen sind.

Ein DTD- oder ein XSD-Dokument beschreiben den strukturellen Aufbau und die logischen Elemente eines XML-Dokumentes (GEEB 2003, S. 50, S. 94). Sie stellen eine formale Grammatik dar, in der die Bezeichnungen der in XML-Instanzdokumenten zulässigen Elemente sowie Attribute und deren Verschachtelung definiert sind (SKULSCHUS & WIEDERSTEIN 2004, S. 23, siehe auch LIAM 2010 b, GEEB 2003, S. 50). Unter XML-Instanzdokument versteht SKULSCHUS & WIEDERSTEIN (2004, S. 23) ein Dokument, dessen Elemente und Struktur den Vorschriften eines bestimmten DTD-/Schema-Dokumentes entsprechen.

Im Gegensatz zu einem DTD-Dokument ist ein XSD-Dokument selbst in XML verfasst und kann damit von beliebiger XML-Software be- und verarbeitet werden (GEEB 2003, S. 50, S. 94, DUMBILL 2001, S. 1). Ein weiterer Vorteil von XSD besteht darin, dass über XSD-Anweisungen eine Überprüfung auf gültige Datentypen und Wertebereiche möglich ist (GEEB 2003, S. 94). GEEB (2003, S. 94, siehe auch WALMSLEY 2002, S. 8) beschreibt weitere Unterschiede zwischen DTD und XSD wie die spezifische Unterstützung von Namensräumen (siehe 3.3.4 XML-Namensräume) und den Objektgedanken (Vererbung). Wegen dieser Nachteile von DTD gegenüber XSD wird XSD entwickelt und im Jahr 2001 vom W3C als „*Recommendation*“ verabschiedet (VLIST 2001, S. 4, BRADLEY 2004, S. 15).

Folgendes Listing 2 zeigt als Beispiel für DTD-Anweisungen einen Quelltext-Ausschnitt aus einem Modul der „*LMML-CS (Computer Science)*“-DTD „LMML11-cmodel-10.mod“. Es spiegelt die Stelle wieder, an der das Element „lmml“ eingeführt wird (Zeile 5).

```
1 <!--================ document element =====================-->
2 <!ENTITY % LMML.content "%Structuremodule.class;
3  %Contentobject.class;">
4 <!ELEMENT lmml (%LMML.content;)>
5 <!ENTITY % LMML.version.attrib "version CDATA #FIXED
6 '%LMML.version;'">
7 <!ATTLIST lmml %LMML.version.attrib;>
8 <!-- end of LMML11-cmodel-10.mod -->
```

Listing 2: Ausschnitt aus einem Modul der LMML-CS DTD (SÜSS 2001, SÜSS 2004, S. 169)

Das Listing 3 zeigt als weiteres Beispiel XSD-Anweisungen aus dem „*LMML-CS (Computer Science)*“-Schema-Dokument mit dem Dateinamen „LMMLStructureModule.xsd“. Auch hier wird die Stelle abgebildet, an der das Element „lmml“ eingeführt wird (Zeile 2).

```
1 <xs:element name="lmml">
2    <xs:complexType>
3       <xs:choice minOccurs="0">
4          <xs:element ref="sgrp_structuremodule"/>
5          <xs:element ref="sgrp_contentmodule"/>
```

```
6       </xs:choice>
7       <xs:attribute name="version" type="xs:string"
8  fixed="-//DE.UNI-PASSAU.IM//XSD LMML-CS 1.1//EN"/>
9     </xs:complexType>
…
10 </xs:element>
```

Listing 3: Ausschnitt aus dem LMML-CS Schema-Dokument mit dem Dateinamen „LMMLStructureModule.xsd“ (SÜSS 2001, SÜSS 2004, S. 158)

Das folgende Kapitel 3.3.4 XML-Namensräume beschreibt, wie Elemente mit identischen Namen aus unterschiedlichen XML-Anwendungen in einem XML-Dokument verwendet werden können.

3.3.4 XML-Namensräume

Das W3C hält in seiner „*Recommendation*“ zu XML-Namensräumen (“*XML namespaces*”) Folgendes fest (BRAY et al. 2009, Abstract): „*XML namespaces provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by URI references.*“

GEEB (2003, S. 88) beschreibt Namensräume als „*virtuelle Bereiche*“, die nach BRAY et al. (2009, 2.1) über eine URI-Referenz identifiziert werden. GEEB (2003, S. 88) weist weiter darauf hin, dass jedes Element und Attribut in einem Namensraum nur einmal vorkommen kann. Mit Hilfe von verschiedenen Namensräumen lassen sich allerdings gleichnamige Elemente und Attribute verschiedener XML-Anwendungen (siehe Kapitel 3.3.2 XML-Anwendung) in einem XML-Dokument mischen (HAROLD 2004, S. 327, S. 345), ohne dass ihre spezifischen Bedeutungen verloren gehen und sie dennoch eindeutig bleiben (GEEB 2003, S. 88 ff).

Namensräume werden in einem Element mit einem Attribut mit dem Namen „xmlns“ deklariert (BRAY et al. 2009, 3). Der „xmlns“-Attributwert besteht aus einer URI. Weist ein „xmlns“-Attribut kein Präfix auf, handelt es sich um einen Standardnamensraum (HAROLD 2004, S. 345, BRAY et al. 2009, 3). Ein Standardnamensraum gilt für das Element, in dem das „xmlns“-Attribut erscheint und für seine Kind-Elemente. Bei Namensräumen mit einem Präfix kann das Präfix in dem Element, in dem das „xmlns“-Attribut erscheint und in seinen Kind-Elementen eingesetzt werden (HAROLD 2004, S. 327 ff). Listing 4 zeigt zwei

Standardnamensräume (definiert im „book“-Element in der dritten Zeile und im „p“-Element in der neunten Zeile) und einen Namensraum mit einem Präfix (definiert im „book“-Element in der vierten Zeile und eingesetzt in der sechsten Zeile).

```
1 <?xml version="1.0"?>
2 <!-- initially, the default namespace is "books" -->
3 <book xmlns='urn:loc.gov:books'
4       xmlns:isbn='urn:ISBN:0-395-36341-6'>
5     <title>Cheaper by the Dozen</title>
6     <isbn:number>1568491379</isbn:number>
7     <notes>
8       <!-- make HTML the default namespace for some commentary -->
9       <p xmlns='http://www.w3.org/1999/xhtml'>
10          This is a <i>funny</i> book!
11      </p>
12    </notes>
13 </book>
```

Listing 4: Standardnamensräume definiert in der dritten und neunten Zeile (Attribut „xmlns“), Namensraum mit Präfix „isbn“ definiert in der vierten Zeile (Attribut „xmlns“) und eingesetzt in der sechsten Zeile (BRAY et al. 2009, 6.6)

3.3.5 XML-Anwendungen des W3C

Im Folgenden werden beispielhaft drei XML-Anwendungen des W3C (MathML, SVG, XHTML) kurz dargestellt, die im E-Learning-Bereich Einsatz finden können.

3.3.5.1 MathML

Das W3C hält in seiner „*Recommendation*“ zu MathML Folgendes fest (CARLISLE et al. 2010, Abstract): „*MathML is an XML application for describing mathematical notation and capturing both its structure and content. The goal of MathML is to enable mathematics to be served, received, and processed on the World Wide Web, just as HTML has enabled this functionality for text.*“

Mit Hilfe der XML-Anwendung MathML können mathematische Ausdrücke (Abbildung 18) in Textform beschrieben werden (Listing 5, zu Abbildung 18 gehörender Ausschnitt aus dem MathML-Quelltext). Die im Beispiel eingesetzte Entität „±“ (Unicode-Zeichen „*PLUS-MINUS SIGN*“ (U000B1), Angabe im

Hexadezimal-Code, siehe RAHTZ & CARLISLE 2013) steht für das Plus-Minus-Zeichen. Die Entität „⁢“ (Unicode-Zeichen „*INVISIBLE TIMES*“ (U02062), Angabe im Hexadezimal-Code, siehe RAHTZ & CARLISLE 2013) steht für den „*unsichtbaren Operator*“ der Multiplikation, der in der traditionellen Mathematik nicht in Erscheinung tritt (CARLISLE et al. 2010, 3.2.5.5). Zu dem Ausdruck Entität hält GEEB (2003, S. 73, siehe auch CARLISLE & ION 2010) fest: „*Entitäten referenzieren (= zeigen) auf eine Zeichenmenge. Die Zeichenmenge ist damit nicht direkt in einem XML-Dokument sichtbar, sondern nur der Verweis durch die Entität. Die Entität ist damit ein Stellvertreter für eine Zeichenmenge*“.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Abbildung 18: Beispiel eines mathematischen Ausdruckes (CARLISLE et al. 2010, 1.4)

```
1 <mrow>
2  <mi>x</mi>
3  <mo>=</mo>
4  <mfrac>
5    <mrow>
6      <mrow>
7        <mo>-</mo>
8        <mi>b</mi>
9      </mrow>
10      <mo>&#xB1;<!--PLUS-MINUS SIGN--></mo>
11      <msqrt>
…
12            <mn>4</mn>
13            <mo>&#x2062;<!--INVISIBLE TIMES--></mo>
14            <mi>a</mi>
…
15      </msqrt>
…
16    </mrow>
17  </mfrac>
18 </mrow>
```

Listing 5: Ausschnitt aus dem MathML-Quelltext (CARLISLE et al. 2010, 1.4) zur Beschreibung des vorhergehenden mathematischen Ausdruckes (Abbildung 18)

3.3.5.2 SVG

Das W3C hält in seiner „*Recommendation*“ zu SVG Folgendes fest (DAHLSTRÖM et al. 2011, 1.1): „*SVG is a language for describing two-dimensional graphics in XML [XML10]. SVG allows for three types of graphic objects: vector graphic shapes (e.g., paths consisting of straight lines and curves), images and text.*“

Die XML-Anwendung SVG erlaubt es zweidimensionale Grafiken (Abbildung 19) zu beschreiben (Listing 6, zu Abbildung 19 gehörender SVG-Quelltext). Mit Hilfe des Attributes „viewbox“ in der Zeile 4 (Listing 6) wird ein neues Koordinatensystem eingefügt, anhand dessen Grafiken in dem SVG-Container positioniert und in ihrer Ausdehnung bestimmt werden können (DAHLSTRÖM et al. 2011, 7.7, EISENBERG 2002, S. 17 ff, ADAM 2002, S. 125 ff). Das erste Rechteck in der Zeile 8 (Listing 6) erzeugt einen dünnen äußeren Rand um das gelbe mit einem eigenen Rand versehene Rechteck (siehe Abbildung 19).

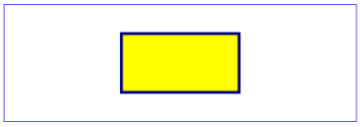


Abbildung 19: Beispiel einer zweidimensionalen Grafik (DAHLSTRÖM et al. 2011, 9.2)

```
1 <?xml version="1.0" standalone="no"?>
2 <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
3   "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
4 <svg width="12cm" height="4cm" viewBox="0 0 1200 400"
5      xmlns="http://www.w3.org/2000/svg" version="1.1">
6   <desc>Example rect01 - rectangle with sharp corners</desc>
7   <!-- Show outline of canvas using 'rect' element -->
8   <rect x="1" y="1" width="1198" height="398"
9         fill="none" stroke="blue" stroke-width="2"/>
10   <rect x="400" y="100" width="400" height="200"
11         fill="yellow" stroke="navy" stroke-width="10"  />
12 </svg>
```

Listing 6: SVG-Quelltext (DAHLSTRÖM et al. 2011, 9.2) zur Beschreibung der vorhergehenden zweidimensionalen Grafik (Abbildung 19)

3.3.5.3 XHTML

Das W3C hält in seiner „*Recommendation*“ zu XHTML Folgendes fest (MCCARRON et al. 2010, 1.1 What is XHTML?): „*XHTML is the reformulation of HTML 4 as an application of XML. XHTML 1.0 [XHTML1] specifies three XML document types that correspond to the three HTML 4 DTDs: Strict, Transitional, and Frameset. XHTML 1.0 is the basis for a family of document types that subset and extend HTML.*”

XHTML 1.0 ist eine XML-Anwendung und stellt eine zu XML konforme Variante von HTML 4.01 dar (GEEB 2003, S. 20). ANDERSON et al. (2000, S. 167) halten fest: „*XHTML gehorcht allen grammatikalischen Regeln von XML (sauber geschachtelte Elemente, Attributwerte in Anführungszeichen etc.) und unterstützt weiterhin alle bisherigen Elemente von HTML.*” Dabei kann es von XML-Software analysiert und verarbeitet werden. Derzeit wird allerdings nicht mehr XHTML (BIRBECK et al. 2010, Status of this Document), sondern HTML 5 vom W3C weiterentwickelt. HTML 5 liegt momentan in einer „*Candidate Recommendation*” des W3C vor (BERJON et al. 2013, siehe auch 3.4 HTML).

Listing 7 zeigt den Quelltext eines XHTML-Dokumentes mit verschiedenen Namensräumen (siehe auch Kapitel 3.3.4 XML-Namensräume) in Zeile 2, 3, 4 und den Quelltext zur Einbindung einer XSD-Datei (siehe auch Kapitel 3.3.3 DTD / W3C XML Schema Definition) in Zeile 6.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <html xmlns="http://www.w3.org/1999/xhtml"
3       xmlns:simpleml="http://www.example.com/xmlns/simpleml1"
4       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5       xsi:schemaLocation="http://www.w3.org/1999/xhtml
6       simpleml-1_0.xsd">
7     <head>
8         <title>An example using defaults</title>
9     </head>
10     <body>
11         <p>This is content in the XHTML namespace</p>
12         <simpleml:element>
13             This is content in the SimpleML namespace.
```

```
14            <simpleml:otherelement/>
15        </simpleml:element>
16        <p>
17            <img src="missing" alt="Missing image"
18            simpleml:myattr="value"/>
19        </p>
20    </body>
21 </html>
```

Listing 7: Beispiel eines XHTML-Dokumentes (MCCARRON et al. 2010, B.4.1. Creating a simple Document Type)

Neben W3C-XML-Anwendungen werden weitere spezielle XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten eingesetzt.

3.3.6 XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten

KORNELSEN et al. (2004, S. 31) beschreiben für die Erstellung von E-Learning-Inhalten drei verschiedene Gruppen von XML-Sprachen (XML-Anwendungen). Als erstes führen sie XML-Sprachen für die „*Textauszeichnung auf Basis von XML*“ an. Sie nennen beispielhaft DocBook, IMS CP, LMML und <ML>³. Die zweite Gruppe wird für die „*Beschreibung von Medienobjekten auf XML-Basis*“ herangezogen. Erwähnt wird unter anderem SVG. Die dritte Gruppe bezeichnen sie als „*XML-Sprache zur Beschreibung mathematischer Formeln*“ und führen MathML als Beispiel auf. Zusätzliche weisen sie auf die Integration anderer Formate (z. B. JPG, WAV, AVI, Flash) in E-Learning-Inhalte hin.

Auch JUNGMANN (2012, S. 102), GRIES et al. (2009, S. 215), EIBL (2006, S. 18 f) und BUNSCHKOWSKI et al. (2004, S. 140) erwähnen als XML-Sprachen zur Erstellung von E-Learning-Inhalten LMML und <ML>³. GRIES et al. (2009, S. 215) sprechen zusätzlich eLML, OUNL-EML und DocBook an. SCHNEIDER (2011 b) führt als „*Languages that model contents*“ nur eLML und LMML auf (Hyperlink von CEN-WS-LT-EML (2013) auf diese Information, zu eLML siehe auch CHIRU et al. 2009).

KLEBL (2004, S. 6 ff) betrachtet allgemeiner den Einsatz von XML in mediengestützten Bildungsprozesse. Er schlägt für eine Unterscheidung von „*XML-Schemata*“ (XML-Anwendungen) zur Kategorisierung von Lehr-/Lernprozessen vor, sich nicht an Anwendungsbereichen (Meta-Daten, didaktische Aspekte, Lehrinhalte),

sondern an ihren Funktionen zu orientieren. Tabelle 6 zeigt einen Überblick dieser Funktionen (siehe auch SCHNEIDER 2011 b).

Tabelle 6: Überblick über Funktionen des Einsatzes von XML in mediengestützten Bildungsprozessen (KLEBL 2004, S. 6)

Funktion	**Beschreibung**	**Beispiele für den Einsatz**
Auswahl (bzw. Unterstützung der Auswahl)	Katalog für Erschließung (Recherche und Auswahl) von Lehr-/Lerneinheiten	Meta-Daten, z. B. IEEE-LOM: Bildungsportale, Lernmodulbibliotheken
Anordnung	Zusammenstellung und Organisation von Ressourcen in einem Container	Export und Import von Inhalten (*Content Packaging*) z. B. SCORM ™
Auszeichnung	Erstellung von Lernmaterialien und Ausgabe auf unterschiedlichen Medien oder Endgeräten	z. B. *Learning Material Markup Language (LMML)* im Projekt PaKMaS (*Passauer Knowledge Management System*)
Prozessbeschreibung	Steuerung des Ablaufs eines Lehr-/Lernprozesse durch Beschreibung von Strukturelementen	*Educational Modelling Language (EML) [OUNL-EML, Anm. d. Verf.], IMS Learning Design* u. a.

Ähnlich dieser Vorstellung definieren RAWLINGS et al. (2002, S. 8) eine „*Educational Modelling Language*“ prozessorientiert: „*An EML is a semantic information model and binding, describing the content and process within a 'unit of learning' from a pedagogical perspective in order to support reuse and interoperability.*“

Im Folgenden wird sich dieser Unterscheidung angeschlossen (Tabelle 6) und es werden verschiedene XML-Anwendungen vorgestellt, deren Schwerpunkt auf der Auszeichnung von E-Learning-Inhalten liegt.

3.3.6.1 LMML

LMML entstand im Rahmen des Forschungsprojekts PaKMaS („*Passauer Knowledge Management System*“) an der Universität Passau (SÜSS & FREITAG 2000 a, S. 110 ff, SÜSS & FREITAG 2000 b) und wurde am IFIS - Institut für Informationssysteme und Softwaretechnik weiterentwickelt.

LMML gilt als „*ein Pionier für den Einsatz von XML zur konzeptuellen und modularen Strukturierung von eLearning Content*“ (FREITAG 2002 a, S. 349, siehe auch SÜSS 2004, S. 58, siehe auch RAWLINGS et al. 2002, S. 10). Die Auszeichnungssprache dient zur Strukturierung von E-Learning-Inhalten und soll die Modularisierung und Wiederverwendung von Lernmaterialen sicherstellen (SÜSS 2004, S. 57 ff, FREITAG 2002 a, S. 349, FREITAG et al. 2002, S. 354 ff, SÜSS & FREITAG 2001 b, S. 1-12, siehe auch JUNGMANN 2012, S. 102). FREITAG (2002 a, S. 349) hält weiter fest, dass LMML „*zur semantischen Beschreibung von eLearning Content*“ eingesetzt wird.

3.3.6.1.1 Passauer Teachware-Modell

Laut SÜSS (2004, S. 44) modelliert das PTM „*für das Anwendungsgebiet eLearning unabhängig von einem konkreten Fachgebiet die sachlogische Fragmentierung von eLearning-Inhalten, Metadaten für eLearning-Informationseinheiten, sowie Beziehungen*“ (siehe Kapitel 3.2.3 Modellierung von E-Learning-Inhalten).

Bei der sachlogischen Fragmentierung lassen sich Lerninhalte in Module fassen, die wiederum Module in sich aufnehmen können (Abbildung 20). Strukturmodule („*StructureModule*“) in LMML ermöglichen - ähnlich der Kapitelangaben in einem Buch - die Strukturierung von Inhaltsmodulen („*ContentModule*“). Inhaltsmodule sind z. B. „*motivation*“, „*example*“ oder „*definition*“. Die Inhaltsmodule selbst können wiederum in Listen oder Tabellen strukturiert sein („*StructureObject*“) und beinhalten letztlich die Medienobjekte („*MediaObject*“) wie Text, Ton, Bild oder Animation (Abbildung 20, FREITAG et al. 2002, S. 355). Die Abbildung 20 stellt die Inhaltsfragmente in Form von UML-Klassen dar (SUESS 2004, S. 27 f, weitere Erläuterungen siehe Kapitel 3.2.3 Modellierung von E-Learning-Inhalten).

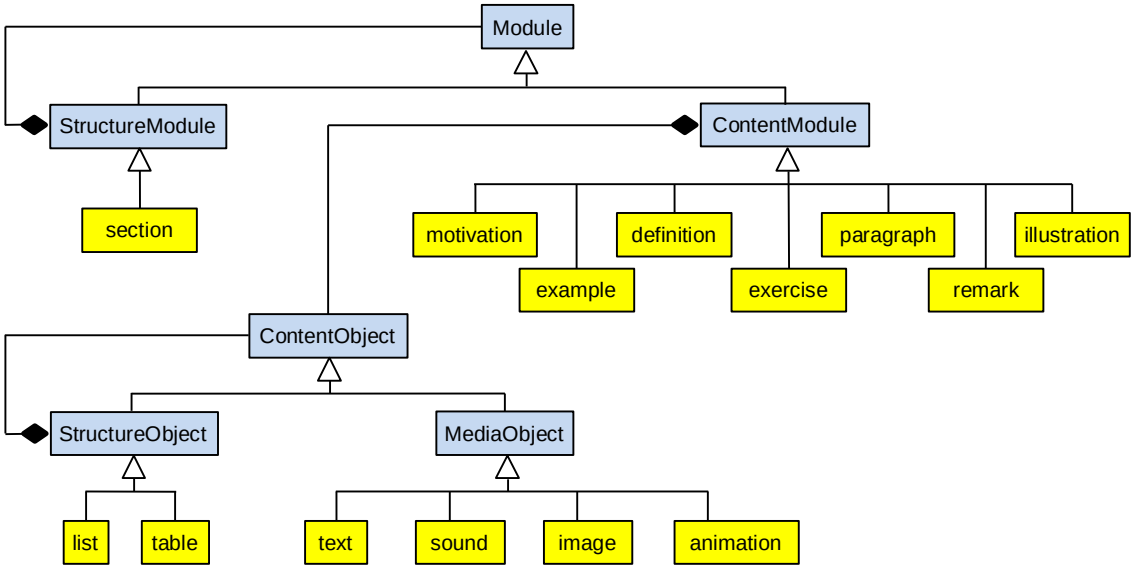


Abbildung 20: Sachlogische Fragmentierung von E-Learning-Inhalten im PTM (SÜSS 2004, S. 45) (identisch mit Abbildung 15)

E-Learning-Inhalte können damit in Module beliebiger Granularität unterteilt werden. Jeder einzelne Baustein, von Strukturmodulen (z. B. ganze Kurse oder Kursabschnitte) über einzelne Inhaltsmodule bis hinunter zu den Medienobjekten, lässt sich mit Metadaten versehen und in anderen Lerneinheiten wiederverwenden (SÜSS 2004, S. 26).

Bezüglich der erwähnten Metadaten hält SÜSS (2004, S. 48) fest: „*Das PTM spezialisiert für die sachlogische Informationsmodellierung von eLearning-Inhalten die Metadaten des ADM*“ (siehe auf Abbildung 14 rechts unten „Metadata“). Er weist darauf hin, dass die eingesetzten Metadaten (beruhend auf Dublin Core (DUBLIN CORE 2013), ARIADNE (ARIADNE 2013), IMS (IMS 2013)) „*die Grundlage für den damals noch nicht existierenden LOM-Standard*“ (HODGINS & DUVAL 2002) darstellen und auf diesem direkt abgebildet werden können.

3.3.6.1.2 LMML-Framework

Das LMML-Framework wurde auf der Basis des „*Passauer Teachware-Modells*“ (bzw. ADM) entwickelt. SÜSS (2004, S. 62) beschreibt es wie folgt: „*Das LMML-Framework ist ein Framework für die XML-basierte Spezifikation von eLearning-Inhalten, das das Passauer Teachware-Modell in den Kern einer XML-Sprachfamilie umsetzt*“ (zur Erläuterung siehe auch Kapitel 3.2.3 Modellierung von E-Learning-Inhalten v. a. mit Abbildung 13, Abbildung 14, Abbildung 15). Als Framework bezeichnet er „*eine mehr oder weniger abstrakte, mehr oder weniger vollständige Architektur einer Klasse von Software*“ (SÜSS 2004, S. 62). Den „*LMML-Kern*“ (Kern

einer XML-Sprachfamilie) definiert er als „*XML-Umsetzung (XML-Binding) des Passauer Teachware-Modells PTM als invarianter Anteil des LMML-Framworks*“ (SÜSS 2004, S. 63). Er ist in eine DTD mit mehreren Moduldateien bzw. in mehrere XSD-Dateien aufgeteilt (siehe z. B. Abbildung 21, Abbildung 22) und „*setzt die Inhaltsfragmente des PTM … als XML-Elemente eines LMML-Dokuments um*“ (SÜSS 2004, S. 64, siehe auch Listing 8).

SÜSS & FREITAG (2001 b, S. 2 f, siehe auch SÜSS 2000, S. 101 ff) beschreiben LMML als ein XML-basiertes Modell für Lerninhalte und bezeichnen es als eine Familie von Auszeichnungssprachen. Eine Instanz des LMML-Framework stellt die LMML-CS (Computer Science) mit ihrer eigenen DTD (bzw. XSD) dar (Abbildung 21, Abbildung 22). Sie dient zur Beschreibung von Lehrinhalten im Fachgebiet Informatik und besteht in der DTD-Version aus 13 Dateien, in der XSD-Version aus 8 Dateien. Durch die Schaffung weiterer spezifischer DTD´s (bzw. XSDs), die in das Framework eingebunden werden, können neue LMML-Instanzen für z. B. verschiedene Fachgebiete geschaffen werden. WEITL et al. (2002) haben darüber hinaus eine LMML-Instanz mit der Zielrichtung einer didaktischen Strukturierung von E-Learning-Inhalten geschaffen (Bezeichnung „*LMML-d*“). Sie halten fest: „*LMML gibt die strukturelle Gliederung Lernmaterial nach inhaltlichen Gesichtspunkten vor. LMML-d fügt dem Contentmodell der LMML ein Modell für die didaktische Funktion hinzu, die Inhalte in einem Lernmodul innehaben (z.B. Lerneinheiten, Präsentationseinheiten,...).*“

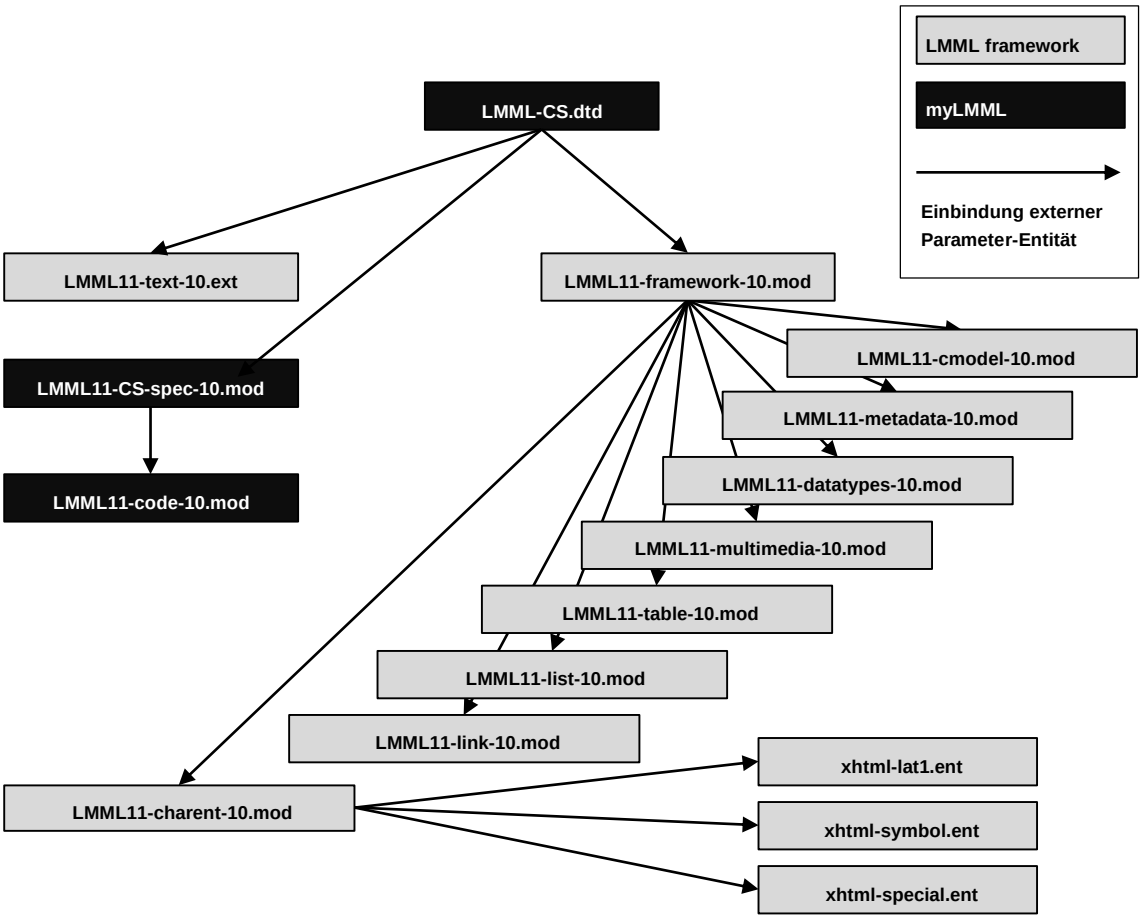


Abbildung 21: DTD-Module der „LMML-CS 1.2“ (SÜSS 2004, S. 77), insgesamt 13 Dateien

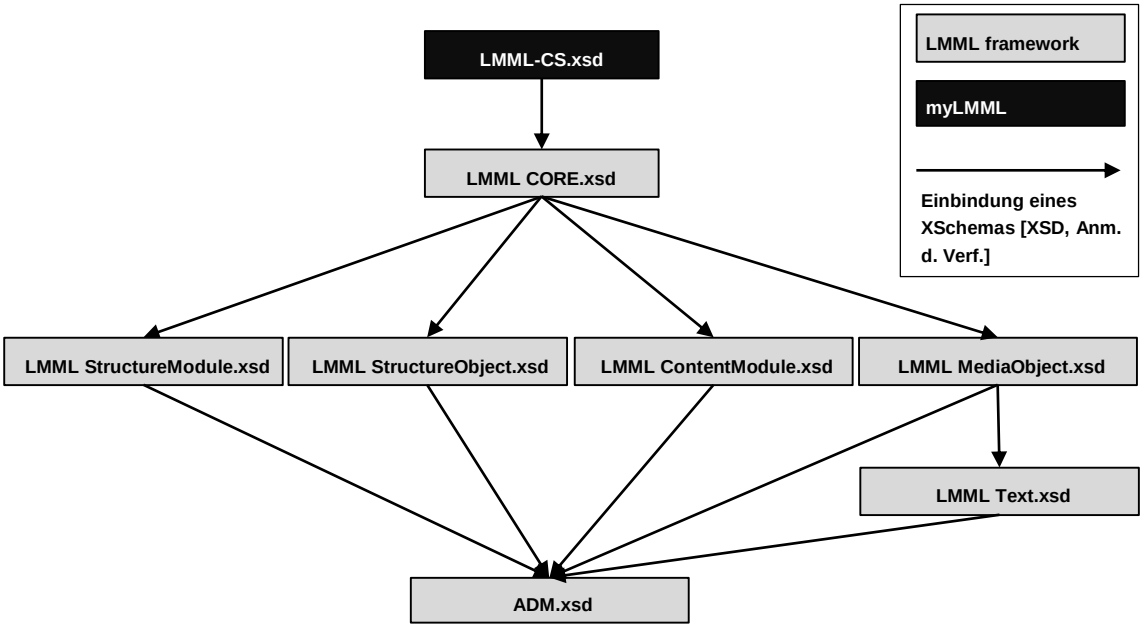


Abbildung 22: W3C-XSD-Module der „LMML-CS 1.2“ (SÜSS 2004, S. 78), insgesamt 8 Dateien

Das folgende Kapitel zeigt ein Beispiel eines LMML-Dokumentes.

3.3.6.1.3 LMML-Beispieldokument

Nach SÜSS (2000, S. 64) ist ein LMML-Dokument „*ein bezüglich einer LMML-Sprachdefinition (DTD, XSchema [entspricht W3C XSD, Anm. d. Verf.] etc.) gültiges XML-Dokument*“. Der strukturelle Aufbau des LMML-Dokumentes - mit seinen als XML-Elementen umgesetzten Inhaltsfragmenten - wird dabei durch seine fachspezifisch erweiterbare DTD (bzw. XSD) festgelegt (siehe auch SÜSS 2004, S. 64).

Folgender XML-Quelltext (Listing 8) zeigt einen Ausschnitt aus einem LMML-Dokument. Einige Inhaltsfragmente sind mit ihren Namen angegeben („StructureModule“, „ContentModule“, „MediaObject“). In Zeile 2 beginnt ein „StructureModule“ „section“ und in Zeile 7 endet es (dunkelgrauer Rahmen). Als Inhalt weist es ein „ContentModule“ „motivation“ auf (Zeile 3 bis 6, hellgrauer Bereich), das wiederum zwei „MediaObject“-Elemente „text“ und „image“ aufweist. Ab Zeile 8 beginnt ein neues „StructureModule“ „section“, das in Zeile 15 endet und zwei „ContentModule“ „definition“ und „example“ mit jeweils einem „MediaObject“ „text“ beinhaltet. Umspannt wird der beschriebene Inhalt von einem weiteren „section“-Element (Zeile 1 bis 16).

```
1   <section title="XML Tutorial">
2     <section title="Introduction">          ← Beginn StructureModule
3       <motivation>                          ← Beginn ContentModule
4         <text>XML has many uses … </text>   ← MediaObject
5         <image src="xml.gif" />             ← MediaObject
6       </motivation>                         ← Ende ContentModule
7     </section>                              ← Ende StructureModule
8     <section title="Basics">
9       <definition>
10        <text>A XML document is … </text>
11      </definition>
12      <example>
13        <text>The following XML … </text>
14      </example>
15    </section>
16  </section>
```

Listing 8: Ausschnitt eines LMML-Dokumentes als Quelltext-Beispiel für das XML-Binding des PTM im LMML-Kern (SÜSS 2004, S. 64)

3.3.6.1.4 „*Passauer Knowledge Management System*“ (PaKMaS)

SÜSS & FREITAG (2001 a) bezeichnen PaKMaS als *„… ein plattformunabhängiges, modulares Softwaresystem, das die Vorzüge von datenbank-basiertem XML-Contentmanagement und adaptiven Hypermedia-Systemen vereint*“. SÜSS (2004, S. 128 f) hält zusätzlich fest, dass es sich bei PaKMaS um „*ein webbasiertes Client-Server-System*“ handelt und „*XML-basierte eLearning-Inhalte … von einem PaKMaS-Server in einer relationalen Datenbank … verwaltet*“ werden. Zur Umsetzung dieser Funktionalitäten beinhaltet die PaKMaS-Distribution laut IFIS (2002) unter anderem Software aus dem „*Apache XML FOP*“-Projekt, dem „*Apache Jakarta*“-Projekt, sowie das „*Sun Java 2 Runtime Environment Plugin*“. Die Lerninhalte lassen sich mit einem herkömmlichen XML-Editor erstellen. XML-basierte Publikationsdokumente können dabei die Beschreibung des zu präsentierenden Inhalts, das Layout, die Navigation, die Personalisierung und die möglichen Anfragen der Lernenden festlegen (IFIS 2002).

PaKMaS kann als Autoren- und Lernumgebung eingesetzt werden und ist offen für die unterschiedlichsten Formate des vorliegenden Lernmaterials. So können z. B. HTML-Seiten, PDF-Dokumente, „Microsoft Word“-Dokumente oder „Microsoft Powerpoint“-Präsentationen eingebunden werden (SÜSS & FREITAG 2000 a, S. 110 ff). Als Ausgabeformate für die Publikation sind gedruckte Skripte im PDF- oder HTML-Format oder ein personalisierbarer Kurs in PaKMaS möglich (Abbildung 23). Dabei werden vom „*PublicationManager*“ auch „*XSL-Stylesheets*“ eingesetzt (SÜSS 2004, S. 133).

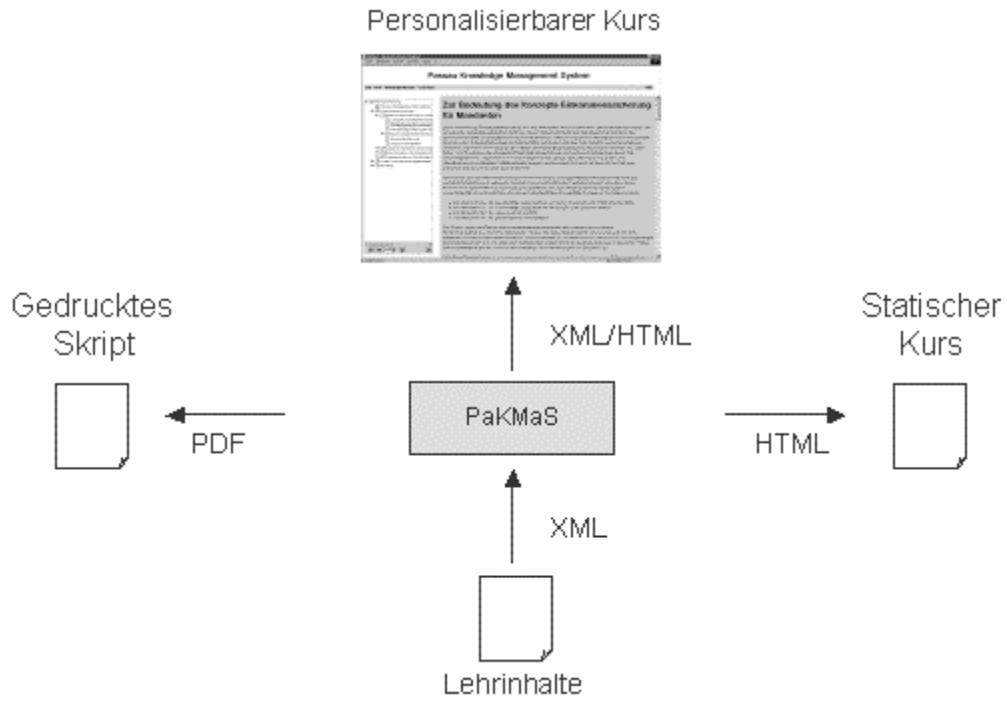


Abbildung 23: PaKMas (IFIS 2002, siehe auch SÜSS 2004, S. 133)

Um die Personalisierung und mehrdimensionale Anfragen geeignet unterstützen zu können, wurde laut SÜSS & FREITAG (IFIS 2002, 2001 a, S. 5 ff, siehe auch SÜSS 2004, S. 19, 24) für PaKMaS eine relationale Speicherung von XML-Dokumenten implementiert. Damit ist es möglich XML-basierte LMML-Lerninhalte in einer relationalen Datenbank abzulegen (SÜSS & FREITAG 2001 a, S. 5). Lernende können damit als Ergebnis in einer auf sie zugeschnittenen Lernumgebung die Auswahl von z. B. Inhalten, Beispielen, Übungen vornehmen.

PaKMaS kann auch in ein SCORM-fähiges LMS integriert werden (zu SCORM siehe Kapitel 3.8 SCORM, zu LMS siehe Kapitel 3.9 LMS). SÜSS (2004, S. 135) erläutert dazu, dass der „*PaKMaS-Client zusammen mit dem angebotenen eLearning-Inhalt … der SCORM-Definition eines SCORM-Inhaltsobjekts*“ genügt. Er führt weiter aus: „*PaKMaS-Client und –Server stehen dabei in Kommunikationsbeziehungen … mit einem SCORM-Lernmanagementsystem, d. h. nach SCORM-Definition mit einem Computersystem, das die Möglichkeit bieten kann, Lerner zu registrieren, Lernaktivitäten zeitlich zu regeln, den Lernprozess zu kontrollieren und zu führen ...* .“

3.3.6.2 <ML>[3]

„*Multidimensional LearningObjects and Modular Lectures Markup Language*“ wurde im Rahmen des vom BMBF geförderten Projektes „Wissenswerkstatt Rechensysteme“ unter der Leitung der Universität Rostock entwickelt (ML3 2005,

JUNGMANN 2012, S. 100). Dabei wurden „*bewährte Konzepte bestehender Sprachen*“ integriert (VOIGT & TAVANGARIAN 2003, S. 5). Neben der Auszeichnung von Inhaltstypen (z. B. Text („text“), Graphik („image“), Sound („sound“)) (Abbildung 24) ist eine zeitliche Skalierung der Inhalte, die Unterstützung von Einsatzszenarien und Zielgruppen sowie eine didaktische Strukturierung der E-Learning-Inhalte möglich (VOIGT & TAVANGARIAN 2003, S. 5).

Die E-Learning-Inhalte werden in Module gegliedert (Abbildung 24 linker Bereich), die jeweils eine Vorlesung von maximal acht Stunden und vier Stunden Übung abbilden. Außerdem wird - im Anhalt an das didaktische Modell von LMML (WEITL et al. 2002) - eine didaktische Strukturierung ermöglicht. Sie soll vor allem die E-Learning-Inhalte für den Lernenden, aber auch Lehrenden, in sinnvolle Einheiten gliedern (VOIGT & TAVANGARIAN 2003, S. 5). Ein Modul wird dafür in Lektionen („lesson“) zerlegt (Abbildung 24 rechter Bereich), die jeweils ca. eine Stunde Zeitaufwand für die Abarbeitung verursachen sollen.

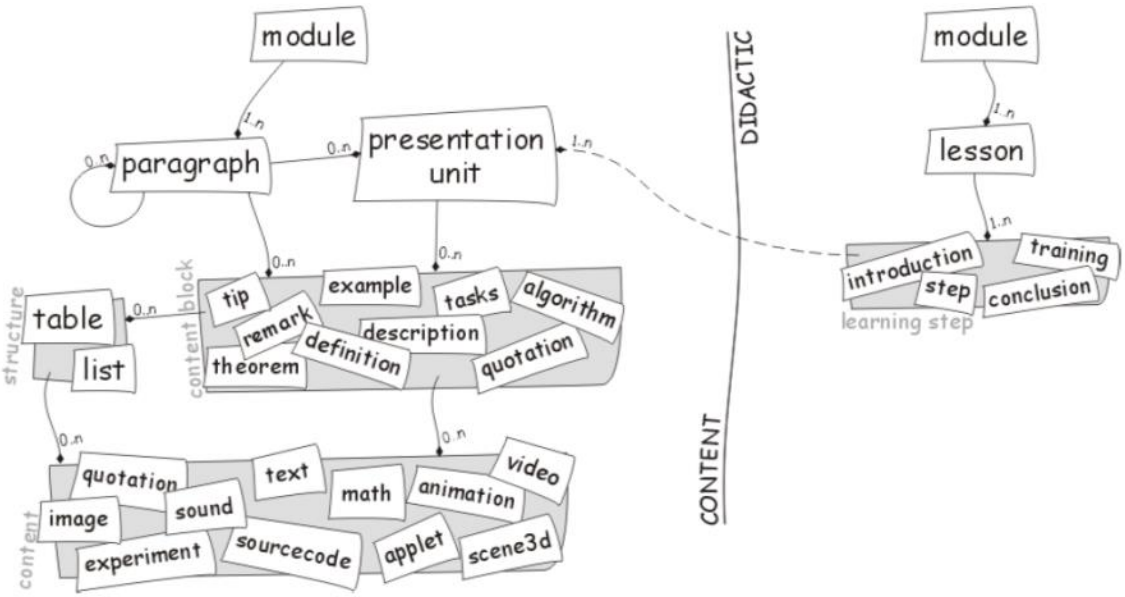


Abbildung 24: Überblick über die Strukturierung eines Moduls in UML-Notation (sachlogische Strukturierung im linken Bereich, didaktische im rechten Bereich) (VOIGT & TAVANGARIAN 2003, S. 6)

Die Module können abschließend zu Kursen zusammengesetzt werden. Um die Flexibilität dabei zu erhöhen, „*ist jedes einzelne Modul bezüglich dreier Dimensionen skalierbar*“ (Intensität bzw. zeitlicher Umfang („intensity“), Zielgruppe („target“), Einsatzszenario bzw. Ausgabeformat („device“)) (VOIGT & TAVANGARIAN 2003, S. 7, Abbildung 25).

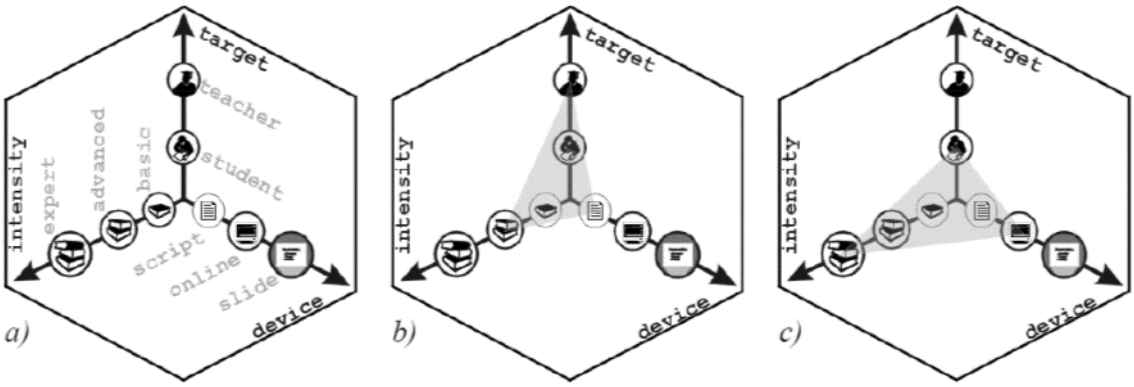


Abbildung 25: a) Moduldimensionen mit ihren jeweiligen Wertebereichen
b) Modulinstanz mit Lehrer als Zielgruppe, druckbar, mittlere Intensität
c) Kombination aus Online-Variante und Expertenmodus für Studenten. (VOIGT & TAVANGARIAN 2003, S. 7)

Über ein XML-Binding lassen sich die Modul-Inhalte in XML abbilden. Dazu stellen zwei XSD-Dokumente (sachlogisches Vokabular in der Datei „ml3c.xsd“, didaktisches Vokabular in der Datei „ml3d.xsd“) „die für den Inhaltsersteller wichtigsten Auszeichnungsmöglichkeiten bereit“ (KORNELSEN & VOIGT 2005, S. 4, VOIGT & TAVANGARIAN 2003, S. 7). Als Ergebnis der Inhaltserstellung besteht ein vollständiges Modul meist aus mehreren XML-Dokumenten und Mediendateien. Über einen Transformationsprozess wird das Modul in ein konkretes Ausgabeformat überführt (VOIGT & TAVANGARIAN 2003, S. 9) (Abbildung 26).

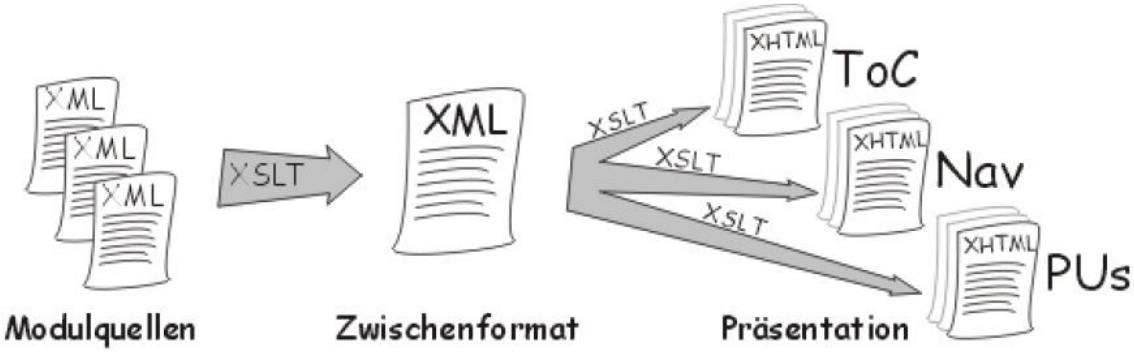


Abbildung 26: Modultransformation (online-Szenario) per XSLT (Erläuterungen zu XSLT siehe Kapitel 3.3.8 XSL) über ein XML-Zwischenformat (ein einzelnes XML-Dokument) hin zu Ausgabe in einem Browser (VOIGT & TAVANGARIAN 2003, S. 10)

Die Transformation wird über das „*WWR Build Tool*“ sichergestellt. Dieses Werkzeug besteht vor allem aus in Java programmierten Komponenten und Bibliotheken (KORNELSEN & VOIGT 2005, S. 2). Die Autoren erläutern auf Seite 9 weiter: „*Das Ergebnis der Modultransformation für die Online-Variante ist grundsätzlich von jedem Browser darstellbar*“.

Zusätzlich stellt <ML>³ „*LOM-konforme Beschreibungsmittel für seine Lehr- und Lernmaterialien*“ zur Verfügung (VOIGT & TAVANGARIAN, 2003, S. 8) und bietet die Möglichkeit SCORM-Pakete zu erzeugen (KORNELSEN & VOIGT 2005, S. 2, S. 4). Durch die Verwendung von Metadaten, die Trennung von Inhalt (Beschreibung auf semantischer Ebene) und Layout können Interoperabilität und Wiederverwendbarkeit sichergestellt werden (VOIGT & TAVANGARIAN, 2003, S. 8).

3.3.6.3 eLML

Nach ELML (2013, glossary) ist eLML ein XML-Framework, das im Rahmen des schweizerischen E-Learning-Projektes GITTA im Jahr 2001 entwickelt wurde. Nach Ende des Projektes 2004 wurde die XML-Struktur als Open-Source-Projekt unter dem Namen eLML veröffentlicht (weitere Unterstützung durch „EduTech“, Organisation zur Förderung des „Swiss Virtual Campus“). Die Auszeichnungssprache beruht auf drei zentralen XSD-Dateien (*„three XML ‚core‘ schemas*“) und verschiedenen XSLT-Dateien, die unterschiedliche Ausgabeformate erzeugen können (XHTML, PDF, LaTeX, SCORM-fähiges Format, „eBooks“-Format „ePub“, „Open Document Format“, DocBook-Format) (FISLER & BLEISCH 2012, S. 3, S. 69 ff). Auch die Ausgabe auf Smartphones wird unterstützt (FISLER & BLEISCH 2012, S. 96 f).

Die „*eLesson Markup Language*“ bildet E-Learning-Inhalte strukturiert in XML auf der Grundlage eines pädagogischen Konzeptes ab (FISLER & BLEISCH 2012, S. 7 f). Dieses pädagogische Konzept (ECLASS-Modell, Anfangsbuchstaben der Bezeichnung siehe Abbildung 27) diente als Ausgangspunkt, wurde in XML abgebildet und eLML benannt.

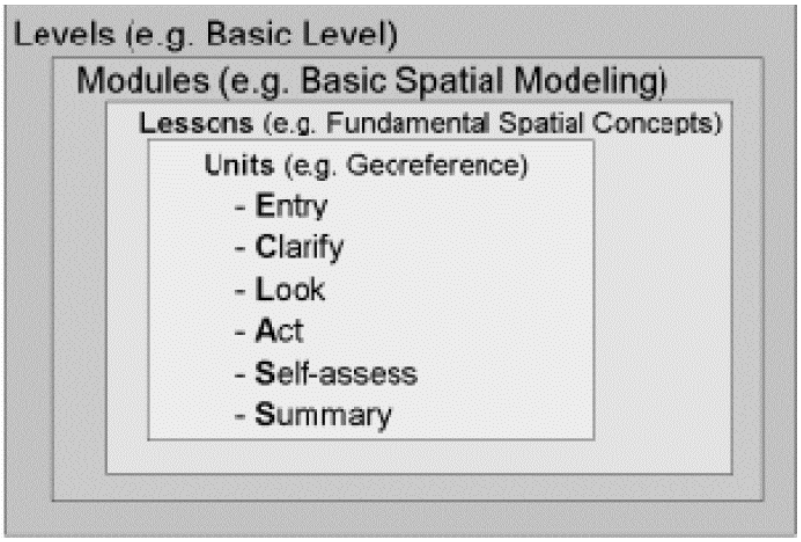


Abbildung 27: ECLASS-Modell, pädagogisches Konzept von eLML (FISLER & BLEISCH 2012, S. 8)

Abbildung 28 zeigt die drei Ebenen der eLML-XML-Struktur. BLEISCH & FISLER (2005, 2.1 eLML Lektionen) halten fest, dass eLML es ermöglicht „*e-Learning Lektionen vollständig zu beschreiben*“. Sie führen weiter aus: „*Damit ist gemeint, dass eine Lektion (lesson) nicht nur die eigentlichen Lerninhalte enthält, sondern auch zusätzliche Informationen wie Glossareinträge (glossary), eine Bibliographie (bibliography) und Metadaten (metadata) über die Lektion*“. Die Metadaten in eLML beruhen auf einer vereinfachten Version der IMS Metadaten-Spezifikation (BLEISCH & FISLER 2005, 2.5 Bibliographie und Metadaten).

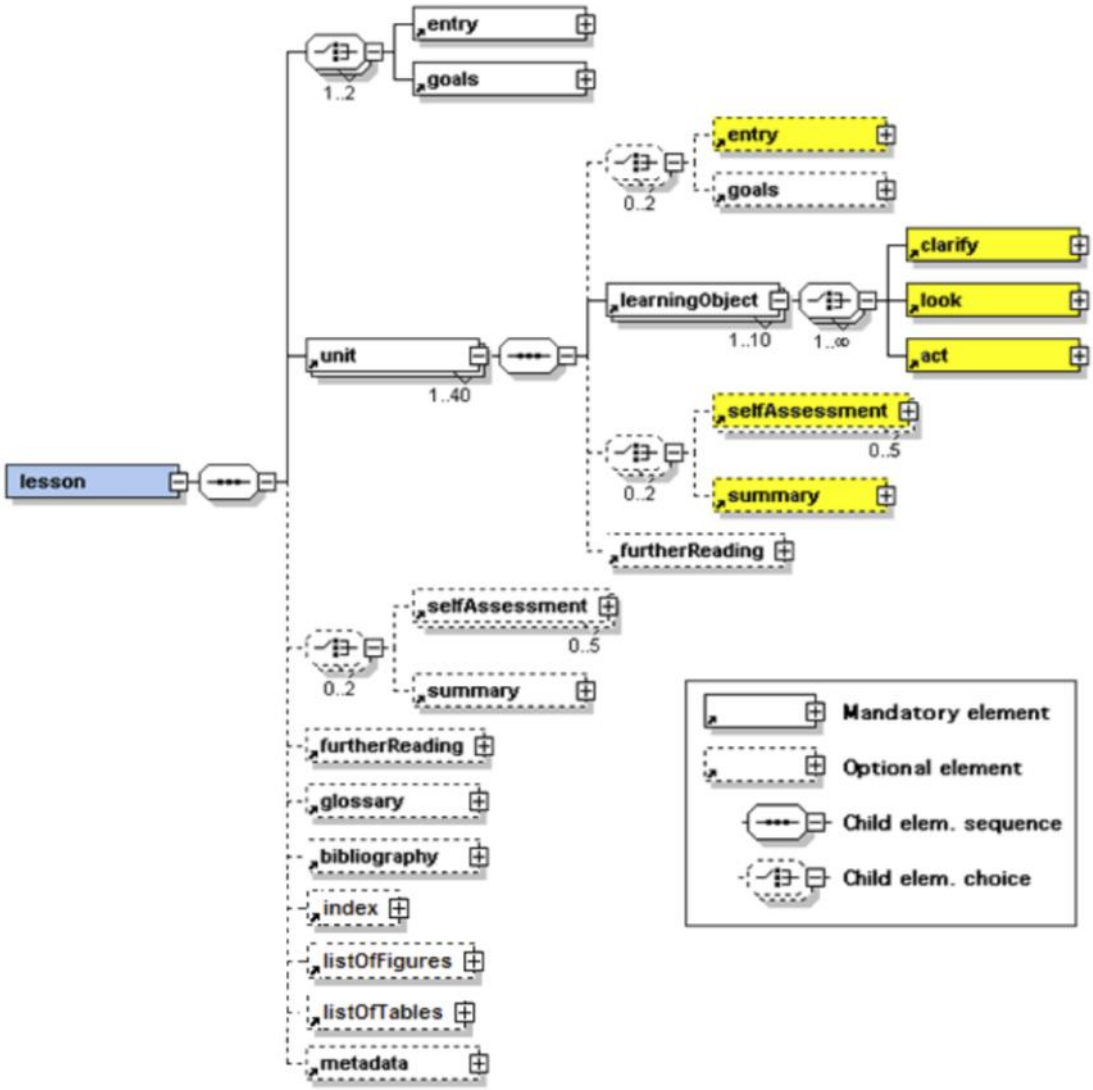


Abbildung 28: Die ersten drei Ebenen der eLML-XML-Struktur, blau eingefärbt das Element „lesson“ als Root-Element, gelb eingefärbt die ECLASS-Elemente (FISLER & BLEISCH 2012, S. 9)

E-Learning-Inhalte können nach dem Download des Frameworks (beinhaltet Ordnerstruktur mit allen Dateien) mit XML-Editoren wie z. B. „oXygen“ oder „Altova XMLSpy“ erstellt werden (FISLER & BLEISCH 2012, S. 29). „Eclipse“ wird für CVS

(„Concurrent Versions System“, Versionsverwaltung von Dateien) und das Projektmanagement vorgeschlagen (Installation von „oXygen“ und „Altova XMLSpy“ als Plugin). Über Einstellungen in dem Konfigurationsdokument mit dem Dateinamen „config.xml“ werden die eLML-Lektionen in das jeweilige Ausgabeformat umgewandelt (FISLER & BLEISCH 2012, S. 70). Über die Ausgabe in ein SCORM-fähiges Format können die eLML-Lektionen in ein LMS wie z. B. „Moodle“ (MOODLE 2014) oder „OLAT“ (OLAT 2014) importiert und dort verwendet werden (FISLER & BLEISCH 2012, S. 91).

3.3.6.4 Weitere XML-Sprachen zur Auszeichnung von E-Learning-Inhalten

Abschließend seien einige weitere XML-Sprachen zur Auszeichnung von E-Learning-Inhalten erwähnt. Auf sie wird aufgrund der Zielrichtung oder des Zeitpunktes des Entstehens der vorliegenden Arbeit oder des geringeren Verbreitungsgrades nicht weiter eingegangen (siehe auch 3.3.6 XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten).

- DocBook (DOCBOOK 2013)
- DITA (DAY et al. 2001)
- GiLES („*Giessen Learning and Education Schema*“, Projekt MiLCA („*Medienintensive Lehrinhalte in der Computerlinguisitk-Ausbildung*“), STÜHRENBERG 2004, S. 403 ff, STÜHRENBERG 2013)
- PALO (ARTACHO 2004)
- TeachML (Projekt „CHAMELEON“ / Projekt „Targeteam“) (MEISSNER et al. 2001, SÜSS 2004, S. 56)

Bezüglich DocBook (siehe auch SCHRAITLE, T. 2004, S. VIII: DocBook als System zur Erstellung technischer Dokumentationen) und DITA (DAY et al. 2001: „*XML-based, end-to-end architecture for authoring, producing, and delivering technical information*“) weisen GRIES et al. (2009, S. 215) darauf hin, dass diese „*Dokument-Beschreibungssprachen ohne eLearning-Fokus*“ zwar für die Abbildung von E-Learning-Inhalten grundsätzlich einsatzfähig wären, sie allerdings erst angepasst werden müssten.

3.3.7 XML-Anwendungen von IMS GLC

Die Organisation „IMS Global Learning Consortium“ stellt verschiedene XML-basierte Sprachen zur Verfügung, die im Bereich E-Learning Einsatz finden.

IMS GLC ist ein nicht kommerzieller Zusammenschluss von über 180 Behörden, Unternehmen und kommerziell ausgerichteten Organisationen (IMS 2013, About us). Es werden einerseits Anbieter von Lernsystemen und Lerninhalten sowie andererseits Bildungsinstitutionen vereint. Die Organisation verfolgt folgende Mission (IMS 2013, About us): „*The mission of the IMS Global Learning Consortium is to advance technology that can affordably scale and improve educational participation and attainment.*“ Es soll die weltweite Entwicklung offener technischer Spezifikationen und deren Einbindung in Programme und technische Dienste im Bildungssektor erreicht werden. An anderer Stelle wird als Motto der Organisation festgehalten: „*Advancing Learning Impact by Enabling the Open Foundation for Seamless, Agile and Information-Rich Educational Technology Integration*“ (IMS 2013, About us).

Das IMS Global hat über 20 „*learning technology standards*“ von weltweiter Bedeutung veröffentlicht (IMS 2013, Background and History). Es werden Folgende genannt: „*Widely-used IMS GLC standards include meta-data, content packaging, common cartridge, enterprise services, question & test, sequencing, competencies, access for all, ePortfolio, learner information, tools interoperability, resource list, sharable state persistence, vocabulary definition, and learning design.*“

Im Folgenden werden aufgrund ihrer Bedeutung im E-Learning-Bereich vier IMS-Spezifikationen kurz beleuchtet (IMS CP, IMS MD, IMS LD und IMS QTI).

3.3.7.1 IMS CP

IMS CP ist eine Entwicklung des IMS GLC und wurde im Jahr 2009 auch in drei Teilen unter „*ISO/IEC 12785, Information technology -- Learning, education, and training -- Content packaging*“ (ISO-12785 2009)) als internationale Norm veröffentlicht („*Part 1: Information model*“, „*Part 2: XML binding*“, „*Part 3: Best practice and implementation guide*“).

SMYTHE et al. (2004, 1.1 Content Packaging Overview) erläutern die Spezifikation „IMS Content Packaging“ wie folgt: „*The IMS Content Packaging specification describes data structures, XML binding and accompanying best practices that are used to provide interoperability for Internet based content with content creation tools, learning management systems (LMS), and run time environments. The scope of the IMS Content Packaging specification is focused on defining interoperability between systems that wish to import, export, aggregate, and disaggregate Packages of content.*“ (siehe auch IMS-CP 2013).

IMS CP beschreibt damit Strukturen, die für den Datenaustausch von E-Learning-Inhalten zwischen verschiedenen Systemen eingesetzt werden können. Abbildung 29 (siehe auch Abbildung 33 und Listing 10) zeigt die Komponenten des IMS CP „*Information Model*“ (SMYTHE & JACKL 2004 a, 2. IMS Content Packaging Conceptual Model).

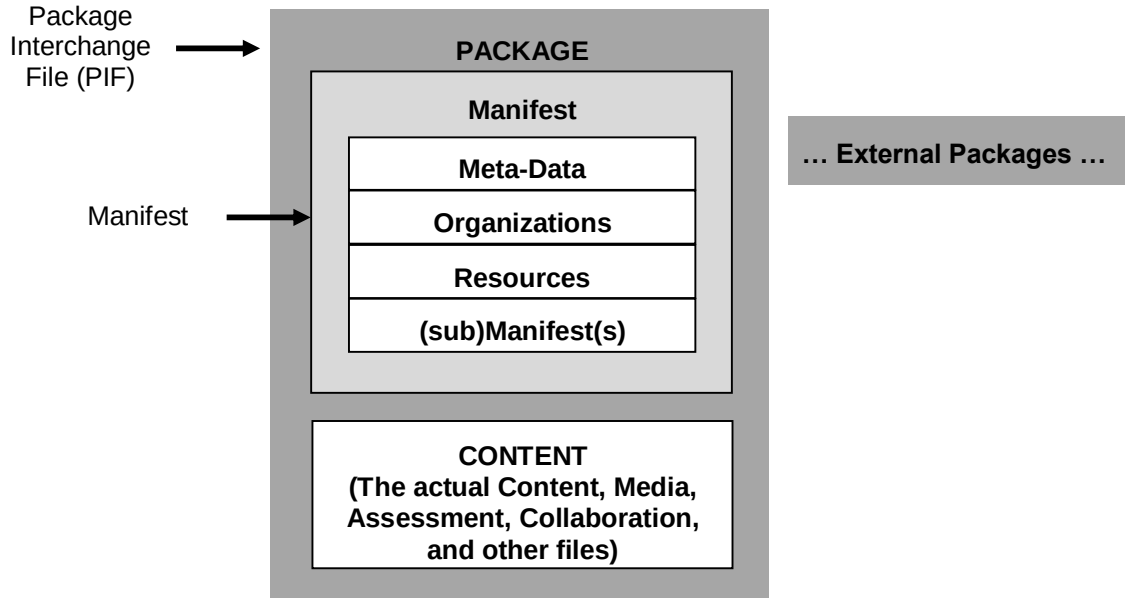


Abbildung 29: Komponenten des IMS CP „*Information Model*“ (SMYTHE & JACKL 2004 a, 2. IMS Content Packaging Conceptual Model)

Nach SMYTHE & JACKL (2004 a, 2.1 Key Elements) besteht das IMS-Paket (Package, Abbildung 29) aus zwei Hauptelementen. Das erste ist eine XML-Datei (Manifest, auch IMS-Manifest-Datei, „imsmanifest.xml“, SMYTHE & JACKL 2004 a, 2.2 Standard Name for the Manifest File), die die Organisation, Metadaten (Meta-Data) und Ressourcen eines Paketes beschreibt. Das zweite sind die für dieses Paket notwendigen Dateien (CONTENT), die als Ressourcen in dem XML-Dokument beschrieben werden. Das Paket stellt eine wieder verwendbare Inhaltseinheit dar und lässt sich z. B. für den Transport (siehe auch SMYTHE & JACKL 2004 c, 3. Relationship to Other Specifications) in eine einzelne Datei (Package Interchange File, z. B. als „.zip“-, „.jar“-, „.cab“-Datei) zusammenfassen (SMYTHE & JACKL 2004 a, 2.1 Key Elements). Das Paket kann einen Teil eines Kurses, einen gesamten Kurs oder mehrere Kurse umfassen.

Listing 9 zeigt den Ausschnitt eines XML-Quelltextes eines IMS-Manifest-Dokumentes (IMS-Manifest-Datei „imsmanifest.xml“). Neben dem Root-Element „manifest“ (Zeile 1), der Einbindung von Metadaten (Zeile 2), wird der Aufbau des

Paketes mit einem „organization“-Element (Zeile 7) inklusive eines „item“-Elementes (Zeile 9) und dem dazugehörigen „ressource“-Element (Zeile 15) sichtbar (siehe auch SMYTHE & JACKL 2004 b).

```
1 <manifest xmlns = "http://www.imsglobal.org/xsd/imscp_v1p1"
…
2    <metadata>
…
3       <imsmd:lom>
…
4       </imsmd:lom >
5    </metadata>
6    <organizations default="TOC1">
7       <organization identifier="TOC1">
8          <title>default</title>
9          <item identifier="ITEM1" identifierref="RESOURCE1">
10             <title>Lesson 1</title>
…
11         </item>
…
12       </organization>
13    </organizations>
14    <resources>
15       <resource identifier="RESOURCE1" type="webcontent"
16        href="lesson1.htm" xml:base="lesson1/">
17          <file href="lesson1.htm"/>
18         <file href="picture1.gif"/>
19       </resource>
…
20    </resources>
21 </manifest>
```

Listing 9: **Ausschnitt eines XML-Quelltextes eines IMS-Manifest-Dokumentes (IMS-Manifest-Datei „imsmanifest.xml“) (Ausschnitt von SMYTHE & JACKL 2004 c, 4.8.3 xml:base)**

3.3.7.2 IMS MD

IMS GLC erklärt (IMS-MD 2013): „*IMS Learning Resource Meta-data Information Model 1.2.1 Final Specification is superseded by IEEE Std 1484.12.1 - 2002, IEEE Standard for Learning Object Metadata (LOM)*“. Zusätzlich wird darauf hingewiesen, dass das „*Information Model 1.2.2 Public Draft*“ und der „*Best Practices and Information Guide 1.2.1*“ durch denselben „IEEE-Standard“ abgelöst wurden. Bezüglich der „*XML Binding Specification*“ wird festgehalten, dass sie durch „*IEEE Std 1484.12.3-2005, IEEE Standard for Learning Technology -- Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata*“ ersetzt wurde.

Die Spezifikation diente zur Abbildung von Metadaten für E-Learning-Inhalte. Da an ihre Stelle „*IEEE Standard for Learning Object Metadata (LOM)*“ tritt, wird hier nicht weiter darauf eingegangen (siehe Kapitel 3.7 IEEE LOM, siehe auch BARKER et al. 2006, SMYTHE & TOWLE 2006).

3.3.7.3 IMS LD

Im Jahr 2003 wurde mit der „*IMS Learning Design Specification*“ Version 1.0 (IMS-LD 2003) eine XML-basierte Möglichkeit (mit eigener Schema-Datei) veröffentlicht Lehrinhalte darzustellen. GLAHN (2002, S. 1 f) hält fest: "*Neben der IEEE Learning Object Metadata Spezifikation (IEEE LOM) (HODGINS et al. 2001) und der Materialbeschreibung des IMS Content Packaging fokussiert IMS LD die didaktischen Aspekte innerhalb eines Lernobjekts. Obwohl IMS LD eigentlich kein Standard, sondern lediglich eine Spezifikation ist, muss diese im Zusammenhang mit dem LOM beachtet werden, da das IMS dem Learning Technology Standardization Committee (LTSC) des IEEE zuarbeitet. IMS LD kann daher als eine Vorstufe eines zukünftigen Standards gesehen werden (Liber 2002).*" SCHAFFERT & KALZ (2009, S. 17) halten fest, dass LOM, SCORM und IMS LD zu den „*existierenden E-Learning-Standards*“ gehören (siehe auch BRUGGER 2002, S. 1).

Die Spezifikation IMS LD ist aus der „*Educational Modelling Language*“ der „Open University of the Netherlands“ hervorgegangen (KOPER 2000, KOPER 2002). Um keine Verwechslungen mit dem allgemeinen Ausdruck EML (Definition EML siehe RAWLINGS et al. 2002, S. 8) zu verursachen, wird analog zu RAWLINGS et al. (2002, S. 9) nicht der Ausdruck EML (ursprünglicher Name der Sprache) sondern OUNL-EML gebraucht.

Das besondere an OUNL-EML bzw. IMS LD ist, dass nicht bloß Lerninhalte, sondern auch Rollen, Beziehungen, Interaktionen und Aktivitäten von Lernenden und Lehrenden integriert werden. Dabei können verschiedene Lernmodelle berücksichtigt (z.B. problemorientiertes lernen, Selbststudium, Frontalunterricht) und ein gesamter Lehr-/Lernprozess abgebildet werden (siehe auch KLEBL 2004, S. 3 ff).

Die „*Learning Design Specification*“ entspricht gemäß dem „*IMS Learning Design Information Model*“ folgenden Anforderungen: (KOPER et al. 2003 a, 2.1 Objective of Learning Design Specification):

- Vollständigkeit (Beschreibung eines gesamten Lehr-Lernprozess in einer Lerneinheit („unit of learning“) inklusive Lernende und Lehrende mit eingesetzten Ressourcen und Diensten, auch Lernprozesse von Einzelnen und Gruppen)
- didaktische Flexibilität (Möglichkeit verschiedene didaktische Varianten abzubilden)
- Personalisierung (Dossier)
- Formalisierung (für automatisierte Verarbeitung)
- Reproduzierbarkeit (wiederholtes Ablaufen mit z. B. anderen Personen)
- Interoperabilität (zwischen verschiedenen Lernentwürfen)
- Kompatibilität (IMS CP, IMS QTI, IMS MD/IEEE LOM, IMS Simple Sequencing)
- Wiederverwendbarkeit (in anderen Lerneinheiten)

IMS LD kann nach der Definition von RAWLINGS et al. (2002, S. 8) (siehe Kapitel 3.3.6 XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten) allgemein als eine EML bezeichnet werden (siehe auch MARTINEZ-ORTIZ et al. 2007, S. 28).

3.3.7.3.1 Vorläufer OUNL-EML

OUNL-EML ist wie IMS LD eine XML-Anwendung zur Beschreibung eines gesamten Lehr-/Lernprozesses und wurde am „Educational Technology Expertise Center der Open University of Netherlands“ (OUNL) entwickelt. Sie diente als Ausgangsbasis für die neuere IMS-Spezifikation IMS LD (RAWLINGS et al. 2002, S. 9, S. 24).

Laut KOPER (2000, S. 30) existierte zum damaligen Zeitpunkt kein vergleichbares Regelwerk (Metamodell), das es erlaubte Lerneinheiten in einer integrierten Art und Weise darzustellen.

3.3.7.3.1.1 Lerneinheit

Der Begründer von OUNL-EML Prof. Koper (2001, S. 4) definiert OUNL-EML wie folgt: „*We called the notation of units of study an 'Educational Modelling Language'*". Eine „*unit of study*" (Lerneinheit) ist dabei „*… the smallest unit providing learning envents for learners, satisfying one or more interrelated learning objectives.*" (KOPER 2001, S. 3). Beispiele sind ein Studiengang, Kurs, Workshop, Praktikum oder Unterrichtseinheit.

In Abbildung 30 wird ein Ausschnitt aus der Struktur einer Lerneinheit mit ihren Lernobjekten gezeigt.

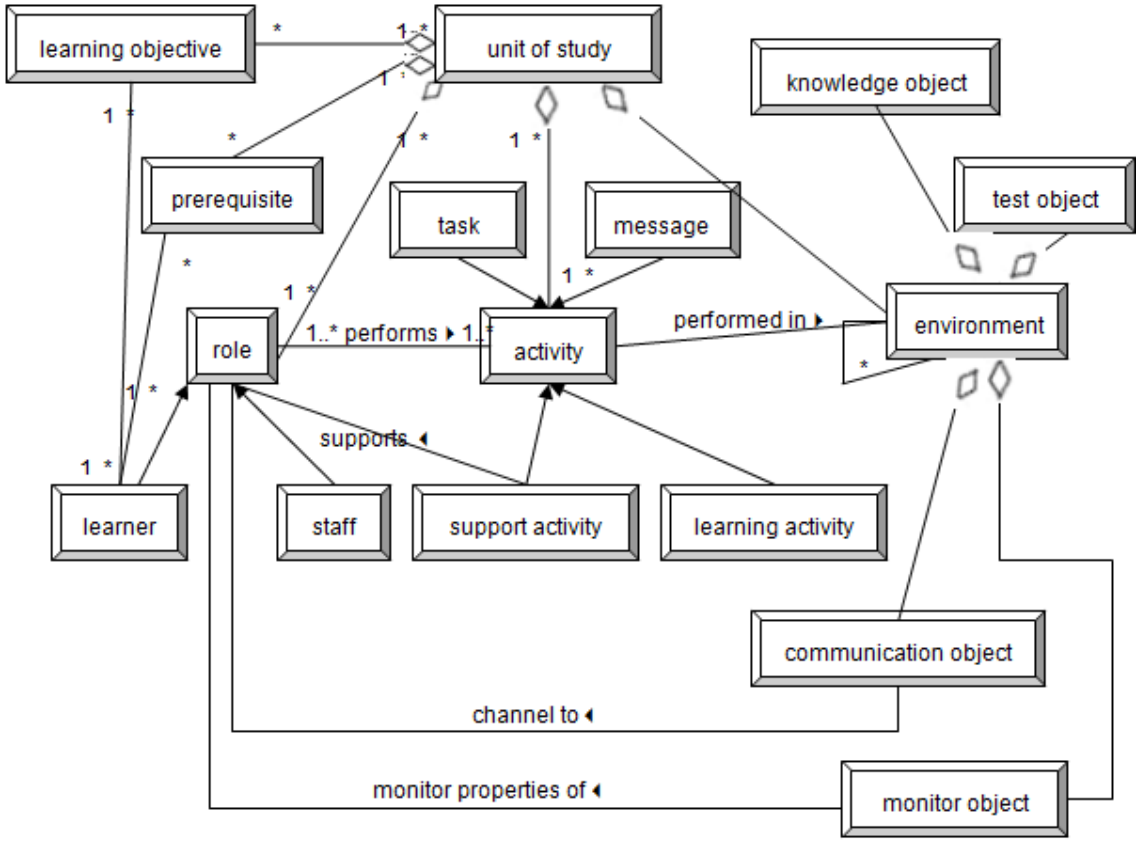


Abbildung 30: Auszug aus dem „*unit of study model*" (KOPER 2001 S. 11)

3.3.7.3.1.2 OUNL-EML-Beispiel

OUNL-EML ist in XML implementiert. Die DTD der OUNL dient dabei als Grundstruktur für die Erstellung von Lerneinheiten (KOPER 2001, S. 17). Abbildung 31 zeigt einen Auszug aus der Darstellung der Basisstruktur des Rahmenwerks für Lernobjekte („*Knowledge-object*", siehe Begriff auch bei WILEY 2000, S. 6) implementiert in der DTD von OUNL-EML.

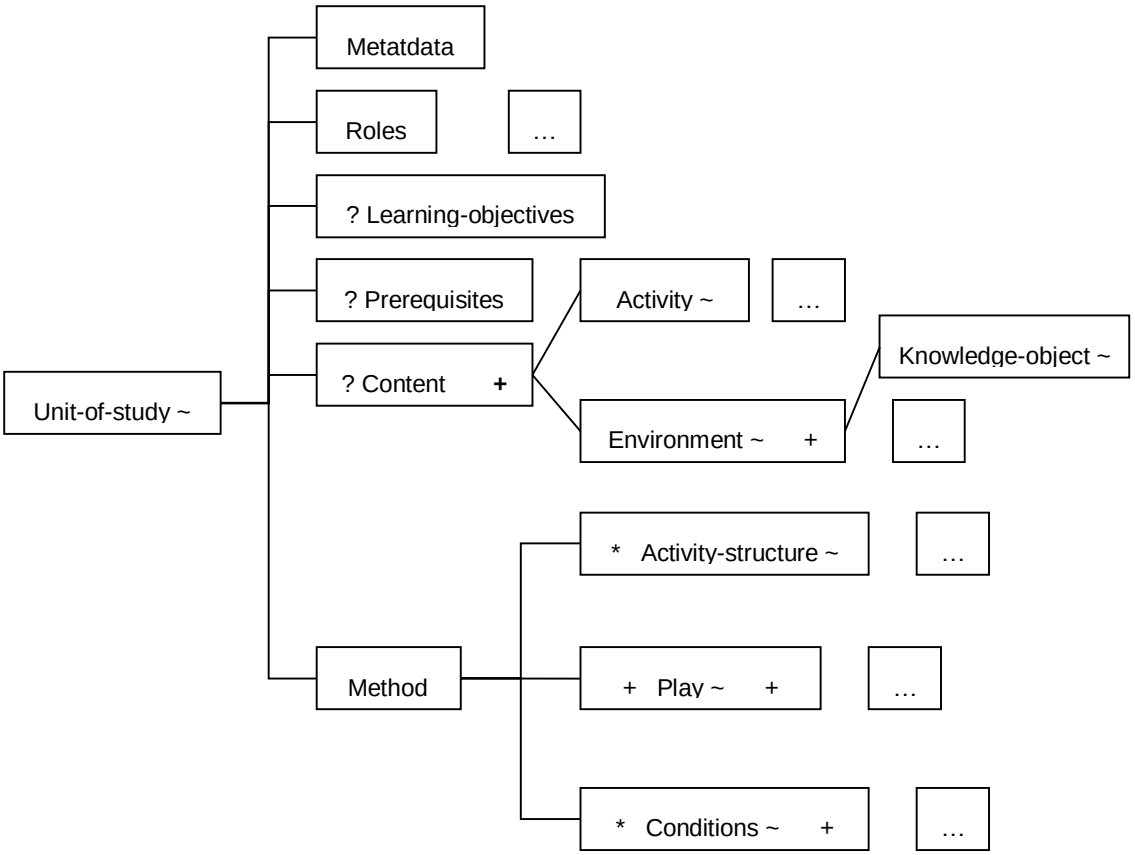
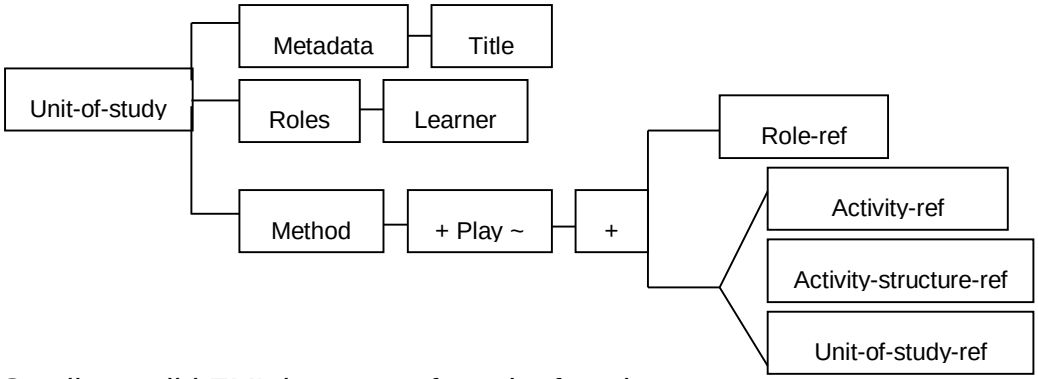


Abbildung 31: Einbindung von Lernobjekten („*Knowledge-object*“) in die DTD von EML (nach KOPER 2001, S. 18)

Abbildung 32 zeigt das kleinste gültige EML-Modell einer „*unit of study*“ und einen möglichen XML-Quelltext (EML-Dokument).

Smallest valid EML model of a unit of study



Smallest valid EML instance of a unit of study

```
<Unit-of-study>
  <Metadata><Title>Course on X</Title></Metadata>
  <Roles><Learner Id=”learner”/></Roles>
   <Method>
     <Play>
       <Role-ref Id-ref=”learner”/>
       <Activity-ref Worldwide-unique-id-ref=”default-student”/>
     <Play>
  </Method>
</Unit-of-study>
```

Abbildung 32: Kleinste OUNL-EML-Modell einer „*unit of study*“ und ein mögliches Beispieldokument in XML (nach KOPER 2001, S. 22)

3.3.7.3.2 IMS LD-Lerneinheit

IMS LD-Lerneinheiten werden als „*Units of Learning*“ bezeichnet (KOPER et al. 2003 a, 2.2.3 Unit of Learning = IMS Content Package + IMS Learning Design, vgl. „*units of study*“ bei OUNL-EML, siehe auch OLIVIER & TATTERSALL 2005, s. 25). Als Inhaltsmodell zur Verpackung von Lerneinheiten wird eine andere IMS-Spezifikation IMS CP vorgeschlagen (IMS-CP 2013). Sie erlaubt es die strukturelle Anordnung einzelner Ressourcen in einem Container zu beschreiben. Die Trennung des didaktischen Überbaus (IMS LD) von der Strukturierung der gesamten Lerneinheit („resource“-Elemente in IMS CP) lässt eine separate Betrachtung und Bearbeitung dieser beiden Einheiten zu. Abbildung 33 zeigt die Einbindung einer „*Unit of Learning*“ in die IMS CP-Struktur. Der didaktische Überbau („*Organizations: IMS LD*“, Implementierung in IMS LD) kann von den physikalischen Dateien (Implementierung in IMS CP) getrennt werden. So ist es auch vorstellbar mit einem vorgefertigten didaktischen Template/Szenario (IMS LD) unterschiedliche Lerninhalte mit ihren

physikalischen Dateien (Angabe über IMS CP) abzubilden (zu didaktischen Szenarien und IMS LD siehe auch CZAPUTA, C. 2008).

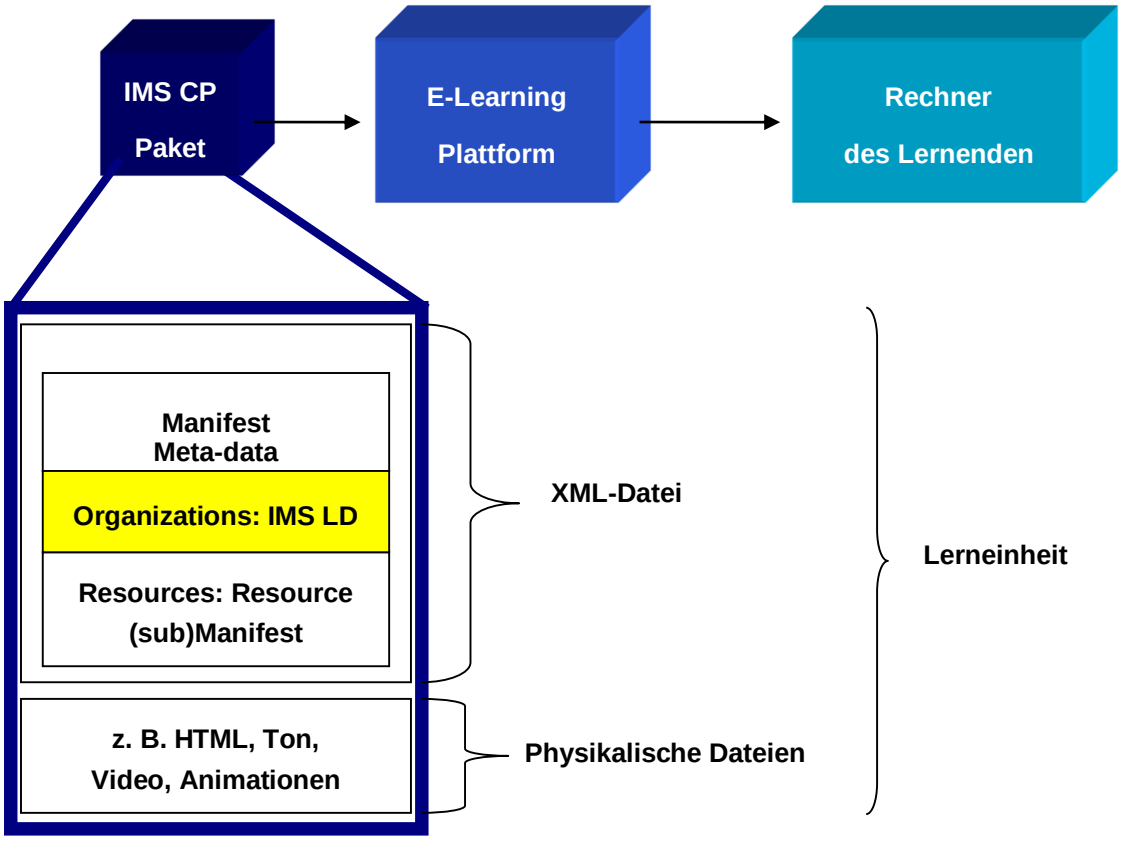


Abbildung 33: IMS CP und IMS LD (erweitert nach KOPER et al. 2003 a, 2.2.3 Unit of Learning = IMS Content Package + IMS Learning Design)

Das Informationsmodell von IMS LD erscheint in drei aufeinander aufbauenden Stufen (Level A, B, C). Sie spiegeln die steigende Komplexität der Anforderung an eine Implementierung wieder. Level C beinhaltet Level A und B. Level B beinhaltet Level A. Alle drei Level sind in verschiedenen XML-Schema-Dateien fixiert (KOPER et al. 2003 a, 1.2 Three Levels of Implementation and Compliance).

3.3.7.3.3 Konzeptionelles Modell Level A

In Level A (Abbildung 34) werden die wesentlichen Bereiche des didaktischen Szenarios in IMS LD eingeführt. Die Elemente „role“ und „activity“ weisen darauf hin, dass nicht bloß Lerninhalte, sondern auch Rollen und Aktivitäten von Lernenden und Lehrenden integriert werden (KOPER et al. 2003 a, 3.1 Level A Information Model). Damit kann ein geplanter Ablauf von Interaktionen im Lehr-/Lernprozess (Work-Flow) formal beschrieben werden.

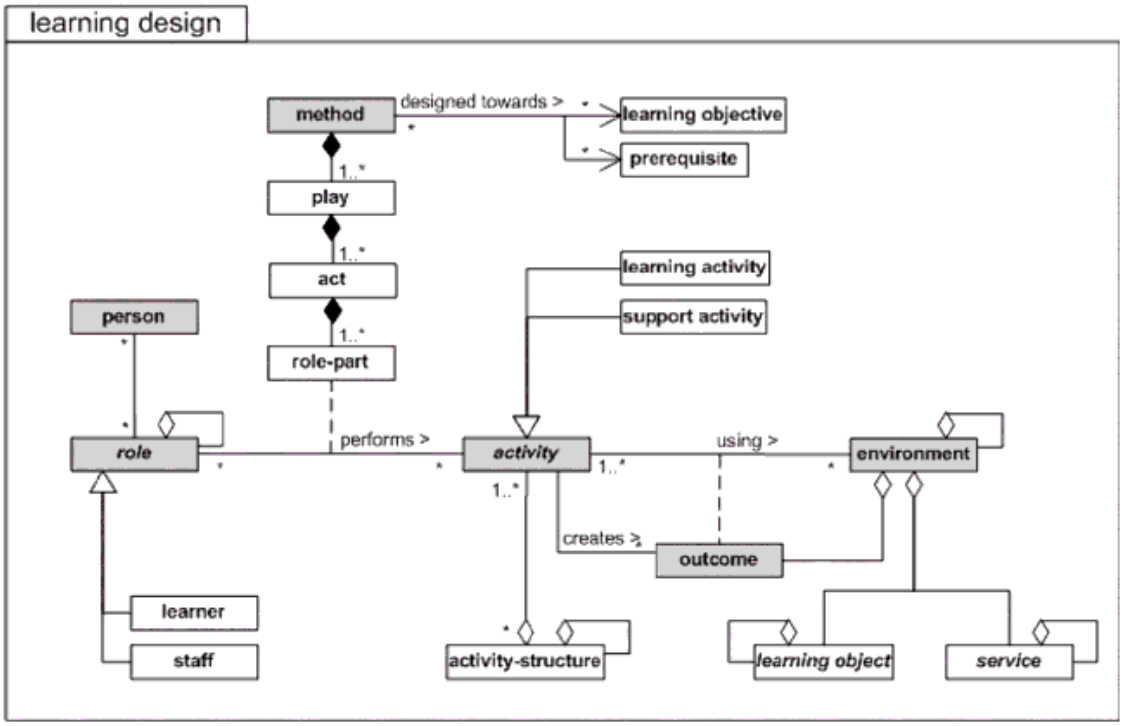


Abbildung 34: konzeptionelles UML-Modell von Level A (KOPER et al. 2003 a, 3.1 Level A Information Model)

3.3.7.3.4 Konzeptionelles Modell Level B

Als zwei weitere Bereiche in einem didaktischen Szenario werden in Level B (Abbildung 35) „property“ und „condition“ eingeführt. „property“ stellt dabei Eigenschaften eines Lernenden dar, die man sich als ein Portfolio oder Dossier vorstellen kann (KOPER et al. 2003 a, KOPER et al. 2003 b). Vorwissen, Lernpräferenz, Testergebnisse, Zertifikate oder individuell erstellte Dokument sind einige Beispiele. „condition“-Elemente sind Bedingungen, die den Lernprozess steuern können. Erst wenn z. B. erfolgreich ein Test bestanden ist, können andere Bereiche einer Lerneinheit aufgerufen werden.

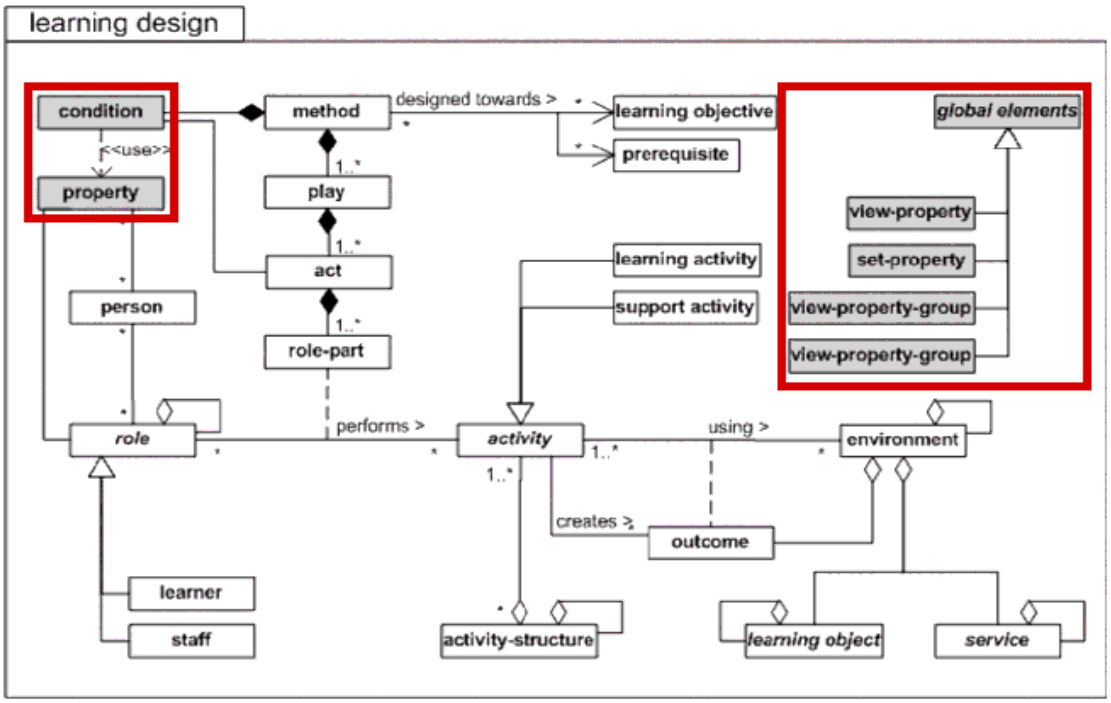


Abbildung 35: konzeptionelles UML-Modell von Level B (KOPER et al. 2003 a, 3.2 Level B Information Model)

3.3.7.3.5 Konzeptionelles Modell Level C

In Level C (Abbildung 36) wird das Element „notification“ eingeführt. Dabei handelt es sich um Benachrichtigungen, die zur Laufzeit zwischen den verschiedenen Rollen oder dem Lernsystem ausgetauscht werden können. Plan- und Rollenspiele wären mögliche Einsatzbereiche (KOPER et al. 2003 b).

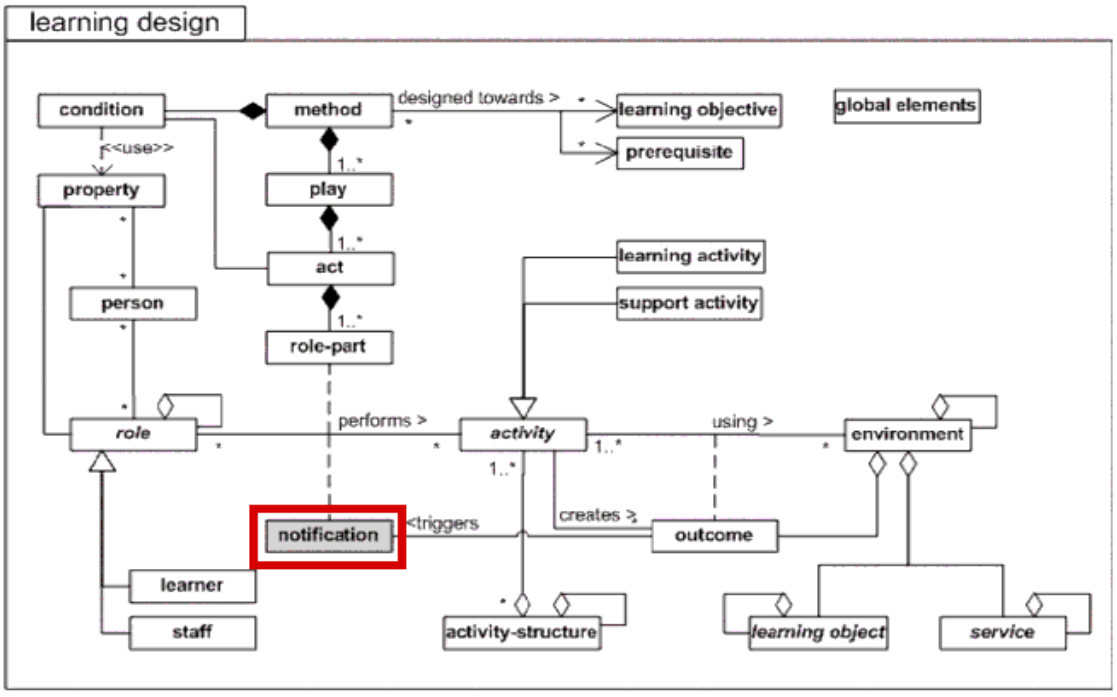


Abbildung 36: konzeptionelles UML-Modell von Level C (KOPER et al. 2003 a, 3.3 Level C Information Model)

3.3.7.3.6 IMS CP und IMS LD in einem XML-Beispiel

IMS CP und IMS LD können über Anweisungen in XSD-Dokumenten in XML abgebildet werden (KOPER et al. 2003 c). Listing 10 zeigt einen Ausschnitt eines Quelltextes eines XML-Dokumentes mit IMS CP- und IMS LD (Level A)-Elementen. Der gesamte Quelltext ist im Anhang A zu finden. Wie im Kapitel 3.3.7.3.2 IMS LD-Lerneinheit angesprochen, ist eine IMS LD Einheit (Zeile 3: „imsld:learning-design“) in die IMS CP-Struktur (Zeile 1: „manifest“) integriert (siehe auch KOPER et al. 2003 b, 3.2.3 Introducing the Elements in Learning Design). Das Element „learning-object“ (Zeile 4) erlaubt es über die Referenzierung einer Ressource externe Dateien (in diesem Fall die HTML-Datei „test.html“, aber auch XML-Dateien usw.) einzubinden (Zeile 7 bzw. 12 bis 14) (KOPER et al. 2003 a, 2.2.2 Conceptual Structure of the Learning Design, 2.3 Conceptual Vocabulary of Learning Design).

```
1 <manifest xmlns=http://www.imsglobal.org/xsd/imscp_v1p1
…
2  <organizations>
3    <imsld:learning-design identifier="LD-1" level="A" uri="">
…
4            <imsld:learning-object identifier="LD-5"
5                  type="knowledge-object">
6              <imsld:title>New Learning Object</imsld:title>
7              <imsld:item identifierref="RES-1" />
8            </imsld:learning-object>
…
9    </imsld:learning-design>
10  </organizations>
11  <resources>
12    <resource identifier="RES-1" type="webcontent" href="test.html">
13      <file href="test.html" />
14    </resource>
15  </resources>
16 </manifest>
```

Listing 10: Quelltext eines IMS LD-XML-Dokumentes („imsmanifest.xml“) mit IMS CP- und IMS LD-Elementen, erstellt mit der Software „RELOAD“

IMS LD-XML-Dateien können z. B. mit einem Standard-XML-Editor oder speziellen IMS LD-Editoren erstellt werden (z. B. RELOAD „Learning Design Editor“ (RELOAD

2008), „ReCourse Learning Design Editor“ (RECOURSE 2010 a)). Anschließend lässt sich die IMS LD-Lerneinheit z. B. über einen IMS LD-Player anzeigen (z. B. RELOAD „Learning Design Player“ (RELOAD 2008), „ReCourse Runtime System“ (RECOURSE 2010 b)). Bezüglich RELOAD hält CZAPUTA (2008, S. 72) fest: „*Überblicksartig sei die Erstellung des Learning Design mit dem Reload-LD-Editor, der nach wie vor als Referenzeditor der Spezifikation gilt, dargestellt.*“ GRIFFITHS et al. (2005, S. 134) haben in einer Tabelle weitere „*Learning Design Tools*“ zusammengestellt.

3.3.7.4 IMS QTI

IMS-QTI (2013) erläutert die „*IMS Question & Test Interoperability Specification*“ wie folgt: „*The IMS Question & Test Interoperability (QTI) specification enables the exchange of item, test and results data between authoring tools, item banks, test constructional tools, learning systems and assessment delivery systems.*“

Die Spezifikation beschreibt ein Datenmodell für die Abbildung von Fragen („assessmentItem“) und Tests („assessmentTest“) mit ihren Ergebnissen und soll den Austausch dieser Daten zwischen verschiedenen Systemen (z. B. LMS, Autoren-Werkzeuge für Prüfungsfragen) erfolgreich ermöglichen (Sicherstellung der Interoperabilität) (KRAAN et al. 2012 a, 1. Question and Test Interoperability, 2. Specification Use Cases). Laut KRAAN et al. (2012 a, 2. Specification Use Cases) wurde IMS QTI entwickelt, um:

- ein dokumentiertes Inhaltsformat für Speicherung und Austausch von Fragen („items“) unabhängig von der Erstellungs-Software („authoring tool“) zu bieten
- die Verwendung von Fragenkatalogen („item banks“) für eine möglichst große Menge von Lern-und Prüfungssystemen („learning and assessment delivery system“) zu unterstützen
- ein dokumentiertes Inhaltsformat für Speicherung und Austausch von Tests („tests“) unabhängig von der Test-Erstellungssoftware zu bieten
- die Verwendung von Fragen („items“), Fragenkatalogen („item banks“) und Tests („tests“) von verschiedenen Quellen in Lern-und Prüfungssystemen zu unterstützen
- Systeme mit der Möglichkeit der konsistenten Darstellung von Testergebnissen anzubieten

Der Ausdruck „item“ wurde der Einfachheit halber mit Frage übersetzt. Definiert wird „item“ in der Spezifikation wie folgt (KRAAN et al. 2012 b, 3. Definitions): „*The smallest exchangeable assessment object within this specification. An item is more than a 'Question' in that it contains the question and instructions to be presented, the*

responseProcessing to be applied to the candidates response(s) and the Feedback that may be presented (including hints and solutions). In this specification items are represented by the assessmentItem class and the term assessment item is used interchangeably for item.“

Abbildung 37 zeigt die IMS QTI-Bausteine Frage („assessmentItem“) und Test („assessmentTest“) im Kontext mit beteiligten Personen und Systemen.

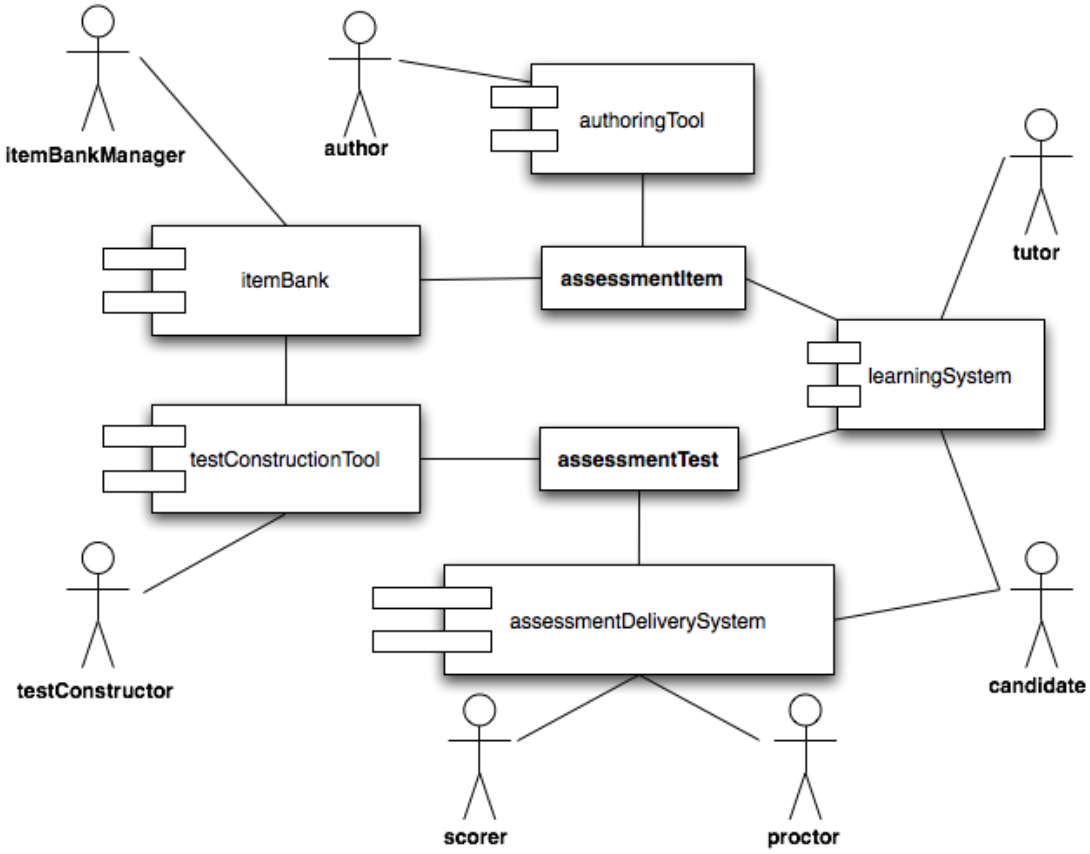


Abbildung 37: Frage („assessmentItem“) und Test („assessmentTest“) im Kontext mit beteiligten Personen und Systemen (KRAAN et al. 2012 a, 2. Specification Use Cases)

IMS QTI kann verschiedene Fragetypen z. B. „Single Choice“, „Multiple Choice“, Anordnungsfragen, Zuordnungsfragen über das XML-Binding abbilden (KRAAN et al. 2012 c, 3. Items). Abbildung 38 zeigt eine „Single Choice“-Frage (nur eine richtige Antwort) mit ihrer potentiellen Darstellung. Listing 11 führt einen Ausschnitt des dazu gehörenden XML-Quelltext auf. Der Quelltext zeigt u. a. das Root-Element „assessmentItem“ (Zeile 4), die einzige richtige Antwort (Zeile 5 Element „responseDeclaration“ mit dem „cardinality“-Attribut-Wert „single“, Zeile 7 Element „correctResponse“) und die drei Auswahlelemente (Zeile 11 bis 16, Elemente „simpleChoice“).

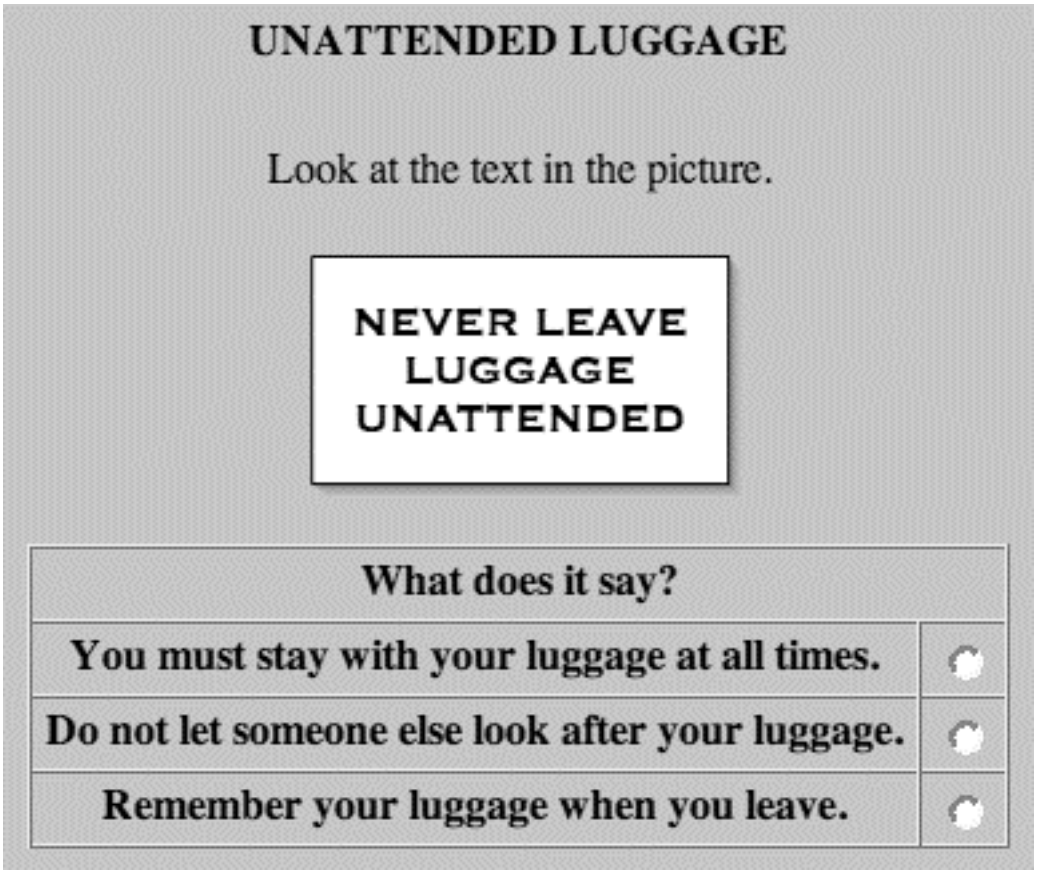


Abbildung 38: mögliche Darstellung einer Single Choice-Frage („simpleChoice“, KRAAN et al. 2012 c, 3. Items)

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- Thie example adapted from the PET Handbook, copyright
3 University of Cambridge ESOL Examinations -->
4 <assessmentItem xmlns="http://www.imsglobal.org/xsd/imsqti_v2p1"
…
5  <responseDeclaration identifier="RESPONSE" cardinality="single"
6  baseType="identifier">
7   <correctResponse>
8    <value>ChoiceA</value>
9   </correctResponse>
10  </responseDeclaration>
…
11    <simpleChoice identifier="ChoiceA">You must stay with your
12    luggage at all times.</simpleChoice>
13    <simpleChoice identifier="ChoiceB">Do not let someone else look
14    after your luggage.</simpleChoice>
```

```
15    <simpleChoice identifier="ChoiceC">Remember your luggage when
16    you leave.</simpleChoice>
…
17 </assessmentItem>
```

Listing 11: Auschnitt des XML-Quelltextes der in Abbildung 38 dargestellten IMS QTI-Frage („simpleChoice“, KRAAN et al. 2012 d, Beispieldatei „choice.xml“)

Wie z. B. die vorher dargestellte IMS QTI-Frage in eine „Unit of Learning“ mit IMS CP (siehe Kapitel 3.3.7.1 IMS CP) und IMS LD (siehe Kapitel 3.3.7.3 IMS LD) integriert werden kann, zeigt ausschnittsweise Listing 12. Die IMS QTI-Frage wird in einem IMS LD-Lernobjekt (Zeile 8) über das IMS-LD-Element „item“ (Zeile 10) eingebunden. Sie ist als eine externe XML-Datei („choice_01.xml“, Zeile 17) referenziert (Zeile 11 „identifierref“, Zeile 16 „identifier“), bei der es sich z. B. um die vorher beschriebene IMS QTI-Frage handeln könnte.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- edited with XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Colin
3 Tattersall (Open University of the Netherlands) -->
4 <imscp:manifest
…
5  <imscp:organizations>
6    <imsld:learning-design identifier="LD-Integration-Example"
7    uri="" level="B">
…
8            <imsld:learning-object identifier="LO-QTI-Item1">
9              <imsld:title>Assign a sign</imsld:title>
10              <imsld:item identifier="I-Question1"
11              identifierref="Question_1"/>
12            </imsld:learning-object>
…
13    </imsld:learning-design>
14  </imscp:organizations>
15  <imscp:resources>
…
16    <imscp:resource identifier="Question_1"
17      type="imsqti_item_xmlv2p1" href="choice_01.xml">
```

```
18      <imscp:file href="choice_01.xml"/>
19      <imscp:file href="sign.png"/>
20    </imscp:resource>
…
21  </imscp:resources>
22 </imscp:manifest>
```

Listing 12: XML-Quelltextausschnitt einer Manifest-Datei (siehe Kapitel 3.3.7.1 IMS CP) mit IMS CP, IMS LD und einem referenzierten (externen) IMS QTI-dokument choice_01.xml (eingekürzt nach KRAAN et al. 2012 e, 4.3. A skeletal example of IMS LD, QTI and CP integration)

Um IMS QTI-Inhalte zu erstellen, bieten sich IMS QTI-Editoren („authoring tools“) an. Neben dieser Software muss zur Darstellung und Sicherstellung der Funktionalität noch ein „Player“ zur Verfügung stehen. Über das Internet sind beide Arten dieser Software (auch für verschiedene LMS wie z. B. „OLAT“ (OLAT 2014) oder „Moodle“ (MOODLE 2014)) zu finden (siehe z. B. KOPER 2013, Selected QTI Tools). Steht keine Software dieser Art zur Verfügung und sollen trotzdem Testfragen mit Funktionalitäten im Browser eingesetzt werden (auch ohne LMS), kann auch auf Werkzeuge zurückgegriffen werden, die auf der Grundlage von HTML und JavaScript diese Funktionalitäten umsetzen (z. B. „Hot Potatoes“ siehe Anhang J, siehe auch WINKELMANN 2006, S. 59 ff, PIOTROWSKI & RÖSNER 2005, 2 Webbasierte Multiple-Choice-Tests).

3.3.8 XSL

Neben den dargestellten XML-Anwendungen gehört auch XSL zur XML-Technologie (W3C-XML-Technology 2013). Die Extensible Stylesheet Language kann zur Verarbeitung, Darstellung oder Konvertierung von XML-Dokumenten herangezogen werden (LIAM 2010 c, What is XSL-FO?, BERGLUND 2006, Abstract) und ist selbst eine mit Hilfe von XML definierte Auszeichnungssprache (LIAM 2010 c, What is XSL-FO?, KAY 2007, 1.1 What is XSLT?). Eine aus XSL-Anweisungen bestehende Style-Sheet-Dokument (gespeichert in einer „.xsl“-Datei) ist damit XML-konform. Anwendungen, die für XML entwickelt wurden, sind somit auch für XSL anwendbar.

XSL zerfällt in zwei Komponenten (LIAM 2010 c, What is XSL-FO?, BERGLUND 2006, 1.1 What is XSLT?):

- „*XSL Transformations*“ (XSLT) (KAY 2007)
  XSLT erlaubt die strukturelle Transformation von einem XML Format in andere Formate.

- “XSL Formatting Objects“ (BERGLUND 2006)
  “*XSL-FO*“ ermöglicht die Formatierung mit Hilfe der „formatting objects“.

LIAM (2010 d, TRANSFORMATION) erläutert dazu: „*XSLT and XSL-FO are W3C Recommendations for defining XML document transformation and presentation. Use XSLT to transform documents into XSL-FO for printing or viewing; you can also use XSLT as a general XML-aware programming and transformation language, and you can use XSL-FO directly without XSLT.*“

3.3.8.1 XSLT

XSLT liegt derzeit in der Version 2.0 vor (KAY 2007). LIAM (2010 d, Wildly Popular) weist allerdings auf Folgendes hin: „*You will find XSLT (version 1) inside most modern Web browsers, so that XML can be transformed on the fly without the user even noticing.*“ Aus diesem Grund und unter Berücksichtigung des Zeitraums der Erstellung der vorliegenden Arbeit (Beginn der Arbeit vor 2006) wird besonders auf die Vorgänger-Version XSLT 1.0 (CLARK 1999) mit XPath Version 1.0 eingegangen (siehe nächsten Absatz). Zusätzlich ist bei XSLT 2.0 und XPath 2.0 eine hohe Abwärts-Kompatibilität vorhanden (KAY 2007, Status of this Document, BONGERS 2004, S. 27, BERGLUND et al. 2010, Abstract).

XSLT („*W3C Recommendation*“ seit 1999) wird vom „World Wide Web Consortium“ (CLARK 1999, Abstract) wie folgt beschrieben: *"... XSLT, which is a language for transforming XML documents into other XML documents*". GEEB (2003, S. 19) erläutert: „*XSLT ist eine Transformationssprache, die Inhalte aus XML-Dokumenten in ein anderes Format transportiert. XSLT ist eine XML-Anwendung und hat damit fest reglementierte Elemente, die in bestimmten Kombinationen vorkommen dürfen oder müssen. XSLT-Anweisungen zu einem XML-Dokument werden damit ihrerseits selbst als XML-Dokumente formuliert*“. In der neuen Version 2.0 wird der gestiegene Funktionsumfang und der Zusammenhang mit der parallelen Entwicklung von XPath Version 2.0 (BERGLUND et al. 2010) betont (KAY 2007, 1.2 What's New in XSLT 2.0?): „*XSLT 1.0 was published in November 1999, and version 2.0 represents a significant increase in the capability of the language. … XSLT 2.0 has been developed in parallel with XPath 2.0 (see [XPath 2.0]), so the changes to XPath must be considered alongside the changes to XSLT.*“

Allgemein erlaubt es XSLT beruhend auf XPath-Ausdrücken (CLARK & DEROSE 1999, BERGLUND et al. 2010, siehe auch CASE et al. 2011) aus einem Quelldokument ein verändertes Zieldokument zu erzeugen. Der XSLT-Prozessor baut bei diesem Vorgang eine Baumstruktur des (XML-)Quell-Dokumentes auf, liest

das XSLT-Dokument ein und erzeugt aus dem Quellbaum einen Zielbaum, der den Angaben des XSLT-Dokument entspricht (Abbildung 39) (CAGLE et al. 2001, S. 62). Das Ergebnis-Dokument kann z. B. ein weiteres XML-, HTML- oder Text-Dokument sein. XPath bietet für diese Transformation eine Syntax an auf bestimmte Knoten in einem XML-Dokument zuzugreifen (CLARK & DEROSE 1999, Introduction). BONGERS (2004, S. 26) beschreibt XPath als eine „*Pfadbeschreibungssprache, in der sich der Weg zu einer, in einem XML-Dokument befindlichen Information formulieren lässt*“. Er führt weiter aus: „*XPath dient der Navigation innerhalb von Dokumentenbäumen und kann beliebige Knotenmengen eines solchen Baumes benennen sowie die dort liegenden Informationen zugänglich machen.*“ XPath wird zusätzlich in der „*XML Linking-Spezifikation*“ (DEROSE et al. 2010) eingesetzt und durch XQuery (CASE et al. 2011) erweitert.

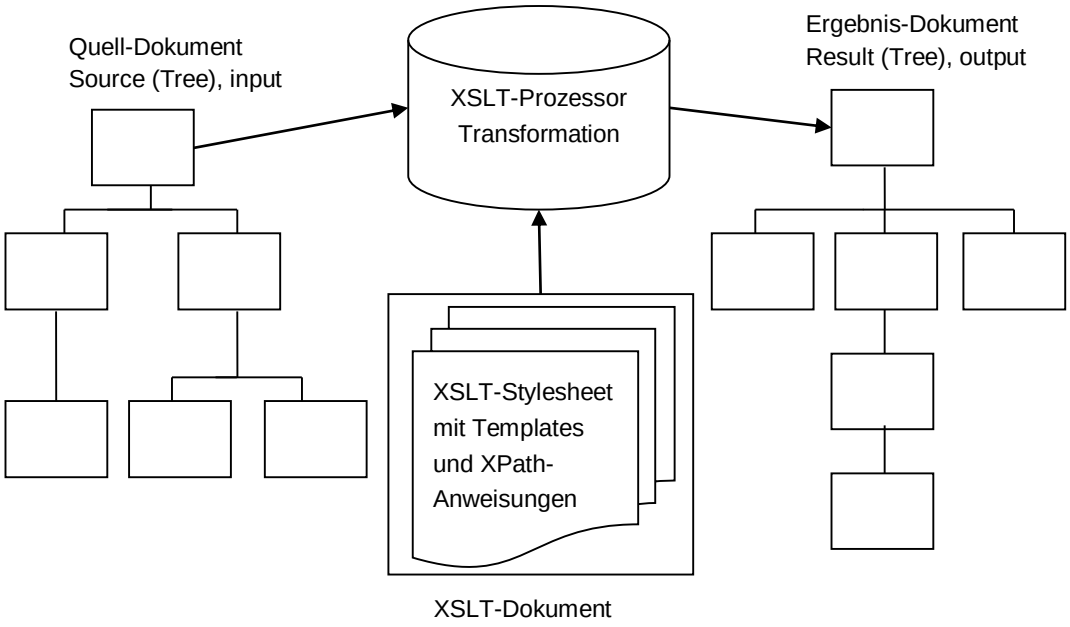


Abbildung 39 Ablauf einer XSLT-Transformation mit veränderten Ausgabe-Dokument (abgeändert nach CAGLE et al. 2001, S. 17)

Ein XSLT-Dokument ist ein XML-Dokument mit dem Root-Element „stylesheet“ und den verschiedenen „template“-Elementen (Listing 13, z. B. Zeile 5). Ein Template (Schablone) gibt die Transformation vor, die auf einen bestimmten Teil eines XML-Dokumentes angewandt werden soll (CLARK 1999, 1 Introduction). Welcher Teil transformiert werden soll, wird mit einer XPath-Anweisung festgelegt. Abbildung 40 stellt den Ablauf einer Transformation mit besonderem Augenmerk auf die Abarbeitung der „templates“ dar.

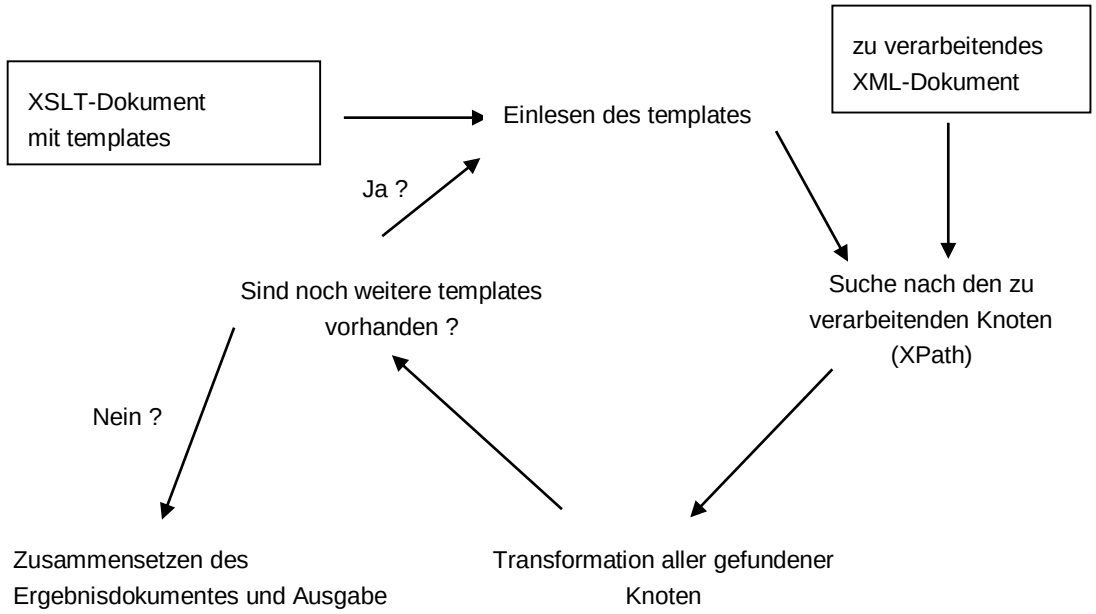


Abbildung 40 Ablauf einer XSLT-Transformation mit besonderem Augenmerk auf die Abarbeitung der „templates“ (abgeändert nach CAGLE et al. 2001, S. 64)

Daneben stehen in XSLT weitere Konstrukte wie z. B. „if“-Bedingungen, „for-each“-Schleifen oder Sortierung von Daten zur Verarbeitung zur Verfügung (CLARK 1999).

Folgendes Beispiel soll eine XSLT-Transformation mit einem XSLT-Dokument (Listing 13), einem XML-Quell-Dokument (Listing 14) und dem XHTML-Ergebnis-Dokument (Listing 15) genauer erläutern. Listing 13 zeigt einen Auszug aus dem XSLT-Dokument. Es transformiert das XML-Quell-Dokument in ein XHTML-Dokument. Über das „output“-Element (Zeile 4) werden Eigenschaften des Ergebnis-Dokumentes angegeben. Das „method“-Attribut legt hierbei den Dokumenttyp fest (z. B. „xml“ oder „html“ (HTML Version 4)), das „indent“-Attribut die Einrückung des Ergebnis-Quelltextes und das „encoding“-Attribut den Zeichensatz (CLARK 1999, 16 Output). In Zeile 6 steht das erste „template“-Element mit dem „match“-Attributwert „doc“. Der Attributwert „doc“ steht für ein XPath-„Pattern“ (Vergleichsmuster, BONGERS 2004, S. 36). Über Patterns werden Knoten im Quell-Dokument gefunden. Mit Hilfe dieses Vergleichsmusters wird ein „doc“-Element im Quell-Dokument gesucht (Listing 14, Zeile 2) und im Ergebnis-Dokment gemäß der Vorgaben dieses Templates nach XHTML transformiert (Listing 15) (CLARK 1999, 2.3 Literal Result Element as Stylesheet). Das im Template angegebene XHTML-Grundgerüst (Listing 13, Zeile 7 bis 12) wird damit in das Ergebnis-Dokument (Listing 15) übernommen. Zusätzlich befinden sich in dem angesprochenen Template zwei weitere XSLT-Anweisungen. Die erste mit dem Namen „value-of select“ und dem XPath-„Pattern“ „title“ (Listing 13, Zeile 9) liest den Inhalt des „title“-Elementes im Quell-Dokument aus (GEEB 2003, S. 134) und erzeugt im Ergebnis-Dokument einen

Textknoten mit diesem Inhalt (CLARK 1999, 7.6.1 Generating Text with xsl:value-of). Die zweite Anweisung „apply-templates“ ohne „select“-Attribut (Listing 13, Zeile 11) sucht für alle Kindknoten (inklusive Textknoten) ausgehend vom aktuellen Knoten („doc“) die passenden Templates und wendet sie an (CLARK 1999, 5.4 Applying Template Rules, BONGERS 2004, S. 645 ff, CAGLE et al. 2001, S. 105, GEEB 2003, S. 132). Auf diese Art und Weise werden im Ergebnis-Dokument die jeweiligen XHTML-Elemente erzeugt und mit Inhalt befüllt. Mit Hilfe dieser Zuweisung von XHTML-Elementen (z. B. „H1“-Element) können zusätzlich z. B. in einem Browser die Transformationsergebnisse indirekt formatiert werden (Standard-Formatierung im Browser ohne explizite CSS- oder XSL-FO-Anweisungen).

```
1 <xsl:stylesheet version="1.0"
2  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
3  xmlns="http://www.w3.org/TR/xhtml1/strict">
…
4  <xsl:output method="xml" indent="yes" encoding="iso-8859-1"/>
5
6  <xsl:template match="doc">
7   <html>
8    <head>
9     <title><xsl:value-of select="title"/></title>
10    </head>
11    <body><xsl:apply-templates/></body>
12   </html>
13  </xsl:template>
14
15  <xsl:template match="doc/title">
16   <h1><xsl:apply-templates/></h1>
17  </xsl:template>
…
18  <xsl:template match="para">
19   <p><xsl:apply-templates/></p>
20  </xsl:template>
```

```
…
21 </xsl:stylesheet>
```

Listing 13: Quelltext eines XSLT-Dokumentes zur Transformation eines XML-Dokumentes (Listing 14) in ein XHTML-Dokument (Listing 15) (Ausschnitt aus CLARK 1999, D.1 Document Example)

```
1 <!DOCTYPE doc SYSTEM "doc.dtd">
2 <doc>
3  <title>Document Title</title>
…
4   <section>
…
5    <para>This is a test.</para>
…
6   </section>
…
7 </doc>
```

Listing 14: Quelltext eines XML-Dokumentes (Input für die XSLT-Transformation) (Ausschnitt aus CLARK 1999, D.1 Document Example)

```
1 <?xml version="1.0" encoding="iso-8859-1"?>
2 <html xmlns="http://www.w3.org/TR/xhtml1/strict">
3  <head>
4  <title>Document Title</title>
5  </head>
6  <body>
7   <h1>Document Title</h1>
…
8   <p>This is a test.</p>
…
9  </body>
10 </html>
```

Listing 15: XHTML-Ergebnis-Dokument (Output der XSLT-Transformation) (Ausschnitt aus CLARK 1999, D.1 Document Example)

Mit Hilfe von XSLT können XML-Dokumente serverseitig oder clientseitig transformiert werden. Die serverseitige XSL-Transformation z. B. nach XHTML (siehe

Kapitel 3.3.5.3 XHTML) oder HTML (siehe Kapitel 3.4 HTML) hat den Vorteil, dass die XML-Inhalte auch mit Browsern betrachtet werden können, die nicht in der Lage sind XML zu verarbeiten. Zum Browser werden in diesem Fall nur (X)HTML-Dateien geschickt. Der Nachteil ist, dass dies nur in Zusammenarbeit mit einem Web-Server funktioniert, aber nicht, wenn die Daten etwa lokal oder auf einer CD-ROM präsentiert werden sollen. Zusätzlich muss der Web-Server eine entsprechende Schnittstelle besitzen, die das Einbinden eines XSL-verarbeitenden Software-Moduls erlaubt. Bei der clientseitigen XSL-Transformation von XML nach (X)HTML übernimmt z. B. der Browser die Transformation und anschließende Darstellung der XML-Inhalte. Das setzt allerdings voraus, dass der Browser XML-Dokumente mit XSLT verarbeiten kann. LIAM (2010 d, What is XSLT Used For?) hält dazu fest: „*XSLT support is shipped with all major computer operating systems today, as well as being built in to all major Web browsers.*“

3.3.8.2 XSL-FO

XSL-FO (BERGLUND 2006) wird vom W3C wie folgt beschrieben (LIAM 2010 c, What is XSL-FO?): „*XSL-FO, the Formatting Objects part of XSL, is an XML markup language for describing page layout and formatting.*“ CAGLE et al. (2001, S. 344, siehe auch SKULSCHUS & WIEDERSTEIN 2005, S. 34 ff) halten zu XSL-FO fest: „*Schematically, XSL-FO output is produced from an XML source document in two steps – an XSLT transformation to add FO markup, followed by the formatting step.*“

XML-Dokumente enthalten nur logische Auszeichnungen. Eine Auszeichnung mit Elementen sagt nichts darüber aus, wie sie darzustellen sind. Die so bezeichneten Daten sind unabhängig vom Ausgabemedium (z. B. Bildschirm, Display, Lautsprecher, Drucker) und enthalten keinerlei Angaben zur Formatierung (z. B. Schriftart, Schriftgröße). Für das Layout von XML-Dokumenten stehen derzeit hauptsächlich zwei Formatierungssprachen zur Verfügung, CSS und XSL-FO. CSS wird meist zur Formatierung von HTML genutzt, kann aber auch zur Darstellung von Elementen eines XML-Dokumentes in einem Browser eingesetzt werden. XSL-FO ermöglicht die sehr differenzierte Formatierung eines XML-Dokumentes mit Hilfe von „formatting objects“. Allerdings wird dazu ein Prozessor benötigt, der auch XSL-FO-Anweisungen umsetzen kann. Ohne CSS und XSL-FO ist zusätzlich durch die Transformation nach HTML eine grundlegende Formatierung möglich (Standard-darstellung von HTML durch den Browser).

Quelltexte eines Input-, eines XSL-FO- und des Ergebnis-Dokumentes einer XSL-FO-Transformation zeigt BERGLUND (2006, 6.5.1.1 Example).

3.4 HTML

Im Gegensatz zu den im vorherigen Kapitel 3.3 XML beschriebenen Auszeichnungssprachen handelt es sich bei HTML um keine XML- sondern eine SGML-Anwendung (RAGGETT et al. 1999, Abstract, zu SGML siehe ISO-8879 1986). Lobin (2000, S. 190) hält HTML für eine der bekanntesten SGML-Anwendungen. HTML und XHTML (siehe Kapitel 3.3.5.3 XHTML) unterscheiden sich zwar in ihrer Ableitung (HTML von SGML, XHTML von XML), sind sich ansonsten allerdings sehr ähnlich (W3C-HTML-CSS 2013, What is XHTML?). XHTML weist jedoch u. a. die stringentere XML-Syntax auf (z. B. exakte Verschachtelung von Elementen, Attributwerte in Anführungszeichen, jedes Element muss geschlossen werden, siehe Unterschied XHTML 1.0 zu HTML 4.01 unter PEMBERTON et al. 2002, 4. Differences with HTML 4).

Die letzte HTML-„*Recommendation*“ mit der Bezeichnung „*HTML 4.01 Specification*“ (RAGGETT et al. 1999) wurde vom W3C weiterentwickelt und liegt derzeit als HTML 5 in Form einer „*Candidate Recommendation*” vor (BERJON 2013). In dieser vorläufigen Empfehlung wird festhalten, dass HTML 5 in HTML- und in XML-Syntax geschrieben werden kann (PIETERS 2013, 1.2 History of HTML). Für die XML-Syntax gilt (PIETERS 2013, 2 Syntax): „*This syntax is compatible with XHTML1 documents and implementations.*“

HTML wird als „*die Sprache*“ des World Wide Web bezeichnet (RAGGETT et al. 1999, Abstract, siehe auch W3C-HTML-CSS 2013, zu Ausdruck World Wide Web siehe RAGGETT et al. 1999, 2.1 What is the World Wide Web?) und aufgrund ihrer weltweiten Verbreitung häufig als Ausgabe-Methode bei XSLT-Transformationen eingesetzt (siehe Listing 13, Zeile 4). KAY (2001, S. 12) hält fest: „*Converting XML to HTML for display is probably the most common application of XSLT today … Once you have the data in HTML format, it can be displayed on any browser.*“ Weiter erläutert er, dass dabei typischerweise HTML 4 zum Einsatz kommt (S. 54). HAUG et al. (2012, 2.1.1.3 HTML) erklären: „*HTML-Dokumente werden mit sogenannten Webbrowsern (z. B. Mozilla Firefox, Microsoft Internet Explorer, Google Chrome) angezeigt. Durch die große Verbreitung der Browser, auch für verschiedene Betriebssysteme, kann HTML von praktisch jedem Computer dargestellt werden.*“ MÜNZ & GULL (2012, S. 22) weisen auf Folgendes hin: „*HTML ist ein so genanntes Klartextformat. HTML-Dateien können Sie mit jedem beliebigen Texteditor bearbeiten, der Daten als reine Textdateien abspeichern kann*“ (siehe auch Kapitel 3.3 XML).

Lobin (2000, S. 190) hält zu HTML fest: „*HTML hat die Aufgabe, Bildschirm-Seiten zu beschreiben und im Zusammenspiel mit dem Hypertext Transmission Protocol (HTTP) über das Internet verfügbar zu machen.*“ Als einen besonders wichtigen Punkt betont er, dass die Verknüpfung mit anderen im Internet abgelegten HTML-Seiten über Hyperlinks möglich ist.

LOBIN et al. (2003, S. 2 f, siehe auch JUNGMANN 2012, S. 100, JUNGMANN et al. 2004, S. 11, vergleiche auch SCHREYER 2002, S. 30) weisen darauf hin, dass HTML im Gegensatz zu XML für die Abbildung von E-Learning-Inhalten weniger geeignet ist: „*HTML ist als Datenhaltungsformat für eLearning-Inhalte, die inhärent gut strukturierbare Informationen darstellen, nur bedingt geeignet, da es kaum Möglichkeiten zur semantischen Auszeichnung und nur wenige Elemente zur expliziten Strukturierung bietet.*“ JUNGMANN (2012, S. 100) hält zusätzlich fest: „*Aus informationstechnischer Sicht erscheint des Weiteren die in den meisten herkömmlichen Lernumgebungen für die Umsetzung genutzte Sprache HTML ungeeignet*“. Sie beschreibt weiter, dass HTML (im Gegensatz zu XML, S. 101 f) im Anwendungsbereich E-Learning für die folgenden Verwendungszwecke untauglich ist:

- *„den Austausch von Komponenten,*
- *die Wiederverwendung von Komponenten,*
- *die automatisierte Verarbeitung und*
- *die Auswertung des Inhalts der Dokumente*“

Darüber hinaus stellt sie fest: „*Trotz viel versprechender Namensgebung ist HTML ungeeignet zum Aufbau komplexer, dynamischer Hypertextstrukturen, deren Potenzial für selbstorganisiertes Lernen damit nur ansatzweise ausgeschöpft werden kann*“. Auf der anderen Seite ist XML nicht als Ersatz für HTML gedacht (BEHME & MINTERT 2000, S. 21, GEEB 2003, S. 12, ANDERSON et al. 2000, S. 36, SEEBOERGER-WEICHSELBAUM 2004, S. 22 ff).

Listing 16 zeigt den Quelltext eines HTML4-Dokumentes. Die Struktur eines HTML-Dokumentes spiegelt sich in den drei Teilen Informationen zur HTML Version (Zeile 1-2), Kopf des Dokumentes mit deklarativen Inhalt („HEAD“-Element, Zeile 4-6) und Körper des Dokumentes mit dem eigentlichen Inhalt wieder („BODY“-Element, Zeile 7-9, oder „FRAMESET“-Element) (RAGGETT et al. 1999, 7.1 Introduction to the structure of an HTML document). Das „HTML“-Element umspannt die letzten beiden Teile mit dem „HEAD“- und „BODY“-Element (Zeile 3-10). Die Element-Namen können in Groß- oder Kleinbuchstaben geschrieben werden.

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
2  "http://www.w3.org/TR/html4/strict.dtd">
3 <HTML>
4  <HEAD>
5   <TITLE>My first HTML document</TITLE>
6  </HEAD>
7  <BODY>
8   <P>Hello world!
9  </BODY>
10 </HTML>
```

Listing 16: Einfaches HTML4-Dokument (RAGGETT et al. 1999, 7.1 Introduction to the structure of an HTML document)

Das folgende Kapitel geht auf CSS ein und schildert wie das Layout eines HTML-Dokumentes beeinflusst werden kann.

3.5 CSS

Neben HTML ist CSS eine Kerntechnologie für die Erstellung von Dokumenten für das World Wide Web („*Web pages*“, W3C-HTML-CSS 2013). Das W3C beschreibt CSS wie folgt (W3C-HTML-CSS 2013, What is CSS?): „*CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. … CSS is independent of HTML and can be used with any XML-based markup language. The separation of HTML from CSS makes it easier to maintain sites, share style sheets across pages, and tailor pages to different environments. This is referred to as the separation of structure (or: content) from presentation.*“ HAUG et al. (2012, 2.1.1.3 HTML (CSS)) weisen bei Ihrer Charakterisierung von CSS auf CSS-Klassen und Identitäten hin: „*Cascading Stylesheets werden dafür verwendet, die Darstellung eines strukturierten Dokuments zu definieren. Es können HTML- oder XHTML-Elementen spezifische Klassen oder Identitäten zugewiesen werden, die genau festlegen, wie und wo das jeweilige Element angezeigt werden soll.*“ Mit anderen Worten lassen sich mit CSS Formatvorlagen zur Gestaltung von Webseiten definieren.

CSS liegt derzeit als „*W3C Recommendation*“ in der Version 2.1 vor (BOS et al. 2011). Das W3C weist bezüglich CSS 1 (LIE & BOS 1996) auf Folgendes hin (BOS et al. 1998, Abstract): „*CSS2 builds on CSS1 (see [CSS1]) and, with very few exceptions, all valid CSS1 style sheets are valid CSS2 style sheets.*“

Listing 17 zeigt den Quelltext eines HTML4-Dokumentes mit den verschiedenen Möglichkeiten CSS in HTML einzubinden (W3C-CSS1 1996, 1.1 Containment in HTML). Grundsätzlich können drei Varianten unterschieden werden: Einbinden einer externen CSS-Datei, Einbinden in den „HEAD“-Bereich über das „STYLE“-Element und Einbinden in den „BODY“-Bereich in ein HTML-Element über das „STYLE“-Attribut (siehe z. B. MÜNZ & GULL 2012, S. 413 ff, MÜNZ & KESSLER 2001, S. 74 ff, LIE & BOS 1997, S. 54 ff). Die erste Variante zeigt Zeile 4 bis 5. Hier wird über das „LINK“-Element von HTML eine externe CSS-Datei über ein „HREF“-Attribut referenziert. Zeile 7 zeigt als Alternative die Referenzierung einer externen Datei über die CSS-Anweisung „@import“ (innerhalb eines „STYLE“-Elementes von HTML). Bei den externen Dateien handelt es sich um reine Textdateien mit der Dateiendung „.css“. Der Vorteil bei der Einbindung einer externen Formatvorlagendatei besteht darin, dass diese Datei von mehreren HTML- oder XML-Dokumenten importiert werden kann und damit eine zentrale Layout-Steuerung möglich wird.

Die zweite Variante über das „STYLE“-Element ist in Zeile 6 aufgeführt. Die CSS-Anweisung in Zeile 8 gibt an, dass der Textinhalt von „H1“-Elementen in diesem HTML-Dokument in blauer Schriftfarbe („color“, „foreground color“, siehe LIE & BOS 1996, 5.3.1 ‚color‘) dargestellt werden soll. Die CSS-Anweisung „H1 { color: blue }“ besteht dabei aus dem Selektor „H1“ und innerhalb der geschweiften Klammern aus der Eigenschaft „color“ mit dem Wert „blue“. Der Selektor gibt an, für welche HTML-Elemente die Formatdefinitionen gelten (MÜNZ & GULL 2012, S. 424). Die dritte Variante über das „STYLE“-Attribut zeigt Zeile 13. Hier soll der Textinhalt des P-Elementes (Absatz) grün formatiert werden.

Wenn mehrere CSS-Anweisungen auf ein Element zutreffen, greifen Mechanismen wie Vererbung und Kaskadierung (inkl. Gewichtung der Reihenfolge der Anweisungen, siehe W3C-CSS1 1996, 1.3 Inheritance, 3 The cascade, siehe auch MÜNZ & GULL 2012, S. 432 ff)

```
1 <HTML>
2  <HEAD>
3   <TITLE>title</TITLE>
4   <LINK REL=STYLESHEET TYPE="text/css"
5   HREF="http://style.com/cool" TITLE="Cool">
6   <STYLE TYPE="text/css">
```

```
7     @import url(http://style.com/basic);
8     H1 { color: blue }
9    </STYLE>
10   </HEAD>
11   <BODY>
12    <H1>Headline is blue</H1>
13    <P STYLE="color: green">While the paragraph is green.
14   </BODY>
15 </HTML>
```

Listing 17: Quelltext eines HTML4-Dokumentes mit verschiedenen Möglichkeiten CSS einzubinden (LIE & BOS 1996, 1.1 Containment in HTML)

3.6 JavaScript

Neben HTML und CSS ist JavaScript ein wesentlicher Baustein bei der Erstellung von Internetauftritten (W3C-WIKI-WEB-STANDARDS-MODEL 2014). JavaScript (Entwicklung der Firma Netscape, Brendan Eich, siehe GOODMAN 2001, Foreword) ist eine meist clientseitig eingesetzte, objektbasierte Skriptsprache, die vom Interpreter des Browsers abgearbeitet werden kann (FLANAGAN 2006, S. 1 ff, ECMA-262 2011, S. 1 ff). Die Grundlagen von JavaScript (und JScript der Firma Microsoft) wurden durch ECMA international unter dem Namen ECMAScript (ECMA-262 2011) veröffentlicht und ein Jahr später 1998 zum ISO-Standard ISO/IEC 16262 (ISO-16262 2011) erhoben (ECMA-262 2011, S. vii, siehe auch ZAKAS 2005, S. 2, GOODMAN 2001, Foreword, WIKIPEDIA-JAVASCRIPT 2013). Im Folgenden wird auf den clientseitigen Einsatz von JavaScript eingegangen.

FLANAGAN (2006, S. 1) beschreibt JavaScript wie folgt: „*JavaScript is most commonly used in web browsers, and, in that context, the general-purpose core is extended with objects that allow scripts to interact with the user, control the web browser, and alter the document content that appears within the web browser window.*“ Er weist weiter darauf hin (S. 3), dass das ECMA eine Erweiterung zu JavaScript mit den Namen E4X („*ECMASCript for XML*“, SCHNEIDER et al. 2005, später ISO-Norm ISO/IEC 22537, siehe ISO/IEC-22537 2006) für den Einsatz mit XML publizierte. Allerdings war E4X in den Browsern kaum implementiert und wurde zugunsten der Unterstützung des (XML-)DOM 2013 eingestellt (WIKIPEDIA-E4X-EN 2013, Browser support, WIKIPEDIA-E4X-DE 2013, Standardisierung, MDN-E4X 2013, W3SCHOOLS-E4X 2013, The Future of E4X, zu „*XML DOM*“ siehe auch ZAKAS 2005, S. 445 ff).

Allgemein hält ZAKAS (2005, S. 3) zu JavaScript fest, dass eine komplette JavaScript-Implementierung aus drei Teilen besteht (Abbildung 41). Er benennt als den Hauptbestandteil das „*ECMAScript*“, als weitere Bestandteile das „*Document Object Model (DOM)*“ und das „*Browser Object Model (BOM)*“.

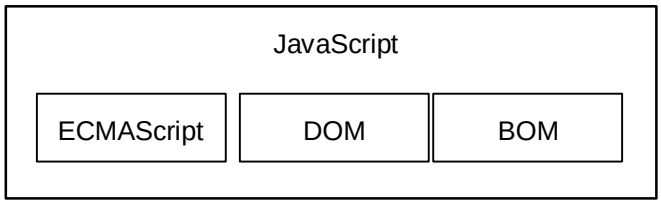


Abbildung 41 Die drei Teile einer vollständigen JavaScript-Implementierung (ZAKAS 2006, S. 3)

Den Ausdruck DOM erklärt er wie folgt (ZAKAS 2005, S. 6, siehe auch HORS et al. 2004, Abstract, FLANAGAN 2006, S. 307 ff, GOODMAN 2001 S. 41 ff): „*The Document Object Model (DOM) is an application programming interface (API) for HTML als well as XML. The DOM maps out an entire page as a document composed of a hierarchy of nodes.*“ Das DOM liegt derzeit als „*Document Object Model (DOM) Level 3*“ in Form einer „*W3C Recommendation*“ vor (Core Spezifikation, HORS et al.2004, siehe Listing 18, Abbildung 42). ZAKAS (2005, S. 7) hält bezüglich dieses „*Level 3*“ fest (siehe auch HORS et al. 2004, Abstract): „*In Level 3, the DOM Core is extended to support all of XML 1.0*“.

```
1 <table>
2  <tbody>
3   <tr>
4    <td>Shady Grove</td>
5    <td>Aeolian</td>
6   </tr>
7   <tr>
8    <td>Over the River, Charlie</td>
9    <td>Dorian</td>
10   </tr>
11  </tbody>
12 </table>
```

Listing 18: Ausschnitt aus dem Quelltext eines XHTML-Dokumentes in Form einer Tabelle (HORS et al. 2004, What the Document Object Model is)

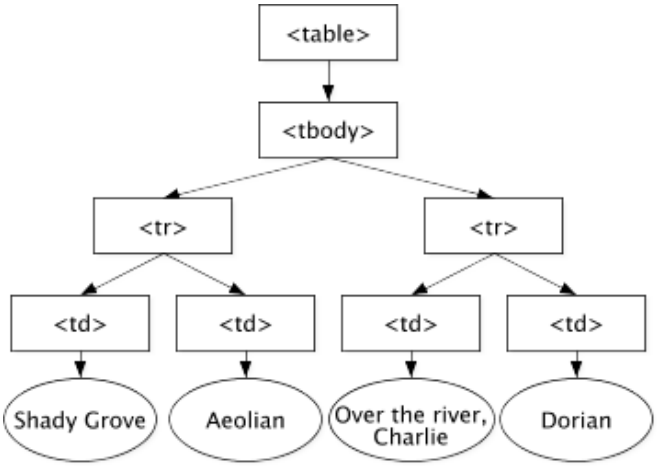


Abbildung 42 Grafische Darstellung des DOM am Beispiel der Tabelle aus Listing 18 (HORS et al. 2004, What the Document Object Model is)

Bezüglich des Ausdrucks BOM führt ZAKAS (2005, S. 9) aus, dass es mit seiner Hilfe möglich ist auf das Browserfenster zuzugreifen und es zu manipulieren (z. B. Erzeugung neuer Browserfenster, Festlegung der Größe des Browserfensters, Zugriff auf das „*navigator object*“ mit der Möglichkeit Informationen über den Browser abzugreifen, ZAKAS 2005, S. 9, S. 136 ff). Allerdings macht er Folgendes deutlich (S. 9): „*Because no standards exist for the BOM, each browser has its own implementation.*“ Trotzdem existieren nach seinen Ausführungen Gemeinsamkeiten bei der Ansprache des BOM in verschiedenen Browsern (S. 9).

Allgemein bietet JavaScript die Möglichkeit HTML und CSS in ihren Funktionalitäten zu erweitern (WIKIPEDIA-JAVASCRIPT 2013, bezüglich dynamische Veränderungen und Dynamic HTML (DHTML) siehe z. B. GOODMAN 2001, Foreword, GOODMAN 1998, S. 3 ff, MÜNZ 2003, S. 11 ff) und erlaubt eine komplexe Verarbeitung von XML mit XSLT (ZAKAS 2005, S. 445 ff, FLANAGAN 2006, S. 502 ff, siehe auch Kapitel 3.6.1 Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript). Es können z. B. XML-Dateien geladen und zusammen mit Parameterwerten an den XSLT-Prozessor des Browsers übergeben werden. Das Ergebnis wird wieder entgegengenommen und kann am Bildschirm dargestellt werden.

Um JavaScript-Anweisungen nutzen zu können, werden Sie meist über das „SCRIPT“-Element im „HEAD“- oder „BODY“-Bereich eines HTML-Dokumentes eingebunden (RAGGETT et al. 1999, 18 Scripts). Der JavaScript-Quelltext kann dabei direkt im HTML-Dokument oder über die Referenzierung einer externen JavaScript-Datei (Dateiendung „.js“) angegeben werden. Zusätzlich lassen sich JavaScript-Anweisungen auch direkt über Ereignisse in verschiedenen HTML-

Elementen etablieren (RAGGETT et al. 1999, 18 Scripts). Folgendes Beispiel (Listing 19) zeigt die verschiedenen Varianten der JavaScript-Einbindung („HEAD“-Bereich Zeile 7-9, „BODY“-Bereich Zeile 13-15, externe Datei Zeile 10, „onclick“-Ereignis Zeile 16). An der Stelle, an der die „alert“-Methode bei der Abarbeitung des Quelltextes durch den Interpreter auftaucht (Zeile 8), wird die Abarbeitung gestoppt, eine Meldefenster mit dem Ausgabeinhalt angezeigt und erst nach einer Bestätigung durch den Anwender wieder weiter fortgeführt. In diesem Meldefenster lassen sich z. B. Werte von Variablen für die Fehlersuche anzeigen (GOODMAN 2001, S. 410 f). In Zeile 10 werden JavaScript-Anweisungen aus der externen JavaScript-Datei „myscript.js“ geladen. In Zeile 14 können JavaScript-Anweisungen im BODY-Bereich der HTML-Datei aufgeführt werden. Zeile 16 bietet eine Schaltfläche mit der Aufschrift „Click“ (BUTTON-Element), die bei ihrer Betätigung die dort angegebenen JavaScript-Anweisungen ausführt.

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
2  "http://www.w3.org/TR/html4/strict.dtd">
3  <HTML>
4  <HEAD>
5   <TITLE>A document with SCRIPT</TITLE>
6   <META http-equiv="Content-Script-Type" content="text/javascript">
7   <SCRIPT type="text/javascript">
8    alert("Text");
9   </SCRIPT>
10   <SCRIPT type="text/javascript" src="myscript.js"> </SCRIPT>
11   </HEAD>
12   <BODY>
13    <SCRIPT type="text/javascript">
14     …some JavaScript…
15    </SCRIPT>
16    <button type="button" onclick="…some JavaScript…">Click</button>
17  </BODY>
18 </HTML>
```

Listing 19: Quelltext eines HTML4-Dokumentes mit den verschiedenen Varianten JavaScript-Anweisungen einzubinden (geändert nach RAGGETT et al. 1999, 18.2.2 Specifying the scripting language)

Zur Erleichterung der Lösung häufig vorkommender, insbesondere einheitlicher browserübergreifender Aufgabenstellungen sind unterschiedliche JavaScript-Bibliotheken („*Sammlung von JavaScript-Funktionen*“, siehe WIKIPEDIA-

JAVASCRIPT 2013, JavaScript-Bibliotheken) wie z. B. „Dojo Toolkit“, „Ext JS“, „jQuery“, „MooTools“, „Prototype“, „Qooxdoo“, „Yahoo User Interface Library“ und „Script.aculo.us“ entstanden (WIKIPEDIA-JAVASCRIPT 2013, JavaScript-Bibliotheken, siehe auch Kapitel 3.6.1 Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript). Die Ausdrücke JavaScript-Bibliothek und JavaScript-Toolkit werden dabei von WIKIPEDIA-JAVASCRIPT (2013, JavaScript-Bibliotheken) inhaltlich gleichgestellt. Zu JavaScript-Frameworks wird festgehalten: „*Toolkits, die nicht nur häufig benutzte Standardfunktionen zur Verfügung stellen, sondern durch ein besonderes Maß an Abstraktion eine grundlegend andere Programmierung nach sich ziehen, werden auch Frameworks genannt.*“ MÜNZ & GULL (2012, S. 26) beschreiben JavaScript-Frameworks wie folgt: „*Das sind Code-Bibliotheken, die für alle wichtigen und häufig verlangten Aufgaben fertige Funktionen bereitstellen und den Entwickler von Browser-Unterschieden fernhalten.*“

Im Folgenden wird auf die clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript eingegangen (insbesondere dazugehörige JavaScript-Frameworks).

3.6.1 Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript

Die meisten Browser (Client-Software) können auch ohne Server-Software und Verbindung in das Internet XML-Dateien mit Hilfe von XSLT und JavaScript verarbeiten (siehe z. B. LEISEGANG & MINTERT 2006, S. 156 ff, ZAKAS 2005, S. 445). Dabei werden z. B. über JavaScript-Anweisungen die benötigten XML- und XSLT-Dateien im Browser geladen, der XSLT-Prozessor (inklusive Parameter) gestartet, die Transformation nach HTML durchgeführt und das HTML-Ergebnis im Browser dargestellt.

STEYER (2007, S. 58) beschreibt, dass sich etwa ab 1995 verschiedene clientseitige Programmiertechniken (wie z. B. die Skriptsprache JavaScript) entwickelten, „*um der mangelnden Performance von Webandwendungen bei gleichzeitiger Belastung der Kommunikationswege und des Servers sowie dem Brachliegen der Client-Ressourcen zu begegnen*“. Er weist allerdings auch auf Sicherheitsbedenken und die Schwierigkeit hin, dass das Skript auch wirklich auf dem jeweiligen Client läuft, da z. B. „*einige proprietäre Standards*“ existieren, die „*nicht in allen Webbrowsern implementiert wurden*“. Auch BEHME & MINTERT (2000, S. 46 ff) gehen auf die clientseitige Verarbeitung von XML-Dateien ein und erklären (S. 55) „*wie ein XML-Dokument zu gebrauchen ist: Entweder serverseitig, mit geeigneter Konvertierung bzw. Formatierung, oder clientseitig, was einen XML-fähigen Browser voraussetzt*“ (siehe auch SCHREYER, M. 2002, S. 29).

ZAKAS (2005, S. 445, siehe auch ZAKAS et al. 2006, S. 107 ff, FLANAGAN 2006, S. 512 ff) erläutert, dass die beiden populärsten Browser „Internet Explorer and Mozilla“ auf unterschiedliche Art und Weise eine clientseitige XML-Manipulation unterstützen („*support client-side XML manipulation*“). Er beschreibt in diesem Zusammenhang die für den „Microsoft Internet Explorer“ spezifische Klasse „ActiveXObject“ mit ihrem „XML-DOM“-Objekt und der „load“-Methode zum Laden von XML- bzw. XSLT-Dateien (siehe auch MSDN-DOM-REFERENCE 2014). Die XML- und die XSLT-Datei werden dabei in separate „XML-DOM“-Objekte geladen (Listing 20, Listing 21, Zeile 1 bis 4), über die „transformNode“-Methode verarbeitet (Listing 21, Zeile 5) und das Transformationsergebnis als „String“ in einer Variablen gespeichert (Listing 21, Zeile 5). Der Ladevorgang von Dateien in „XML-DOM“-Objekte kennt dabei zwei Modi synchron und asynchron. Lädt man die Datei synchron (Listing 21, Zeile 1 und 2), wird die Abarbeitung der im Quelltext folgenden JavaScript-Anweisungen so lange gestoppt, bis die Datei fertig geladen ist. Lädt man die Datei asynchron, wird die Abarbeitung der im Quelltext folgenden JavaScript-Anweisungen nicht gestoppt und es müsste zur Fehlervermeidung z. B. über eine Ereignisbehandlungsroutine vor dem Zugriff auf die Datei überprüft werden, ob sie bereits vollständig geladen ist (ZAKAS 2005, S. 447, siehe auch MSDN-ASYNC 2014, siehe auch „event handler“ mit dem Namen „onreadystatechange“ im Kapitel 3.6.2 Ajax). Ab Version 5 des Browsers „Microsoft Internet Explorer“ wird das dazu benötigte „ActiveX-based library MSXML“ mit ausgeliefert (ZAKAS 2005, S. 446, siehe auch MSDN-MSXML3 2000, XSLT 1.0).

```
1 var oXmlDom = new ActiveXObject("Microsoft.XmlDom");
2 var oXslDom = new ActiveXObject("Microsoft.XmlDom");
```

Listing 20: Erzeugen zweier „Microsoft.XmlDom“-Objekte als Behälter für ein XML- und ein XSLT-Dokument (nach ZAKAS 2005, S. 445, S. 473, vergleiche auch MICROSOFT-CREATEPROCESSOR-METHOD 2013)

```
1 oXmlDom.async = false
2 oXslDom.async = false
3 oXmlDom.load("employees.xml");
4 oXslDom.load("employees.xsl");
5 var sResult = oXmlDom.transformNode(oXslDom);
```

Listing 21: Synchrones Laden einer XML- und einer XSLT-Datei in die „Microsoft.XmlDom“-Objekte und Transformation über die „transformNode“-Methode (nach ZAKAS 2005, S. 473, vergleiche auch MICROSOFT-CREATEPROCESSOR-METHOD 2013)

ZAKAS et al. (2006, S. 143) weisen bezüglich des Umgangs mit XSLT-Dateien darauf hin, dass der „Microsoft Internet Explorer“-Browser ab Version 6 eine zweite, aufwendigere, aber leistungsfähigere Möglichkeit bietet XML-Dateien mit XSLT und JavaScript zu verarbeiten. Mit der Browser-Version 6 steht das dazu notwendige „MSXML 3.0“ automatisch zur Verfügung (ZAKAS 2005, S. 473). Zusätzlich unterstützt „MSXML 3.0“ komplett XSLT der Version 1.0 (MSDN-MSXML3 2000, XSL versus XSLT 1.0, ZAKAS 2005, S. 473). Bei dieser zweiten Vorgehensweise wird die XML-Datei wie bei der vorherigen Variante in einem separaten „XML-DOM“-Objekt abgelegt (Listing 20, Zeile 1, Listing 21, Zeile 1, 3). Die XSLT-Datei hingegen wird in ein zu erzeugendes „FreeThreadedDOMDocument“-Objekt synchron geladen (Listing 22) und anschließend in einem „XSLTemplate“-Objekt zwischengespeichert (Listing 23). Damit kann schnell auf das kompilierte, zwischengespeicherte „XSL-Stylesheet“ zurückgegriffen werden (ZAKAS et al. 2006, S. 144).

```
1 var oXslDom =
2  new ActiveXObject("Msxml2.FreeThreadedDOMDocument.3.0");
3 oXslDom.async = false;
4 oXslDom.load("books.xsl");
```

Listing 22: Synchrones Laden einer XSLT-Datei in ein „FreeThreadedDOMDocument“-Objekt (ZAKAS et al. 2006, S. 143, siehe auch MICROSOFT-CREATEPROCESSOR-METHOD 2013)

```
1 var oXslTemplate =
2  new ActiveXObject("Msxml2.XSLTemplate.3.0");
3 oXslTemplate.stylesheet = oXslDom;
```

Listing 23: Zwischenspeichern einer XSLT-Datei in ein „XSLTemplate“-Objekt (ZAKAS et al. 2006, S. 144, siehe auch MICROSOFT-CREATEPROCESSOR-METHOD 2013)

Das „XML-DOM“-Objekt (mit dem XML-Dokument) kann zur weiteren Verarbeitung an das „XSL-Processor”-Objekt (mit dem XSLT-Dokument) übergeben werden (Listing 24).

```
1 var oXslProcessor = oXslTemplate.createProcessor();
2 oXslProcessor.input = oXmlDom;
```

Listing 24: Aufruf der „createProcessor“-Methode des „XSLTemplate“-Objektes und Übergabe des XML-Dokumentes über die input-Eigenschaft des „XSLProcessor”-Objektes (ZAKAS et al. 2006, S. 144, siehe auch MICROSOFT-CREATEPROCESSOR-METHOD 2013)

Darüber hinaus lassen sich mit Hilfe des „XSLProcessor”-Objektes über die “addParameter“-Methode Parameterwerte an das “*XSL-Stylesheet*” übergeben (Listing 25).

```
1 oXslProcessor.addParameter("message","Meine Bücherliste");
```

Listing 25: Übergabe eines Wertes an den Parameter „message” im “*XSL-Stylesheet*” (ZAKAS et al. 2006, S. 147, siehe auch MICROSOFT-CREATEPROCESSOR-METHOD 2013)

Die XSLT-Verarbeitung kann anschließend über die “transform”-Methode des „XSLProcessor”-Objektes angestoßen werden (Listing 26, Zeile 1). Das Ergebnis lässt sich abschließend aus der “output”-Eigenschaft des „XSLProcessor“-Objektes auslesen (Listing 26, Zeile 2).

```
1 oXslProcessor.transform();
2 document.body.innerHTML = oXslProcessor.output;
```

Listing 26: Aufruf der „transform“-Methode des „XSLProcessor“-Objektes und Ausgabe des Ergebnisses (ZAKAS et al. 2006, S. 145, MICROSOFT-CREATEPROCESSOR-METHOD 2013)

Bei „Mozilla“-Browser (z. B. „Mozilla Firefox“) erklärt ZAKAS (2005, S. 450 ff), muss anders als bei „Microsoft Internet Explorer“-Browser bezüglich der clientseitigen Verarbeitung von XML-Dateien vorgegangen werden. In diesem Fall werden über die „createDocument“-Methode „XML-DOM“-Objekte erstellt (Listing 27, Zeile 1 und 2), über die „load“-Methode die XML- und XSLT-Datei synchron geladen (Listing 28, Zeile 1 und 2, ab Mozilla 1.4 ist ein synchrones Laden möglich, siehe ZAKAS 2005, S. 451) und die XSLT-Datei in ein „XSLTProcessor“-Objekt zur Verarbeitung importiert (Listing 29).

```
1 var oXmlDom = document.implementation.createDocument("","",null);
2 var oXslDom = document.implementation.createDocument("","",null);
```

Listing 27: Erzeugen zweier „XML-DOM“-Objekte als Behälter für ein XML- und ein XSLT-Dokument (nach ZAKAS 2005, S. 450, vergleiche auch HEARN 2005)

```
1 oXmlDom.async = false
2 oXslDom.async = false
3 oXmlDom.load("employees.xml");
4 oXslDom.load("employees.xsl");
```

Listing 28: Synchrones Laden einer XML- und einer XSLT-Datei (nach ZAKAS 2005, S. 451, siehe auch ZAKAS et al. 2006, S. 148, vergleiche auch HEARN 2005)

```
1 var oProcessor = new XSLTProcessor();
2 oProcessor.importStylesheet(oXslDom);
```

Listing 29: Aufruf der „XSLTProcessor"-Methode und Import des XSLT-Dokumentes (ZAKAS 2005, S. 477)

Zusätzlich lassen sich mit Hilfe des „XSLTProcessor"-Objektes über die "setParameter"-Methode Parameterwerte an das „*XSL-Stylesheet*" übergeben (Listing 30, Zeile 1). Die abschließende XSLT-Verarbeitung kann z. B. über die "transformToDocument"-Methode des „XSLTProcessor"-Objektes angestoßen (Listing 30, Zeile 2) werden. Das Ergebnis lässt sich als DOM-Objekt in einer Variablen speichern (Listing 30, Zeile 2), weiterverarbeiten bzw. ausgeben.

```
1 oProcessor.setParameter(null,"message","Hello World!");
2 var oResultDom = oProcessor.transformToDocument(oXmlDom);
```

Listing 30: Parameter-Übergabe an das „XSLTProcessor"-Objekt und Ablage des Transformationsergebnisses in das DOM-Objekt mit dem Variablennamen „oResultDom" (nach ZAKAS 2005, S. 479)

Im Jahr 2005 tauchte in diesem fachlichen Zusammenhang der Ausdruck Ajax auf.

3.6.2 Ajax

Der Ausdruck Ajax – „*Asynchronous JavaScript and XML*" – wird erstmalig in einem Artikel von GARRETT (2005) beschrieben. Er hält fest, dass es sich bei Ajax um keine einzelne Technologie, sondern um verschiedene bereits vorhandene Technologien handelt (siehe auch WENZ 2010, S. 7, WENZ 2008, S. 391, GAMPERL 2007, S. 11, CRANE & PASCARELLO 2006, S. xix, ZAKAS et al. 2006, S. 25, WIKIPEDIA-JAX-EN 2013):

„*Ajax incorporates:*

- *standards-based presentation using XHTML and CSS;*
- *dynamic display and interaction using the Document Object Model;*
- *data interchange and manipulation using XML and XSLT;*

- *asynchronous data retrieval using XMLHttpRequest;*
- *and JavaScript binding everything together.*"

GARRETT (2005) beschreibt eine Ajax-Anwendung als etwas, das die „*start-stop-start-stop nature*" der Web-Interaktionen durch das Dazwischenschalten einer „*Ajax engine*" zwischen Anwender und Server beendet (Abbildung 43). Anstatt eine Webseite zu laden (linke Seite Abbildung 43), lädt der Browser eine in JavaScript geschriebene „*Ajax engine*" (rechte Seite Abbildung 43). Diese JavaScript-Anweisungen sind für die Darstellung im Browser und die Interaktion mit dem Server gemäß des Verhaltens des Anwenders verantwortlich. Sie erlauben eine asynchrone Interaktion des Anwenders mit der Ajax-Anwendung, unabhängig von der Kommunikation mit dem Server. So können im Hintergrund z. B. bereits neue Daten vom Server geholt werden, während der Anwender mit der Anwendung interagiert (z. B. Vorladen des an den sichtbaren Bereich angrenzenden Kartenmaterials in der Ajax-Anwendung „Google Maps", BERGMANN & BORMANN 2005, S. 15 ff, PFEIFFER 2006, S. 13, ZAKAS et al. 2006, S. 70, „Predictive Fetch"). Ein Neuladen der Webseite ist nicht notwendig (ZAKAS et al. 2006, S. 28). Damit reduziert sich die Zeitdauer des Ladens einer Webseite und der Anwender kann schneller auf die gewünschten Inhalte zugreifen.

Abbildung 43 zeigt im Vergleich links das traditionelle und rechts das Ajax-„*web application model*".

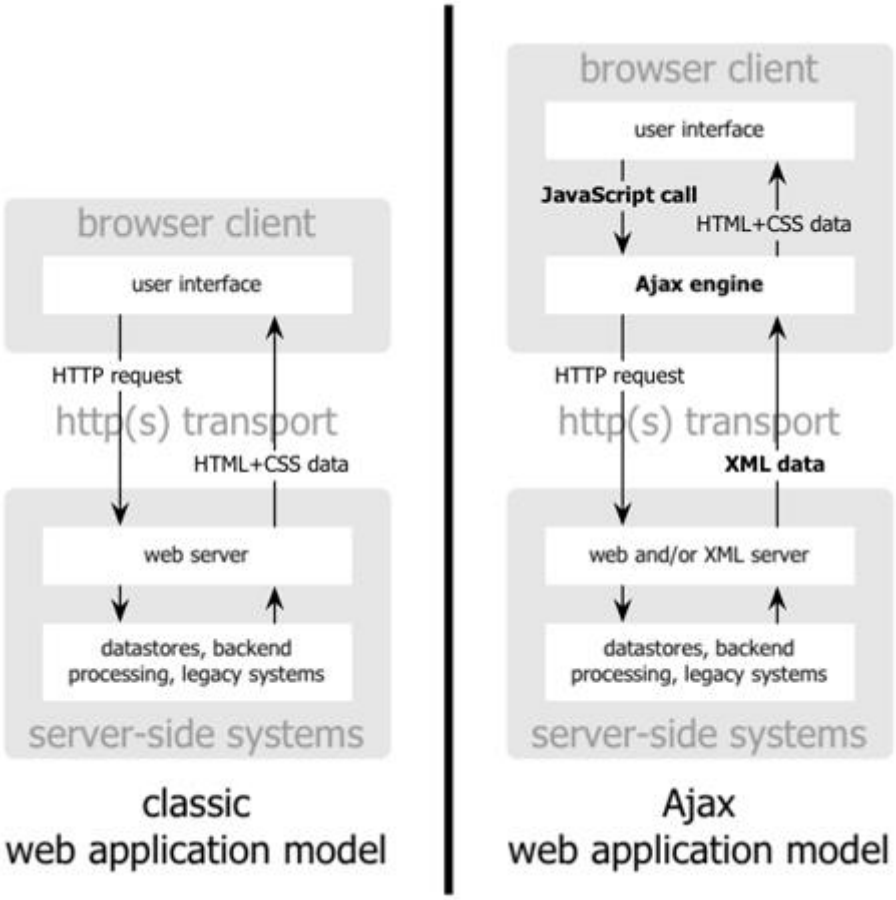


Abbildung 43 Links „*classic web application model*“, rechts „*Ajax web application model*“ (GARRETT 2005)

WANG (2005, S. 156) erläutert, dass im Ajax-Modell zuerst eine „*Initialseite aus HTML, CSS und JavaScript*“ im Browser geladen, aufgrund einer Benutzeraktion bzw. eines Ereignisses (abgefangen durch einen „EventListener“, auch „event handler“ im Folgenden Event-Handler, siehe WIKIPEDIA-EREIGNIS 2013) eine JavaScript-Funktion aufgerufen, darüber ein „*HTTP Request asynchron an den Server*“ geschickt (während der Benutzer weiter arbeitet), das Ergebnis entgegengenommen, über das DOM ausgelesen und ohne einen neuen Seitenaufbau im Browser angezeigt wird. Er weist zusätzlich darauf hin (siehe auch GAMPERL 2007, S. 178): „*Kern von Ajax ist der Einsatz von XMLHttpRequest, das eine versteckte und asynchrone Kommunikation mit dem Server ermöglicht*“. Ähnlich äußert sich WENZ (2010, S. 14): „*Grundpfeiler von Ajax ist das bereits angesprochene XMLHttpRequest-Objekt, das den Datenaustausch mit einem Webserver ermöglicht, jedoch ganz ohne den gefürchteten Page Refresh (dem Neuladen und vor allem Neuaufbau der Seite).*“ Weiter erklärt er (S. 37), dass über JavaScript unter anderem drei Möglichkeiten bestehen, die über das „XMLHttpRequest“-Objekt gelieferten XML-Daten weiter zu verarbeiten. Er beschreibt dabei (S. 37 ff) den „*direkten DOM-Zugriff auf die die XML-Daten*“, „*XPath-Suchen in*

den Daten“ und „*XSLT-Umwandlungen der Daten“* und betont den Vorteil dieser einfacheren clientseitigen Verarbeitung strukturierter Daten im Vergleich zur Verarbeitung servergelieferter „*simpler String*“-Daten. Auch das W3C greift die Bedeutung des „XMLHttpRequest“-Objektes auf. Inzwischen liegt eine „*XMLHttpRequest specification*“ als „*W3C Working Draft*“ vor (AUBOURG et al. 2012).

Nach Ausführungen von WANG (2005, S. 156) wurde das „XMLHttpRequest“-Objekt zuerst im Microsoft Internet Explorer 5.0 als „ActiveX“-Objekt eingebunden und erst anschließend von anderen Browsern (z. B. Firefox, Safari, Opera) integriert (siehe auch GAMPERL, 2007, S. 178 f, WENZ 2010, S. 14 ff., WENZ 2008, S. 394). WIKIPEDIA-AJAX-DE (2013, Status/Verbreitung, siehe auch WENZ 2008, S. 395) hält ausführlicher fest, welche Browser unter anderem Ajax unterstützen: „*Microsoft Internet Explorer (ab Version 5.0), Mozilla Firefox (ab Version 1.0), Opera (ab Version 8.0), Apple Safari (ab Version 1.2), Google Chrome (ab Version 0.2).*“

Die folgenden Quelltexte Listing 31, Listing 32 und Listing 33 (zur allgemeinen Erläuterung der JavaScript-Anweisungen siehe auch MSDN-XMLHTTPREQUEST 2014) zeigen den Einsatz von Ajax mit XSLT mit einer „*Fallunterscheidung zwischen den beiden Fronten Internet Explorer und Rest der Welt*“ nach WENZ (2010, S. 56 f). Werden alle drei JavaScript-Blöcke in eine HTML-Datei eingebunden (siehe Kapitel 3.6 JavaScript), ermöglichen ihre Anweisungen eine XML-Datei mit Hilfe einer XSLT-Datei zu transformieren und das Transformationsergebnis im Browser anzuzeigen.

Dazu erzeugen zunächst die JavaScript-Befehle in Listing 31 das „XMLHttpRequest“-Objekt. In Zeile 2 wird zuerst in einer „Browserweiche“ geprüft, ob der Browser die Klasse „window.XMLHttpRequest“ kennt (für andere Browser als den „Microsoft Internet Explorer“ unter Version 7). Ist dies der Fall, erstellt der Quelltext in Zeile 3 eine Instanz des „XMLHttpRequest“-Objektes. Seit der Version 7 unterstützt auch der „Microsoft Internet Explorer“ dieses Objekt „nativ“ ohne „ActiveX“ (WENZ 2010, S. 16, siehe auch MSDN-XMLHTTPREQUEST 2014). In Zeile 4 bis 13 wird für die „Microsoft Internet Explorer“-Browser unter Version 7 eine Instanz des „XMLHttpRequest“-Objektes in Form eines „ActiveXObject“ erzeugt.

```
1 var XMLHTTP = null;
2 if (window.XMLHttpRequest) {
3   XMLHTTP = new XMLHttpRequest();
```

```
4 } else if (window.ActiveXObject) {
5   try {
6     XMLHTTP = new ActiveXObject("Msxml2.XMLHTTP");
7   } catch (ex) {
8     try {
9       XMLHTTP = new ActiveXObject("Microsoft.XMLHTTP");
10     } catch (ex) {
11     }
12   }
13 }
```

Listing 31: Ajax-Beispiel, Erzeugung eines „XMLHttpRequest“-Objektes, JavaScript-Quelltext nach WENZ (2010, S. 18 f)

Der „Event-Handler“ „onload“ in Listing 32 Zeile 1 stellt sicher, dass die folgende Funktion mit ihrer HTTP-Anfrage erst dann abgesetzt wird, wenn die HTML-Seite (in der sich am Schluss alle JavaScript-Blöcke aus Listing 31, Listing 32 und Listing 33 befinden) komplett geladen wurde. Auf diese Art und Weise soll sichergestellt werden, dass vorherige Anweisungen nicht ins Leere laufen, wenn z. B. die Antwort vom Server schneller eintrifft als das Laden und Verarbeiten der HTML-Seite im Browser (WENZ 2010, S. 25). Die Methode „open“ des „XMLHttpRequest“-Objektes (Listing 31) erstellt in Listing 32 Zeile 2 eine „HTTP-Anfrage“. Mit dem ersten Parameter „GET“ (häufigste eingesetzte „HTTP-Methode“, siehe GAMPERL 2007, S. 183) gibt sie an, wie die Anfrage erstellt werden soll und mit dem zweiten Parameter, welche URL die XML-Datei aufweist („daten.xml“). Der dritte Parameter legt fest, ob die Kommunikation asynchron oder synchron verlaufen soll. Fehlt er (wie im vorliegenden Beispiel), so wird von einer asynchronen Kommunikation ausgegangen und ein „Event-Handler“ „onreadystatechange“ eingesetzt, um feststellen zu können, ob die Kommunikation abgeschlossen ist (MSDN-ASYNC 2014). Der „Event-Handler“ „onreadystatechange“ (Zeile 3) des „XMLHttpRequest“-Objektes wird angestoßen, wenn sich der Status der „HTTP-Anfrage“ ändert (genauer die „readyState“-Eigenschaft des „XMLHttpRequest“-Objektes). Bei jeder Veränderung wird die „DatenAusgeben“-Funktion aufgerufen (Zeile 3). Die Methode „send“ des „XMLHttpRequest“-Objektes (Zeile 4) stellt schließlich sicher, dass die „HTTP-Anfrage“ an den Server geschickt und die Antwort entgegengenommen wird.

```
1 window.onload = function() {
2   XMLHTTP.open("GET", "daten.xml");
3   XMLHTTP.onreadystatechange = DatenAusgeben;
4   XMLHTTP.send(null);
5 }
```

Listing 32: Ajax-Beispiel, nach kompletten Laden der HTML-Datei („Event-Handler“ „onload“) Erstellung einer „HTTP-Anfrage“ und Aufruf der JavaScript-Funktion „DatenAusgeben“, JavaScript-Quelltext nach WENZ (2010, S. 56 f)

Die durch den bereits angesprochenen „Event-Handler“ „onreadystatechange“ angestoßene „DatenAusgeben“-Funktion in Listing 33 prüft zuerst, ob der Wert der „readyState“-Eigenschaft 4 erreicht hat (Zeile 2). Der Wert 4 bedeutet, dass die „HTTP-Anfrage“ vollständig gesendet wurde und ein Ergebnis vorliegt (WENZ 2010, S. 22, siehe auch MSDN-XMLHTTPREQUEST 2014). Ist dieser Wert erreicht, nimmt die „responseXML“-Eigenschaft des erstellten „XML-DOM“-Objektes das Ergebnis der „HTTP-Anfrage“ entgegen (XML-Datei „daten.xml“) und legt es in der Variablen „xml“ ab (Zeile 3). Über eine Browserweiche (Zeile 4 für „Mozilla-Browser“ und „Opera“ Version 9, Zeile 13 für andere Browser wie „Microsoft Internet Explorer“, WENZ 2010, S. 54 f) wird anschließend die Verarbeitung mit unterschiedlichen JavaScript-Anweisungen fortgeführt.

Die Zeilen 5 bis 12 spiegeln die in Kapitel 3.6.1 (Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript) bereits beschriebene Variante für „Mozilla“-Browser wieder. Allerdings wird anstatt der „transformToDocument”-Methode des „XSLTProcessor”-Objektes die „transformToFragment”-Methode eingesetzt (Zeile 11). Die „transformToFragment”-Methode nimmt als Parameter die zu transformierenden XML-Daten entgegen und das Dokument, das das Transformationsergebnis aufnehmen soll. Die „appendChild“-Methode fügt letztlich das Transformationsergebnis (Fragment) in das im Browser-Fenster sichtbare Dokument ein (Zeile 12).

Die Zeilen 14 bis 25 geben die in Kapitel 3.6.1 (Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript) bereits beschriebene Variante für „Microsoft Internet Explorer“-Browser ab Version 6 wieder. Für die Ausgabe wird allerdings ein „div“-Element erzeugt (Zeile 23), in das das Transformationsergebnis eingehängt wird (Zeile 24). Das Endergebnis erscheint in dem im Browser-Fenster sichtbaren Dokument.

```
1 function DatenAusgeben() {
2   if (XMLHTTP.readyState == 4) {
3     var xml = XMLHTTP.responseXML;
4     if (window.XSLTProcessor) {
5      var xslt =
6      document.implementation.createDocument("", "", null);
7       xslt.async = false;
8       xslt.load("daten.xsl");
9       var process = new XSLTProcessor();
10       process.importStylesheet(xslt);
11       var ergebnis = process.transformToFragment(xml, document);
12       document.body.appendChild(ergebnis);
13     } else {
14       var xslt =
15       new ActiveXObject("MSXML2.FreeThreadedDOMDocument");
16       xslt.async = false;
17       xslt.load("daten.xsl");
18       var template = new ActiveXObject("MSXML2.XSLTemplate");
19       template.stylesheet = xslt;
20       var process = template.createProcessor();
21       process.input = xml;
22       process.transform();
23       var d = document.createElement("div");
24       d.innerHTML = process.output;
25       document.body.appendChild(d);
26     }
27   }
28 }
```

Listing 33: Ajax-Beispiel, JavaScript-Funktion „DatenAusgeben()“, Transformation einer XML-Datei mit Hilfe einer XSLT-Datei und Anzeige des Transformationsergebnisses im Browser, JavaScript-Quelltext nach WENZ (2010, S. 56 f)

Das Beispiel funktioniert im „Microsoft Internet Explorer“ nur (einschließlich der derzeitigen Version 11), wenn die Beispiel-HTML-Datei von einem Server geladen wird. Der Grund liegt darin, dass das „XMLHTTP“-Objekt Server-Informationen erwartet (MIME-Typ, „Multipurpose Internet Mail Extensions“), ob es sich bei der angeforderten Datei um eine XML-Datei handelt (ZAKAS 2005, S. 494).

Um den Einsatz von Ajax, trotz der unterschiedlichen Implementierungen der Ajax-Technologien in den verschiedenen Browsern, zu erleichtern, entstanden verschiedene clientseitige Ajax-Frameworks wie z. B. „jQuery“, „Prototype”, „Dojo Toolkit”, „Spry (Adobe)”, „Yahoo UI Library”, „Sarissa” (WIKIPEDIA-AJAX-FRAMEWORKS-EN 2013, siehe auch GAMPERL 2007, S. 233, LEISEGANG et al. 2007, S. 30 ff, SPANNEBERG & MINTERT 2007, S. 8 ff). Sie können als browserübergreifendes Fundament für weitere Entwicklungen herangezogen werden (HARNISCH 2013, S. 51). Neben clientseitigen existieren nach WENZ (2010, S. 101, siehe auch STIERAND S. 52 ff, LEISEGANG et al. 2007, S. 35 ff) auch serverseitige Ajax-Frameworks. Er erklärt dazu, dass es zwar serverseitige “*Ajax-Frameworks*” wie z. B. „Sajax” und „ASP.NET AJAX” gibt, sie aber eigentlich nicht notwendig sind, da es bei Ajax vor allem um das “*Absetzen von HTTP-Anforderungen an den Server und Auswertung der HTTP-Rückgabewerte*” (durch den Browser = Client-Software) geht. Auch STEYER (2007, S. 61) weist in diesem Sinne darauf hin: „*AJAX fordert zwar im Client bestimmte Techniken, jedoch … keine explizit einzuhaltenden Voraussetzungen auf dem Webserver.*“

WENZ 2010 (S. 59 ff, siehe auch HARNISCH 2013, S. 50) erläutert, dass sich nicht nur über XML, sondern auch über „JSON“ („JavaScript Object Notation“) Objektdaten als String an eine JavaScript-Anwendung übergeben lassen. Er legt dar (WENZ 2010, S. 59 f): „*Viele Ajax-Anwendungen und –Frameworks setzen intern auf JSON, da es etwas einfacher zu handhaben ist als das im vorherigen Kapitel gezeigte XML*“.

Das folgende Kapitel verlässt den Themenbereich der Programmierung und wendet sich der Abbildung von Metadaten mit XML über IEEE LOM zu.

3.7 IEEE LOM

Wie bereits in Kapitel 3.3.7.2 IMS MD angesprochen, löst der „*IEEE Standard for Learning Object Metadata*“ (IEEE LOM, siehe IMS-MD 2013) die Spezifikation IMS MD zur Abbildung von Metadaten für Lernobjekte ab (inklusive E-Learning-Inhalte). Dieser „IEEE-Standard“ ist in der Veröffentlichung „*IEEE Std 1484.12.1 – 2002,*

(Final) Draft Standard für Learning Object Metadata“ beschrieben (HODGINS & DUVAL 2002). HODGINS (2005, siehe auch HODGINS & DUVAL 2002, S. 5) hält bezüglich dieses „Standards“ Folgendes fest: „*This standard will specify the syntax and semantics of Learning Object Metadata, defined as the attributes required to fully/adequately describe a Learning Object.*“ Ähnlich beschreibt SCHREYER (2002, S. 55) den Ausdruck Metadaten als „*strukturierte Beschreibungen von Informationsobjekten in Textform*“. MONTANDON (2004, S. 4) führt die folgenden Punkte auf, die in seinen Augen viele Definitionen des Begriffes Metadaten gemeinsam aufweisen: „*Wiedergabe von strukturierten Informationen über Objekte, Unterstützung und Vereinfachung der Identifikation von Objekten, Zweck eines Ordnungssystems zur Gewährleistung der Wiederauffindbarkeit von Objekten*“. Weiter definiert HODGINS (2005) im „Standard“ den Ausdruck Lernobjekte (LO, siehe auch HODGINS & DUVAL 2002, S. 5): „*Learning Objects are defined here as any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning.*“ Durch IEEE LOM wird das Lernobjekt selbst und nicht der Inhalt in seiner Struktur beschrieben. Dazu können z. B. XML-Anwendungen (siehe Kapitel 3.3.6 XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten) eingesetzt werden. Als Absicht des „Standards“ beschreiben HODGINS & DUVAL (2002, S. 5): „*The purpose of this multi-part Standard is to facilitate search, evaluation, acquisition, and use of learning objects, for instance by learners or instructors or automated software processes.*“ Ein weiteres Ziel ist die gemeinsame Nutzung und der Austausch von Lernobjekten. Die Lernobjekte können dabei mit vielfältigen Metadaten versehen werden. Beispielsweise können die Metadaten eines Online-Kurses festhalten, wer der Autor des Kurses ist, für welche Zielgruppe er gedacht ist usw. (siehe auch SANCHEZ-ALONSO et al. 2011, S. 146, KERRES). Die Mehrsprachigkeit wird in diesem Zusammenhang besonders betont.

SCHNEIDER (2005, S. 20 f, siehe auch SANCHEZ-ALONSO et al. 2011, S. 146) weist auf die Tatsache hin, dass IEEE LOM unter anderem auf der Grundlage eines gemeinsamen Standardisierungsvorschlages des ARIADNE-Projektes und der IMS-Initiative an das IEEE entstand. Auch HODGINS & DUVAL (2002, S. iii) betonen, dass der IEEE LOM-„Standard“ seine Wurzeln in ARIADNE, DUBLIN CORE und IMS Projekten hat. Die „Ariadne Foundation“ erklärt (ARIADNE-METADATA 2014): „*For describing learning content, we mainly rely on IEEE LTSC LOM but we support other standards like Dublin Core (DC) and ISO/IEC MLR as well, by automatically transforming metadata from one format into another.*“ „ISO/IEC MLR“ wird in ISO/IEC 19788 („*Information technology - Learning, education and training - Metadata for learning resources*“) beschrieben. BAILEY (2012) erwähnt zusätzlich weitere Metadaten-Initiativen wie z. B. „*Learning Resource Metadata Initiative*“ (LRMI) und

bemerkt: „*That is, IEEE LOM, ANZ-LOM, Dublin Core, MLR etc, against LRMI. Who knows how it will end?*“

Allgemein betont JUNGMANN (2012, S. 66), dass die Vergabe und Pflege von Metadaten Voraussetzung für die Wiederverwendung von E-Learning-Inhalten ist. Auch SÜSS (2004, S. 24, siehe auch KERRES 2012, S. 447) hebt hervor, dass die „*Voraussetzung für die kooperative, modulare Erstellung und flexible technische Wiederverwendung von eLearning-Inhalten*“ unter anderem ihre Beschreibung mit Metadaten ist. Nach SCHAFFERT & KALZ (2009, S. 17) gehört IEEE LOM (neben SCORM und IMS LD) zu den „*existierenden E-Learning-Standards*“ (siehe auch BRUGGER 2002, S. 1, SÜSS 2004, S. 25).

Zur Beschreibung von LOs werden die Metadaten in Gruppen eingeteilt. Folgende Aufzählung gibt die Gruppen mit einer kurzen Beschreibung an (HODGINS & DUVAL 2002, S. 6 f).

1. „*General*“: Ressource als Ganzes, Titel, Sprache, Beschreibung ...
2. „*Life Cycle*“: Version, Auflage, Mitwirkende …
3. „*Meta-Metadata*“: Metadaten über ein LO von einem Nicht-Autor …
4. „*Technical*“: Format, Größe, Installations- und Bedienungsvoraussetzungen …
5. „*Educational*“: Interaktivität, Schwierigkeitsgrad …
6. „*Rights*“: Urheberrecht, Lizenzrechte, Nutzungsbedingungen …
7. „*Relation*“: Beziehung zwischen LOs („is-part-of“, „is-basis-for“ ...)
8. „*Annotation*“: Besondere Bemerkungen zu einer LO
9. „*Classification*“: Einordnung in den Kontext eines Fachgebietes

Jede Gruppe kann wiederum unterteilt werden. Die unten aufgeführte Aufzählung gibt Informationen aus der Gruppe „General“ an (HODGINS & DUVAL 2002, S. 10 ff). Sie beschreiben eine Ressource als Ganzes.

„*General*“:

„*1.1 Identifier*“

„1.2 Title“

„1.3 Language“

„1.4 Description“

„1.5 Keywords“

„1.6 Coverage“ (Zeitalter, Kultur, Region auf das sich ein LO bezieht …)

„1.7 Structure“ (hierarchisch, linear (Navigation per weiter, zurück) …)

„1.8 Aggregation Level“ (atomar, Seite, Kurs …)

Abbildung 44 zeigt einen graphischen Überblick über die Gruppierung der Metadaten im IEEE LOM.

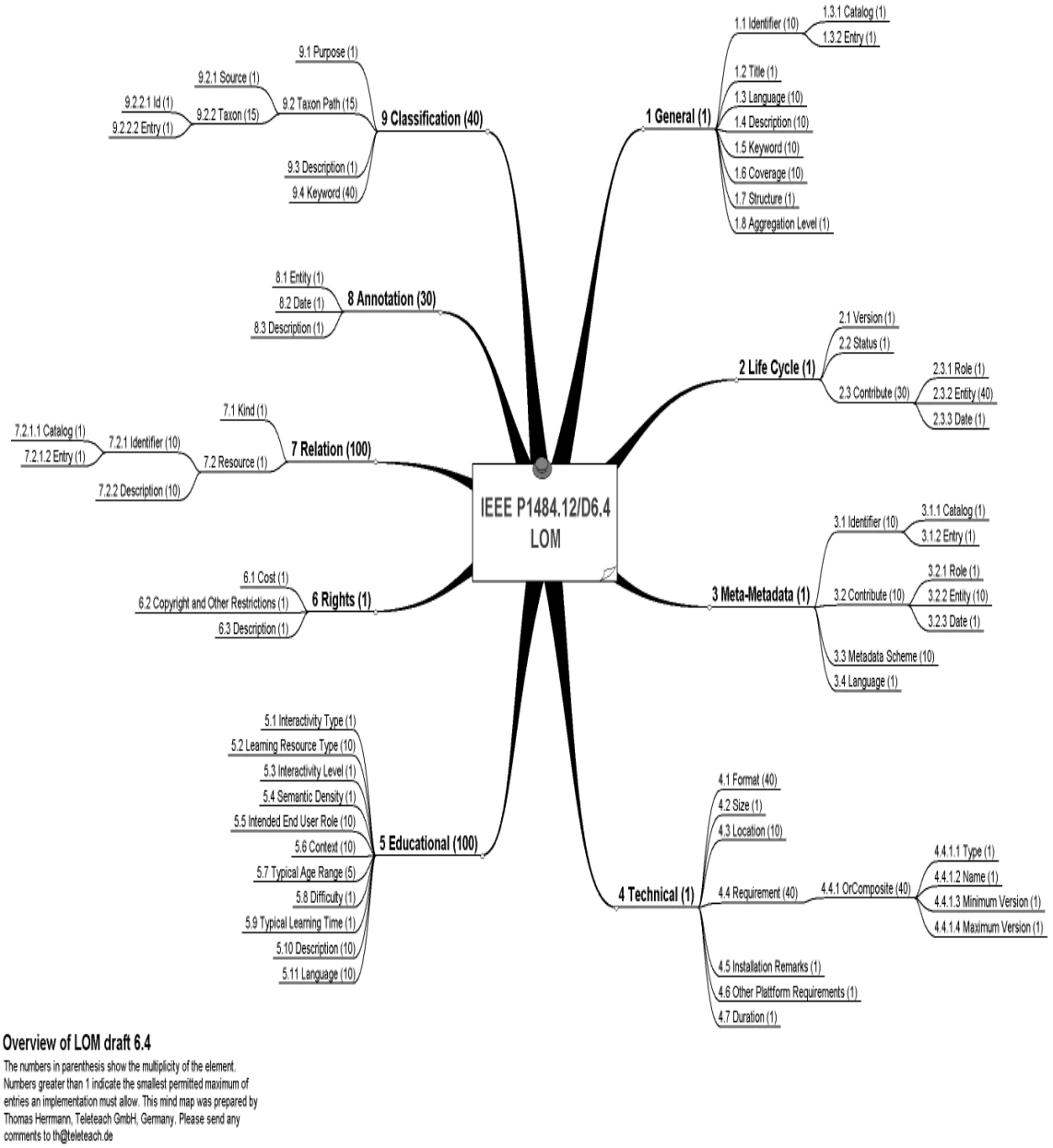


Abbildung 44 Graphischer Überblick über die Gruppierung der Metadaten im IEEE LOM (DUVAL 2002 a, p. 10)

Die Metadaten können in XML abgebildet werden (siehe auch WIKIPEDIA-LOM-EN 2013). Dieses „*XML Binding*“ wird in „*IEEE Std 1484.12.3-2005, IEEE Standard for Learning Technology -- Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata*“ beschrieben (HODGINS et al. 2004). Als Beispiel dazu zeigt Anhang B einen Ausschnitt eines Quelltextes eines IEEE LOM-XML-Dokumentes (General Gruppe).

Um IEEE LOM-XML-Dateien einfacher erstellen und mit Metadaten befüllen zu können, wurden „Metadaten-Editoren“ entwickelt (z. B. RELOAD 2008, RWTHAACHEN LOM-EDITOR (2005), siehe auch WIKIPEDIA-LOM-EN 2013). Mit ihrer Hilfe können die Metadaten einer LO verwaltet und z. B. in einer separaten XML-Datei abgespeichert werden. SCHNEIDER (2011 a, 2 History and version) weist darauf hin, dass die Metadaten nach dem „*IEEE LOM-Standard*“ (IEEE 1484.12.1) auch in SCORM (siehe folgendes Kapitel) verwendet werden können (SCORM Version 1.2 und SCORM Version 1.3 (entspricht SCORM 2004)). Die Initiative ADL (Herausgeber von SCORM) bemerkt dazu (JESUKIEWICZ 2009 a, S. CAM-3-22, CAM-4-65): „*If using metadata to describe SCORM Content Model Components, ADL highly recommends, at a minimum, the use of the IEEE LOM metadata scheme.*“

3.8 SCORM

Wie im vorherigen Kapitel erwähnt, gehört nach SCHAFFERT & KALZ (2009, S. 17) neben IEEE LOM auch SCORM zu den „*existierenden E-Learning-Standards*“ (siehe auch MONTANDON 2004, S. 13, BRUGGER 2002, S. 1, SÜSS 2004, S. 25). Ähnlich äußert sich JUNGMANN (2012, S. 96) zu SCORM: „*Erfahrungen von AICC, IMS und LOM fließen in diesen XML-basierten Standard ein*“ (siehe auch Abbildung 45, DODDS & THROPP 2006, S. SCORM-1-9 ff, GEEB 2003, S. 18). Auch nach KERRES (2012, S. 450) ist SCORM „*ein wichtiger Standard für den Austausch von gekapselten Lernobjekten zwischen verschiedenen Lernplattformen*“. SCORM dient damit nicht der Erstellung von Lerninhalten, wie es in Kapitel 3.3.6 XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten beschrieben wurde. ADLNET-SCORM (2014, Synopsis) erklärt: „*SCORM content can be delivered to your learners via any SCORM-conformant Learning Management System (LMS) using the same version of SCORM.*” Weiter hält die ADL-Initiative fest: „*This specification promotes reusability and interoperability of learning content across Learning Management Systems (LMSs).*“

Hinter der Entwicklung von SCORM steht die Initiative ADL. Die aktuellste SCORM-Version „*SCORM 2004 4th Edition*“ (SCORM 1.3, im Folgenden „SCORM2004“ genannt) stammt aus dem Jahr 2009 und wird derzeit überarbeitet („*minor update to SCORM 2004 4th Edition*“) (ADLNET-SCORM 2014, Synopsis). SCORM wurde 2009 auch als ISO-Standard „*Information technology - Sharable Content Object Reference Model (SCORM) 2004 3rd Edition*“ in vier Teilen veröffentlicht (ISO-29163 2009).

ADLNET-SCORM (2014, Overview) beschreibt SCORM wie folgt: „*The SCORM (Sharable Content Object Reference Model) is a collection and harmonization of specifications and standards that defines the interrelationship of content objects, data models and protocols such that objects are sharable across systems that conform to the same model.*“ Weiter wird festgehalten: “*This specification promotes reusability and interoperability of learning content across Learning Management Systems (LMSs).*“ Außerdem werden die Anforderungen an SCORM aufgeführt: „*accessible, interoperable, durable, and reusable content and systems*“ (ADLNET-SCORM 2014, Synopsis).

Nach DODDS & THROPP (2006, S. SCORM-1-9, siehe auch Abbildung 45) werden in SCORM („*SCORM 2004 3rd Edition*“) folgende Spezifikationen („Standards“) eingesetzt:

- „*IEEE Data Model For Content Object Communication*
- *IEEE ECMAScript Application Programming Interface for Content to Runtime Services Communication*
- *IEEE Learning Object Metadata (LOM)*
- *IEEE Extensible Markup Language (XML) Schema Binding for Learning Object Metadata Data Model*
- *IMS Content Packaging*
- *IMS Simple Sequencing.*“

In seiner Dokumentation weist SCORM drei Hauptthemen auf („*technical books*“, im Folgenden Bücher genannt): „*Content Aggregation Model (CAM)*“, „*Run-time Environment (RTE)*“ und „*Sequencing and Navigation (SN)*“ (DODDS & THROPP 2006, S. SCORM-1-11, Abbildung 45, siehe auch ADLNET-SCORM-SPECIFICATION 2012).

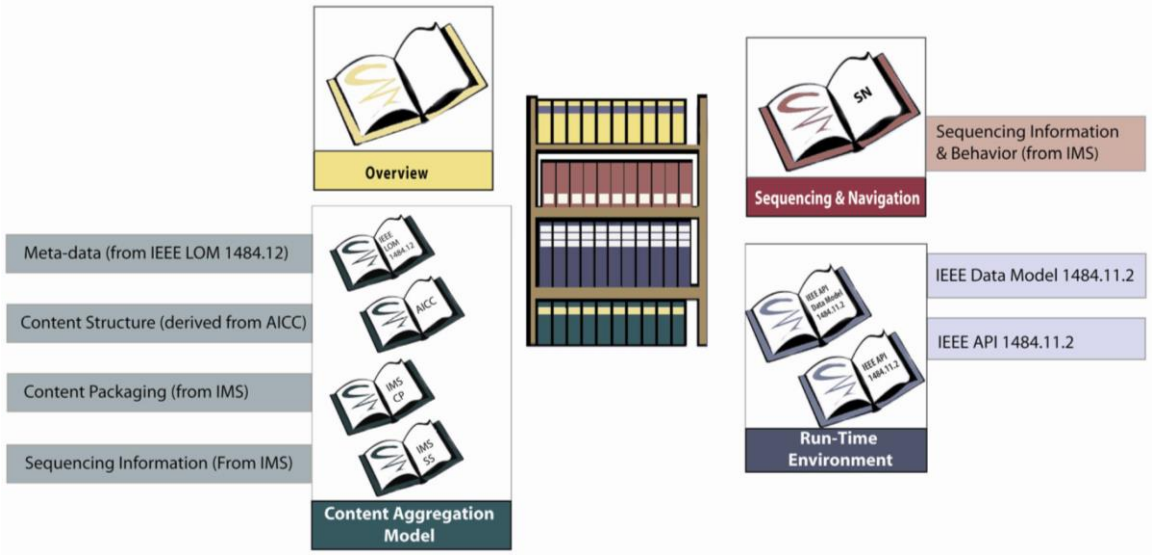


Abbildung 45 SCORM-Bücher (DODDS & THROPP 2006, S. SCORM-1-11, [Spezifikationsnummer für „*IEEE Data Model For Content Object Communication*“ lautet IEEE 1484.11.1, Anm. d. Verf.])

Im ersten Buch „Content Aggregation Model (CAM)“ werden die unterschiedlichen SCORM-Komponenten eines Lernpaketes („*content package*“) beschrieben, wie sie für den Datenaustausch gepackt („IMS Content Packaging“), für die Suche mit Metadaten ausgestattet (IEEE LOM) und wie die Reihenfolge von Komponenten im Lernablauf festgelegt („*IMS Simple Sequencing*“) werden können (JESUKIEWICZ 2009 a, S. CAM-1-3 f). Bei den Komponenten des „*SCORM Content Model*“ werden „*assets*“, „*sharable content objects (SCOs)*“, „*activities*“, „*content organization*“ und „*content aggregations*“ unterschieden (JESUKIEWICZ 2009 a, S. CAM-2-3). Als „asset“ wird der kleinste Baustein eines Lerninhaltes bezeichnet, der von einem Browser („Web client“) dargestellt und dem Lernenden zur Verfügung gestellt werden kann (JESUKIEWICZ 2009 a, S. CAM-2-3). Darunter fallen Medienobjekte wie z. B. Audio-, Video-, Bild-, XML-, oder HTML-Dateien (JUNGMANN 2012, S. 97, siehe auch Abbildung 46). Ein SCO besteht aus einem oder mehreren „assets“, kann einzeln gestartet werden und ist im Gegensatz zu einem „asset“ in der Lage über eine SCORM-Schnittstelle („*Application Programming Interface (API)*“, JESUKIEWICZ 2009 b, S. RTE-3-3) mit einem LMS zu kommunizieren (Abbildung 46, JESUKIEWICZ 2009 a, S. CAM-2-4). Zu dem letzten erwähnten Punkt hält JESUKIEWICZ (2009 a, S. CAM-2-5) fest: “*A SCO … must have a means to locate an LMS provided API Instance and must invoke the minimum API methods (Initialize(“”) and Terminate(“”)).*”

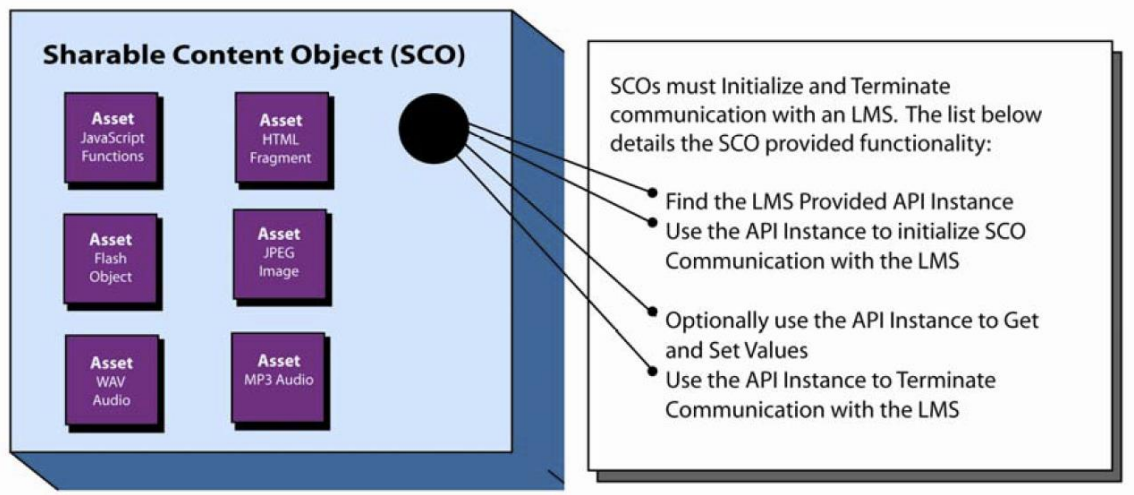


Abbildung 46 Konzeptionelle Darstellung eines SCO (JESUKIEWICZ 2009 a, S. CAM-2-4)

Eine *„(learning) activity*“ ist ein Vorgang, den der Lernende während des Lernprozesses durchführt (JESUKIEWICZ 2009 a, S. CAM-2-5). Sie stellt dem Lernenden Lerninhalt („*learning resource*“, „*SCO*“ oder „*asset*“) zur Verfügung oder besteht selbst wieder aus „*activities*“ (JESUKIEWICZ 2009 a, S. CAM-2-5). Die „*activities*“ werden in der (IMS-)Manifest-Datei als „item“- oder „organization“-Elemente abgebildet (JESUKIEWICZ 2009 a, S. CAM-5-3, siehe auch Listing 9 in Kapitel 3.3.7.1 IMS CP). Als weitere SCORM-Komponente legt die „*content organization*“ die Struktur fest, die für die Abarbeitung des Lerninhaltes vorgesehen ist (Abbildung 47, siehe auch Listing 9 in Kapitel 3.3.7.1 IMS CP).

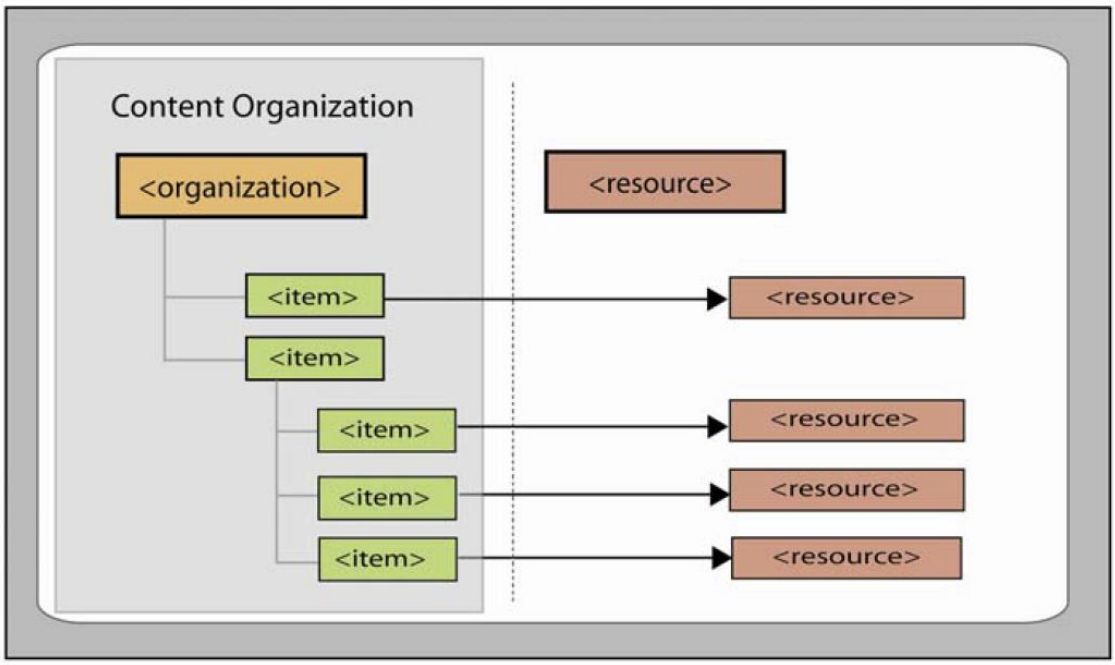


Abbildung 47 Konzeptionelle Darstellung einer „*content organization*“ (JESUKIEWICZ 2009 a, S. CAM-2-6)

Die letzte Komponente des „*SCORM Content Model*“ ist die „*content aggregation*“. Sie beschreibt den Prozess Lerninhalte („SCOs“, „assets“) gemäß einer vorgegebenen Struktur in eine Lerneinheit einzubinden (JESUKIEWICZ 2009 a, S. CAM-2-7, Abbildung 48). Nach JESUKIEWICZ (2009 a, S. CAM-2-8) wird der Ausdruck „*content aggregation*“ auch benutzt, um das „*content package*“ zu beschreiben. Das „*content package*“ soll einen standardisierten Austausch von Lerninhalten zwischen verschiedenen Systemen ermöglichen (z. B. LMS, Entwicklersoftware, Datenbank, JESUKIEWICZ 2009 a, S. CAM-3-3 ff). Es beinhaltet die eigentlichen Inhaltsdateien und die in der IMS-Manifest-Datei festgelegte Struktur (inklusive Verhalten) der Lerneinheit. Auf das „*content package*“ wird nicht mehr weiter eingegangen, da das „*SCORM Content Packaging*“ mit IMS CP bereits im Kapitel 3.3.7.1 IMS CP beschrieben wurde.

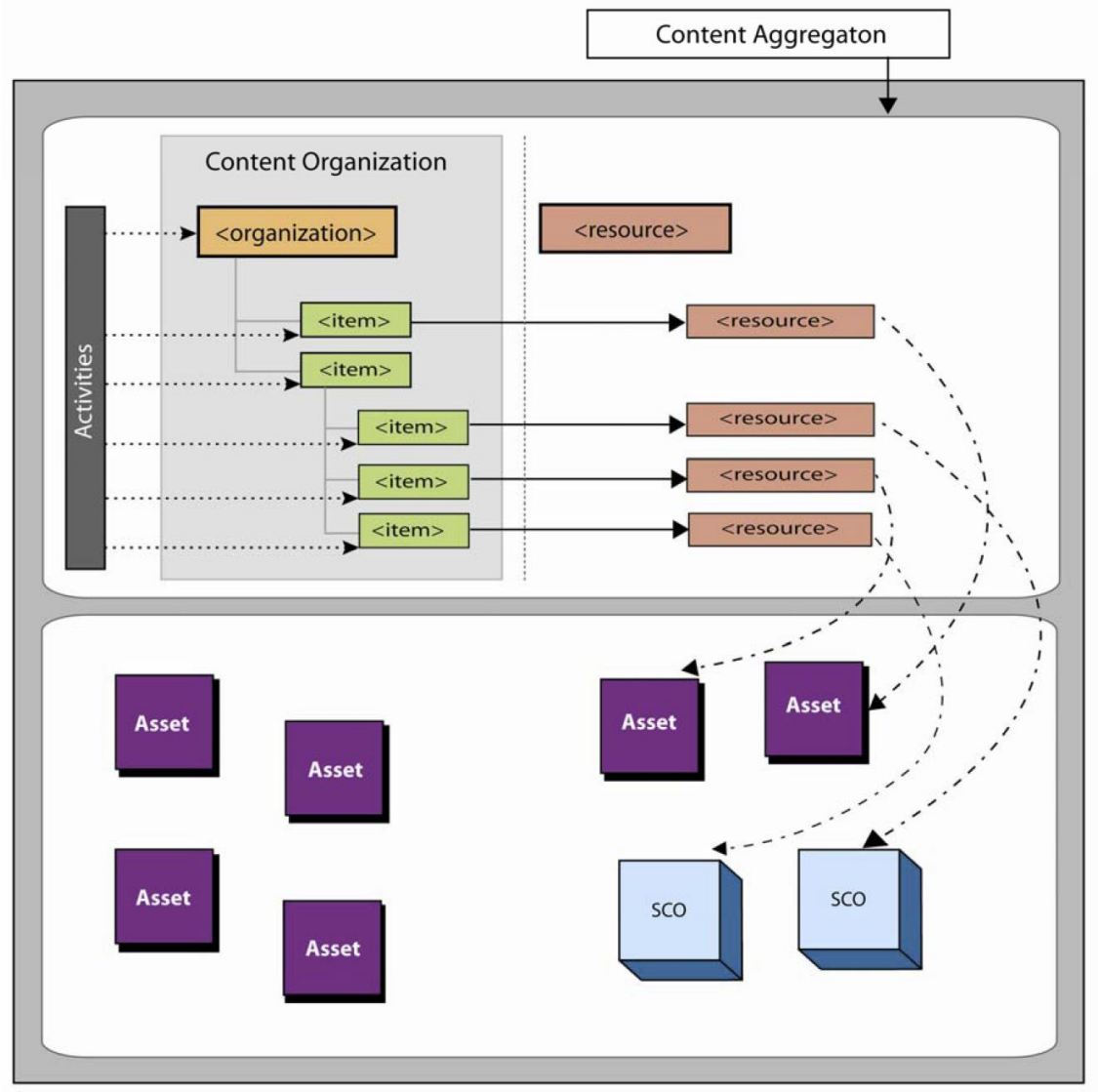


Abbildung 48 Konzeptionelle Darstellung einer „*content aggregation*“ (JESUKIEWICZ 2009 a, S. CAM-2-8)

Abbildung 49 zeigt SCORM-Inhaltsbausteine vom kleinsten („*Asset*“) bis zum größten („*Curriculum*“) Baustein.

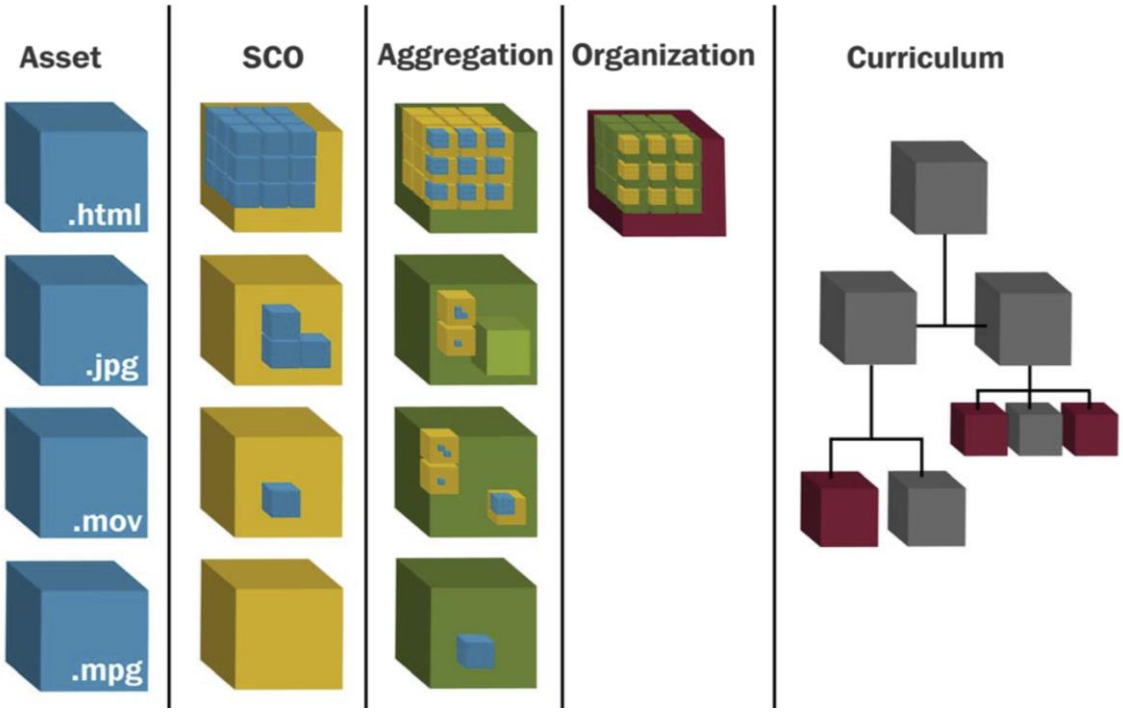


Abbildung 49 SCORM-Inhaltsbausteine (DEIBLER et al. 2008, S. 3-2)

Im zweiten Buch „*Run-Time Environment (RTE)*“ werden die Voraussetzungen für die Interaktion von LMS und SCORM-Inhalten geklärt (z. B. Suche der API-Implementation, Meldung über Start / Ende der Anzeige des SCORM-Inhaltes, allgemeine Festlegung der Kommunikation) (JESUKIEWICZ 2009 b, S. RTE-1-3, JESUKIEWICZ 2009 a, S. CAM-1-4 f).

Das dritte Buch „*Sequencing and Navigation (SN)*“ thematisiert die Festlegung der Reihenfolge (Ablaufsteuerung) in der SCORM-konforme Lerninhalte dem Lernenden angeboten werden können und wie er darin navigieren kann (JESUKIEWICZ 2009 , S. SN-1-3).

Trotz dieser Funktionalität („*Sequencing and Navigation (SN)*“) bringt JUNGMANN (2012, S. 98) zum Ausdruck, dass bei SCORM pädagogische Aspekte kaum berücksichtigt werden (siehe auch KERRES 2012, S. 453). Eine andere Spezifikation des IMS („*IMS Common Cartridge*“) mag diesem Gesichtspunkt zum Teil Rechnung tragen und legt einen Schwerpunkt auf die Umsetzung interaktiver und kollaborativer Lernsituationen (IMS-CC-FAQ 2014, 2. What problem is Common Cartridge meant to solve?, siehe auch ZIMMERMANN 2012, S. 70 ff, vergleiche auch Ausdruck PLE im nächsten Kapitel 3.9 LMS). „*IMS Common Cartridge*“ integriert dabei IEEE LOM, IMS

CP, IMS QTI, „*IMS Authorization Web Service*“ und „*IMS Basic Learning Tools Interoperability*“ (MATTSON 2014, 1 Introduction).

Zusätzlich bemerkt JUNGMANN (2012, S. 98), dass SCORM zu wenige Möglichkeiten bietet, „*die eigentliche redaktionelle Komponente, d. h. die Entwicklung wiederverwendbarer Lerninhalte, zu unterstützen*“ (siehe auch BRITAIN 2011, S. 60 ff). Daneben weist KERRES (2012, S. 451) darauf hin, dass die „*SCORM-Kompatibilität*“ zwar eine erfolgreiche Austausch- und Übertragbarkeit zwischen Systemen (z. B. LMS) wahrscheinlich macht, aber keine Garantie geben kann. Trotzdem hält er fest, dass SCORM „*eine große Akzeptanz gefunden*“ hat (KERRES 2012, S. 448).

Um den Einsatz von SCORM zu erleichtern, hat ADLNET-SCORM (2014, „*SCORM Certified Products*“, „*SCORM Adopters*“) zwei Listen SCORM-kompatibler Software-Produkte zusammengestellt. Diese Produkte erlauben es unter anderem SCORM-kompatible Inhalte zu erzeugen und darzustellen. Die Software „RELOAD“ wird dabei zur Erstellung von SCORM-Paketen besonders hervorgehoben (ADLNET-SCORM 2014). Auch verschiedene LMSs wie z. B. „ILIAS“ und „Moodle“ sind aufgeführt.

Im Folgenden wird auf den Ausdruck LMS eingegangen.

3.9 LMS

Der Ausdruck LMS wird in der Fachliteratur unterschiedlich beschrieben.

DEIBLER et al. (2008, S. 11-3) charakterisieren ein LMS wie folgt: „*An LMS is a Software package used to administer one or more courses to one or more learners. An LMS ist typically a web-based system that allows learners to authenticate themselves, register for courses, complete courses and take assessments. The LMS stores the learner's performance records and can provide assessment information to instructors.*“ Ähnlich äußern sich BAUMGARTNER et al. (2002 b, S. 309, siehe auch S. 24, Abbildung 50, ARNOLD et al. 2011, S. 69 f, WIKIPEDIA-LMS 2013): „*LMS ist ein Softwaretool, auf welches im Intranet/Internet zugegriffen werden kann, und das über eine entsprechende Oberfläche bestimmte Funktionalitäten, wie den Aufruf und die Administration von Lernerinnen, Lerninhalten, Übungsaufgaben, Kommunikationstools usw. von einer zentralen Stelle aus ermöglicht. Sie ist die zentrale Schnittstelle einer Lernumgebung zwischen Trainingsanbieterinnen und*

Trainingskundinnen.“ Die Autoren unterscheiden in ihren weiteren Ausführungen zwischen LMS und LCMS (BAUMGARTNER et al. 2002 b, S. 23, siehe auch BONEU 2011, S. 116, EIBL 2006, S. 12 ff). Nach ihren Aussagen kombiniert ein LCMS „*die typischen Funktionen von LMS mit den Funktionen zur Content-Erstellung und zur Content-Personalisierung der Content Management Systeme (CMS).*“ BAUMGARTNER et al. (2002 b, S. 43) geben als Definition eines LCMS an: „*Ein Learning Content Management System ist eine Software, die die Erstellung, Speicherung und Verwaltung von wieder verwendbaren Lernobjekten (RLO’s) sowie die Organisation und Betreuung webunterstützten Lernens ermöglichen.*“

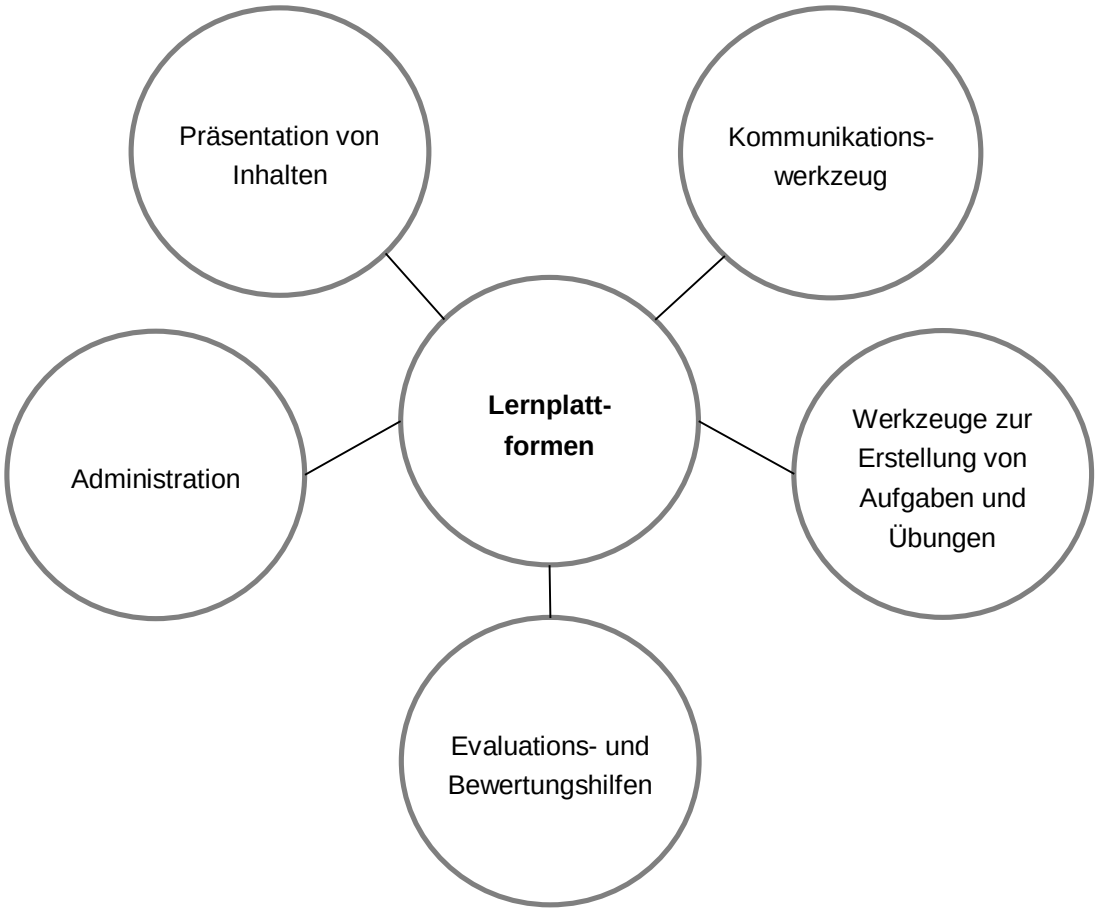


Abbildung 50 Fünf wichtigsten Funktionsbereiche von Lernplattformen nach BAUMGARTNER et al. (2002 b, S. 27), Lernplattform entspricht LMS (BAUMGARTNER et al. 2002 b, S. 30)

Diese Vorstellung der Unterscheidung eines LMS und eines LCMS übernimmt SCHULMEISTER nicht (2003, S. 15, vergleiche auch BRITAIN 2011, S. 58). Aufgrund der fließenden Übergänge beider Systeme gibt er die Unterscheidung auf (SCHULMEISTER 2003, S. 15, siehe auch ETEACHING-LMS 2012). Dieser Ansicht und der anschließenden Definition wird sich in der folgenden Arbeit angeschlossen. SCHULMEISTER (2003, S. 10) definiert ein LMS wie folgt (Abbildung 51): *„Als*

Lernplattform oder Learning Management System (LMS) werden – im Unterschied zu bloßen Kollektionen von Lehrskripten oder Hypertext-Sammlungen auf Web-Servern – Software-Systeme bezeichnet, die über folgende Funktionen verfügen:

- *Eine Benutzerverwaltung (Anmeldung mit Verschlüsselung)*
- *Eine Kursverwaltung (Kurse, Verwaltung der Inhalte, Dateiverwaltung)*
- *Eine Rollen- und Rechtevergabe mit differenzierten Rechten*
- *Kommunikationsmethoden (Chat, Foren) und Werkzeuge für das Lernen (Whiteboard, Notizbuch, Annotationen, Kalender etc.)*
- *Die Darstellung der Kursinhalte, Lernobjekte und Medien in einem netzwerkfähigen Browser.*“

SCHULMEISTER (2003, S. 10 f) legt dar, dass ein LMS in wesentlichen drei Schichten aufweist (Abbildung 51). Die erste Schicht bezeichnet er als „*Datenbankschicht*“, in der sich z. B. alle Lernobjekte und Benutzerdaten wiederfinden. Die zweite Schicht beschreibt er als „*Schicht mit Schnittstellendefinitionen zu anderen Systemen (API, application programmer's interface)*“. Als andere Systeme sieht er z. B. Studentenverwaltung, Raumverwaltung, Abrechnungssysteme oder Bibliotheken an. Die dritte Schicht schließlich stellt „*die Inhalte für den Administrator, den Dozenten oder den Studierenden*“ dar.

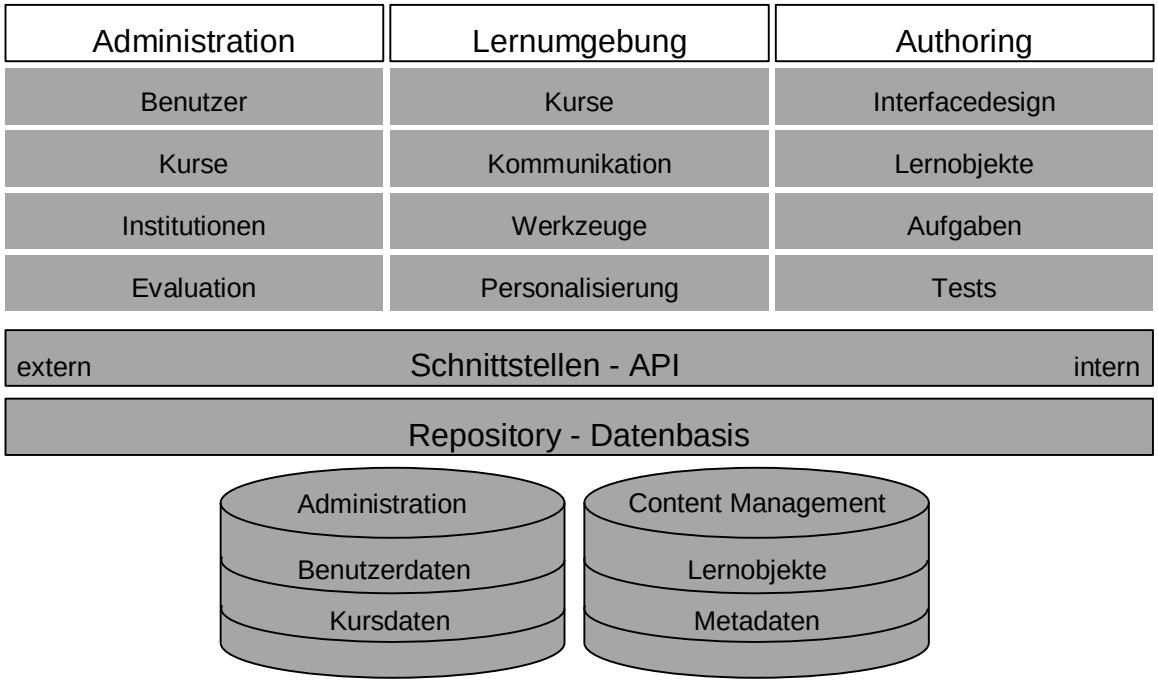


Abbildung 51 Wichtigste Elemente eines LMS (SCHULMEISTER 2003, S. 11, siehe auch MEIER 2006, S. 45 ff, KERRES 2012, S. 438 ff)

Die Ausdrücke Lernplattform, LMS (LCMS), Virtual Learning Environment werden in der vorliegenden Arbeit als identisch betrachtet. Im Folgenden wird der Ausdruck LMS benutzt (siehe ETEACHING-LMS 2012, BAUMGARTNER et al. 2002 b, S. 30, BRITAIN 2011, S. 58, SCHULMEISTER 2003, S. 12, WILBERS 2007, S. 5, HÄFELE & MAIER-HÄFELE 2008, S. 17, WIKIPEDIA-LMS 2013, BÖR 2003).

Eine Liste verschiedener LMSs bietet z. B. ADLNET-SCORM (2014, „*SCORM Certified Products*“, „*SCORM Adopters*“), SCHUMANN (2008) oder WIKIPEDIA-LMS-LISTE 2013). Zum Thema „Evaluation von Lernplattformen“ bietet z. B. JELITTO (2012) eine Hyperlinksammlung zu unterschiedlichen Veröffentlichungen.

KERRES (2012, S. 438) weist darauf hin, dass sich die Vielfalt der LMSs bis „*auf einige wenige Produkte mit einer hohen Verbreitung und relativ ähnlicher Funktionalität reduziert*“ hat. Insbesondere weist er auf die Bedeutung von „open source“-Lösungen hin. Er erläutert weiter, dass 2008 nur mehr zwei Produkte (ein kommerzielles und ein „open source“-Produkt) in insgesamt 88% der britischen Hochschulen eingesetzt werden (JISC 2008, S. 13). Nach JISC (2008, S. 13) handelt es sich dabei um die Software-Produkte „Blackboard“ (kommerziell) und „Moodle“ („open source“). „The 24th National Survey of Computing and Information Technology in US Higher Education“ ergab 2013 (GREEN 2013, S. 23), dass ebenfalls diese beiden LMS den untersuchten LMS-Markt der US-amerikanischen Hochschulen beherrschen („Blackboard“ 41%, „Moodle“ 23%, 62% der Klassen nutzen LMS (leicht steigend im Vergleich zum Jahr 2011)). Für den deutschsprachigen Raum liegen nach KERRES (2012, S. 438) zu diesem Thema keine vergleichbaren Zahlen vor.

KALZ et al. (2011, S. 3) halten bezüglich der Verbreitung von LMS fest: „*In nahezu jeder Hochschule in Mitteleuropa sowie bei vielen Schulen und Bildungseinrichtungen werden derzeit LMS eingesetzt.*“ Sie führen weiter aus: „*Zu den am weitesten verbreiteten gehören hier Moodle, Ilias und Blackboard.*“ Auch HÄFELE & MAIER-HÄFELE (2008, S. 17) heben die beiden bewährten, häufig im Einsatz befindlichen „Open-Source“-Lernplattformen „ILIAS“ und „Moodle“ hervor (siehe auch FLORIAN 2012, S. 5, WILBERS 2007, S. 19, EIBL 2006, S. 14 f). FISLER & BLEISCH (2012, S. 10, KOPER 2013) erwähnen zusätzlich als weitere „Open-Source“-Lernplattform „OLAT“ (OLAT 2014).

WILBERS (2011, S. 20) führt aus, dass sich virtuelle Plattformen (z. B. Lernplattformen, Community-Plattformen) weiter entwickeln, sich dabei aber immer

mehr in ihren Funktionen einander annähern. Er erklärt (siehe auch KERRES 2006, S. 7): „*Als Alternative zu Plattformen, die vor allem einer zentralen Metapher – Kurs, Community oder Projekt – folgen, können persönliche Lernumgebungen (PLE, personal learning environment) begriffen werden.*“ SCHAFFERT & KALZ (2009, S. 6, siehe auch KALZ et al. 2011, S. 5 f) definieren PLE wie folgt: „*Persönliche Lernumgebungen, kurz PLE, sind Lernanwendungen, bei denen Lerner verteilte Online-Informationen, -Ressourcen oder -Kontakte einerseits selbst in ihre PLE integrieren können und andererseits auch ihre im Rahmen der PLE vollzogenen Aktivitäten und deren Produkte in anderen Online-Umgebungen auf der Basis von Standards zur Verfügung stellen können.*“ Sie legen dar, dass der Lernende im Fokus einer PLE steht (selbstgesteuertes Lernen, WILBERS 2011, S. 20) und „*sich selbst Webinhalte, Lernressourcen und Lernwerkzeuge so arrangiert und sie so nutzt, dass sie sein persönliches Wissensmanagement und Lernen unterstützen*“ (SCHAFFERT & KALZ 2009, S. 1). Eine PLE kann dabei in einer Oberfläche „Web 2.0“-Anwendungen wie z. B. Weblogs, Wikis, Community-Funktionalitäten oder „Content Sharing“ integrieren (VAN HARMELEN 2008, KERRES 2006, S. 7, WILBERS 2011, S. 20, WIKIPEDIA-PLE 2013). VAN HARMELEN (2008) betont dabei im Zusammenhang mit PLE und selbstgesteuerten Lernen in sozialen Netzwerken: „*Social constructivism has as a central precept that knowledge is created by learners in the context of, and as a result of social interaction. Social constructivist approaches are particularly aided by using social networking service based PLEs as mediating mechanisms between learners*“ (siehe auch Kapitel 3.1.1.3 Konstruktivismus und Ausdruck Konnektivismus in Kapitel 3.1.1.4 Zusammenfassung).

Um eine PLE mit ihren „Web 2.0“-Anwendungen oder allgemein ein LMS nutzen zu können, lassen sich als Oberfläche Browser einsetzen. Im nächsten Kapitel werden Marktanteile verschiedener Browser betrachtet.

3.10 Marktanteile Browser

Die clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript und deren Darstellung (siehe z. B. Kapitel 3.6.1 Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript) sowie die Nutzung von Lerninhalten über ein LMS sind unter anderem vom jeweiligen Browser (Client-Software) abhängig (WIKIPEDIA-LMS 2013). Um die Marktanteile und Tendenz der führenden Browser grob einschätzen zu können, werden im Folgenden die Ergebnisse verschiedener Auswertungen betrachtet (siehe auch WIKIPEDIA-BROWSER-STATISTIK 2014).

Nach STATCOUNTER-FAQ (2014) handelt es sich bei „StatCounter“ um ein „*web analytics service*“, das ihren „*tracking code*“ auf global über 3 Millionen Internetauftritten („sites“) installiert hat und die Ergebnisse monatlich auswertet. Bei Aufruf der Webseiten („page views“) lassen sich z. B. der Browser mit seiner Version, das Betriebssystem oder die Plattform (Desktop-PC/Laptop, mobiles Gerät, Tablet-PC, Konsole) feststellen. Die folgenden Prozentangaben (inklusive der Werte in Tabelle 7) außerhalb der Klammern beziehen sich auf den Anteil des jeweiligen Browsers im Monat Januar des Jahres 2014. Sie wurden aus der von STATCOUNTER (2014) für jedes Diagramm angebotenen CSV-Datei („comma-separated values“-Datei) ausgelesen. Die Prozentangaben innerhalb der Klammern beziehen sich auf den Anteil des jeweiligen Browsers im Monat Dezember des Jahres 2008. Beide Zahlen sollen eine grobe Tendenz zeigen. Der Zeitrahmen („Period“) von Dezember 2008 bis Januar 2014 stellt bei den vorgenommenen Einstellungen („Platform: Desktop, Mobile, Tablet, Console“, „Region: Germany“) die maximal auswählbare Periode dar (STATCOUNTER 2014). Alle für die jeweilige Region aufgeführten Browser liegen im Januar 2014 jeweils über 1 % Marktanteil. Außerdem sind alle Prozent-Angaben auf ganze Prozent gerundet.

Abbildung 52 zeigt die festgestellten Browseranteile für Deutschland. „Firefox“ („Mozilla Firefox“) ist nach den Angaben von STATCOUNTER (2014) mit ca. 38 % (57 %) der führende Browser. „Chrome“ („Google Chrome“) liegt mit ca. 24 % (1 %) über dem Anteil von „IE“ („Microsoft Internet Explorer“) mit ca. 16 % (35 %). Die restlichen Browser weisen im Januar 2014 jeweils Werte unter 10 % auf.

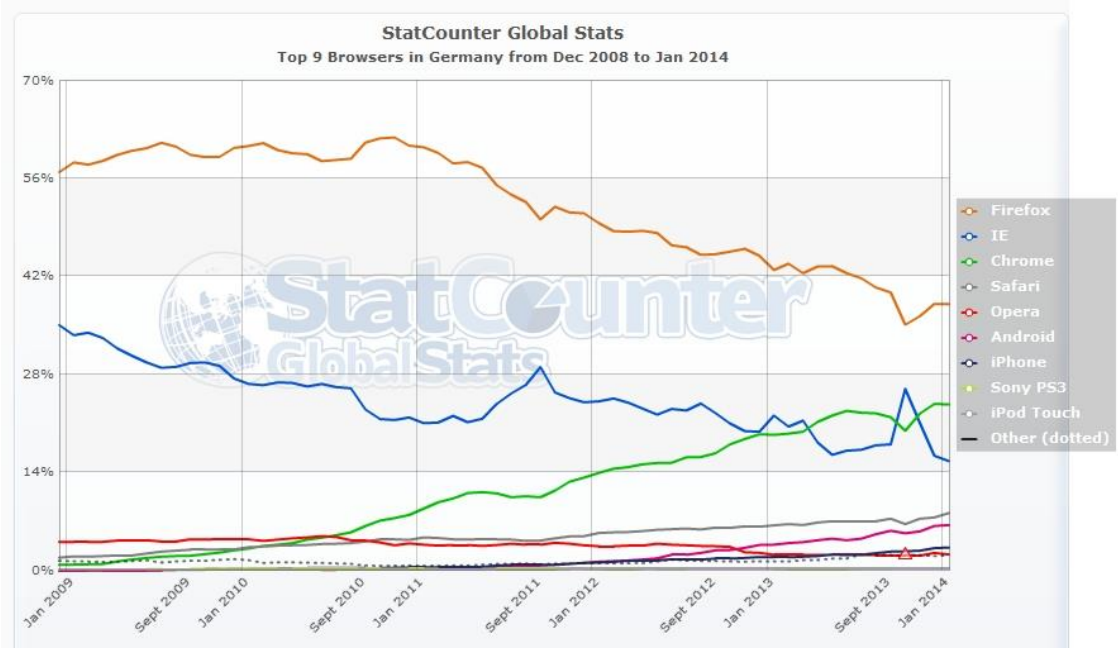


Abbildung 52 Browseranteile („Top 9“) in Deutschland inklusive mobiler Geräte (Period: „Dez 2008 to Jan 2014 (Monthly)“, Platform: „Desktop, Mobile, Tablet, Console“, Region: „Germany“, STATCOUNTER 2014, http://gs.statcounter.com/)

Abbildung 53 zeigt die festgestellten Browseranteile für Europa. Hier führt nach den Angaben von STATCOUNTER (2014) im Gegensatz zu den Zahlen für Deutschland „Chrome“ mit ca. 37 % (1 %) vor „Firefox“ mit ca. 22 % (38 %) und vor „IE“ mit ca. 16 % (50 %). Die restlichen Browser weisen im Januar 2014 jeweils Werte unter 10 % auf.

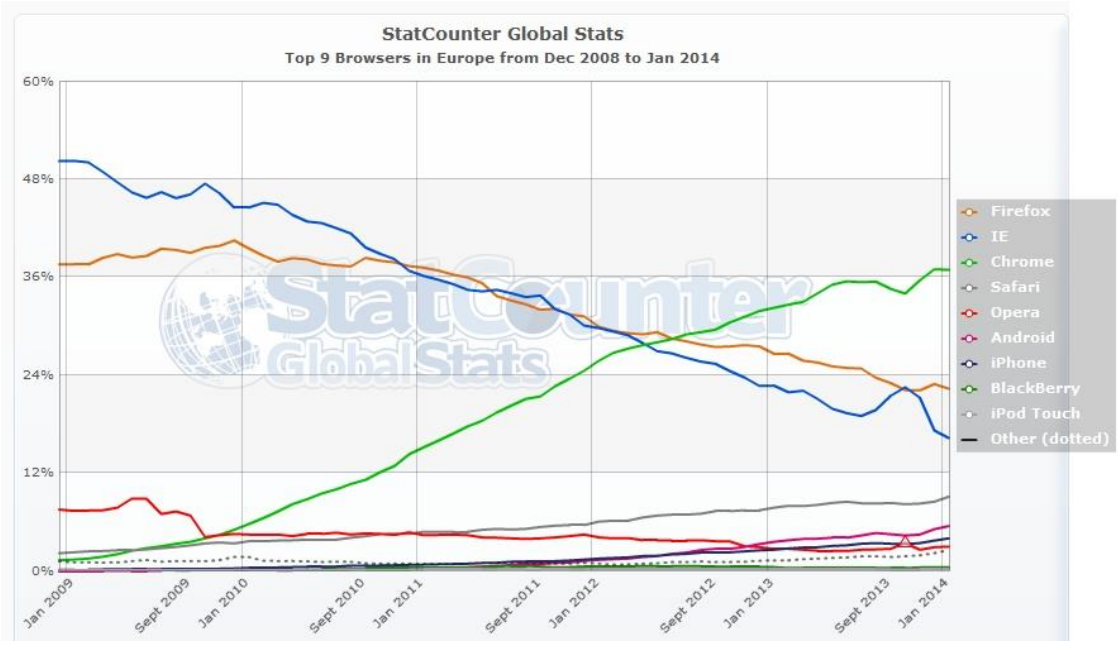


Abbildung 53 Browseranteile („Top 9“) in Europa inklusive mobiler Geräte (Period: „Dez 2008 to Jan 2014 (Monthly)“, Platform: „Desktop, Mobile, Tablet, Console“, Region: „Europe“, STATCOUNTER 2014, http://gs.statcounter.com/)

Abbildung 54 zeigt die weltweit festgestellten Browseranteile. Hier führt nach den Angaben von STATCOUNTER (2014) wie in Europa „Chrome“ mit ca. 36 % (1 %), allerdings (im Gegensatz zu den Zahlen für Europa) gefolgt von „IE“ mit ca. 17 % (67 %) und „Firefox“ mit ca. 15 % (25 %). Die restlichen Browser weisen im Januar 2014 jeweils Werte unter 10 % auf.

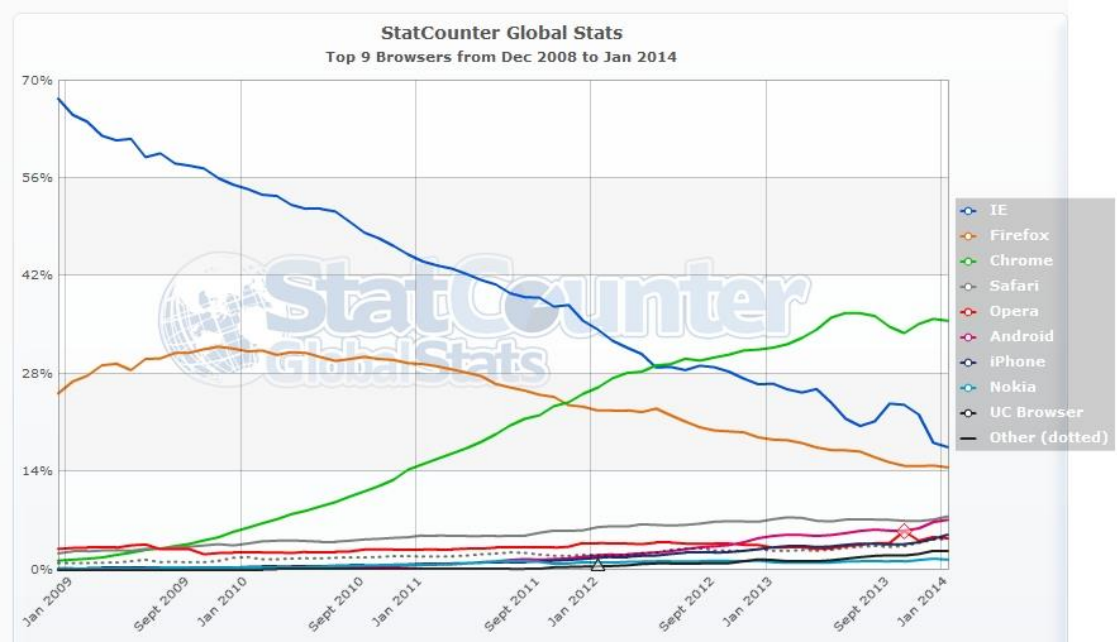


Abbildung 54 Browseranteile („Top 9“) weltweit inklusive mobiler Geräte (Period: „Dez 2008 to Jan 2014 (Monthly)“, Platform: „Desktop, Mobile, Tablet, Console“, Region: „Worldwide“, STATCOUNTER 2014, http://gs.statcounter.com/)

Tabelle 7 zeigt in einer Übersicht die jeweiligen Angaben der Browseranteile (inklusive Browser mit Anteilen von 1 % bis 10 % im Januar 2014). Die vier führenden Browser in allen Regionen sind „Chrome“, „Firefox“, „IE“ und „Safari“ (alphabetische Reihenfolge).

Tabelle 7: Browser („Top 9“) in alphabetischer Reihenfolge und ihre Anteile in % in den Regionen Deutschland, Europa, weltweit (STATCOUNTER 2014, http://gs.statcounter.com/)

Browser	Deutschland in % Jan 2014 (Dez 2008)	Europa in % Jan 2014 (Dez 2008)	weltweit in % Jan 2014 (Dez 2008)
Android	6 (0)	5 (0)	7 (0)
Chrome	24 (1)	37 (1)	36 (1)
Firefox	38 (57)	22 (38)	15 (25)
IE	16 (35)	16 (50)	17 (67)
iPhone	3 (0)	4 (0)	5 (0)
Nokia			1 (0)
Opera	2 (4)	3 (7)	4 (3)
Safari	8 (2)	9 (2)	8 (2)
UC Browser*			3 (0)
Summe	**97 (99)**	**96 (98)**	**96 (98)**

* Browser auf mobilen Geräten, vor allem in China, Indien (WIKIPEDIA-UC-BROWSER 2014)

Ähnliche Erhebungen wie die durch STATCOUNTER (2014) ergeben ebenfalls nach WIKIPEDIA-BROWSER-STATISTIK (2014, Summary table) die führenden vier Browser „Chrome“, „Firefox“, „IE“ und „Safari“ (alphabetische Reihenfolge). Insgesamt steigt nach diesen Angaben (Tabelle 7, WIKIPEDIA-BROWSER-STATISTIK 2014, siehe auch WEBTREKK WEBSTATISKTIK 2014) der Anteil der Browser mobiler Geräte (vor allem „Android“, „iPhone“, „Nokia“, „UC-Browser“), steigt der Anteil von „Google Chrome“ und fällt der Anteil des „Microsoft Internet Explorer“ sowie des „Mozilla Firefox“. Allerdings sind die angegebenen Zahlen aufgrund der vielfältigen Einflussfaktoren auf die Erhebungen mit Vorsicht zu beurteilen (WIKIPEDIA-BROWSER-STATISTIK 2014, Accuracy).

Abbildung 55 zeigt abschließend die weltweiten Browseranteile in Form einer Karte im Zeitrahmen von November 2013 bis Januar 2014 (zur Legende: Browser „Iron“ („SRWare Iron“) basiert auf dem „open-source browser project Chromium“, das „open source“-Projekt zu „Google Chrome“, WIKIPEDIA-IRON 2014). Die hier maximal auswählbare Zeitdauer liegt bei 90 Tagen.

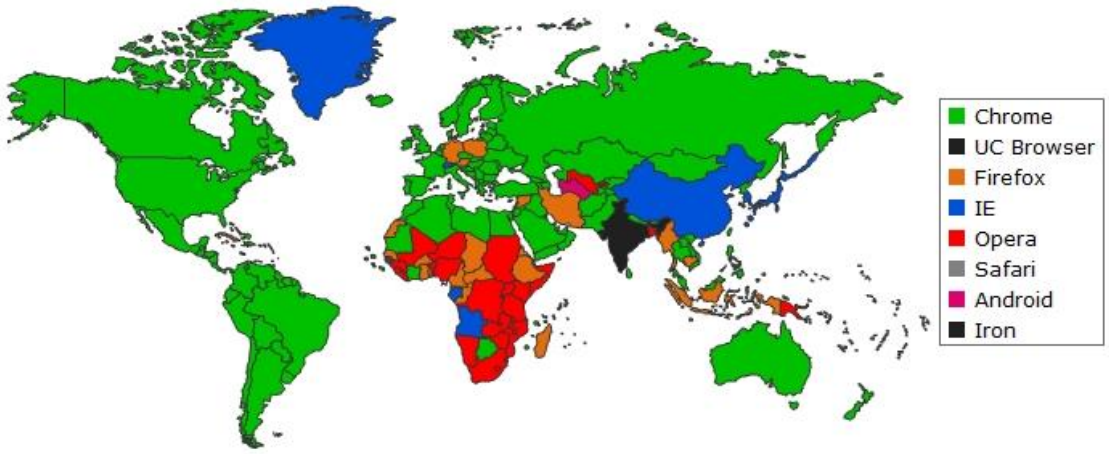


Abbildung 55 **„Top Browsers Per Country from Nov 2013 to Jan 2014 (StatCounter Global Stats)“, Browseranteile weltweit (dargestellt als Karte) inklusive mobiler Geräte (Period: „Nov 2013 to Jan 2014 (Monthly)“, Platform: „Desktop, Mobile, Tablet, Console“, Region: „Worldwide“, STATCOUNTER 2014, http://gs.statcounter.com/)**

4 Ergebnisse

Im Folgenden werden die Ergebnisse der Arbeit vorgestellt. Beginnend mit der Aufführung der eingesetzten Software wird die im Rahmen dieser Arbeit entwickelte XML-Sprache WLMML mit seinem Framework erläutert (siehe auch QUEDNAU et al. 2007, PAAR et al. 2006, STREHL et al. 2005). Abschließend werden unterschiedliche lerntheoretische sowie didaktische Ansätze beschrieben.

Ziel war es eine auf „Standards“ beruhende Möglichkeit zu finden, mehrsprachige E-Learning-Inhalte strukturiert abbilden und darstellen, sie mit Standard-Software mit und ohne Server-Verbindung dem Lernenden anbieten (offline wie online) und in ein LMS importieren zu können. Die XML-Anwendung (siehe Kapitel 3.3.2 XML-Anwendung) WLMML (inkl. Framework) mit ihrer clientseitigen Verarbeitung kann dies leisten. Zusätzlich ermöglicht sie die Modularisierung und Wiederverwendung von E-Learning-Inhalten. Um diese Stärken möglichst nachhaltig zu bewahren, wird im WLMML-Projekt konsequent auf den Einsatz etablierter internationaler Standards/Normen und Spezifikationen geachtet (siehe Kapitel 3.2.1 Standardisierungsbemühungen, siehe auch STREHL et al. 2005, PAAR 2003). So verwenden z. B. alle WLMML- und WLMML-Framework-Dateien den internationalen Unicode-Zeichensatz in der Kodierung „UTF-8“ (siehe Kapitel 3.3.1 Zeichensätze, Internationalisierung von Lernmaterial). Die folgende Aufzählung fasst die eingesetzten internationalen „Standards“ zusammen:

- Unicode „UTF-8“ (siehe Kapitel 3.3.1 Zeichensätze)
- XML (siehe Kapitel 3.3 XML)
- W3C XSD (siehe Kapitel 3.3.3 DTD / W3C XML Schema Definition)
- XSLT (siehe Kapitel 3.3.8.1 XSLT)
- JavaScript (siehe Kapitel 3.6 JavaScript)
- HTML (siehe Kapitel 3.4 HTML)
- CSS (siehe Kapitel 3.5 CSS)
- SCORM (siehe Kapitel 3.8 SCORM) mit
    - IMS CP (siehe Kapitel 3.3.7.1 IMS CP)
    - IEEE LOM (siehe Kapitel 3.7 IEEE LOM)

4.1 Eingesetzte Software

Für die Erstellung der WLMML-Schema-Datei wurde die kommerzielle Software „Altova XMLSpy Professional Edition“ (Version 2006) eingesetzt (siehe z. B. Hinweis auf diese Software zur XML-Daten-Bearbeitung von GEEB 2003, S. 23).

WLMML-Inhaltsdateien wurden in der vorliegenden Arbeit „Altova XMLSpy Professional Edition“ (Versionen 2005/2006/2007/2008/2009), mit „Microsoft Word“ (Versionen 2003/2007/2010 (Vertrieb der Software vor 10.01.2010, siehe MICROSOFT-WORD-XML-MARKUP 2014) und „Eclipse“ (Version 3.2) erstellt und von diesen Programmen gegen die WLMML-Schema-Datei validiert.

Neben diesen Programmen sind auch Standard-XML-Editoren wie z. B. „XMetal“, „Oxygen XML Editor“, „Exchanger XML Lite 3.2“ (für Privatanwender und akademische Einrichtungen kostenlos) oder „Microsoft XML Notepad 2007“ (kostenlos) in der Lage WLMML-Inhaltsdateien zu generieren (siehe auch Liste von XML-Editoren WIKIPEDIA-XML-EDITOREN 2014). Da WLMML-XML-Dateien reine Textdateien sind, kann daneben jeder Texteditor, der in der Lage ist Dateien im UTF-8-Format abzuspeichern, WLMML-Dateien erzeugen. Allerdings erleichtern spezifische Programme mit XML-Unterstützung wie z. B. „Altova XMLSpy“ und „Eclipse“ das Erstellen von WLMML-Dateien (siehe Anhang E).

Für die Bildbearbeitung wurde in der vorliegenden Arbeit die Software „Ulead Photo Impact“ (Version 6 und 10) eingesetzt.

Die UML-Abbildungen wurden in erster Linie mit dem Programm „Microsoft Visio 2012“ erstellt.

Für die JavaScript-Programmierung, die Erstellung der CSS-Dateien und sonstige Quelltextarbeiten wurde „Microsoft Notepad“ (unter „Microsoft Windows XP und Windows 7“) und „Notepad++“ (Version 5.8.2) herangezogen.

SCORM-Module (siehe Kapitel 3.8 SCORM) wurden mit der Software „RELOAD“-Editor (Versionen 2.5.1, 2.5.5 und 2.5.6) (RELOAD 2008) und der sehr ähnlichen Software „ADL SCORM 2004 RELOAD Editor“ (Version 1.1) (ADLNET-RELOAD 2009) erstellt. „RELOAD“ ist ein Editor („open-source“-Software), der es u. a. erlaubt „ADL SCORM 2004“-Lerninhalte mit Metadaten zusammenzustellen, zu importieren und zu exportieren (BERKING 2011, ADLNET-SCORM 2014).

Als Testumgebungen für die Abarbeitung der Anweisungen in den XML-/XSLT-/HTML-/CSS-/JavaScript-Dateien sowie Überprüfung der Funktionstauglichkeit

weiterer Dateiformate (z. B. Bild-, Ton-, Video-Dateien) dienten die folgenden auf dem Betriebssystem „Microsoft Windows“ installierten Browser:

- „Microsoft Internet Explorer“ (Versionen mit den Anfangsziffern 6 bis 11)
- „Mozilla Firefox“ (Versionen mit den Anfangsziffern 2, 3, 4, 5, 10, 12, 17, 24, 26, 28, 29, 30)
- „Google Chrome“ (Versionen mit den Anfangsziffern 2, 4, 5, 6, 7, 8, 9, 12, 13, 20, 24, 31, 34, 35)
- „Apple Safari“ (Versionen mit den Anfangsziffern 4 und 5)
- „Opera“ (Versionen mit den Anfangsziffern 10, 11, 12)

Um die Ladereihenfolge beim Aufruf von Inhalten von einem Server anzeigen zu können, wurde die Software „Fiddler Web Debugger (v2.4.8.0)“ (FIDDLER 2014) eingesetzt.

Als Server-Testumgebung kam XAMPP (Versionen 1.7.3, 1.8.3) (XAMPP 2014) zum Einsatz.

Folgende LMSs wurden für Testzwecke eingesetzt:

- „ADL Sample Run-Time Environment“ Version 1.1.1, „SCORM2004“-konformes Beispiel-LMS für Testzwecke (ADLNET-SCORM-SRTE 2009)
- „Moodle“ in den Versionen 1.9, 2.0.2, 2.6.1, 2.7 (MOODLE 2014, neues LMS der „Technischen Universität München“ TUM-MOODLE 2014)
- „CLIX® Campus“ (CLIX) (CLIX 2005, RATHMAYER 2005)
- „ILIAS“ in der Version 4.4.3 (ILIAS 2014)

Als Betriebssysteme wurden in erster Linie die Microsoft Produkte „Windows XP“ und „Windows 7“ sowie als Textverarbeitungsprogramm „Microsoft Word“ eingesetzt. Herzlichen Dank an dieser Stelle an die Firma Microsoft und an alle weiteren Firmen, deren Produkte genutzt wurden.

4.2 WLMML

Die im Rahmen der vorliegenden Arbeit entwickelte XML-Sprache WLMML entstand als ein Projekt innerhalb von WELPE (Weihenstephaner E-Learning-Projekte). WLMML steht für die im Projekt vorliegende XML-Sprache, die es erlaubt E-Learning-Inhalte in XML abzubilden. XML wurde ausgewählt, da die Meta-Sprache

als Datenformat im E-Learning-Bereich - auch gegenüber HTML - Vorteile bietet (siehe Kapitel 3.3 XML, 3.4 HTML). WLMML orientiert sich an LMML (siehe Kapitel 3.3.6.1 LMML) der Universität Passau. LMML wurde als Anhalt ausgewählt, da sie zum Entstehungszeitpunkt der vorliegenden Arbeit bereits eingesetzt wurde, der Modellierungsansatz überzeugte und sie mit am weitesten für die strukturierte Abbildung von E-Learning-Inhalten etabliert war (siehe Kapitel 3.3.6.1 LMML). Das WLMML-Modell (vergleichbar mit dem PTM) kann als neu modellierte Instanz des ADM nach SÜSS (2004, S. 32 ff) betrachtet werden (siehe Kapitel 3.2.3 Modellierung von E-Learning-Inhalten). Die XML-Sprache WLMML (vergleichbar mit LMML-CS) ist die Umsetzung des WLMML-Modells in XML (XML-Binding) und bildet die sachlogische Fragmentierung von E-Learning-Inhalten in Form eines „XML Schema“ ab (siehe Kapitel 3.3.6.1.2 LMML-Framework). Abbildung 56 zeigt eine Übersicht über diese sachlogische Fragmentierung in WLMML.

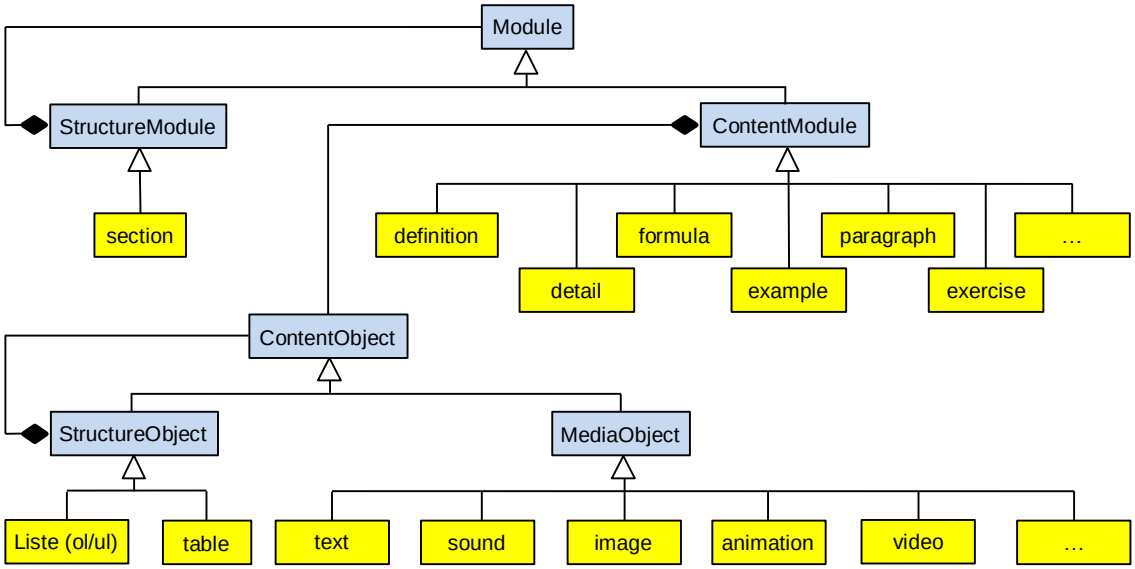


Abbildung 56: Sachlogische Fragmentierung von E-Learning-Inhalten in WLMML (vergleiche Kapitel 3.3.6.1 LMML)

Bezüglich des in Abbildung 56 (siehe auch Kapitel 3.3.6.1 LMML) verwendeten Ausdrucks „*Module*“ halten WEITL & HÖCK (2003, S. 4) Folgendes fest: "*Der Begriff Module im Metamodell* [ADM, siehe Kapitel 3.2.3 Modellierung von E-Learning-Inhalten, Anm. d. Verf.] *steht nicht für ein abgeschlossenes Lernmodul im Sinne der WWR* [BMBF-Projekt Wissenswerkstatt Rechensysteme, Anm. d. Verf.], *sondern bezeichnet eine größere zusammenhängende inhaltliche Einheit.*“ Sie führen weiter aus: „*Ganz oben in der Elementhierarchie ist das Dokument (im WWR-Jargon als Modul bezeichnet). Ein Dokument besteht aus mehreren Modulen, die entweder Strukturmodule oder Contentmodule sind. Strukturmodule können rekursiv weitere Strukturmodule oder Contentmodule enthalten. Contentmodule bestehen aus*

Contentobjekten, welche Strukturobjekte oder Medienobjekte sein können. Strukturobjekte können rekursiv weitere Struktur- und Medienobjekte enthalten." Auch das WLMML-Modell spiegelt diese Vorstellung wieder (Abbildung 56).

Als eigenständige XML-Sprache weist WLMML einen nach semantischen Gesichtspunkten gegliederten Aufbau auf (W3C-SEMANTIC-WEB 2011, siehe auch W3C-OWL 2011). So tragen die WLMML-Elemente eine bestimmte Bedeutung (z. B. „definition“-, „detail“-, „formula“-, „example“-Element) und können Lerninhalte in strukturierter Form nach didaktischen Gesichtspunkten abbilden (Abbildung 56). Die E-Learning-Inhalte (insbesondere Inhalte des „Life Sciences“) können darüber in einfacher Form modular abgebildet werden. Elemente anderer XML-Sprachen wie MathML (siehe Kapitel 3.3.5.1 MathML), SVG (siehe Kapitel 3.3.5.2 SVG) und XHTML (siehe Kapitel 3.3.5.3 XHTML) lassen sich über spezielle WLMML-Elemente einbinden. Unabhängig von der Darstellung werden die E-Learning-Inhalte (außer z. B. Bilder, Töne, Videos) in reiner Textform (XML-Format) im Unicode in der Kodierung „UTF-8“ abgespeichert (siehe Kapitel 3.3.1 Zeichensätze). Erst Anweisungen in Dateien des WLMML-Frameworks übernehmen die optische Aufbereitung.

Die XML-Anwendung (siehe Kapitel 3.3.2 XML-Anwendung) WLMML wurde eigenständig entwickelt (siehe auch QUEDNAU et al. 2007, S. 6 f), um neue Elemente und Attribute (sowie auch „alte“ LMML-Elemente und -Attribute) an spezifischen Stellen im WLMML-Dokument einsetzen zu können. Zusätzlich sollte Software wie z. B. „Eclipse“ oder „Altova XMLSpy“ (ab Version 2006) bei der Erstellung von WLMML-Dateien (mit dem identischen WLMML-Schema) verwendet werden können. Darüber hinaus war es ein Ziel die acht LMML-Schema-Dateien durch eine einzige Schema-Datei zu ersetzen. Zur weiteren Vereinfachung hinsichtlich Erlernbarkeit und Übersichtlichkeit wurden in WLMML nicht alle Elemente und Attribute von LMML übernommen. Zusätzlich ergab sich bei der Erstellung verschiedener Lerninhalte der Bedarf einige Elemente bzw. Attribute neu einzuführen. Auf der Grundlage dieser Überlegungen entstand eine einzelne neu entwickelte XSD-Datei (siehe Kapitel 3.3 DTD / W3C XML Schema Definition, siehe auch PAAR et al. 2006, S. 2). Bezüglich der Erstellung eines eigenen „XML Schema“ halten TORSTEN et al (2003, S. 436) fest: „*Keinem der verfügbaren Ansätze zum Einsatz von XML im Bereich E-Learning (z. B. EML, LMML, TeachML) kann eine dominierende Stellung als Standard zugesprochen werden. Ein eigenes Schema ermöglicht die Fokussierung auf die originären Projektinhalte, die Berücksichtigung*

der neuesten Entwicklungen und Standards im Bereich XML sowie die schnelle Einflussnahme auf den Entwicklungsprozess.“

Im nächsten Kapitel wird auf das WLMML-Schema eingegangen.

4.2.1 Schema

Das WLMML-Schema beschreibt die zulässigen Elemente, Attribute und deren Verschachtelung (siehe Kapitel 3.3 DTD / W3C XML Schema Definition). Sie liegt in Form einer einzelnen neu entwickelten XSD-Datei vor (Dateiname „wlmml.xsd“, siehe Anhang C und Anhang D) und verwendet wie alle WLMML-Dateien den Zeichensatz „UTF-8“ (Listing 34, Zeile 1, siehe Kapitel 3.3.1 Zeichensätze).

Jedes WLMML-Dokument ist dann valide, wenn es gegen das WLMML-Schema fehlerfrei geprüft werden kann. Dafür muss das Dokument z. B. das Root-Element „wlmml“ (Listing 34, Zeile 6) und mindestens ein Struktur-Modul-Element aufweisen („choice“-Anweisung, Listing 34, Zeile 9). Das „bibliography“-Element (Literaturverzeichnis, Zeile 10) stellt neben dem „section“-Element (Inhaltsabschnitt, Zeile 11) das einzige Struktur-Modul-Element dar (Listing 34). Diese beiden Struktur-Modul-Elemente spiegeln die hierarchische Grobstruktur eines Dokumentes wieder. Über das Element „complexType“ (Listing 34, Zeile 8) können andere Elemente oder Attribute integriert werden (GAO et al. 2012, 2.2.1 Type Definition Components ff). Das Element „simpleType“ (Listing 34, Zeile 14, siehe auch Listing 36, Zeile 2) hingegen kann Einschränkungen („constraints“) und Informationen über Attributwerte und reine Text-Elemente einbinden (GAO et al. 2012, 2.2.1 Type Definition Components ff). Ein Beispiel für den Einsatz des Elementes „simpleType“ zeigt Listing 34 (Zeile 14). In Zeile 13 wird das erforderliche („required“) Attribut „languageSystem“ eingeführt. Dieses Attribut im Root-Element „wlmml“ legt fest in welcher Systemsprache die WLMML-Datei angezeigt wird (z. B. die Sprache der Textbausteine auf Systemschaltflächen). Damit wird sichergestellt, dass E-Learning-Inhalte in einer Sprache für die keine Systemsprache existiert, mit Systemschaltflächen in einer vorhandenen Systemsprache angezeigt werden können. Die Sprache muss in Form einer Zeichenkette nach „ISO 639-1:2002 Codes for the representation of names of languages -- Part 1: Alpha-2 code“ (ISO-639-1 2002, WIKIPEDIA-ISO-639 2014) mit zwei Buchstaben angegeben werden. Über das Element „restriction“ (mit dem „base“-Attributwert „xs:string“) und sein Kindelement „pattern“ können die erlaubten Text-Werte eingeschränkt werden (Listing 34, Zeile 15, 16). Der „value“-Attributwert „[a-zA-Z][a-zA-Z]“ des „pattern“-Elementes legt fest,

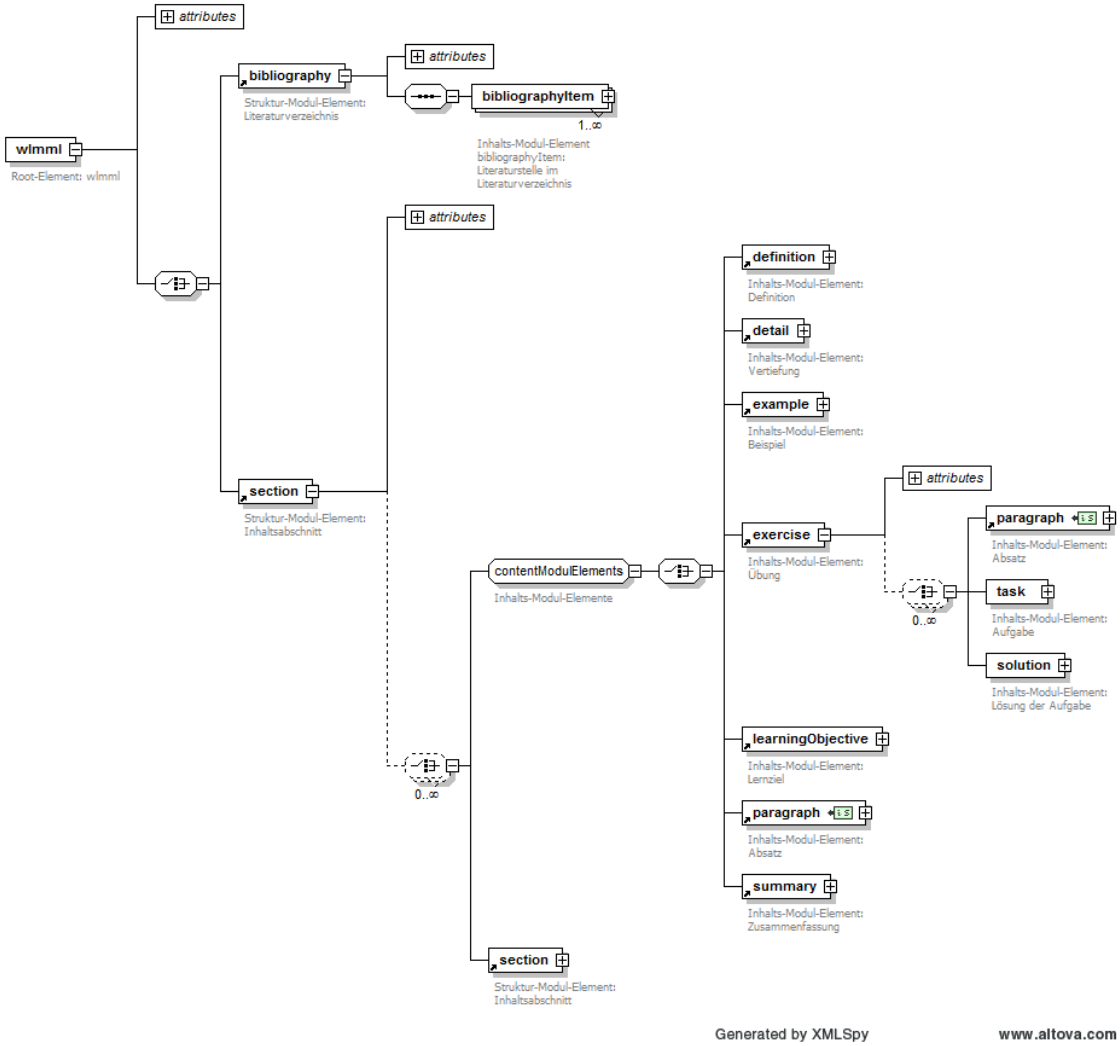
dass nur zwei Buchstaben in Groß- oder Kleinschreibung eingesetzt werden dürfen (Listing 34, Zeile 16).

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 …
3 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="wlmml"
4     targetNamespace="wlmml" elementFormDefault="qualified">
5 …
6    <xs:element name="wlmml">
7 …
8       <xs:complexType>
9         <xs:choice>
10          <xs:element ref="bibliography"/>
11          <xs:element ref="section"/>
12        </xs:choice>
13        <xs:attribute name="languageSystem" use="required">
14          <xs:simpleType>
15            <xs:restriction base="xs:string">
16              <xs:pattern value="[a-zA-Z][a-zA-Z]"/>
17            </xs:restriction>
18          </xs:simpleType>
19        </xs:attribute>
20 …
21     </xs:complexType>
22   </xs:element>
```

Listing 34: Zeichensatz „UTF-8“, Root-Element „wlmml“, Struktur-Modul-Elemente „bibliography“ und „section“ in der WLMML-XSD-Datei

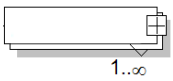
Die WLMML-XSD-Datei lässt sich in Textform (Listing 34), aber auch in Form eines Schema-Diagramms darstellen. Abbildung 57 und Abbildung 58 zeigen Ausschnitte aus der grafischen Darstellung des WLMML-Schema (WLMML-Schema-Diagramm) mit allen Elementen, die in einem WLMML-Dokument verwendet werden können. Abbildung 57 macht wie bereits beschrieben z. B. deutlich, dass im Root-Element „wlmml“ nur die beiden Struktur-Modul-Elemente „bibliography“ (Literaturverzeichnis) und „section“ (Inhaltsabschnitt) erlaubt sind. Das Element „section“ kann selbst wieder ein „section“-Element oder ein bzw. mehrere Inhalts-Modul-Elemente

enthalten (Abbildung 57, siehe auch Abbildung 56). Solche Inhalts-Modul-Elemente sind „definition“ (Definition), „detail“ (Vertiefung), „example“ (Beispiel), „exercise“ (Übung), „learningObjective“ (Lernziel), „paragraph“ (Absatz) und „summary“ (Zusammenfassung). Die Elemente „task“ (Aufgabe) und „solution“ (Lösung) werden im Inhalts-Modul-Element „exercise“ eingebunden (Abbildung 57). Das Element „formula“ erscheint nur innerhalb der Elemente „paragraph“ und „td“. Inhalts-Modul-Elemente führen die von den Struktur-Modul-Elementen vorgegebene hierarchische Strukturierung eines Dokumentes weiter fort.



Legende

Obligatorisches Element, das ein globales Element referenziert (Pfeil unten links), Plus-Zeichen = complex content (d.h. mindestens ein Child-Element oder -Attribut) Linien oben links = simple content (z. B. Text) oder complex content (z. B. Text mit Elementen)



Obligatorisches mehrmals vorkommendes Element, Häufigkeit des Auftretens 1 bis ∞

Choice-Kompositor, nur ein einzelnes Element der Auswahl darf auftreten, gestrichelte Umrandung = optional (bei durchgezogener Linie obligatorisch) Häufigkeit des Auftretens 0 bis ∞

Sequence-Kompositor, das Element muss genau in der im Schema-Diagramm angegebenen Reihenfolge erscheinen

Abbildung 57: Ausschnitt aus dem WLMML-Schema bis Element „paragraph“ (WLMML-Schema-Diagramm erzeugt von XML-Software „Altova XMLSpy“) mit Legende, siehe auch PAAR et al. (2006, S. 3)

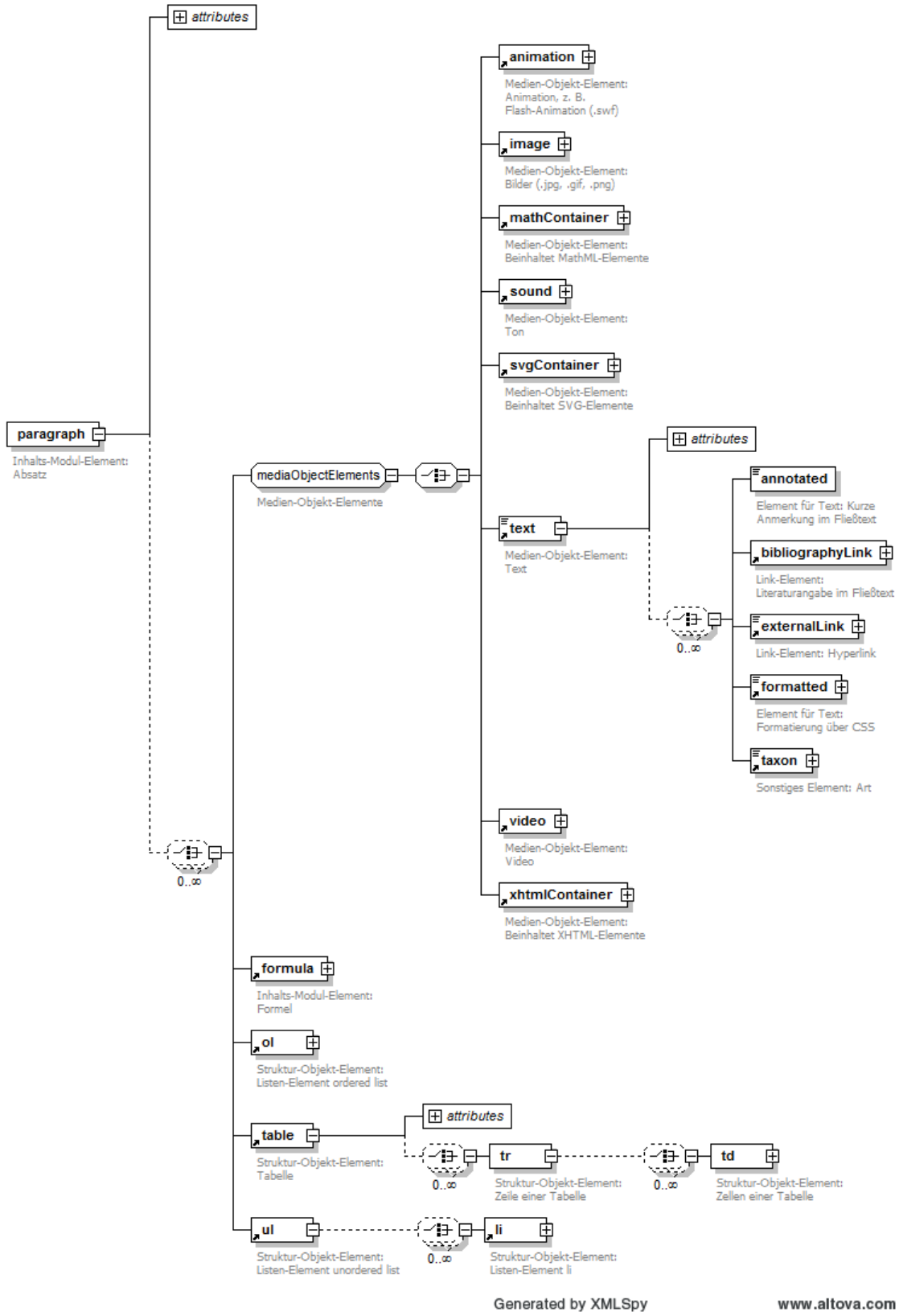


Abbildung 58: Ausschnitt aus dem WLMML-Schema ab Element „paragraph“ (WLMML-Schema-Diagramm erzeugt von XML-Software „Altova XMLSpy“), Legende siehe Abbildung 57

Für das bereits angesprochene Struktur-Modul-Element „section“ stellen die Anweisungen in Listing 35 sicher, dass in allen Kind-Elementen aller „section“-Elemente die „label“-Attributwerte nur einmal vorkommen (GAO et al. 2012, 3.11 Identity-constraint Definitions, BINSTOCK et al. 2003, S. 343 ff, WALMSLEY 2002, S. 380 ff, SKULSCHUS & WIEDERSTEIN 2004, S. 218 ff). Damit ist in einem WLMML-Dokument jeder „label“-Attributwert eindeutig („unique“, Listing 35, Zeile 3). Für diese Funktionalität wird das „unique“-Element eingesetzt (Listing 35, Zeile 3). Über den XPath-Ausdruck des „xpath“-Attributwertes im „selector“-Element werden alle Kindelemente des aktuellen Knoten (entspricht „section“-Knoten) angesprochen (Listing 35, Zeile 4). Der XPath-Ausdruck „@label“ als „xpath“-Attributwert im „field“-Element wählt schließlich das Attribut „label“ aus, das einen eindeutigen Attributwert aufweisen soll (Listing 35, Zeile 5). Das „label“-Attribut kann nur im „paragraph“-Element verwendet werden.

```
1 <xs:element name="section">
2 …
3  <xs:unique name="labelMustBeUnique">
4    <xs:selector xpath=".//*"/>
5    <xs:field xpath="@label"/>
6  </xs:unique>
7 </xs:element>
```

Listing 35: Eindeutiger „label“-Attributwert in allen Kind-Elementen des Struktur-Modul-Elementes „section“ (Ausschnitt aus der WLMML-XSD-Datei)

Das Inhalts-Modul-Element „paragraph“ ermöglicht es im Lerninhalt einen Absatz anzugeben. Der Inhalt dieses Elementes wird übersetzt. Das Attribut „label“ dieses Elementes muss einen „integer“-Wert über Null aufweisen. Dies wird mit Hilfe des „restriction“-Elementes mit seinem „base“-Attribut und des „minExclusive“-Elementes mit seinem „value“-Attribut erreicht (Listing 36, Zeile 3 und 4). Der „label“-Attributwert ist zusätzlich im gesamten WLMML-Dokument eindeutig (siehe Listing 35). Auf der Grundlage dieses eindeutigen Integer-Wertes wird später mit Hilfe von Anweisungen im WLMML-Framework bei der Auswahl eines Absatzes der zugehörige anderssprachige Absatz in der jeweiligen Sprachdatei gesucht und angezeigt (siehe Kapitel 4.3 WLMML-Framework).

```
1 <xs:attribute name="label" use="required">
2  <xs:simpleType>
```

```
3    <xs:restriction base="xs:integer">
4       <xs:minExclusive value="0"/>
5    </xs:restriction>
6 </xs:simpleType>
```

Listing 36: Attribut „label“ des Elementes „paragraph“, „label“-Attributwert als „integer“-Wert über Null, (Ausschnitt aus der WLMML-XSD-Datei)

Ob ein Absatz zur Übersetzung angeboten wird, kann über ein weiteres Attribut des Elementes „paragraph“ mit dem Namen „translation“ gesteuert werden. Der Standardwert „yes“ muss in einem WLMML-Dokument nicht explizit angegeben werden (Listing 37, Zeile 1). Er stellt sicher, dass jeder Absatz per Standardvorgabe mit einem Hyperlink zur Übersetzung angeboten wird (über Anweisungen im WLMML-Framework). Gibt man „no“ an, erscheint dieser Hyperlink nicht. Über das Element „restriction“ (mit dem „base“-Attributwert „xs:NMTOKEN“) und sein Kindelement „enumeration“ können die einzig einsetzbaren Schlüssel-Werte vorgegeben werden (Listing 37, Zeile 1).

```
1 <xs:attribute name="translation" default="yes">
2  <xs:simpleType>
3    <xs:restriction base="xs:NMTOKEN">
4       <xs:enumeration value="no"/>
5       <xs:enumeration value="yes"/>
6    </xs:restriction>
7  </xs:simpleType>
8 </xs:attribute>
```

Listing 37: Eindeutiger „label“-Attributwert in allen Kind-Elementen des Struktur-Modul-Elementes „section“ (Ausschnitt aus der WLMML-XSD-Datei)

Das zweite Struktur-Modul-Element „bibliography“ beinhaltet nur ein oder mehrere Inhalts-Modul-Elemente „bibliographyItem“ (Abbildung 57).

Als Struktur-Objekt-Elemente dienen „ol“ („ordered list“, nummerierte Liste), „ul“ („unordered list“, unsortierte Liste) und „table“ (Tabelle) (Abbildung 58). Diese von (X)HTML übernommenen Elemente werden in WLMML umgesetzt, um eine Validierung gegen die WLMML-Schema-Datei zu ermöglichen und sollen helfen Inhalte anschaulich zu strukturieren. Sie könnten aber auch mit ihrer Funktion über ein „xhtmlContainer“-Element als Standard-XHTML-Elemente eingebracht werden.

Allerdings lassen sich Inhalte von xhtmlContainer“-Elementen nicht gegen das WLMML-Schema validieren.

Medien-Objekt-Elemente werden durch die Elemente „animation“ (Animation), „image“ (Bild), „mathContainer“ (Container für MathML-Elemente), „sound“, „svgContainer“ (Container für SVG-Elemente), „text“ (Text), „video“ (Video) und „xhtmlContainer“ (Container für XHTML-Elemente) abgebildet (Abbildung 58). Sie stellen den eigentlichen Inhalt dar. Die Elemente „animation“, „image“, „sound“, „video“ referenzieren andere externe Dateien. Über die Medien-Objekt-Elemente „mathContainer“ können MathML-Elemente, über „svgContainer“ SVG-Elemente und über „xhtmlContainer“ XHTML-Elemente ohne WLMML-Elemente eingebunden werden. Es wird allerdings innerhalb der Elemente „mathContainer“, „svgContainer“ und „xhtmlContainer“ nicht validiert und kein WLMML-Element berücksichtigt. Listing 38 zeigt am Beispiel des „mathContainer“-Elementes wie diese drei Elemente in die WLMML-XSD-Datei integriert sind. Das Element kann nicht („minOccurs“ gleich 0) oder mehrfach („maxOccurs" gleich „unbounded“) vorkommen (Listing 38, Zeile 4). Das „any“-Element erlaubt es beliebige Elemente einzubinden, die nicht durch das WLMML-Schema spezifiziert sind. Ihr Namespace kann mit angegeben werden (Listing 38, Zeile 5, GAO et al. 2012, 3.10 Wildcards, 3.4.2.6 Examples of Complex Type Definitions).

```
1 <xs:element name="mathContainer">
2 …
3  <xs:complexType>
4    <xs:choice minOccurs="0" maxOccurs="unbounded">
5      <xs:any namespace=http://www.w3.org/1998/Math/MathML
6        processContents="lax"/>
7    </xs:choice>
8 …
9  </xs:complexType>
10 </xs:element>
```

Listing 38: „mathContainer“-Element (Ausschnitt aus der WLMML-XSD-Datei)

Die Elemente „annotated“ (Anmerkung), „formatted“ (Formatierung), „taxon“ (Taxon), „bibliographyLink“ (Literaturstelle) und „externalLink“ (Hyperlink) werden im Medien-Objekt-Element „text“ eingebunden (Abbildung 58). Das Element „taxon“ soll z. B. für

Inhalte der „Life Sciences“ die Möglichkeit schaffen taxonomische Angaben (z. B. Art) besonders zu kennzeichnen.

Evtl. mit WLMML-Elementnamen identische MathML-/SVG-/XHTML-Elementnamen werden durch ihren “XML namespace” unterschieden (3.3.4 XML-Namensräume).

Neben Angaben für Elemente sind in der WLMML-Schema-Datei spezifische Angaben für Attribute festgelegt (siehe auch Erläuterungen zu den Elementen „wlmml“ und „paragraph“ weiter oben). Attribute werden zur leichteren Handhabung in der WLMML-Schema-Datei meist in Form von Attributgruppen („xs:attributeGroup“) angelegt (Listing 39, Zeile 1 ff) und in dem jeweiligen Element, in dem sie Verwendung finden sollen, referenziert (Listing 40, Zeile 5). Listing 39 (Zeile 7 bis 10) zeigt zusätzlich wie über das „restriction“-Element mit seinen Kindelementen „minInclusive“ sowie „maxInclusive“ die einsetzbaren „integer“-Werte für das Attribut „width“ auf den Wertebereich von 1 bis 1600 eingegrenzt wird.

```
1  <xs:attributeGroup name="size">
2  …
3     <!-- ......... Attribut width ......... -->
4     <xs:attribute name="width">
5  …
6        <xs:simpleType>
7           <xs:restriction base="xs:integer">
8              <xs:minInclusive value="1"/>
9              <xs:maxInclusive value="1600"/>
10             </xs:restriction>
11          </xs:simpleType>
12       </xs:attribute>
13       <!-- ......... Attribut height ......... -->
14       <xs:attribute name="height">
15 …
16       </xs:attribute>
17    </xs:attributeGroup>
```

Listing 39: Attributgruppe „size“ mit den Attributen „width“ und „height“ (Ausschnitt aus der WLMML-XSD-Datei)

```
1 <xs:element name="animation">
2 …
3    <xs:complexType>
4 …
5       <xs:attributeGroup ref="size"/>
6    </xs:complexType>
7 </xs:element>
```

Listing 40: Referenzierung der Attributgruppe „size“ im Element „animation“ (Ausschnitt aus der WLMML-XSD-Datei)

Alle Attribute („title“, „bibliographyKey“, „type“, „align“, „colspan“, „height“, „rowspan“, „style“, „uri“, „valign“, „width“, „label“, „translation“, „languageSystem“, „translated“) werden im Kapitel 4.2.2 Elemente bei dem jeweiligen Element bzw. im Kapitel 4.2.3 Attribute beschrieben.

Anhang C zeigt den vollständigen Quelltext des WLMML-Schema (WLMML-XSD-Datei). Anhang D verweist auf die beiliegende CD-ROM mit einer detaillierten digitalen WLMML-Schema-Dokumentation im HTML- und „Microsoft Word“-Format („.docx“-Datei).

In den nächsten Kapiteln wird genauer auf die verschiedenen Elemente und Attribute eingegangen.

4.2.2 Elemente

Im Folgenden werden die einzelnen WLMML-Elemente mit ihren Kind-Elementen und Attributen beschrieben (siehe auch Anhang C und Anhang D). Erforderliche Attribute („required“) sind in den Tabellen kursiv hervorgehoben. Auf die Attribute selbst wird im Kapitel 4.2.3 Attribute eingegangen.

4.2.2.1 Root-Element

Das Element „wlmml“ ist das Root-Element (auch Wurzel-Element oder Dokument-Element). Jedes XML-Dokument weist nur ein Root-Element auf, das alle anderen Elemente beinhaltet. Listing 41 zeigt einen Ausschnitt aus einem WLMML-Dokument mit der Angabe des Zeichensatzes (Zeile 1, siehe Kapitel 3.3.1 Zeichensätze), des Root-Elementes (Zeile 3) und eines Kind-Elementes „section“ (Zeile 4). Das Wurzel-Element „wlmml“ darf nur die beiden Kind-Elemente „bibliography“ und „section“ beinhalten (Tabelle 8).

```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 …
3 <wlmml xmlns="wlmml" … >
4 <section title="WLMML-Elemente">
5 …
6 </section>
7 </wlmml>
```

Listing 41: Root-Element „wlmml“ mit einem Kind-Element „section“ (Ausschnitt aus einem WLMML-Dokument)

Tabelle 8: Root-Element (Wurzel-Element) „wlmml“

Elementname	Mögliche Kind-Elemente	Attribute
wlmml	bibliography, section	*languageSystem*, translated

4.2.2.2 Struktur-Modul-Elemente

Das Struktur-Modul-Element „bibliography“ bildet (mit seinem Kind-Element „bibliographyItem“) ein Literaturverzeichnis ab (Listing 42, Tabelle 9).

```
1 <bibliography title="LITERATURVERZEICHNIS">
2  <bibliographyItem bibliographyKey="STREHL et al. 2005">
3    <text>STREHL, O., QUEDNAU, H. D., PAAR, S. (2005): … </text>
4  </bibliographyItem>
5  <bibliographyItem bibliographyKey="IEEE LTSC 2005">
6    <text>IEEE LTSC (2005): … </text>
7  </bibliographyItem>
8 </bibliography>
```

Listing 42: Element „bibliography“ mit Kind-Elementen „bibliographyItem“ (Ausschnitt aus einem WLMML-Dokument)

Das Struktur-Modul-Element „section“ gliedert den E-Learning-Inhalt in Inhaltsabschnitte und strukturiert ihn weiter mit seinen Kind-Elementen (Listing 43, Tabelle 9).

```
1 <section title="WLMML-Elemente">
2  <paragraph label="1"> … </paragraph>
3 </section>
```

Listing 43: Element „section“ mit einem Kind-Element „paragraph“ (Ausschnitt aus einem WLMML-Dokument)

Tabelle 9: Struktur-Modul-Elemente „bibliography“ und „section“

Elementname	Mögliche Kind-Elemente	Attribute
bibliography	bibliographyItem	*title*
section	section, definition, detail, learningObjective, example, exercise, paragraph, summary	*title*

4.2.2.3 Inhalts-Modul-Elemente

Das Element „bibliographyItem“ spiegelt eine Literaturangabe im Literaturverzeichnis wieder. Das Element „definition“ erlaubt es eine Definition wiederzugeben. Soll eine Vertiefung zu bestimmten E-Learning-Inhalten angeboten werden, kann dies über das Element „detail“ erreicht werden. Der „title“-Attributwert des Elementes „detail“ wird als farbig hinterlegter Text angezeigt (mit Hilfe von Anweisungen im WLMML-Framwork). Klickt der Anwender mit der linken Maustaste darauf, wird unterhalb dieses Textes der Inhalt des Elementes „detail“ angezeigt. Das Element „example“ bildet ein Beispiel ab und das Element „exercise“ eine Übung. Zu jeder Übung kann ein Absatz, eine Aufgabe und eine Lösung angeboten werden (Elemente „paragraph“, „solution“ und „task“).

Das Element „formula“ ermöglicht es eine mathematische Formel bzw. Gleichung anzugeben. Es tritt nur innerhalb der Elemente „paragraph“ und „td“ auf. Bei extern zur Verfügung gestellten Plugins kann eine „.swf“-Datei, „.svg“-Datei oder MathML-Datei z. B. auch über das „animation“-Element oder über das „externalLink“-Element mit einem „type“-Attributwert “formula“ eingebunden werden.

Das Element „learningObjective“ erlaubt es ein Lernziel anzugeben (vgl. das Element „learning-objectives“ unter KOPER et al. 2003 b).

Das unspezifizierte Inhaltselement „paragraph“ ermöglicht es im E-Learning-Inhalt einen Absatz zu erzeugen (Listing 44, siehe auch Ausführungen zum Element „paragraph“ im Kapitel 4.2.1 Schema). Ein leerer Absatz erzeugt einen

Zeilenumbruch. Für mehrere Zeilenumbrüche können leere „text“-Elemente eingesetzt werden. Mehrere leere Absätze werden nicht in mehrere Zeilenumbrüche umgesetzt.

Über die Elemente „task“ und „solution“ innerhalb des Elementes „exercise“ können eine Aufgabenstellung („task“) und eine Aufgabenlösung („solution“) angeboten werden. Der „title“-Attributwert des Elementes „solution“ wird als farbig hinterlegter Text dargestellt. Klickt der Anwender mit der linken Maustaste darauf wird unterhalb dieses Textes die Aufgabenlösung angezeigt (mit Hilfe von Anweisungen im WLMML-Framwork).

Das Element „summary“ bietet die Möglichkeit eine Zusammenfassung abzubilden.

Als Beispiel für den Einsatz eines Inhalts-Modul-Elementes zeigt Listing 44 das „paragraph“-Element mit den Kind-Elementen „text“ und „image“.

```
1 <paragraph label="1">
2    <text>Beispiel</text>
3    <image title="Holz" uri="../../media/10003_00003_holz.jpg"/>
4 </paragraph>
```

Listing 44: Inhalts-Modul-Element „paragraph“ mit Kind-Elementen „text“ und „image“ (Ausschnitt aus einem WLMML-Dokument)

In der Tabelle 10 werden die Inhalts-Modul-Elemente mit ihren Kind-Elementen und Attributen aufgeführt.

Tabelle 10: Inhalts-Modul-Elemente

Elementname	Mögliche Kind-Elemente	Attribute
bibliographyItem	externalLink, text	*bibliographyKey*
definition	paragraph	*title*
detail	paragraph	*title*
example	paragraph	*title*
exercise	paragraph, task, solution	*title*
formula	image, mathContainer, text	*title*

learningObjective	paragraph	*title*
paragraph	animation, image, mathContainer, sound, svgContainer, text, video, xhtmlContainer, formula, ol, ul, table	*label*, translation
solution	paragraph	*title*
summary	paragraph	*title*
task	paragraph	*title*

4.2.2.4 Struktur-Objekt-Elemente

In WLMML können Listen und Tabellen zur Strukturierung von Lerninhalten eingesetzt werden.

4.2.2.4.1 Listen

Die Struktur-Objekt-Elemente „ol“ („ordered list“, nummerierte Liste) und „ul“ („unordered list“, unsortierte Liste) bieten die Möglichkeit Listen darzustellen. Die einzelnen Aufzählungsinhalte werden über das Element „li“ („list item“, Aufzählungspunkt) abgebildet (Tabelle 11). Der Wert des optionalen Attributes „type“ legt fest (Listing 45, Zeile 2), von welchem Typ das Element „li“ ist („definition“, „detail“, „example“, „exercise“, „formula“). Sollen z. B. Definitionen in einer Zusammenstellung aufgelistet werden, kann dies auf diese Art und Weise erreicht werden (mit Hilfe von Anweisungen im WLMML-Framework). Listen können auch verschachtelt eingesetzt werden (Listing 45, Zeile 4).

```
1 <ol>
2 <li type="definition"><text> … </text></li>
3 <li><text>Aufzählungspunkt </text>
4    <ul>
5       <li><text> … </text></li>
6    </ul>
7 </li>
8 </ol>
9 <ul>
10   <li><text> … </text></li>
11   <li><text> … </text></li>
12 </ul>
```

Listing 45: Struktur-Objekt-Elemente „ol“ und „ul“ (Ausschnitt aus einem WLMML-Dokument)

In der Tabelle 11 werden die Listen-Elemente mit ihren Kind-Elementen und Attributen aufgeführt.

Tabelle 11: Listen-Elemente

Elementname	Mögliche Kind-Elemente	Attribute
li	animation, image, mathContainer, sound, svgContainer, text, video, xhtmlContainer, ol, ul	type
ol	li	
ul	li	

4.2.2.4.2 Tabellen

Über das Struktur-Objekt-Element „table“ lässt sich eine Tabelle in den Lerninhalt einbinden (Listing 46, Tabelle 12). Das Attribut „align“ dieses Elementes wird eingesetzt, um Text um eine Tabelle schreiben zu können (Listing 46, Zeile 3). Der Standardattributwert ist „left“. Daneben sind noch „right“ und „center“ möglich. Rahmen, „cellpadding“ und „cellspacing“ werden über XSLT-Anweisungen im WLMML-Framework gesetzt. Das Element „tr“ spiegelt eine Zeile und das Element „td“ eine Zelle einer Tabelle wieder (Listing 46, Tabelle 12). Die Attribute colspan und rowspan verbinden Zellen (Listing 46, Zeile 3 und 7, siehe Kapitel 4.2.3.4 Attribute u.a. für Tabellen und Medienobjekte).

```
1 <table align="center" title="Titel der Tabelle">
2  <tr>
3    <td align="center" colspan="2"><text>erste Zelle</text></td>
4    <td><text>zweite Zelle</text></td>
5  </tr>
6  <tr>
7    <td rowspan="2"><text>dritte Zelle</text></td>
8    <td><text>vierte Zelle</text></td>
9    <td><text>fünfte Zelle</text></td>
10   </tr>
11   <tr>
12     <td><text>sechste Zelle</text></td>
13     <td><text>siebte Zelle</text></td>
```

```
14   </tr>
15 </table>
```

Listing 46: Struktur-Objekt-Element „table“ (Ausschnitt aus einem WLMML-Dokument)

Tabelle 12: Elemente einer Tabelle

Elementname	Mögliche Kind-Elemente	Attribute
table	tr	*title*, align
td	animation, image, mathContainer, sound, svgContainer, text, video, xhtmlContainer, formula, ol, ul	colspan, rowspan, valign, align, width, height
tr	td	

4.2.2.5 Medien-Objekt-Elemente

Über Medien-Objekt-Elemente können die eigentlichen E-Learning-Inhalte abgebildet werden.

Das Animationsobjekt „animation“ erlaubt es im Fließtext („inline“) Animationen einzubinden. Die Animationsdatei (Flash-Animation, „.swf“-Datei) wird über das Attribut „uri“ mit ihrer Pfadangabe referenziert. Auch Bild- („image“-Element) und Ton-Dateien („sound“-Element) werden über ihr „uri“-Attribut in das WLMML-Dokument integriert (siehe z. B. Listing 44, Zeile 3). Bei den Elementen „sound“ und „video“ wird der „title“-Attributwert des Elementes als Hyperlink in den Fließtext eingebunden (mit Hilfe von Anweisungen im WLMML-Framework). Grundsätzlich kann zusätzlich jede externe Datei über das „externalLink“-Element eingebunden werden (siehe Kapitel 4.2.2.6 „Link“-Elemente).

Das Medien-Objekt-Element „mathContainer“ ermöglicht es MathML-Elemente ohne den Einsatz von WLMML-Elementen einzubinden. Gleiches gilt für die Medien-Objekt-Elemente „svgContainer“ (Einbindung von SVG-Elementen) und „xhtmlContainer“ (Einbindung von XHTML-Elementen). Der Inhalt dieser drei Medien-Objekt-Elemente wird allerdings nicht gegen das WLMML-Schema validiert und es wird kein WLMML-Element durch Anweisungen im WLMML-Framework berücksichtigt.

Listing 47 zeigt als Beispiel wie ein „mathContainer“-Element mit MathML-Elementen (Listing 47, Zeile 2 bis 7) eingebunden werden kann. Der Attributwert „title“ erscheint unter dem mathematischen Ausdruck (Text: „Abbildung: …“). Soll der mathematische Ausdruck im Fließtext angezeigt wird, müssen keine weiteren Angaben gemacht werden. Bei der Anzeige in einer separaten Zeile kann dem Attribut „display“ der Wert “block“ zugewiesen werden (Listing 47, Zeile 2). Der Namespace ist wie beschrieben anzugeben (Listing 47, Zeile 2, „xmlns“). Bei einem „svgContainer“-Element lautet der bei dem Kind-Element „svg“ einzutragende Namespace „http://www.w3.org/2000/svg“ (vgl. Listing 47, Zeile 2).

```
1 <mathContainer title="MathContainer - mathematischer Ausdruck">
2  <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
3    <mfrac>
4      <mi>a</mi>
5      <mi>b</mi>
6    </mfrac>
7  </math>
8 </mathContainer>
```

Listing 47: „mathContainer“-Element mit MathML-Elementen (Ausschnitt aus einem WLMML-Dokument)

Listing 48 stellt dar, wie ein „xhtmlContainer“-Element mit XHTML-Elementen in einem WLMML-Dokument eingebunden werden kann. Der Namespace für das eingebundene XHTML-Element lautet „http://www.w3.org/1999/xhtml“ (Listing 48, Zeile 2).

```
1 <xhtmlContainer>
2  <table xmlns="http://www.w3.org/1999/xhtml" border="1">
3    <tr>
4      <td>Zelle 1</td>
5      <td>Zelle 2</td>
6    </tr>
7  </table>
8 </xhtmlContainer>
```

Listing 48: „xhtmlContainer“-Element mit XHTML-Elementen (Ausschnitt aus einem WLMML-Dokument)

Das „text“-Element stellt sicher, dass Textinhalt angegeben werden kann. Die „align“-Attributwerte „left“, „center“ und „right“ richten den Text in die gewünschte Richtung aus. Die Verschachtelung von „formatted“-Elementen in einem „text“-Element ist möglich.

In der Tabelle 13 werden die Medien-Objekt-Elemente mit ihren Kind-Elementen und Attributen aufgeführt.

Tabelle 13: Medien-Objekt-Elemente

Elementname	Mögliche Kind-Elemente	Attribute
animation	EMPTY (kein Inhalt)	*title*, *uri*, width, height
image	EMPTY	*title*, *uri*
mathContainer	MathML -Elemente	*title*
sound	EMPTY	*title*, *uri*
svgContainer	svg-Elemente	*title*
text	annotated, bibliographyLink, externalLink, formatted, PCDATA (reiner Text, BRAY et al. 2008, 3.2.2 Mixed Content), taxon	align
video	EMPTY	*title*, *uri*
xhtmlContainer	xhtml-Elemente	

4.2.2.6 „Link“-Elemente

Das Element „bibliographyLink“ stellt eine Literaturstelle im Fließtext dar. Das Element verweist über sein Attribut „bibliographyKey“ auf eine Stelle im Literaturverzeichnis. Das „bibliographyLink“-Element tritt nur im Element „text“ auf (Listing 49).

```
1 <text>
2 <bibliographyLink bibliographyKey="STREHL et al. 2005"/> weisen
3   darauf hin, dass ...
4 </text>
```

Listing 49: „bibliographyLink“-Element (Ausschnitt aus einem WLMML-Dokument)

Das Element „externalLink“ bietet die Möglichkeit einen Hyperlink einzubinden. Über die Attribute „width“ und „height“ kann die Fenstergröße des per JavaScript zu öffnenden Fensters angegeben werden (Anweisungen im WLMML-Framework, Funktionalität je nach Browser unterschiedlich). Es wird auf ein externes Objekt referenziert. Werden „width“ und „height“ nicht angegeben wird ohne JavaScript ein neues Browserfenster geöffnet. Das Element „externalLink“ tritt nur im Element „text“ auf (Listing 50). Das optionale Attribut „type“ legt fest, von welchem Typ der Hyperlink ist. Über den „type“-Attributwert „www“ kann mit Hilfe von Anweisungen im WLMML-Framework eine Hyperlink-Liste zusammengestellt werden. Die Werte „animation“, „image“ und „video“ werden eingesetzt, um die externe Fenstergröße definieren zu können. Ohne die Angaben von „width“ und „height“ werden diese Objekte im Fließtext („inline“) eingebunden.

```
1 <text>
2 Die <externalLink uri="http://www.laerche.de/" title="Lärche">
3 Lärche</externalLink> ist eine Pionierbaumart.
4 </text>
```

Listing 50: „externalLink“-Element (Ausschnitt aus einem WLMML-Dokument)

In der Tabelle 14 werden die Link-Elemente mit ihren Kind-Elementen und Attributen aufgeführt.

Tabelle 14: Link-Elemente

Elementname	Mögliche Kind-Elemente	Attribute
bibliographyLink	EMPTY	*bibliographyKey*, width, height
externalLink	image, PCDATA	*title*, *uri*, width, height, type

4.2.2.7 Elemente für Text

Das Element „annotated“ ermöglicht es eine kurze Anmerkung im Fließtext („inline“) anzugeben. Es tritt nur im Element „text“ auf (Listing 51, Zeile 3). Über Anweisungen im WLMML-Framework erscheint im Fließtext ein Symbol, das beim Überfahren mit der Maus den Textinhalt des „annotated“-Elementes anzeigt.

Zur Formatierung von Text kann das Element „formatted“ herangezogen werden. Es tritt nur im Element „text“ auf (Listing 51, Zeile 2). Eine Verschachtelung von

„formatted“-Elementen ist möglich und erlaubt auf diese Art und Weise die Kombination verschiedener Formatierungen. Das Attribut „style“ gibt eine CSS-Klasse an, die über Anweisungen im WLMML-Framework die Formatierung mit dieser CSS-Klasse umsetzt.

```
1 <text>
2  <formatted style="bold">Hervorgehobener</formatted>
3  Text <annotated>Kurze Anmerkung im Fließtext</annotated>
4 </text>
```

Listing 51: „formatted“- und „annotated“-Element (Ausschnitt aus einem WLMML-Dokument)

In der Tabelle 15 werden die Elemente für Text aufgeführt.

Tabelle 15: Elemente für Text

Elementname	Mögliche Kind-Elemente	Attribute
annotated	PCDATA	
formatted	formatted, PCDATA	style

4.2.2.8 Sonstige Elemente

Über das Element „taxon“ kann ein taxonomischer Ausdruck (z. B. Art) angegeben werden. Das Element tritt nur im Element „text“ auf (Listing 52). Über Anweisungen im WLMML-Framework erscheint der Text-Inhalt des Elementes „taxon“ (hier: „Fichte“, Listing 52, Zeile 2) als Hyperlink, der beim Überfahren mit der Maus den „title“-Attributwert (hier: „picea abies“, Listing 52, Zeile 2) anzeigt.

```
1 <text>
2 Die <taxon title="picea abies">Fichte</taxon> …
3 </text>
```

Listing 52: „taxon“-Element (Ausschnitt aus einem WLMML-Dokument)

In der Tabelle 16 wird das „taxon“-Element mit seinem Kind-Element und Attribut aufgeführt.

Tabelle 16: Element „taxon“

Elementname	Mögliche Kind-Elemente	Attribute
taxon	PCDATA	*title*

4.2.3 Attribute

Im Folgenden werden die WLMML-Attribute beschrieben (siehe auch Anhang C und Anhang D). Bei welchen Elementen die Attribute eingesetzt werden können geht aus den Tabellen (Spalte Attribute) der vorangegangenen Kapitel zu den einzelnen Elementen hervor (siehe Kapitel 4.2.2 Elemente).

4.2.3.1 Allgemeines Attribut „title“

Das allgemeine Attribut „title“ ermöglicht eine Beschreibung des Element-Inhaltes (Tabelle 17). Das Attribut kann bei verschiedenen Elementen auftreten (z. B. bei Element „section“ Listing 42 Zeile 1, bei Element „bibliography“ Listing 43 Zeile 1, bei Element „image“ Listing 44 Zeile 3, siehe allgemeinen Hinweis in Kapitel 4.2.3 Attribute). Über Anweisungen im WLMML-Framework wird der Attributwert z. B. als Überschrift über das jeweilige Element gesetzt oder erscheint beim Überfahren des jeweiligen Elementes mit der Maus (Anweisungen in der XSLT-Datei „main.xsl“, siehe Kapitel 4.3 WLMML-Framework).

Tabelle 17: Allgemeines Attribut „title“

Attributname	Werte	Beispiel
title	Zeichenkette	title = "Überschrift, Kurzbeschreibung"

4.2.3.2 Attribut für die Elemente „bibliographyItem“ und „bibliographyLink“

Das Attribut „bibliographyKey“ wird benötigt, um Querverweise aus Literaturangaben im Fließtext zum Literaturverzeichnis herstellen zu können (Tabelle 18). Es tritt nur in den Elementen „bibliographyItem“ (Listing 42, Zeile 2 und 5) und „bibliographyLink“ (Listing 49, Zeile 2) auf. Der Wert des „bibliographyKey“-Attributes eines „bibliographyLink“-Elementes im Fließtext muss mit dem Wert des „bibliographyKey“-Elementes des „bibliographyItem“-Elementes im Literaturverzeichnis übereinstimmen. In diesem Fall wird durch Anweisungen im WLMML-Framework der „bibliographyKey“-Attributwert als Hyperlink angezeigt und führt zur gewünschten Literaturstelle im Literaturverzeichnis.

Tabelle 18: Attribut „bibliographyKey“

Attributname	Werte	Beispiel
bibliographyKey	Zeichenkette	bibliographyKey = "STREHL et al. 2005“

4.2.3.3 Attribut für die Elemente „externalLink “und „li“

Das Attribut „type“ tritt bei den Elementen „externalLink“ und „li“ auf und legt mit vordefinierten Werten den „Typ“ dieser beiden Elemente fest.

Wird das Attribut im Element „externalLink“ verwendet, sind die in Tabelle 19 (Spalte Werte, siehe „externalLink“) vordefinierten Attributwerte möglich. Einer dieser Werte ist „definition“. So kann im Fließtext ein Hyperlink („externalLink“) vom Typ „definition“ z. B. zu einer Datei mit einer Definition führen. Alle Hyperlinks dieses Typs lassen sich anschließend mit Hilfe von Anweisungen im WLMML-Framework zu einer Hyperlink-Liste verschiedener Definitionen zusammenstellen. Allgemein können damit externe Dateien verschiedenen Inhaltstyps (je nach „type“-Attribut-Wert) über einen Hyperlink referenziert und über ihren „type“-Attributwert zusammengefasst werden. Gleiches gilt für „li“-Elemente, die das „type“-Attribut aufweisen. Die möglichen Attributwerte im Element „li“ sind ebenfalls in Tabelle 19 (Spalte Werte, siehe „li“) aufgeführt.

Tabelle 19: Attribut „type“ für die Elemente „externalLink “und „li“

Attributname	Werte	Beispiel
type	externalLink: • animation • definition • detail • example • exercise • formula • image • learningObjective • solution • summary • task • taxon • translation • video • www li: • definition • detail • example • exercise • formula	type = "definition“

4.2.3.4 Attribute u.a. für Tabellen und Medienobjekte

Die Attribute „align“, „colspan“, „rowspan“ und „valign“ können in Tabellen eingesetzt werden. Über das Attribut „align“ lässt sich eine Tabelle im Ganzen (siehe z. B. Listing 46, Zeile 1), sowie der Inhalt einer Tabellenzelle ausrichten (siehe z. B. Listing 46, Zeile 3). Es wird im „table“-Element und im „td“-Element eingesetzt. Wie bei (X)HTML-Tabellen lassen sich über das Attribut „colspan“ Zellen waagrecht (siehe z. B. Listing 46, Zeile 3) und über das Attribut „rowspan“ senkrecht verbinden (siehe z. B. Listing 46, Zeile 7). Das Attribut „valign“ ermöglicht es den Inhalt einer Tabellenzelle vertikal auszurichten (Einsatz im „td“-Element, Listing 53, Zeile 3).

Die Attribute „height“ und „width“ können im Element „td“ eingesetzt werden, um die Höhe („height“) bzw. Breite („width“) einer Zelle in Pixel festzulegen (Listing 53, Zeile 3). Außerdem lassen sich darüber in den Elementen „animation“, „bibliographyLink“ und „externalLink“ die Höhe und Breite eines per JavaScript zu öffnenden Fensters festlegen (Listing 54, Zeile 2). Das Fenster wird dabei über Anweisungen im WLMML-Framework erzeugt (Funktionalität ist je nach Browser unterschiedlich). Allgemein sind die beiden Attribute in den Elementen „animation“, „bibliographyLink“, „externalLink“ und „td“ erlaubt.

```
1 <table title="Titel der Tabelle">
2  <tr>
3    <td height="200" width="400" valign="top">
4      <text>Zelle</text>
5    </td>
6  </tr>
7 </table>
```

Listing 53: Attribute „height“, „width“ und „valign“ für das Element „table“ (Ausschnitt aus einem WLMML-Dokument)

```
1 Die <externalLink uri="http://www.laerche.de/" title="Lärche"
2 height="200" width="400"> Lärche</externalLink> ist …
```

Listing 54: Attribute „height“, „width“ in einem „externalLink“-Element (Ausschnitt aus einem WLMML-Dokument)

Über das Attribut „style“ können vor allem Formatierungen von Elementinhalten vorgenommen werden (über Anweisungen im WLMML-Framework). Das Attribut

findet nur im Element „formatted“ Verwendung. Der Attributwert stellt den Namen einer CSS-Klasse innerhalb der CSS-Datei („layout.css“) des WLMML-Frameworks dar. Die beiden Attributwerte „blank“ und „hr“ bilden Ausnahmen. Der Attributwert „blank“ ergibt über XSLT-Anweisungen im WLMML-Framework ein Leerzeichen, „hr“ eine horizontale Linie. Der Attributwert „font-lucida-sans-unicode“ wurde eingeführt, da z. B. das Unicode-Zeichen „⇨“ (nach rechts weisender Pfeil, UNICODE 2013) im Browser „Microsoft Internet Explorer“ der Versionen 6 und 7 mit der Schriftart Arial (Standard-Schriftart für Fließtext im WLMML-Framework) nicht als nach rechts weisender Pfeil angezeigt wird. Stellt man die Schriftart für dieses besondere Zeichen auf „Lucida Sans Unicode“ um (style=“ font-lucida-sans-unicode“), wird das Zeichen entsprechend dargestellt.

Mit Hilfe des Attributes „uri“ („Uniform Resource Identifier“) kann eine relative Pfadangabe zu einem Objekt angegeben werden. Diese Pfadangabe wird ausgehend von der WLMML-Datei gesetzt, in der das Objekt eingefügt werden soll (siehe z. B. Listing 44, Zeile 3).

In Tabelle 20 sind die verschiedenen angesprochenen Attribute zusammen aufgeführt.

Tabelle 20: Attribute u.a. für Tabellen und Medienobjekte

Attributname	Werte	Beispiel
align	left, center, right	align = “left“
colspan	integer	colspan = “2“
height	1 - 1200	height = “500“
rowspan	integer	rowspan = “2“
style	Zeichenkette, insbesondere: • bg_blue • bg_green • bg_red • bg_yellow • blank • blue • bold • border • font-lucida-sans-unicode • green • hr • italic • red • sub • sup • underline	style = "bg_blue"
uri	URI	uri = "../../media/ 10003_00003_himalaya.jpg"

valign	top, middle, bottom	valign = “top“
width	1 -1600	width = “500“

4.2.3.5 Attribute für das Element „paragraph“

Die Attribute „label“ und „translation“ sind nur im Element „paragraph“ erlaubt (Tabelle 21).

Das Attribut „label“ muss einen im WLMML-Dokument eindeutigen (nur einmal vorkommenden) „integer“-Wert über Null aufweisen (siehe Ausführungen zum Element „paragraph“ im Kapitel 4.2.1 Schema). Über diesen eindeutigen Wert wird mit Hilfe von Anweisungen im WLMML-Framework bei der Auswahl eines Absatzes der zugehörige anderssprachige Absatz in der jeweiligen Sprachdatei gesucht und angezeigt (siehe Kapitel 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen, 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei). Ob ein Absatz zur Übersetzung angeboten wird, kann über das Attribut „translation“ mit seinen Attributwerten „yes“ und „no“ gesteuert werden. Dieses Attribut stellt mit seinem Standardwert „yes“ sicher, dass jeder Absatz mit einem Hyperlink zur Übersetzung angeboten wird (siehe Ausführungen zum Element „paragraph“ im Kapitel 4.2.1 Schema). Gibt man „no“ als Attributwert an, erscheint für diesen Absatz kein Hyperlink zur Übersetzung. Das Attribut muss nicht explizit angegeben werden (Standard-Attributwert „yes“).

Tabelle 21: Attribute „label“ und „translation“ für das Element „paragraph“

Attributname	Werte	Beispiel
label	Ganzzahlige eindeutige „integer“-Zahl	label = "1"
translation	yes, no	translation = “no“

4.2.3.6 Attribute für das Element „wlmml“

Das erforderliche Attribut „languageSystem“ im Root-Element „wlmml“ legt fest in welcher Systemsprache z. B. die Textbausteine auf Systemschaltflächen angezeigt werden. Existiert im WLMML-Framework noch keine Übersetzung der System-Textbausteine, wird die hier angegebene System-Sprache verwendet. Die Sprache muss in Form einer Zeichenkette nach „ISO 639-1:2002“ (ISO-639-1 2002, WIKIPEDIA-ISO-639 2014) mit zwei Buchstaben angegeben werden (siehe auch Ausführungen zum Root-Element „wlmml“ im Kapitel 4.2.1 Schema).

Das Attribut „translated“ des Root-Elementes „wlmml“ gibt an, in welchen Inhaltssprachen außer der aktuellen Sprache das WLMML-Dokument übersetzt vorliegt. Ein leerer Attributwert verhindert, dass in diesem WLMML-Dokument unter vordefinierten Abschnitten anklickbare Sprachkürzel-Hyperlinks angezeigt werden (über Anweisungen im WLMML-Framework). Die Anzeige der anklickbaren Sprachkürzel-Hyperlinks erfolgt über die am unteren Browserrand verborgene Hilfsleiste. Gibt man durch ein Leerzeichen getrennt verschiedene Sprachen an, werden diese als Sprachkürzel-Hyperlinks unter den vordefinierten Abschnitten angezeigt und nicht die in der XML-Datei „imsmanifest.xml“ über Metadaten angegebenen Sprachen. Die jeweilige Sprache muss in Form eines Sprachordners (z. B. „de“) mit einem identisch lautenden Ordnernamen existieren (siehe auch Kapitel 4.3.2.3.2 Sprachordner im WLMML-Framework).

Liegt etwa ein Lernmodul z. B. komplett in deutscher und englischer Sprache vor, kann ein finnischsprachiges WLMML-Dokument mit deutscher Systemsprache und ihren Übersetzungen in z. B. die Sprachen „eo“ und „fr“ (translated=“eo fr“) in die deutschen Inhalte eingebunden werden. In diesem Fall erscheinen unter den Absätzen dieses finnischsprachigen WLMML-Dokumentes (falls es vom Benutzer aktiviert wird) die Hyperlinks zu den Absätzen in den Sprachen „eo“ und „fr“. Gibt man keinen Attributwert an (translated=““), verschwindet der Hyperlink in der am unteren Browserrand verborgenen Hilfsleiste, der es erlaubt verschiedene Sprachen unter den Absätzen anzuzeigen. Das WLMML-Dokument wird damit nur in einer Sprache angeboten. Wird das Attribut „translated“ nicht angegeben, werden die Sprachen, in die Absätze übersetzt werden können, aus den Metadaten-Angaben der „organization“-Elemente in der XML-Datei „imsmanifest.xml“ geholt.

Tabelle 22 führt die beiden - nur im Element „wlmml“ erlaubten - Attribute auf.

Tabelle 22: Attribute „languageSystem“ und „translated“ für das Root-Element (Wurzel-Element) „wlmml“

Attributname	Werte	Beispiel
languageSystem	Zeichenkette nach „ISO 639-1:2002“, zwei Buchstaben	languageSystem = “de“
translated	Zeichenkette nach „ISO 639-1:2002“, zwei Buchstaben	translated=“eo fr“

4.2.4 Beispiel-WLMML-Datei

Das folgende Listing 55 zeigt ein Beispiel eines gültigen („validen“) WLMML-Dokumentes mit verschiedenen Elementen und Attributen (vergleiche auch PAAR et al. 2006, S. 3 f, zu Dateinamenskonvention siehe Kapitel 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM)). Im Prolog des WLMML-Dokumentes ist der Unicode-Zeichensatz (UTF-8) eingetragen (Zeile 1, siehe Kapitel 3.3.1 Zeichensätze). Wird die WLMML-Datei im Unicode-Format abgespeichert, können z. B. die deutschen Umlaute „ä“, „ö“ oder „ü“ direkt im Fließtext angegeben werden. In Zeile 2 wird auf die XSLT-Datei „main.xsl“ zur Transformation nach HTML verwiesen (siehe Kapitel 3.3.8.1 XSLT). Danach taucht in der Zeile 3 Root-Element „wlmml“ auf. Es beinhaltet:

- verschiedene Namespace-Angaben („wlmml“, „xsi“ (siehe GAO et al. 2012, 2.7 Schema-Related Markup in Documents Being Validated), „xhtml“, „math“, „svg“, siehe allgemein Kapitel 3.3.4 XML-Namensräume)
- einen Verweis auf die WLMML-Schemadatei („xsi:schemaLocation …“, siehe GAO et al. 2012, 2.7 Schema-Related Markup in Documents Being Validated)
- das Attribut „languageSystem“

Das auf das „wlmml“-Element folgende Element „section“ (Zeile 9) beinhaltet einen Absatz („paragraph“, Zeile 10) mit Textinhalt („text“, Zeile 11), einem Hyperlink („externalLink“, Zeile 11) und einer Grafik („image“, Zeile 13).

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet type="text/xsl" href="../../system/xsl/main.xsl"?>
3 <wlmml xmlns="wlmml"
4 xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
5 xsi:schemaLocation="wlmml ../../system/xsd/wlmml/wlmml.xsd"
6 xmlns:xhtml=http://www.w3.org/1999/xhtml
7 xmlns:math=http://www.w3.org/1998/Math/MathML
8 xmlns:svg="http://www.w3.org/2000/svg" languageSystem="de">
9  <section title="Erster Inhaltsabschnitt">
10     <paragraph label="1">
11       <text>Im <externalLink uri="http://www.himalaya.de/"
12         title="Himalaya">Himalaya</externalLink> ist ...</text>
13       <image title="Himalaya"
14         uri="../../media/10003_00003_himalaya.jpg"/>
15     </paragraph>
```

```
16   </section>
17 </wlmml>
```

Listing 55: Quelltext eines „validen“ WLMML-Dokumentes (siehe auch PAAR et al. (2006, S. 4)

Anhang E bietet eine vertiefende Anleitung (im Wordformat, „.docx-Datei“) für die Erstellung von WLMML-Dateien. Dabei werden folgende Programme behandelt:

- „Altova XMLSpy Professional Edition“ Version 2007 (Testversion)
- „Microsoft Word“ Versionen 2003/2007/2010 (Vertrieb der Software vor 10.01.2010, siehe MICROSOFT-WORD-XML-MARKUP 2014)
- „Eclipse“ Version 3.2 (inklusive „Web Tools“ Version 1.5.2)

Die Abbildung der Lerninhalte im Browser und alle weiteren Funktionalitäten werden durch das WLMML-Framework sichergestellt.

4.3 WLMML-Framework

Das im Rahmen der vorliegenden Arbeit entwickelte WLMML-Framework erlaubt es v. a. mit WLMML abgebildete Lerninhalte im Browser darzustellen. Es stellt sozusagen einen „Rahmen“ („Frame“) zur Verfügung in dem Autoren Lerninhalte erstellen können ohne sich um die Umsetzung unterschiedlicher Funktionalitäten (siehe Kapitel 4.3.2 Funktionalität) kümmern zu müssen (vergleiche auch WIKIPEDIA-FRAMEWORK 2013). Für die Erläuterung des Ausdruckes „Framework“ wird auf die von MÜNZ & GULL (2012, S. 26) für JavaScript-Frameworks festgelegte Beschreibung zurückgegriffen: „*JavaScript-Frameworks … sind Code-Bibliotheken, die für alle wichtigen und häufig verlangten Aufgaben fertige Funktionen bereitstellen und den Entwickler von Browser-Unterschieden fernhalten.*“ Der Begriff Code wird dabei auf alle im WLMML-Framework eingesetzten Programmier- und Auszeichnungssprachen wie z. B. JavaScript, XML, XSLT, HTML und CSS bezogen.

Das WLMML-Framework ist an sich keine Lernplattform (siehe Kapitel 3.9 LMS), sondern bietet mit seiner inhaltlichen Ausrichtung (WLMML-Dateien) eine Möglichkeit Lerninhalte abzubilden. Es besteht aus verschiedenen Ordnern und Dateien (XML-, XSLT-, HTML-, CSS-, JavaScript-Dateien) (siehe Kapitel 4.3.1 Verzeichnisstruktur), die eine clientseitige Verarbeitung erlauben. Die Trennung von Inhalt (XML), Grammatik (Schema), Formatierung (CSS) und Verarbeitung (XSLT, JavaScript) spiegelt sich in den verschiedenen Dateien des Frameworks wieder.

Über die WLMML-Inhaltsdateien und das WLMML-Framework ist eine Verarbeitung durch den Client (Client Software Browser), strukturierte Abbildung und Darstellung, Modularisierung und Wiederverwendung, Einbindung von Metadaten und konzeptionelle Integration der Mehrsprachigkeit bezüglich Lernmaterials umsetzbar (QUEDNAU et al. 2007, PAAR et al. 2006, siehe auch STREHL et al. 2005). Es wird dabei konsequent auf den Einsatz etablierter internationaler Standards/Normen und Spezifikationen geachtet (siehe Kapitel 4 Ergebnisse).

Die mehrsprachigen Lerninhalte sollten mit gängigen Browsern ohne zusätzliche Software-Installation online und offline betrachtet und auch in Lernmanagementsysteme wie z. B. „Clix“ (CLIX 2005), „Moodle“ oder „ILIAS“ (ILIAS 2005) eingesetzt werden können.

4.3.1 Verzeichnisstruktur

Das WLMML-Framework besteht aus mehreren Ordnern mit verschiedenen Dateien, die für die Funktionalität der Lerninhalte verantwortlich sind. Es beinhaltet sämtliche benötigten XML-, XSLT-, HTML-, CSS-, Medien- und JavaScript-Dateien (siehe Anhang F). Abbildung 59 zeigt die Verzeichnisstruktur des WLMML-Frameworks und die Dateien im „Root-Verzeichnis“ (z. B. „imsmanifest.xml“, „index.htm“).

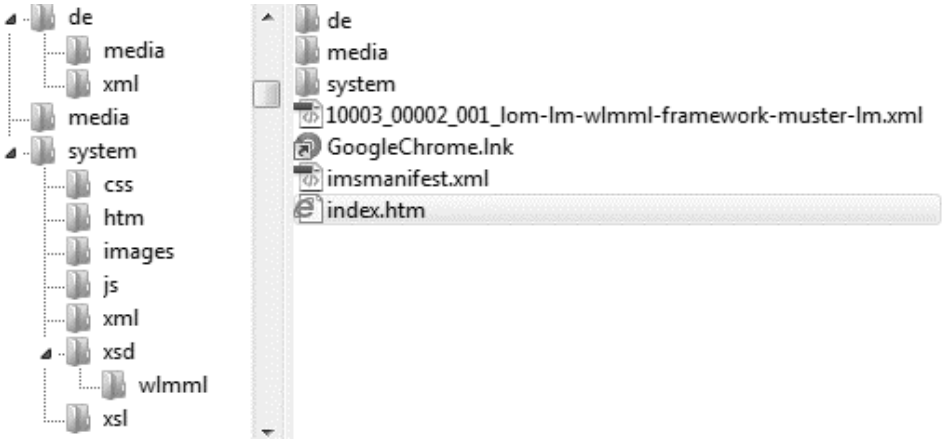


Abbildung 59: Verzeichnisstruktur (linke Seite) und Dateien im „Root-Verzeichnis“ (rechte Seite) des WLMML-Frameworks (am Beispiel des Muster-WLMML-Lernmoduls, siehe Anhang F)

In das Framework ist bereits ein Standard-Sprachordner „de“ (deutsch, Länderbezeichnungen nach ISO-639-1 2002) integriert. Dieser weist zwei Unterordner „media“ und „xml“ auf, in die sprachspezifische (in diesem Fall deutschsprachige) Inhalte abgelegt werden sollen. Der Unterordner „media“ soll dabei alle deutschsprachigen Medien-Dateien (z. B. Bild-, Video-, Ton-, PDF-

Dateien, Bild-Dateien z. B. mit deutschen Text), der Unterordner „xml“ alle deutschsprachigen WLMML-Dateien beinhalten. Zusätzlich befindet sich im Unterordner „xml“ die HTML-Datei „constant.htm“, die als aufrufende Datei für die sprachspezifische Anzeige der Verzeichnisse und der Volltextsuche benötigt wird.

Im nicht sprachspezifischen übergeordneten „media"-Ordner (in Abbildung 59 unter der Pfadangabe „media/“ zu finden) werden die Medien-Dateien abgelegt, die in allen Sprachen identisch sind. Auf diese Art und Weise sollen Redundanzen vermieden werden. Eine sprachunabhängige Mediendatei muss nicht in jedem sprachspezifischen sondern nur einmal im übergeordneten „media“-Ordner vorliegen. Alle WLMML-Dokumente, die in unterschiedlichen Sprachen gehalten sind, können diese Mediendateien einbinden.

Der Ordner „system“ beinhaltet alle Ordner und Dateien, die den Systemkern des WLMML-Frameworks ausmachen. Die folgende Liste führt sie auf und gibt einen kurzen Überblick über ihren Inhalt und dessen Funktionen:

- Ordner „css“
  (CSS-Dateien zur Formatierung der XSLT-Transformationsergebnisse und der Hilfsleiste am unteren Browserrand von WLMML-Lerninhaltsdokumenten, allgemeine zentrale Steuerung des Aussehens der Lerninhalte im Browserfenster)
- Ordner „htm“
  (HTML-Dateien, die von der Start-Datei „index.htm“ zum „Start-Frameset“ mit Sprachauswahl, Navigationsleiste und Inhaltsbereich zusammengesetzt werden)
- Ordner „images“
  (WLMML-Framework spezifische Bild-Dateien, z. B. Pfeilbilder der Hilfsleiste am unteren Browserrand von WLMML-Lerninhaltsdokumenten)
- Ordner „js“
  (JavaScript-Dateien für komplexere Funktionalitäten, z. B. Browsererkennung, Ermittlung/Übergabe/Anzeige der spezifischen Sprach-Kürzel (z. B. „de“), Volltextsuche, Steuerung der XSLT-Transformationen)
- Ordner „xml“
  (XML-Dateien, die an verschiedenen Stellen geladen werden und XSLT-Dateien zur weiteren Verarbeitung referenzieren (z. B. Anzeige der Navigationsleiste, Anzeige der Verzeichnisse))
- Ordner „xsd“ mit dem Unterordner „wlmml“

(WLMML-Schema-Datei (in Abbildung 59 unter der Pfadangabe „xsd/wlmml/“ zu finden) zur Validierung der WLMML-Dateien)

- Ordner „xsl“
  (XSLT-Dateien für die Transformation der WLMML-Dateien)

Die Dateien im „Root-Verzeichnis“ des WLMML-Frameworks (Abbildung 59 rechte Seite) übernehmen unterschiedliche Aufgaben. Die Datei „10003_00002_001_lom-lm-wlmml-framework-muster-lm.xml“ dient als Beispieldatei zur Aufnahme von Metadaten. Die Verknüpfung „Google Chrome.lnk“ soll als Muster für den erleichterten Aufruf von lokalen Lerninhalten in „Google Chrome“ dienen (Inhalt der Verknüpfung: „Pfadangabe zur ‚.exe‘-Datei von Google Chrome --allow-file-access-from-files“). Ansonsten verweigert dieser Browser den Zugriff auf lokale Dateien. Die SCORM-konforme XML-Datei „imsmanifest.xml“ stellt eine Art Inhaltsverzeichnis des Lerninhaltes dar (mit Pfadangabe zu den einzelnen Dateien = Ressourcen, z. B. WLMML-, Bild-, Video-, Ton-, PDF-Dateien). Der Inhalt dieser Datei spiegelt sich u. a. beim Aufruf der Startseite des Lerninhaltes als Navigationsstruktur im linken Teil des Browserfensters wieder. Die HTML-Datei „index.htm“ ist die Startseite des Lerninhaltes. Sie kann aufgerufen werden, wenn keine SCORM-fähige Umgebung zur Verfügung steht (z. B. lokal oder auf einem Server).

Für die Erstellung SCORM-kompatibler Lerninhalte („SCORM 2004“) wurde die Software RELOAD (RELOAD 2008) eingesetzt. Sie fügte als Ergebnis dem Framework automatisch einige weitere SCORM-spezifische Ordner und Dateien hinzu, die nicht integraler Bestandteil des WLMML-Frameworks sind. Dazu gehören die Ordner „common“, „extend“, „unique“ und „vocab“, die jeweils nur Schema-Dateien beinhalten und die Schema-Dateien im Root-Verzeichnis selbst. Alle diese Dateien sind von RELOAD zur Validierung SCORM-2004-kompatibler Inhalte vorgesehen. Zusätzlich kann die in SCORM eingesetzte XML-Datei „imsmanifest.xml“ (siehe Kapitel 3.3.7.1 IMS CP und 3.8 SCORM) erstellt oder, wenn sie bereits vorhanden ist, bearbeitet werden. Um „SCORM 2004“-Funktionalitäten sicher stellen zu können, wird die JavaScript-Datei „scorm.js“ im Systemordner „js“ eingesetzt (siehe Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei, Anmerkungen zur JavaScript-Datei „scorm.js“). Ihr Inhalt stammt (mit Anpassungen) von OSTYN (2007, S. 35 ff) und OSTYN (2006).

Abbildung 60 zeigt die Verzeichnisstruktur und die Dateien im „Root-Verzeichnis“ am Beispiel des Muster-Lernmoduls „10003_00003_001_lm-wlmml-framework-

einfaches-beispiel-lm“. Dieses Muster-Lernmodul ist als Grundlage für die Erstellung aller WLMML-Lerninhalte gedacht (inkl. „SCORM-2004“-kompatibler Lerninhalte).

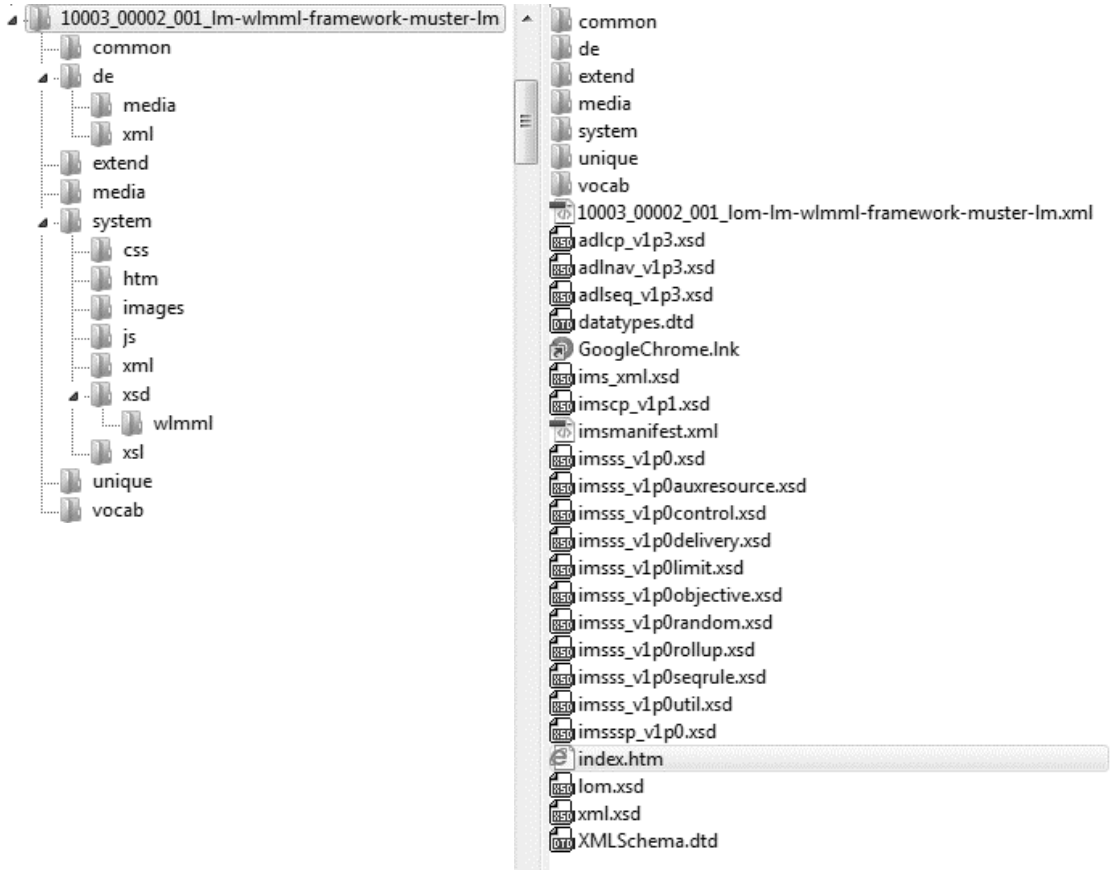


Abbildung 60: Verzeichnisstruktur (linke Seite) und Dateien im „Root-Verzeichnis“ (rechte Seite) des Muster-WLMML-Lernmoduls („SCORM 2004“-Lernpaket, siehe Anhang G)

Im Folgenden wird auf die besonderen Funktionalitäten des WLMML-Frameworks eingegangen.

4.3.2 Funktionalitäten

Die besonderen Funktionalitäten des WLMML-Frameworks spiegeln sich in folgenden Punkten wieder:

- Modularisierung und Wiederverwendung von E-Learning-Inhalten
- Metadatenkonzept
- konzeptionelle Integration von Mehrsprachigkeit
- clientseitige Verarbeitung

Auf diese Themen wird in den kommenden Kapiteln eingegangen.

4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM)

Insbesondere aufgrund des hohen Aufwandes bei der Erstellung von E-Learning-Inhalten sollten sie vielseitig einsetzbar und wiederverwendbar sein. KERRES (2012, S. 447) betont: „*Nur eine konsequente Mehrfachnutzung rechtfertigt es in vielen Fällen, hohe Mittel in die Entwicklung von E-Learning-Angeboten zu investieren.*“ Er weist weiter darauf hin: „*Die Mehrfachnutzung und Wiederverwertbarkeit erscheint damit als eine zentrale Hürde, aber auch die wesentliche Chance für die Zukunft von E-Learning. Kann man dieses Problem lösen, so sollte eine wesentlich breitere Verfügbarkeit von Contents möglich werden.*“ Damit erscheint auch der Austausch auf verschiedenen Granularitätsebenen und möglichst einfache Import in LMSs (z. B. über SCORM) sinnvoll. Das im Folgenden beschriebene technische Konzept setzt diese Vorstellungen für die im WLMML-Framework abgelegten E-Learning-Inhalte um (insbesondere WLMML-Dateien). STREHL et al. (2005, S. 1) halten bezüglich dieses technischen Konzeptes fest: „*Hierbei werden E-Learning-Veranstaltungen so konzipiert, dass sie sich aus kleinen, wiederverwertbaren E-Learning-Modulen zusammensetzen lassen.*“ Diese Lernmodule (siehe Kapitel 3.2.3 Modellierung von E-Learning-Inhalten) sollten thematisch abgeschlossen, austauschbar, technisch lauffähig, evtl. mehrsprachig, mit Metadaten versehen und mit Lernzielen versehen sein (siehe auch RADULESCU 2005, S. 36). Der Autor von E-Learning-Inhalten erstellt sie einzeln, technisch und inhaltlich unabhängig voneinander. Als Voraussetzung für diese Vorgehensweise sollte das zu vermittelnde Wissensgebiet in der Phase der Lehrveranstaltungsplanung in Lernziele zerlegt werden (STREHL et al. 2005, S. 4, siehe auch Kapitel 3.1 Lerntheoretische Grundlagen, 3.1.2 Didaktik). Die einem Lernziel zugeordneten E-Learning-Inhalte werden anschließend mit WLMML (siehe Kapitel 4.2 WLMML) beschrieben und in einer festgelegten, „SCORM2004“-kompatiblen Verzeichnisstruktur abgelegt (Abbildung 60, siehe Kapitel 4.3.1 Verzeichnisstruktur, zu SCORM siehe Kapitel 3.8 SCORM). Das Ergebnis ist ein „SCORM2004“-kompatibles WLMML-Lernmodul (im Folgenden Lernmodul genannt). Es entspricht einem „Sharable Content Object“ (SCO) von SCORM und ist in der Lage über eine SCORM-Schnittstelle („Application Programming Interface (API)“) mit einem LMS zu kommunizieren (siehe Kapitel 3.2.3 Modellierung von E-Learning-Inhalten und 3.8 SCORM). Das Lernziel kann z. B. über das WLMML-Element „learningObjective“ in einer WLMML-Inhaltsdatei oder über Metadaten (LOM, HODGINS et al. 2004, S. 22, 5.4.9.1 Purpose (educational objective)) in der externen Metadaten-Datei im „Root-Verzeichnis“ angegeben werden (Abbildung 60, Datei „10003_00002_001_lom-lm-wlmml-framework-muster-lm.xml“, siehe Kapitel 4.3.2.2 Metadaten). Die erste Variante ermöglicht es über Anweisungen im WLMML-Framework das Lernziel (auf Lernmodulebene) bzw. die Lernziele (auf Lerneinheit- oder Kurs-Ebene) in einem Verzeichnis anzeigen zu

lassen (siehe auch Kapitel 4.3.2.4.3 Automatische Erstellung von Verzeichnissen). Darüber hinaus lässt sich jedes WLMML-Lernobjekt benutzerfreundlich aufbereiten, indem es z. B. mit einem oder mehreren einführenden Lernmodulen versehen wird. Als Inhalte dieser Lernmodule bieten sich neben der Angabe der Lernziele z. B. folgende Punkte an (siehe auch Kapitel 3.1.2 Didaktik, als Beispiel siehe QUEDNAU 2010):

- Benutzungsanleitung
- Angabe der Lehrinhalte
- Angabe der empfohlenen Vorkenntnisse
- Angabe sonstiger Hinweise

Die zweite Variante erlaubt es z. B. einem „SCORM2004“-kompatiblen LMS, das LOM-Angaben in einer externen XML-Datei verarbeiten kann, auf diese Daten zuzugreifen. Allgemein können zusätzlich dem Lernmodul weitere Metadaten hinzugefügt werden (siehe Kapitel 3.7 IEEE LOM, 4.3.2.2 Metadaten).

Aus den Lernmodulen lassen sich abschließend die anderen beiden WLMML-Lernobjekt-Typen Lerneinheiten und Kurse (Vorlesungen) in „SCORM2004“-kompatiblen Paketen zusammensetzen (siehe auch Kapitel 3.2.3 Modellierung von E-Learning-Inhalten, siehe auch RADULESCU 2005, S. 36 ff, GEIGER 2005, S. 18 ff). STREHL et al. (2005, S. 4, Wiederverwertbarkeit auf der Ebene von Lernmodulen) erklären in diesem Zusammenhang: „*Sollen mehrere Lernmodule zu einem Kurs bzw. einer Lerneinheit zusammengesetzt werden, muss eine Gliederung, ein Ablaufplan definiert werden.*“ Sie führen weiter aus, dass diese Gliederung über die „SCORM2004“-Manifestdatei realisiert wird. Jede WLMML-Inhaltsdatei eines WLMML-Lernobjektes (Lernmodul/Lerneinheit/Kurs) kann in dieser Datei als Gliederungspunkt auftreten. Für das Erstellen einer Lerneinheit oder eines Kurses müssen abschließend die Dateien der einzelnen Lernmodule noch übereinander kopiert werden (Abbildung 61). Ist das geschehen, gilt es eine externe Metadaten-Datei im „Root-Verzeichnis“ für die neue Lerneinheit (bzw. Kurs) anzulegen und die „SCORM2004“-Manifestdatei „imsmanifest.xml“ anzupassen (siehe auch Kapitel 3.3.7.1 IMS CP).

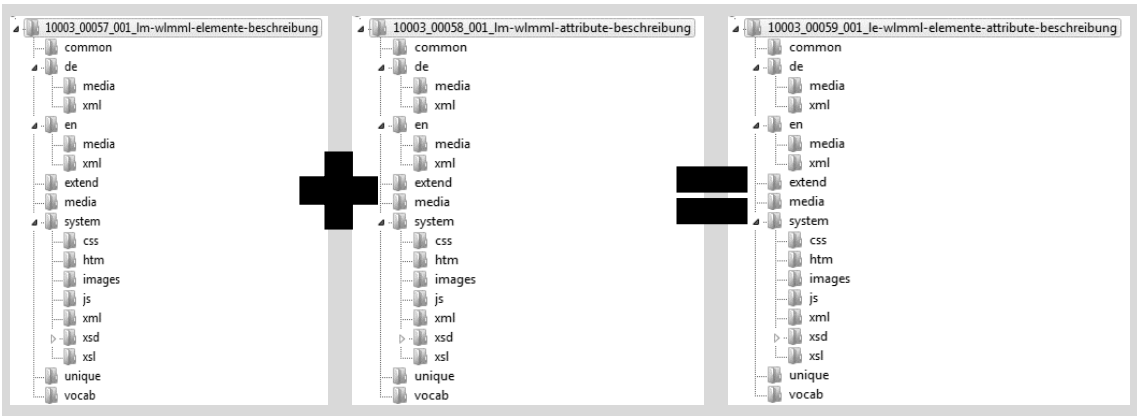


Abbildung 61: Übereinander Kopieren von zwei Lernmodulen, Ergebnis eine Lerneinheit (rechts), Verzeichnisstruktur nach STREHL et al. (2005, S. 5)

Die E-Learning-Inhalte in einem Lernmodul/Lerneinheit/Kurs lassen sich sofort über einen kompatiblen Browser (siehe Kapitel 4.3.3 Voraussetzungen der Browser-Software) nach dem Aufruf der Start-HTML-Datei „index.htm“ (Abbildung 60, rechte Seite) betrachten und nutzen (siehe Kapitel 4.3.2.4 Clientseitige Verarbeitung). Sie sind damit unabhängig von einem LMS und können über die Start-HTML-Seite „index.htm“ („Root-Verzeichnis“) „*eigenständig wie Web-Seiten*“ offline wie online verwendet werden (siehe auch STREHL et al. 2005, S. 6). „SCORM2004“-Funktionalitäten gehen allerdings bei der Nutzung ohne ein unterstützendes System verloren. WLMML-Inhalte sollten auch in IMS LD über dessen Element „learning-object“ integriert werden können (siehe Kapitel 3.3.7.3.6 IMS CP und IMS LD in einem XML-Beispiel). Allerdings müsste der gesamte Ordner des Lernmoduls (bzw. der Lerneinheit oder des Kurses) zur Verfügung gestellt werden.

Da WLMML-Lernobjekte (Lernmodul/Lerneinheit/Kurs) von sich aus „SCORM2004“-kompatibel sind, lassen sie sich auch in „SCORM2004“-fähige LMS importieren. Dazu muss die gesamte Verzeichnisstruktur für den Import in das LMS gepackt werden („.zip“-Datei, „Package Interchange File“, siehe Kapitel 3.3.7.1 IMS CP, 3.8 SCORM). Um „SCORM2004“-fähige WLMML-Lernobjekte zu erstellen, anzupassen und zu packen (inkl. der Kennzeichnung einer WLMML-Datei in der XML-Datei „imsmanifest.xml“ als SCO), bietet sich Software wie z. B. der „RELOAD“-Editor (RELOAD 2008) an (siehe Kapitel 3.8 SCORM). Eine ausführliche Anleitung dafür findet sich in Anhang I.

Alle Dateien und Verzeichnisse eines WLMML-Lernobjektes werden in einem Ordner abgelegt, der einen spezifischen Namen wie z. B. „10003_00002_001_lm-wlmml-framework-muster-lm“ trägt (Abbildung 60 links oben). Die ersten fünf Ziffern geben

den Autor wieder, die folgenden fünf Ziffern die eindeutige Nummer des Lernmoduls (bzw. Lerneinheit oder Kurs) des jeweiligen Autors, die folgenden drei Ziffern die Version und die letzten Zeichen eine individuelle Benennung. Innerhalb der Benennung lassen sich die ersten zwei Buchstaben für den Typ des WLMML-Lernobjektes („ku“ = Kurs, „le“ = Lerneinheit, „lm“ = Lernmodul) verwenden. Zwischen den vier Bezeichnungsblöcken kann ein Unterstrich als Trennzeichen, innerhalb der individuellen Bezeichnung ein Bindestrich gesetzt werden (z. B. Ordnername: „10003_00002_001_lm-wlmml-framework-lm-muster“). Umlaute, Leerzeichen und Sonderzeichen müssen wie auch bei Dateinamen vermieden werden. Abbildung 62 zeigt Beispiele der Ordnernamensvergabe.

Neben den Ordnernamen werden alle Dateinamen mit der Autor- und Lernmodul-Nummer des Lernmoduls versehen (siehe auch QUEDNAU 2007, S. 7 f), in welchem sie zuerst auftauchen (z. B: „10003_00003_himalaya.jpg“). Einleitungsdateien oder zum WLMML-Lernobjekt gehörende Ergänzungsdateien werden ebenfalls mit der Lernmodul/Lerneinheit/Kurs-Nummer versehen, in der sie auftauchen. Der Typ und die Versionsnummer wird außer in der LOM-Datei des WLMML-Lernobjektes in Dateinamen nicht mit aufgenommen. Eine Literaturverzeichnis-Datei erhält im Namen zusätzlich den Text „literatur“ (z. B. „10003_00007_literatur-editor.xml“). Es können mehrere Literaturverzeichnis-Dateien in einer Lerneinheit vorliegen. Identische Literaturangaben werden vom System ausgefiltert. Ein Literaturverzeichnis wird als einem Lernmodul zugehörig betrachtet und trägt im Dateinamen die Nummer des Lernmoduls. Die LOM-Datei des Lernmoduls erhält im Namen den Text „lom“ und die Versionsnummer des Lernmoduls (z. B. „10003_00002_001_lom-lm-wlmml-framework-muster-lm.xml“). Zusätzlich lässt sich in der Benennung der Typ des Lernobjektes mit aufnehmen. Neben dem Dateinamen sollte ihr Inhalt bei der Erstellung eines neuen WLMML-Lernobjektes angepasst werden.

Es bietet sich an die Namen der Ordner der Lernmodule (bzw. der Lerneinheiten oder der Kurse) mitzuprotokollieren (Abbildung 62).

Übersicht Lernmodule/Lerneinheiten/Kurse
Autor 10003 (Sebastian Paar)

Typ: lm = Lernmodul, le = Lerneinheit, ku = Kurs

Lernmodule/Lerneinheiten/Kurse

Autor	Nummer	Version	Typ	Name	Ordnername
10003	00001	001	lm	wlmml-framework-lerninhalte-benutzung-hilfe	10003_00001_001_lm-wlmml-framework-lerninhalte-benutzung-hilfe
10003	00002	001	lm	wlmml-framework-muster-lm	10003_00002_001_lm-wlmml-framework-muster-lm
10003	00003	001	lm	wlmml-framework-einfaches-beispiel-lm	10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm
10003	00004	001	lm	wlmml-elemente	10003_00004_001_lm-wlmml-elemente
10003	00005	001	lm	wlmml-framework-einleitung	10003_00005_001_lm-wlmml-framework-einleitung
10003	00006	001	lm	wlmml-framework-download-inhalt	10003_00006_001_lm-wlmml-framework-download-inhalt
10003	00007	001	lm	wlmml-namenskonventionen	10003_00007_001_lm-wlmml-namenskonventionen
10003	00008	001	lm	wlmml-dateien-allgemein	10003_00008_001_lm-wlmml-dateien-allgemein
10003	00009	001	lm	wlmm-framework-sprachordner	10003_00009_001_lm-wlmm-framework-sprachordner
10003	00010	001	lm	wlmml-sprachangaben	10003_00010_001_lm-wlmml-sprachangaben
10003	00011	001	lm	wlmml-framework-metadaten	10003_00011_001_lm-wlmml-framework-metadaten
...					
10003	00047	001	le	wlmml-framework-muster-lm	10003_00047_001_le-wlmml-framework-muster-lm
10003	00048	001	le	wlmml-elemente	10003_00048_001_le-wlmml-elemente
10003	00049	001	le	wlmml-framework-ueberblick	10003_00049_001_le-wlmml-framework-ueberblick
10003	00050	001	le	wlmml-framework-mehrsprachigkeit	10003_00050_001_le-wlmml-framework-mehrsprachigkeit
10003	00051	001	le	wlmml-altova	10003_00051_001_le-wlmml-altova
10003	00052	001	le	wlmml-word	10003_00052_001_le-wlmml-word
10003	00053	001	le	wlmml-eclipse	10003_00053_001_le-wlmml-eclipse
10003	00054	001	le	scorm-wlmml-framework-reload	10003_00054_001_le-scorm-wlmml-framework-reload
10003	00055	001	ku	lernmodule-erstellen	10003_00055_001_ku-lernmodule-erstellen

Abbildung 62: Übersicht Lernmodule/Lerneinheiten/Kurse

Grundsätzlich lassen sich alle Inhalte in einer einzigen Datei abgelegen und sukzessiv WLMML-Elemente zuweisen. Es sollte allerdings bei umfangreicheren Inhalten geprüft werden, ob die Inhalte nicht besser in mehrere separate WLMML-Dateien abgelegt werden können. Als Vorgabe, welche Dateien mit welchem Inhalt erstellt werden, könnte bei bereits vorhandenem Lernmaterial das Inhaltsverzeichnis der Lernmaterialvorlage dienen. Jeder Inhaltsverzeichniseintrag würde dann in etwa eine WLMML-Datei und damit ein selbständiges Lernmodul mit einem Lernziel widerspiegeln. Ergeben mehrere Inhaltsverzeichniseinträge zusammen eine in sich abgeschlossene sinnvolle inhaltliche Einheit (mit einem Lernziel), sollten sie zu einer WLMML-Datei zusammengefasst werden (Lernmodul). Wenn noch keine Inhalte vorhanden sind, erscheint es sinnvoll zuerst das Wissensgebiet über eine Sachanalyse sinnvoll zu strukturieren (Modularisierung der E-Learning-Inhalte), den Inhalten Lernziele zuzuordnen und anschließend die Lerninhalte zu erstellen (siehe auch GEIGER 2005, S. 54 ff, siehe auch Kapitel 3.2.1.2 Standardisierung de jure <-> de facto, ISO-Norm zur Entwicklung und Qualitätssicherung von E-Learning). KORNELSEN et al. (2004, S. 32, siehe auch JUNGMANN 2012, S. 109 ff) gehen darüber hinaus und beschreiben in ihren „Grundlagen des Authoring-Prozesses“ in 8 Phasen die Erstellung von XML-basierten E-Learning-Inhalten:

1. Informationssammlung und Strukturierung in einem Drehbuch
2. Formalisierung der Drehbuch-Inhalte in die ausgewählte XML-Notation
3. Einfügen der Medienobjekte
4. Hinzufügen didaktischer Aspekte
5. Festlegen des Layouts

6. Transformation des Dokumentes in ein für den Endanwender geeignetes Format
7. Evaluation zur Qualitätssicherung
8. Publikation der Inhalte für den Endanwender

Eine ausführliche technische Anleitung für die Erstellung eines WLMML-„SCORM2004“-Lernmoduls findet sich im Anhang I (siehe auch QUEDNAU et al. 2007, QUEDNAU & PAAR 2007). Ein Beispiel eines einfachen Lernmoduls bietet Anlage Anhang H.

Weitere Beispiele für Lernobjekte wurden begleitend zu zwei Diplomarbeiten entwickelt (GEIGER 2005, RADULESCU 2005). Diese Lernobjekte entstanden auf der Grundlage einer Vorgängerversion von WLMML und des WLMML-Frameworks. Zusätzlich wurde von QUEDNAU (2010) ein kompletter WLMML-Kurs in drei Sprachen erstellt und in einer Vorlesung im Themenbereich der beschreibenden und schließenden Statistik verwendet (siehe auch QUEDNAU et al. 2007, S. 8). Darüber hinaus entstanden im Rahmen dieser Arbeit unterschiedliche Kurse, Lerneinheiten und Lernmodule.

4.3.2.1.1 Beispiel-LMS von ADL

Die ADL-Initiative bietet ein „SCORM2004“-konformes Beispiel-LMS für Testzwecke an („*ADL Sample Run-Time Environment*“ (ADL-SRTE) Version 1.1.1, ADLNET-SCORM-SRTE 2009). Es fungiert als einfaches LMS und gibt Meldungen des LMS sowie des „SCORM2004“-kompatiblen E-Learning-Inhaltes während ihrer Kommunikations-Session in Echtzeit an. Abbildung 63 zeigt das importierte und registrierte WLMML-Lernmodul „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ in der „*ADL Sample Run-Time Environment*“ (ADLNET-SCORM-SRTE 2009) im Browser „Microsoft Internet Explorer 6“ (Betriebssystem „Microsoft Windows XP Service Pack 2“). Das Lernmodul wurde als gepackte Datei („10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm.zip“) importiert. Die beiden „organization“-Elemente des WLMML-Lernmoduls erscheinen als zwei Kurse (Abbildung 63 unter „Registered Courses“).

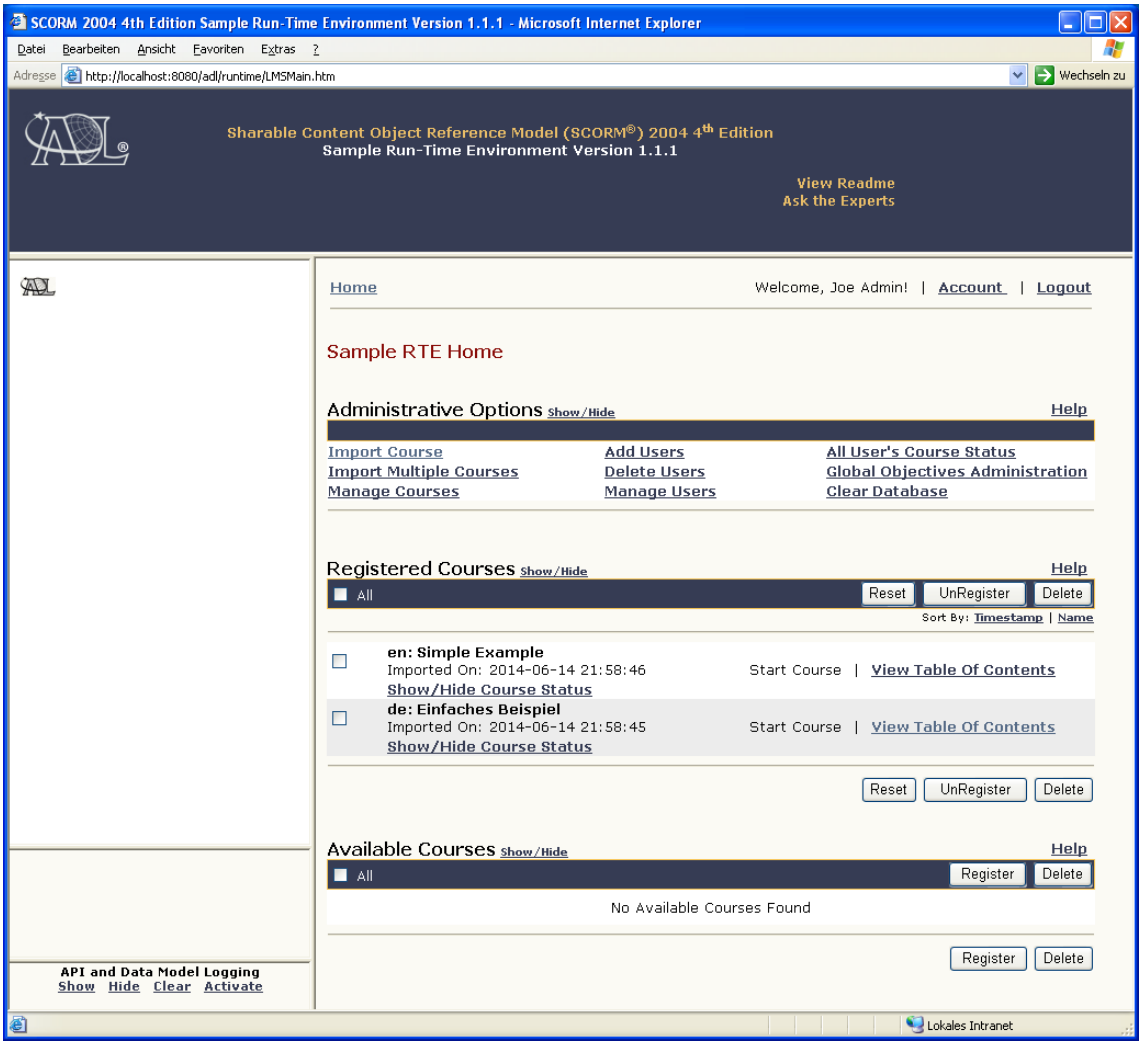


Abbildung 63: Importiertes WLMML-Lernmodul, die beiden „organization“-Elemente erscheinen als zwei Kurse, „ADL Sample Run-Time Environment“

Nach der Auswahl des deutschen Kurses (Abbildung 63, Hyperlink „View Table Of Contents“ im Kurs „de: Einfaches Beispiel“) zeigt Abbildung 64 das Ergebnis. Im linken unteren Teil dieser Abbildung („API and Datamodel Calls“-Bereich) sind die Kommunikationsmeldungen zu sehen, die nach dem Aufruf der WLMML-Inhaltsdatei (SCO) über den Hyperlink „Einfaches Beispiel“ erscheinen. Zuerst kommt die Meldung „Called Initialize“. Die positive Rückmeldung „Initialize Returned Error Code 0“ (kein Fehler) bestätigt, dass ein SCORM-kompatibler Lerninhalt vorhanden ist und die Kommunikations-Session gestartet werden kann. Die Kommunikation baut sich deshalb mit Erfolg auf, weil Anweisungen in der Funktion „ScormInitialize()“ aus der JavaScript-Datei „scorm.js“ des WLMML-Frameworks erfolgreich über die vorhandene „SCORM2004-API“-Instanz mit dem LMS in Verbindung treten (siehe auch Kapitel 3.8 SCORM, Initialize(“”)). Diese Funktion wird bei jedem Aufruf einer WLMML-Inhaltsdatei gestartet.

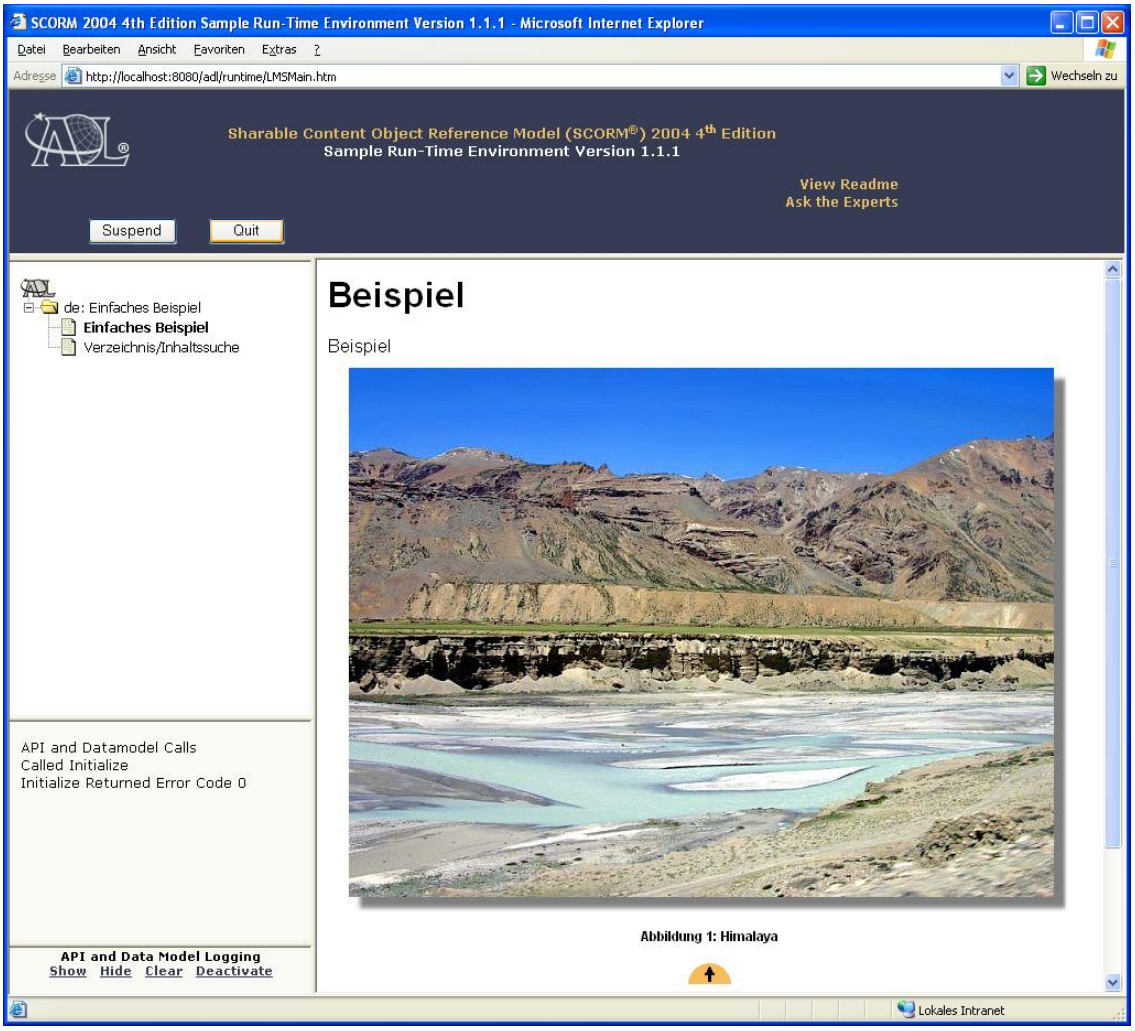


Abbildung 64: „ADL Sample Run-Time Environment“ mit einem WLMML-Lernmodul, Kommunikationsmeldungen (linker unterer Bereich), Browser „Microsoft Internet Explorer 6“

Wenn die aufgerufene Datei verlassen wird, stellen Anweisungen in der Funktion „ScormTerminate()“ (ebenfalls aus der JavaScript-Datei „scorm.js“ des WLMML-Frameworks, siehe auch Kapitel 3.8 SCORM, Terminate(“”)) weitere Rückmeldungen an das LMS sicher:

- Setzen des Status ("cmi.completion_status") des SCO auf „completed“ (Lerninhalt wurde „fertig“ betrachtet)
- Festhalten der Zeitdauer, wie lange das SCO geöffnet war („cmi.session_time“)
- Beenden der Kommunikations-Session zwischen LMS und SCO („Terminate“)

Beide JavaScript-Funktionen „ScormInitialize()“ und „ScormTerminate()“ sind genauer im Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei (Anmerkungen zur JavaScript-Datei „scorm.js“) beschrieben.

Abbildung 65 (links unten) zeigt die angesprochenen Kommunikationsmeldungen (Rückmeldungen) nach dem Verlassen der WLMML-Inhaltsdatei. Gleichzeitig macht Abbildung 65 deutlich, wie das Ergebnis einer Volltextsuche mit dem Suchbegriff „Beispiel“ im Browser („Microsoft Internet Explorer 6“, Betriebssystem „Microsoft Windows XP Service Pack 2“) aussieht und der markierte Suchbegriff in einer Ergebnisdatei dargestellt wird (siehe auch Kapitel 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten).

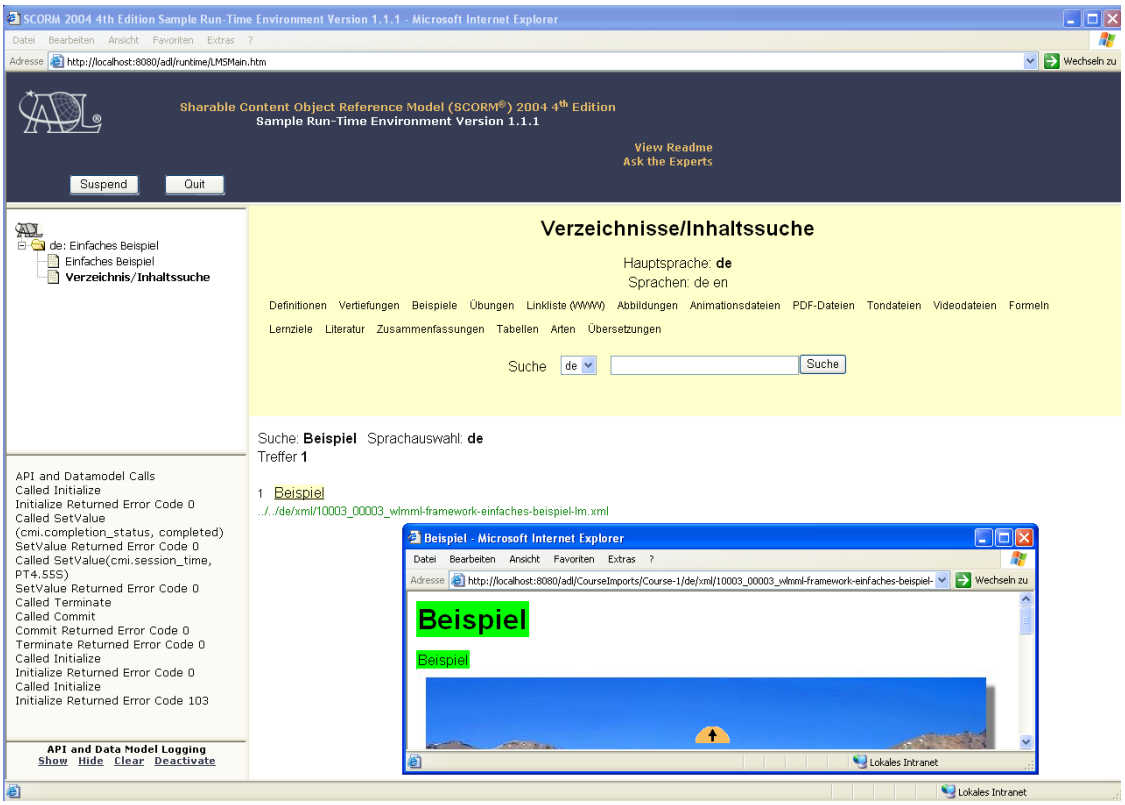


Abbildung 65: „ADL Sample Run-Time Environment“, Kommunikationsmeldungen (linker unterer Bereich), Aufruf eines neuen WLMML-Inhaltes („Verzeichnisse/Inhaltssuche“) mit anschließender Volltextsuche, Browser „Microsoft Internet Explorer 6“

Alle SCORM2004-Funktionalitäten (außer die Aufrufe der JavaScript-Funktionen „ScormInitialize()“ und „ScormTerminate()“) werden über JavaScript-Anweisungen in der JavaScript-Datei „scorm.js“ umgesetzt. Die Datei ist in jedem „system“-Ordner des WLMML-Frameworks zu finden (siehe z. B. Anhang H).

4.3.2.1.2 LMS „Moodle“

Das LMS „Moodle“ (MOODLE 2014, ETEACHING-MOODLE 2013, siehe auch Kapitel 3.9 LMS) ist ein „open source“-LMS und wird derzeit z. B. an der „Technischen Universität München“ eingesetzt (TUM-MOODLE 2014).

Das Lernmodul mit dem Namen „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ wurde wieder als gepackte Datei („10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm.zip“) importiert. Abbildung 66 zeigt nach dem Import zwei „Organisations“ an, die die beiden „organization“-Elemente in der Manifestdatei „imsmanifest.xml“ des WLMML-Lernobjektes wiedergeben. Nach der Auswahl der ersten Moodle-„Organisation“ im Aufklappmenü mit dem Namen „de: Einfaches Beispiel“ wird der Lerninhalt des WLMML-Lernobjektes angezeigt (Abbildung 67).

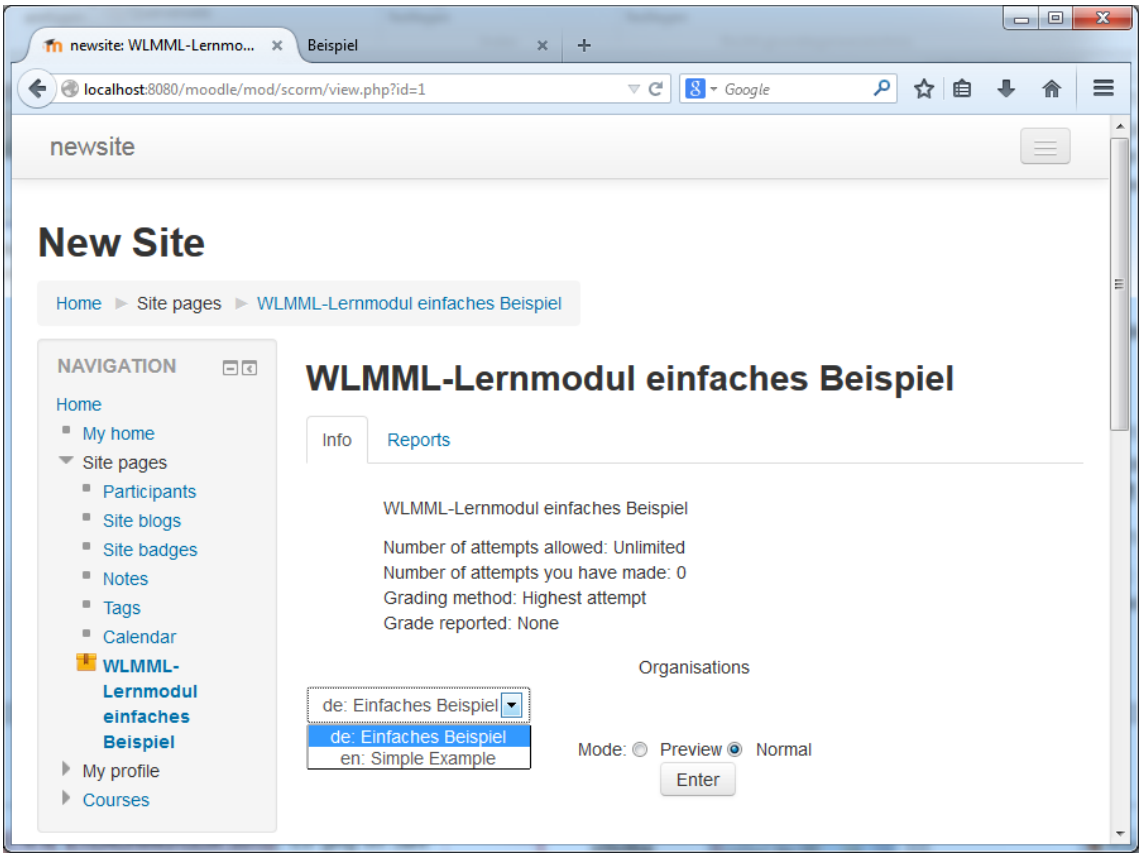


Abbildung 66: Anzeige zweier „Organisations“ nach Aufruf des importierten WLMML-Lernmoduls im LMS „Moodle“ („Moodle 2.7“, Browser „Mozilla Firefox 29.0.1“)

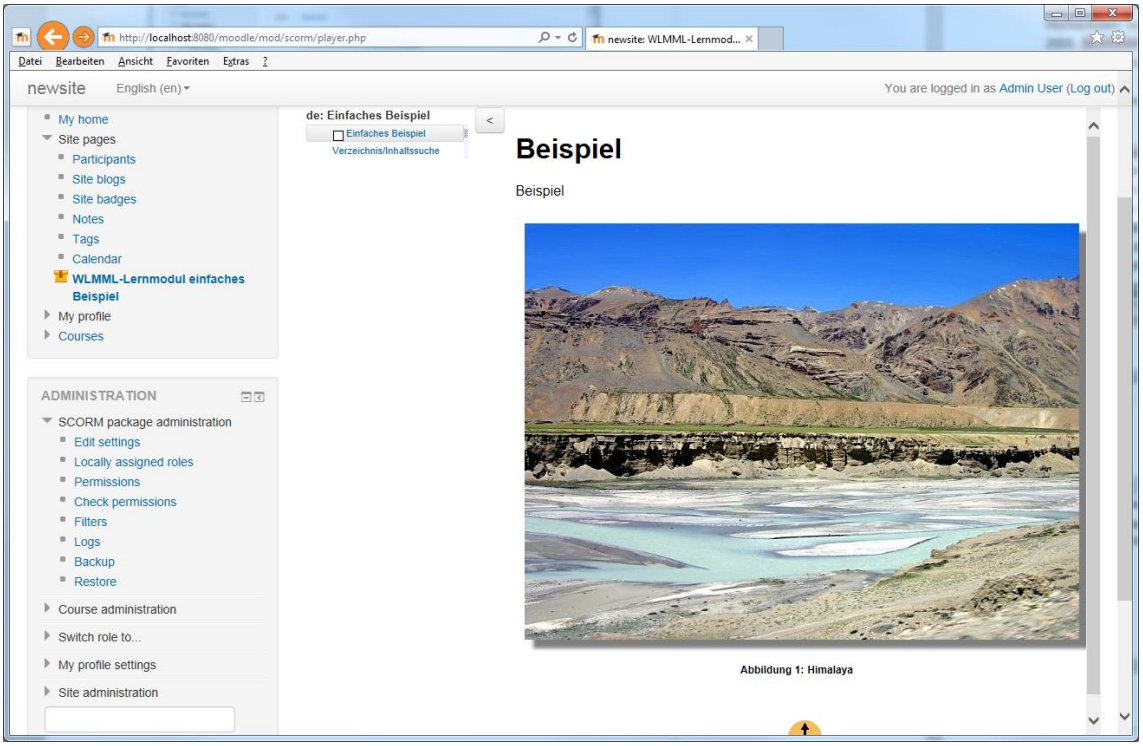


Abbildung 67: Anzeige des Lerninhaltes nach Auswahl der Moodle-„Organisation“ mit dem Namen „de: Einfaches Beispiel“ (siehe Abbildung 66) („Moodle 2.7“, Browser „Mozilla Firefox 29.0.1“)

Wählt der Anwender anschließend den Hyperlink „Verzeichnis/Inhaltssuche“ aus, wird die vorherige WLMML-Datei (SCO) verlassen und als „fertig“ („completed“) gekennzeichnet (siehe auch Kapitel 4.3.2.1.1 Beispiel-LMS von ADL).

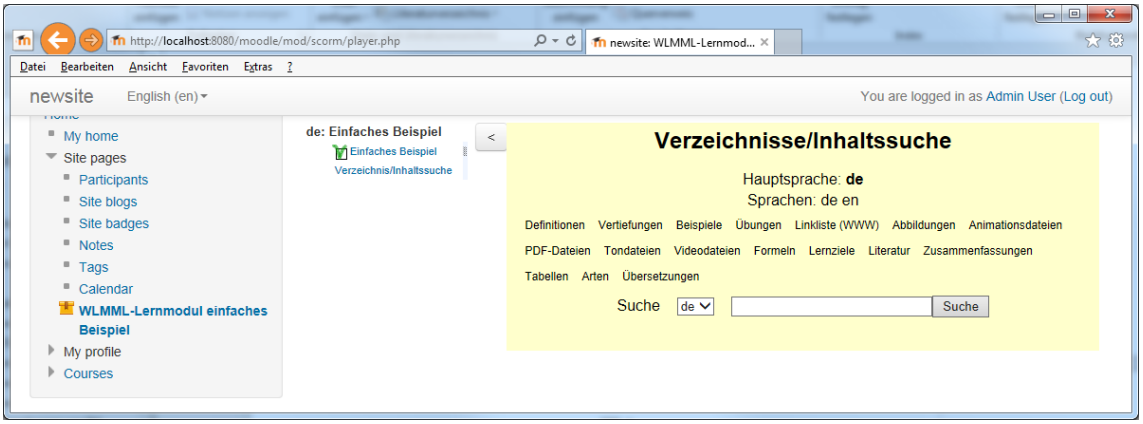


Abbildung 68: Anzeige des Lerninhaltes nach Auswahl des Hyperlinks „Verzeichnis/Inhaltssuche“ im LMS „Moodle“, erstes SCO als „completed“ gekennzeichnet (grüner Haken bei „Einfaches Beispiel“) („Moodle 2.7“, Browser „Mozilla Firefox 29.0.1“)

4.3.2.1.3 LMS „ILIAS“

„ILIAS“ (ILIAS 2014, ETEACHING-ILIAS 2012, siehe auch Kapitel 3.9 LMS) ist neben „Moodle“ ein weiteres „open source“-LMS. Es erlaubt Lerninhalte als „Learning Module HTML“ in gepackter Form zu importieren (z. B. „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm.zip“). Abbildung 69 zeigt das auf diese Weise importierte WLMML-Lernmodul mit dem Namen „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“.

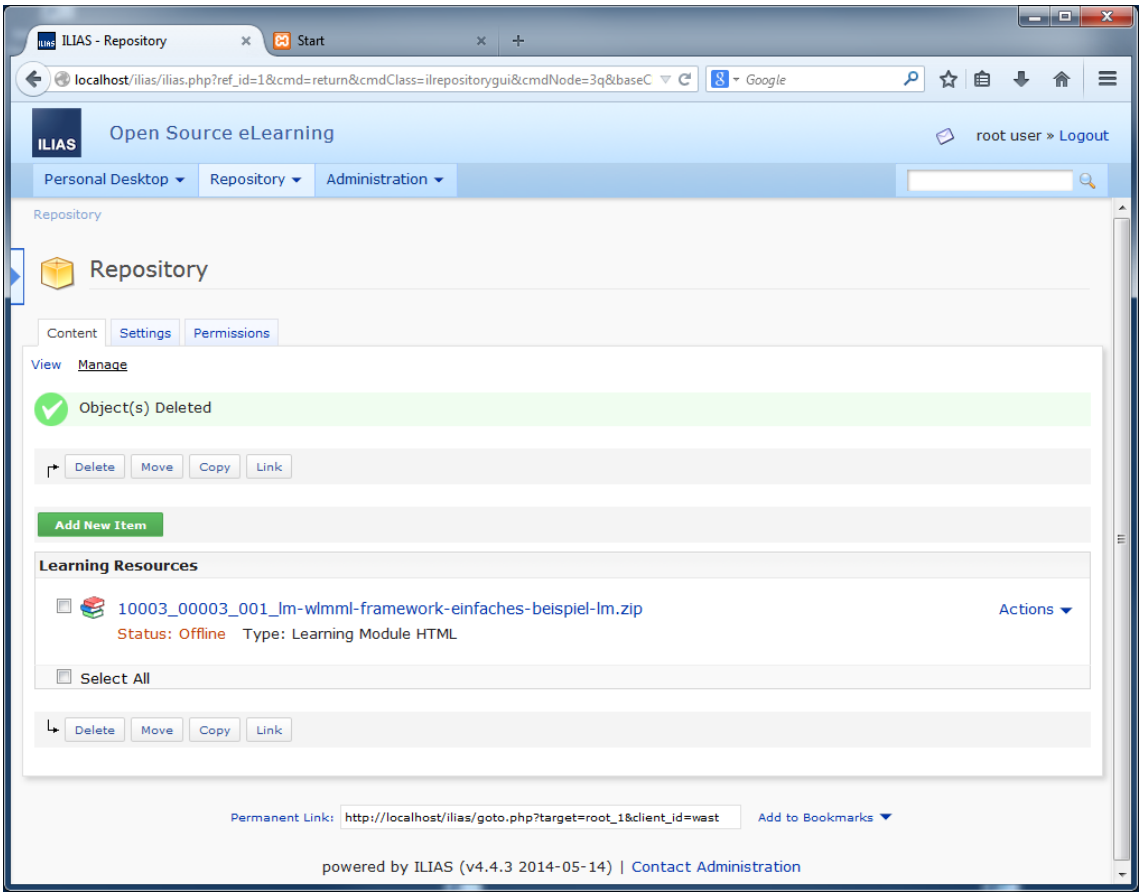


Abbildung 69: LMS „ILIAS“, Import des WLMML-Lernmoduls als „Learning Module HTML“ („ILIAS 4.4.3“, Browser „Mozilla Firefox 29.0.1“)

Nach der Auswahl des Hyperlinks „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm.zip“ erscheint der Lerninhalt in einem eigenen Browser-Fenster (Abbildung 70).

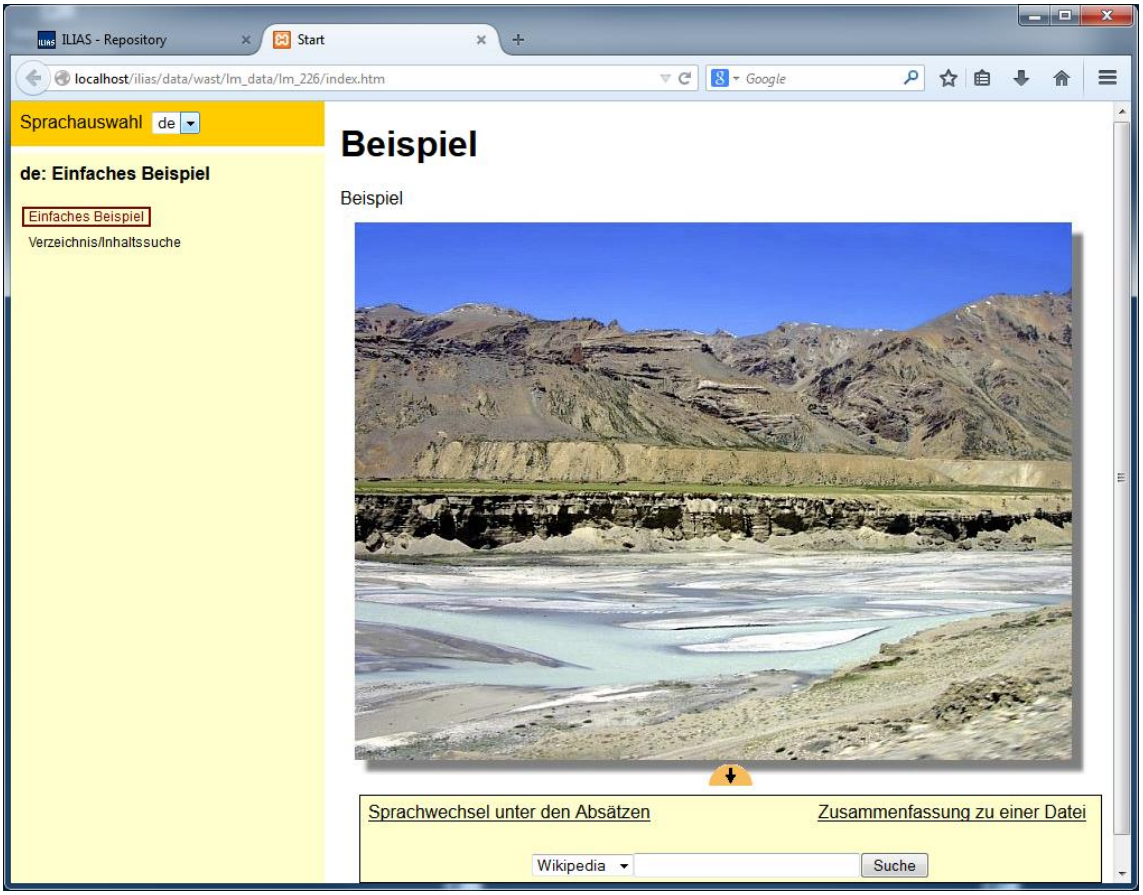


Abbildung 70: Anzeige des WLMML-Lernmoduls mit sichtbar geschaltener Hilfsleiste im LMS „ILIAS“ (siehe Adressleiste) („ILIAS 4.4.3“, Browser „Mozilla Firefox 29.0.1“)

4.3.2.1.4 LMS „CLIX Campus“

„CLIX Campus“ (CLIX) ist ein kommerzielles LMS (siehe auch Kapitel 3.9 LMS) für Hochschulen von dem Unternehmen „IMC AG“ (CLIX 2005, RATHMAYER 2005, SCHUMANN 2008, S. 4 ff, ETEACHING-CLIX 2012). Es ging z. B. im Jahr 2005 in der Version 4.5 an der Technischen Universität München in Betrieb (RADULESCU 2005, S. 27, SCHARNBECK 2005, S. 19 ff) und lief bis ca. Ende des Jahres 2010. Anschließend wurde es dort durch das LMS „Moodle“ ersetzt (TUM-MOODLE 2014).

Nach dem Import des gepackten WLMML-Lernmoduls „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ als SCORM-Paket und dem anschließenden Aufruf des importierten Lerninhaltes erscheinen zwei CLIX-Haupttypen („SCORM Organisation“) (Abbildung 71). Sie spiegeln die beiden „organization“-Elemente in der Manifestdatei „imsmanifest.xml“ des WLMML-Lernobjektes wieder. Nach der Auswahl der ersten „SCORM Organisation“ (Name „de: Einfaches Beispiel“) wird deren Inhalt angezeigt (Abbildung 72).

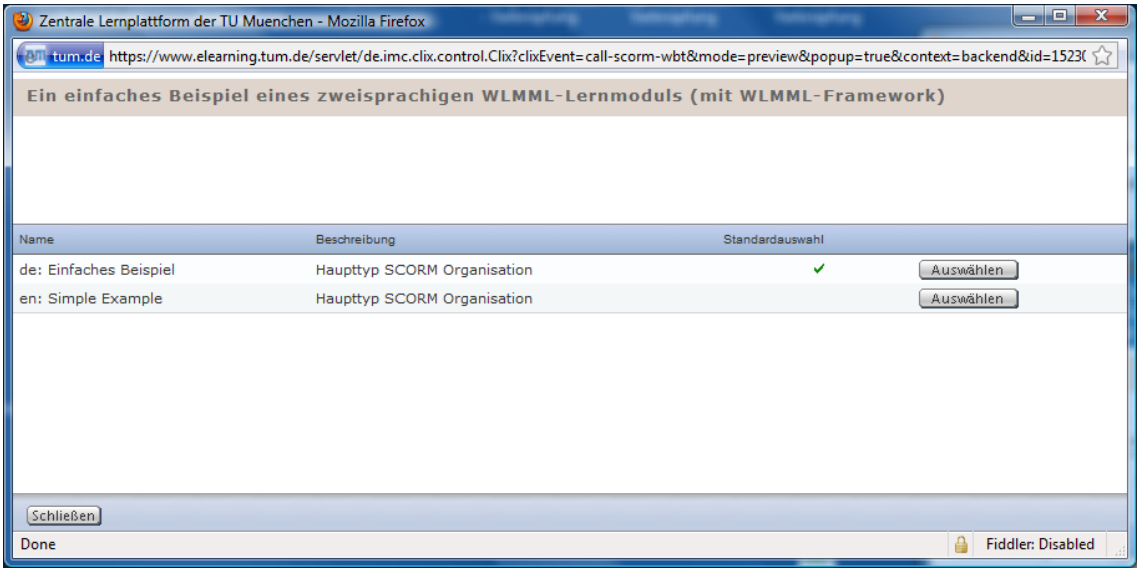


Abbildung 71: Anzeige zweier „SCORM Organisation“ nach Aufruf des importierten WLMML-Lernmoduls (CLIX Version 5, Browser „Mozilla Firefox 3.6“)

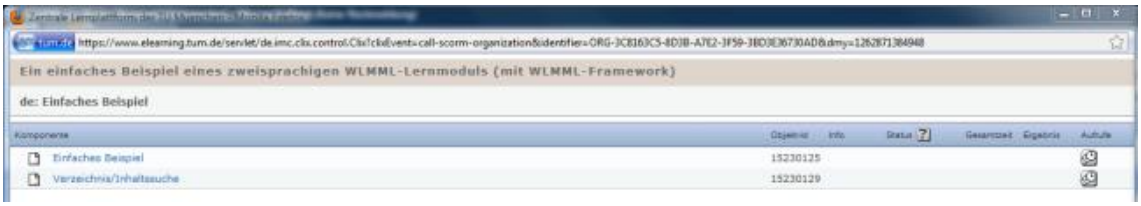


Abbildung 72: Anzeige nach Auswahl der ersten „SCORM Organisation“ (Name „de: Einfaches Beispiel“) (CLIX Version 5, Browser „Mozilla Firefox 3.6“)

Nutzt man den Hyperlink, der auf dem Text „Einfaches Beispiel“ liegt und ruft damit die WLMML-Datei mit dem Namen „10003_00003_wlmml-framework-einfaches-beispiel-lm.xml“ auf, zeigt sich das Ergebnis in Abbildung 73.

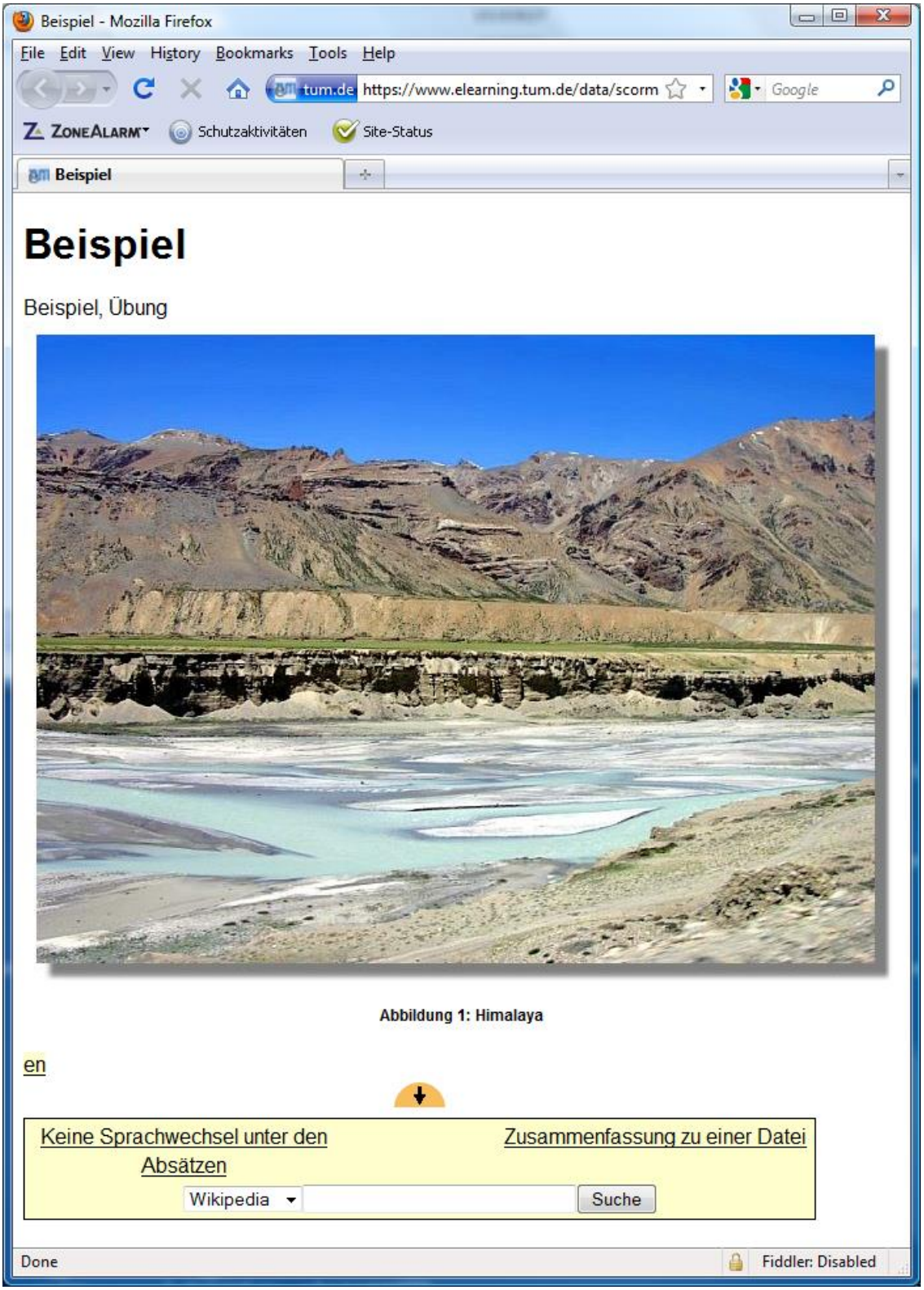


Abbildung 73: Anzeige des Lerninhaltes (WLMML-Datei) nach Auswahl des Hyperlinks mit dem Text „Einfaches Beispiel“ (siehe Abbildung 72), aktivierter Sprachwechsel mit aufgeklappter Hilfsleiste (CLIX Version 5, Browser „Mozilla Firefox 3.6“)

Auch die Metadaten-Angaben in der externen LOM-Datei eines WLMML-Lernobjektes werden in CLIX eingelesen (Abbildung 74).

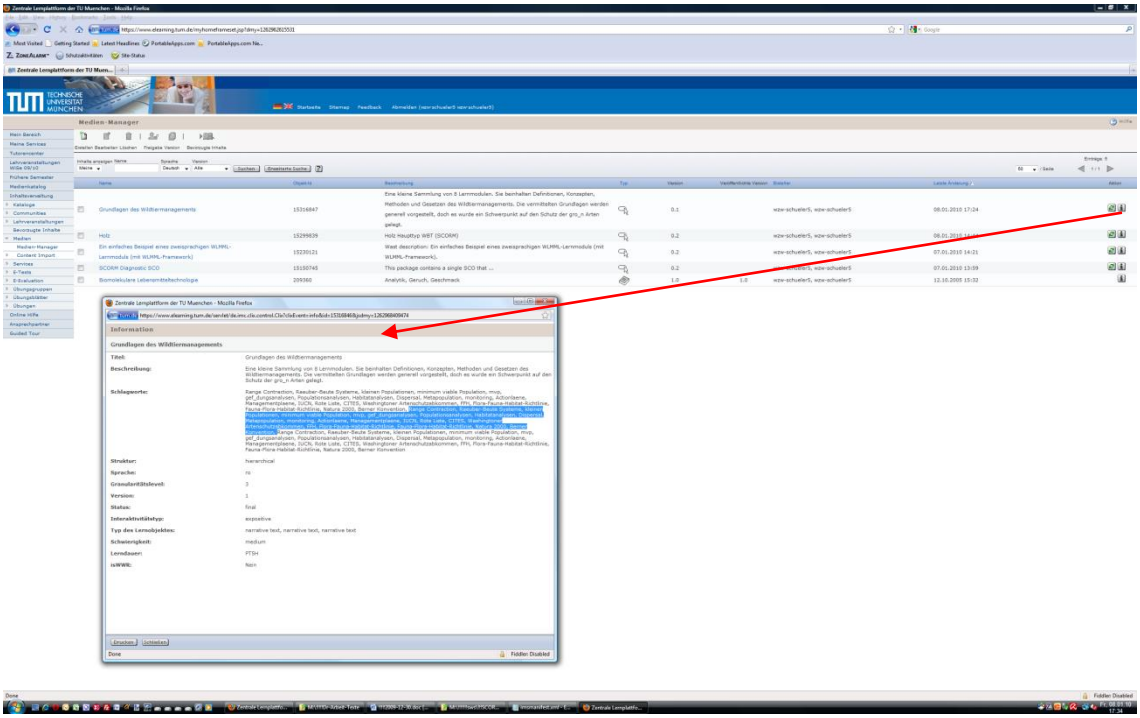


Abbildung 74: Metadaten-Angabe in CLIX aus der externen LOM-Datei eines WLMML-Lernobjektes (CLIX Version 5, Browser „Mozilla Firefox 3.6“)

Weitere Informationen zu Metadaten-Angaben bietet das nächste Kapitel.

4.3.2.2 Metadaten

Für die Wiederverwertbarkeit von Lerninhalten ist ihre Ausstattung mit beschreibenden Informationen (Metadaten) von zentraler Bedeutung (siehe Kapitel 3.7 IEEE LOM, siehe auch QUEDNAU & STREHL 2007, STREHL et al. 2005, S. 3). WLMML-Lernobjekte unterliegen einem durchgängigen Metadatenansatz, in dem auf allen Ebenen (Lernmodul, Lerneinheit, Kurs) Metadaten in Form von externen LOM-Dateien erstellt werden (STREHL et al. 2005, S. 6, STREHL et al. 2007). Besonders WLMML-Lernmodule - als grundlegende Bausteine für die Zusammensetzung von Lerneinheiten und Kursen - sollten damit ausgestattet sein. Jedes Lernmodul (wie auch alle anderen WLMML-Lernobjekte) wird mit einer externen LOM-Datei in XML beschrieben (siehe Kapitel 3.7 IEEE LOM). Die Metadaten in dieser Datei können z. B. von einem LOM-fähigen LMS ausgelesen werden (siehe Kapitel 4.3.2.1.4 LMS „CLIX Campus“). Die externe eigenständige LOM-Datei eines jeden Lernobjektes (siehe Kapitel 4.3.1 Verzeichnisstruktur, 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM)) ist in der Manifest-Datei „imsmanifest.xml“ referenziert (siehe Kapitel 4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-Datei) und beschreibt das Lernobjekt allgemein. Sie kann von LOM-fähigen

Systemen z. B. für die Recherche von Lernobjekten genutzt werden. Grundsätzlich lassen sich auch weitere Metadaten gemäß SCORM (Kapitel 3.8 SCORM) in der Manifest-Datei „imsmanifest.xml“ angeben. Sie werden allerdings vom WLMML-Framework nicht berücksichtigt. Eine Ausnahme bilden die Sprachangaben in Form von Metadaten in „organization“-Elementen. Sie müssen für die Umsetzung der Mehrsprachigkeit im WLMML-Framework eingetragen werden (siehe Kapitel 4.3.2.3.1 Sprachangaben über Metadaten in der Manifest-Datei).

Wie Metadaten allgemein in WLMML-Lernobjekte eingebunden werden können, zeigt Anhang I (siehe auch GEIGER 2005, S. 84 f, RADULESCU 2005, S. 71 ff).

4.3.2.3 Mehrsprachigkeit

Das WLMML-Framework ist für eine mehrsprachiges Angebot der Lernobjekte konzipiert (STREHL et al. 2005, S. 6, QUEDNAU et al. 2007, QUEDNAU & PAAR 2007, PAAR et al. 2006, S. 11, siehe auch GEIGER 2005, S. 53, RADULESCU 2005, S. 56 f, Kapitel 2.4 Mehrsprachigkeit). Es stellt sicher, dass die Mehrsprachigkeit auch offline (siehe Kapitel 4.3.2.4 Clientseitige Verarbeitung) ohne spezifische SCORM-Unterstützung (durch z. B. einen SCORM-Player oder ein LMS) in den meisten zeitgemäßen Browsern funktioniert (siehe Kapitel 4.3.3 Voraussetzungen der Browser-Software). Auch der konsequente Einsatz des Unicode in der Kodierung „UTF-8“ in WLMML und im WLMML-Framework unterstützt diese Ausrichtung auf ein mehrsprachiges E-Learning-Angebot (siehe auch Kapitel 3.3.1 Zeichensätze, 4.2 WLMML).

Wenn ein Lernobjekt in mehreren Sprachen vorliegt, wird dem Anwender eine Sprachauswahl angeboten. Er kann die Sprache für das gesamte Lernobjekt auswählen (bzw. ändern, siehe auch Kapitel 4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-Datei) oder während des Lernprozesses sich einzelne Absätze in den verfügbaren Sprachen anzeigen lassen (siehe auch Kapitel 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen). Abbildung 75 zeigt diese Möglichkeiten der Sprachauswahl am Beispiel des Lernmoduls „10003_-00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ (siehe Anhang H). Daneben lässt sich für die Volltextsuche in WLMML-Inhaltsdateien die Sprache vom Anwender angeben (siehe Kapitel 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten). Zusätzlich ist eine sprachabhängige Suche in verschiedenen Online-Diensten über die Hilfsleiste am unteren Browserrand möglich (siehe Kapitel 4.3.2.4.2.3 Suche in verschiedenen Online-Diensten (nur mit Internet-Zugang möglich)).

Das WLMML-Lernobjekt kann auch zweimal in verschiedenen Browserfenstern geöffnet und nebeneinander in unterschiedlichen Sprachen angezeigt werden. Auf diese Art und Weise lassen sich die Inhalte parallel zweisprachig betrachten (QUEDNAU et al. 2007, S. 4).

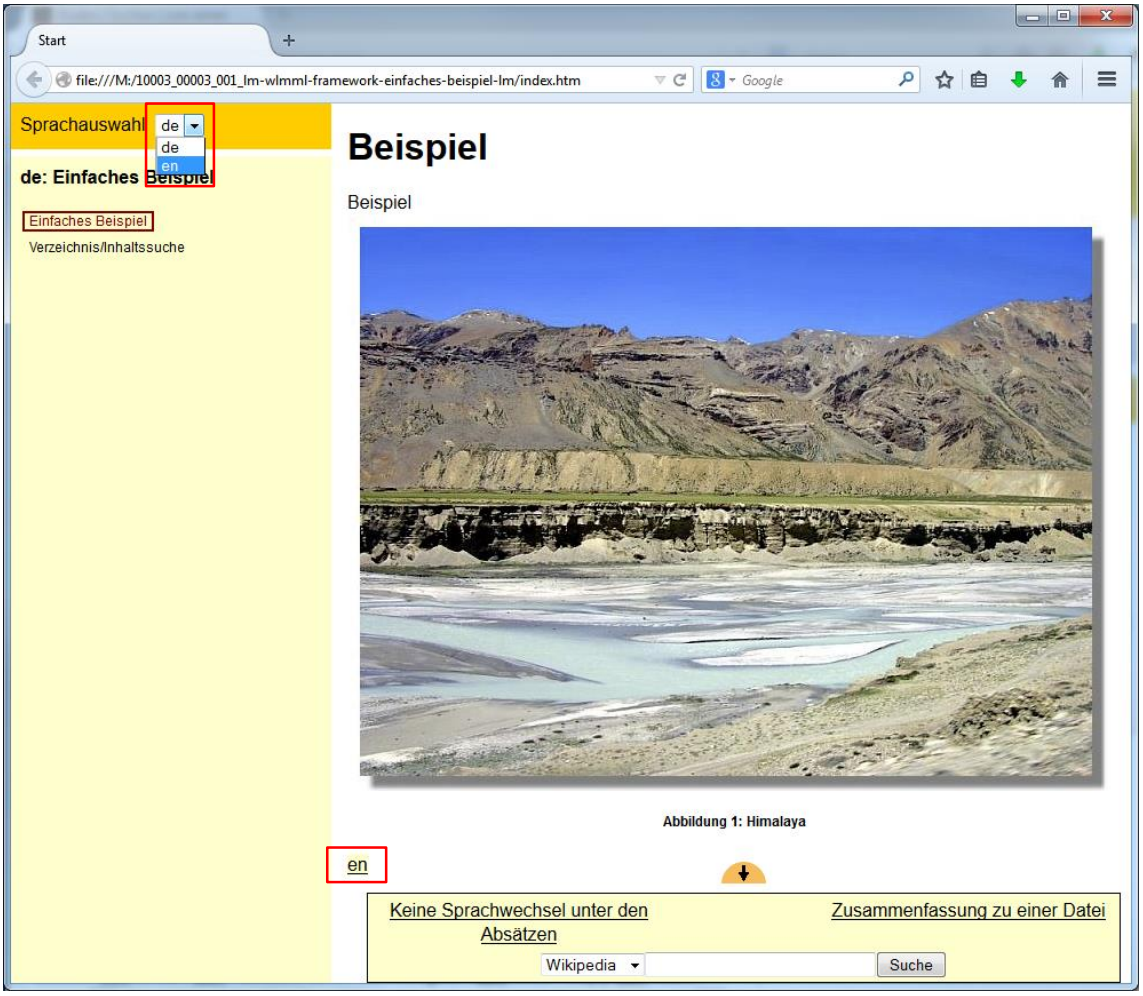


Abbildung 75: Umsetzung der Mehrsprachigkeit in einem WLMML-Lernobjekt (links oben Sprachauswahl für gesamtes Lernobjekt, unten Sprachauswahl für einen Absatz über die Hilfsleiste, Darstellung im Browser „Mozilla Firefox 29.0.1“

Die Sprachauswahl wird automatisch über das WLMML-Framework erzeugt. Der Übersetzer muss die Inhalte in den verschiedenen Sprachen zur Verfügung stellen und braucht keine technischen Anpassungen zur mehrsprachigen Darstellung vornehmen (STREHL et al. 2005, S. 6). Einem Lernobjekt lassen sich damit jederzeit weitere Übersetzungen hinzufügen.

Zur Umsetzung der Mehrsprachigkeit werden Sprachangaben im WLMML-Framework an folgenden Stellen vorgenommen (siehe die kommenden Kapitel):

- über Metadaten-Angaben in der Manifest-Datei „imsmanifest.xml“

- bei der Benennung von Sprachordnern
- in WLMML-Inhaltsdateien
- bei der Einführung einer neuen Systemsprache in der XML-Datei „lang_system.xml“

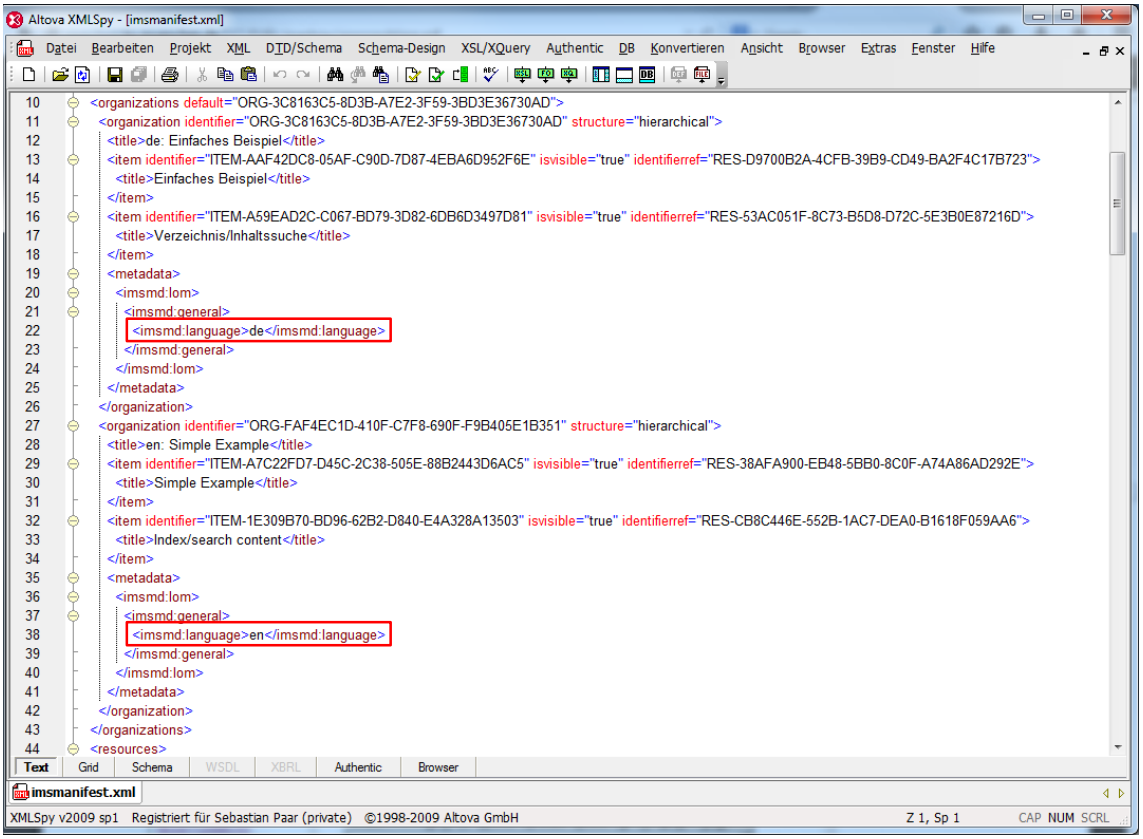
Eine ausführliche Anleitung für die Erstellung eines mehrsprachigen WLMML-SCORM 2004-Lernmoduls findet sich in der Anhang I. Als Beispiel eines einfachen mehrsprachigen Lernmoduls wird in den folgenden Unterkapitel das WLMML-Lernmodul „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ herangezogen (siehe Anhang H).

4.3.2.3.1 Sprachangaben über Metadaten in der Manifest-Datei

Das WLMML-Framework wertet für die Umsetzung der Mehrsprachigkeit die in Form von Metadaten in der Manifest-Datei „imsmanifest.xml“ bei den einzelnen „organization“-Elementen angegebenen Sprachen aus. Dazu muss für jede Inhaltssprache in der „imsmanifest.xml“ ein „organization“-Element angelegt werden. Jedem dieser „organization“-Elemente lässt sich anschließend eine Metadaten-Angabe zuordnen, in welcher Inhaltssprache der Lerninhalt angeboten werden soll (Listing 56 Zeile 19 bis 25, Zeile 35 bis 41, siehe auch Kapitel 4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-Datei, 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei und 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen).

Die Einteilung in „organization“-Elemente mit ihren Sprach-Metadaten-Angaben stellt zusätzlich sicher, dass der Anwender über eine von einem „SCORM 2004-Player“ oder einem SCORM2004-fähigen LMS automatisch erstellten Navigationsmöglichkeit (z. B. Aufklapp-Menü oder Auswahl des jeweiligen „organization“-Elementes) in ein andere Sprache, d. h. ein anderes „organization“-Element wechseln kann (siehe die Unterkapitel zu 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM)). Auf diese Art und Weise wird auf der Grundlage von SCORM indirekt die Mehrsprachigkeit erreicht. Grundsätzlich wird das „organization“-Element von JESUKIEWICZ (2009 a, S. CAM-3-26 f) wie folgt beschrieben: „*The content organization is defined by the <organization> element. The content organization is a conceptual term. The content organization can be a lesson, module, course, chapter, etc. What a content organization defines is dependent on an organization’s curricular taxonomy.*“ Das „organization“-Element kann mehrfach in einer Manifest-Datei auftreten (JESUKIEWICZ 2009 a, S. CAM-3-26, SMYTHE & JACKL 2004 b, 3.3.1 <organization>).

Listing 56 zeigt am Beispiel des Lernmoduls „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“, wie dem ersten „organization“-Element in Zeile 22 die Sprache „de“ und dem zweiten in Zeile 38 die Sprache „en“ zugewiesen werden. Die Kürzel richten sich nach den Sprachbezeichnungen für „deutsch“ und „englisch“ gemäß ISO-639-1 (2002).



Listing 56: Sprachangaben über Metadaten der „organization“-Elemente (Zeile 22 und 38), Ausschnitt aus der XML-Datei „imsmanfest.xml“ des Lernmoduls „10003_-00003_001_lm-wlmml-framework-einfaches-beispiel-lm“

Die zugewiesenen Sprachangaben gelten für das gesamte WLMML-Lernobjekt und müssen den Sprachordnernamen in diesem Lernobjekt entsprechen (siehe Kapitel 4.3.2.3.2 Sprachordner im WLMML-Framework). Davon abweichende Sprachangaben können nur in einer WLMML-Datei vorgenommen werden und gelten dann auch nur für diese (siehe Kapitel 4.3.2.3.3 Sprachangaben in WLMML-Dateien).

4.3.2.3.2 Sprachordner im WLMML-Framework

Damit die Metadaten-Angaben für die Umsetzung der Mehrsprachigkeit greifen, ist es zwingend notwendig, dass entsprechend benannte Sprachordner in der Verzeichnisstruktur des Lernobjektes vorhanden sind. Sollen z. B. die Sprachen „deutsch“ und „englisch“ für das WLMML-Lernobjekt angeboten werden (siehe Listing 56), müssen die Sprachordner mit den Namen „de“ und „en“ im „Root-Verzeichnis“

des Lernobjektes vorliegen (Abbildung 76). Die Kürzel „de“ und „en“ entsprechen den Sprachbezeichnungen für „deutsch“ und „englisch“ gemäß ISO-639-1 (2002) und sind damit identisch mit den Metadaten-Sprachangaben in der Manifest-Datei (siehe Kapitel 4.3.2.3.1 Sprachangaben über Metadaten in der Manifest-Datei). Zur Unterstützung ist ein deutscher Sprach-Musterordner im WLMML-Framework bereits enthalten (siehe Kapitel 4.3.1 Verzeichnisstruktur).

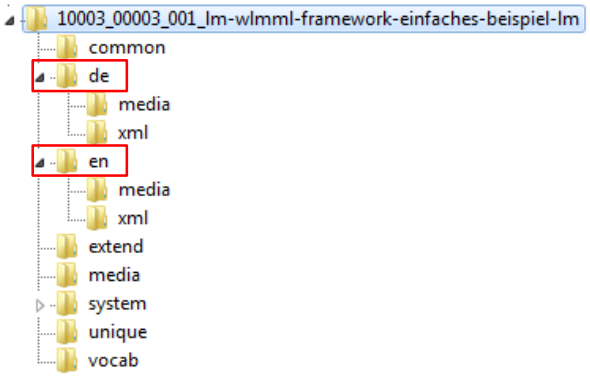


Abbildung 76: Zwei Sprachordner „de“ und „en“

Als praktische Vorgehensweise für die Übersetzung von Lerninhalten wird vorgeschlagen, zuerst die Inhalte einer Sprache fertig zu stellen, den gesamten Sprachordner zu kopieren, neben dem ersten einzufügen, in die neue Sprache umzubenennen (Sprachbezeichnungen nach ISO-639-1 2002) und anschließend die Dateien in den sprachspezifischen Ordnern „media“ und „xml“ zu übersetzen (siehe auch QUEDNAU et al. 2007, S. 6 f). Im Beispiel in der Abbildung 76 sind diese sprachspezifischen Ordner im Verzeichnis mit der Pfadangabe „de/“ bzw. „en/“ zu finden.

Im Ordner „xml“ der jeweiligen Sprache befinden sich die sprachspezifischen WLMML-Dateien (Abbildung 76, siehe auch 4.3.1 Verzeichnisstruktur). Die Dateinamen der WLMML-Dateien mit einem anderssprachigen Inhalt müssen identisch lauten. D. h. in jedem „xml“-Sprachordner befinden sich Dateien mit gleichem Namen. Abbildung 77 (rechte Seite) und Abbildung 78 (rechte Seite) zeigen den Inhalt zweier Sprachordner „de/xml/“ und „en/xml/“.

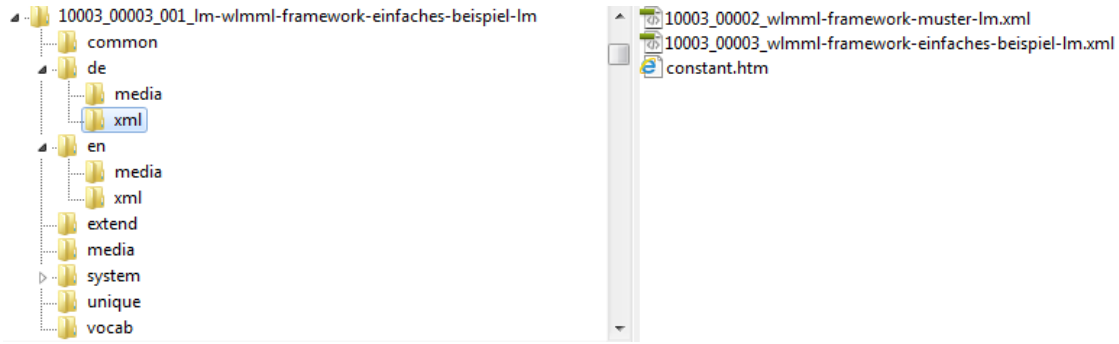


Abbildung 77: Deutsche WLMML-Inhaltsdateien im Ordner „de/xml/“

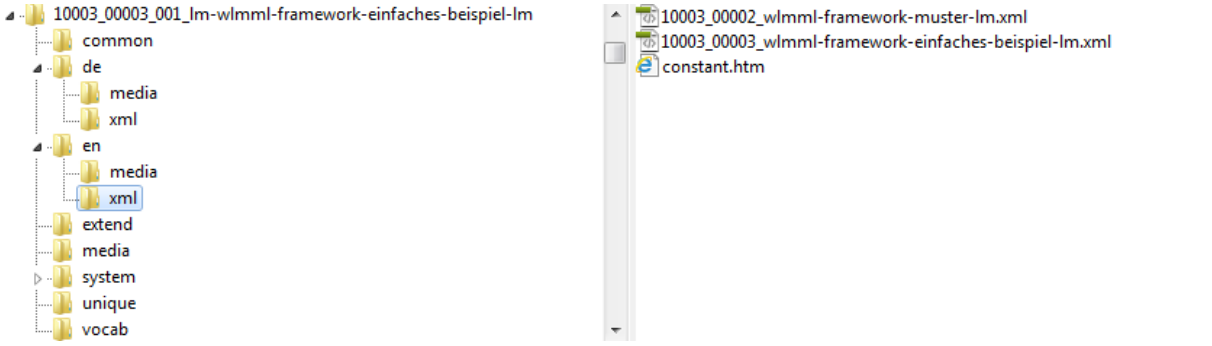


Abbildung 78: Gleichnamige englische WLMML-Inhaltsdateien im Ordner „en/xml/“

4.3.2.3.3 Sprachangaben in WLMML-Dateien

Das im „wlmml“-Element einer jeden WLMML-Datei erforderliche Attribut „languageSystem“ gibt an, mit welcher Systemsprache die WLMML-Datei angezeigt wird (z. B. für die Sprache der Texte auf Systemschaltflächen bzw. von Hyperlink-Texten in der Hilfsleiste am unteren Browserrand, siehe Kapitel 4.3.2.4.2 Funktionalitäten in der Hilfsleiste am unteren Browserrand). Die Systemsprache kann von der Inhaltssprache der WLMML-Datei abweichen (falls z. B. die Systemsprache noch nicht existiert). Die Auswirkungen einer individuellen Änderung der Angabe einer Systemsprache zeigen sich nur für diese WLMML-Datei und wirken sich nicht auf andere WLMML-Dateien aus. Die Systemsprachen selbst sind in der XML-Datei „lang_system.xml“ des WLMML-Frameworks festgehalten (siehe Kapitel 4.3.2.3.4 Systemsprachen im WLMML-Framework).

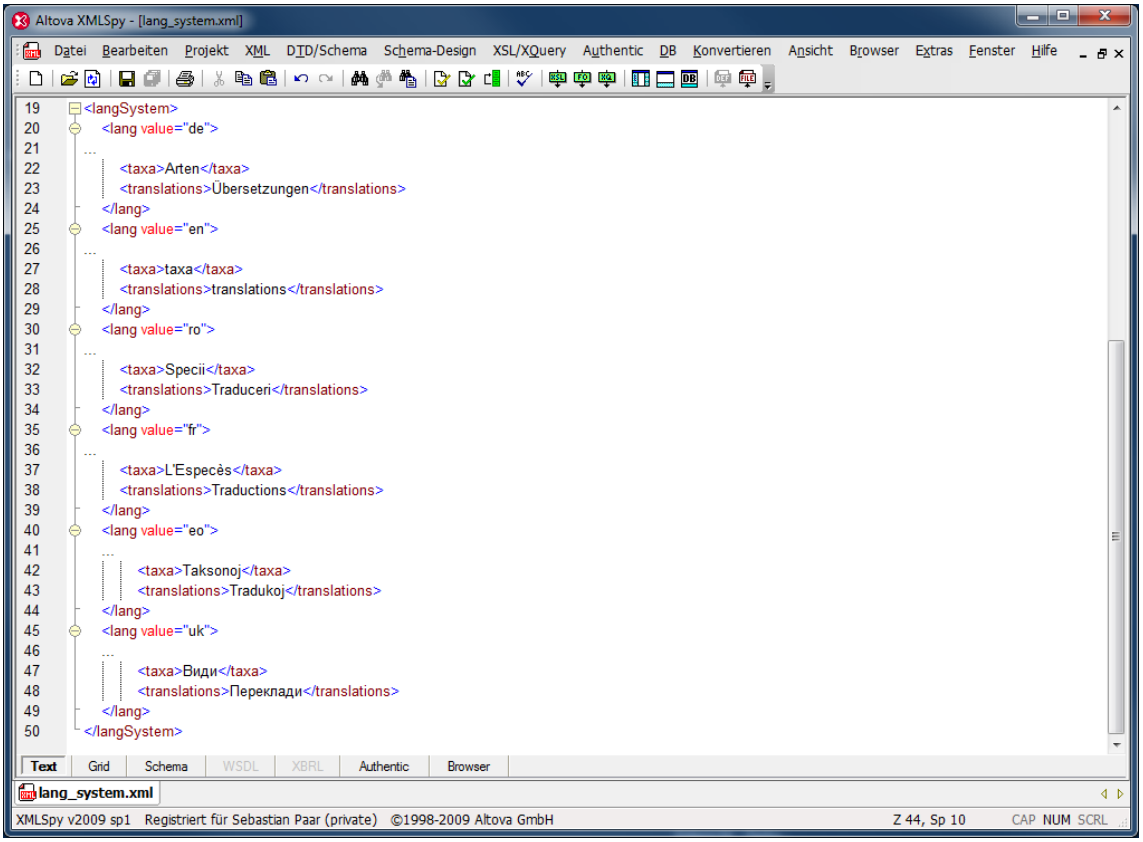
Das optionale Attribut „translated“ im „wlmml“-Element einer WLMML-Datei hält speziell fest, in welchen Sprachen die Inhaltsdatei übersetzt vorliegt (z. B. translated=“eo fr“, Sprachbezeichnungen nach ISO-639-1 (2002)). Liegt etwa ein Lernmodul komplett in deutscher und englischer Sprache vor, könnte z. B. eine finnischsprachige WLMML-Datei mit deutscher Systemsprache und ihren Übersetzungen in „eo“ und „fr“ (translated=“eo fr“) in die deutschen Inhalte eingebunden werden (weitere Details siehe Anhang I).

Damit spezifisch jeder Absatz einer WLMML-Inhaltsdatei im Browser in einer anderen Sprache angezeigt werden kann, muss er im WLMML-Dokument über seinen „label“-Attributwerte eindeutig identifizierbar sein (siehe Kapitel 4.2.3.5 Attribute für das Element „paragraph“, 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen, 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei). Außerdem müssen die übersetzte WLMML-Inhaltsdatei den gleichen Dateinamen und die korrespondierenden Absätze die gleichen „label“-Attributwerte

aufweisen. Die Reihenfolge der Absätze in den jeweiligen übersetzten WLMML-Inhaltsdateien spielt keine Rolle. Ist der ausgewählte Absatz in der gleichnamigen übersetzten Datei nicht vorhanden (bzw. kein identischer „label“-Attributwert), öffnet der Browser nur diese WLMML-Inhaltsdatei. Existiert die per Hyperlink aufgerufene übersetzte Datei nicht, gibt der Browser seine Standardmeldung aus, dass die Datei nicht gefunden werden kann.

4.3.2.3.4 Systemsprachen im WLMML-Framework

Die vorhandenen Systemsprachen sind in der XML-Datei „lang_system.xml“ im Verzeichnis „system/xml/“ des WLMML-Frameworks aufgeführt (siehe auch QUEDNAU et al. 2007, S. 8). Sie beinhaltet sprachspezifische Systemausdrücke (z. B. „Zusammenfassung zu einer Datei“, „Arten“, „Übersetzungen“) in 6 Sprachen (Listing 57). Diese Ausdrücke verwendet das WLMML-Framework bei sprachspezifischen Angaben z. B. innerhalb der Hilfsleiste am unteren Browserrand (siehe Kapitel 4.3.2.4.2 Funktionalitäten in der Hilfsleiste am unteren Browserrand) oder bei der sprachspezifischen Benennung von Verzeichnissen (siehe Kapitel 4.3.2.4.3 Automatische Erstellung von Verzeichnissen).



Listing 57: Sprachspezifische Systemausdrücke in 6 Sprachen, Ausschnitt aus der XML-Datei „lang_system.xml“

Existiert eine System-Sprache noch nicht (z. B. für die Sprache der Texte auf Systemschaltflächen bzw. von Hyperlink-Texten in der Hilfsleiste am unteren

Browserrand, siehe Kapitel 4.3.2.4.2 Funktionalitäten in der Hilfsleiste am unteren Browserrand), kann sie über die XML-Datei „lang_system.xml“ mit einem XML-Editor oder beliebigen Text-Editor angelegt werden. Es wird vorgeschlagen eine existierende Sprache komplett zu markieren, kopieren, einzufügen und zu übersetzen. Als Anhalt dienen die bereits eingetragenen System-Sprachen. Ansonsten sind keine weiteren Aktionen für den Einsatz der neuen Systemsprache notwendig (außer ihrer Angabe im Attribut „languageSystem“ des „wlmml“-Elementes der jeweiligen WLMML-Datei). Die Umsetzung wird automatisch durch das WLMML-Framework übernommen.

4.3.2.4 Clientseitige Verarbeitung

Die komplette Funktionalität des auf der Basis von WLMML erstellten E-Learning-Inhaltes wird vom Client-Rechner (Client-Software Browser) sichergestellt. Auf serverseitige Programmiersprachen und die Anbindung an einen Server kann deshalb verzichtet werden. Es muss damit z. B. keine Internetverbindung vorhanden sein, um die lokal abgelegten Lerninhalte zu nutzen. Auch wenn die WLMML-Framework-Dateien z. B. ursprünglich auf einem Server liegen, werden sie bei Bedarf über den Aufruf der WLMML-Dateien (Lerninhalte) im Browserfenster komplett auf den Client-Rechner geladen und dort für die Verarbeitung der WLMML-Dateien eingesetzt (siehe auch QUEDNAU et al. 2007, S. 8). Abbildung 79 zeigt, dass beim Aufruf der Start-HTML-Seite „index.htm“ (hier z. B. von einem Webserver im Internet unter „http://www.selfinternet.de/10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm/index.htm“) von diesem Web-Server über das Internet die benötigten WLMML-Framework-Dateien zum Client-Rechner geladen werden. Sie stehen damit zur clientseitigen Verarbeitung zur Verfügung (siehe Kapitel 4.3.2.4.1 Verarbeitungsvorgang für die Darstellung im Browser). In der Auflistung der geladenen Dateien sind die benötigten HTML-, CSS-, JavaScript-, XML-, XSL-, JPG- und GIF-Dateien zu finden. Die Dateinamenserweiterung „htm“ taucht häufig unter Windows-Betriebssystemen auf (frühere „8.3“-Windows-Regel bei Dateinamen). Abbildung 80, Abbildung 81 und Abbildung 82 zeigen das Verarbeitungsergebnis in verschiedenen Browser. Im vorliegenden Fall dient als Beispiel das WLMML-Lernmodul „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ (siehe Anhang H). Auf den Ausdruck Lernmodul wurde näher im Kapitel 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM) eingegangen.

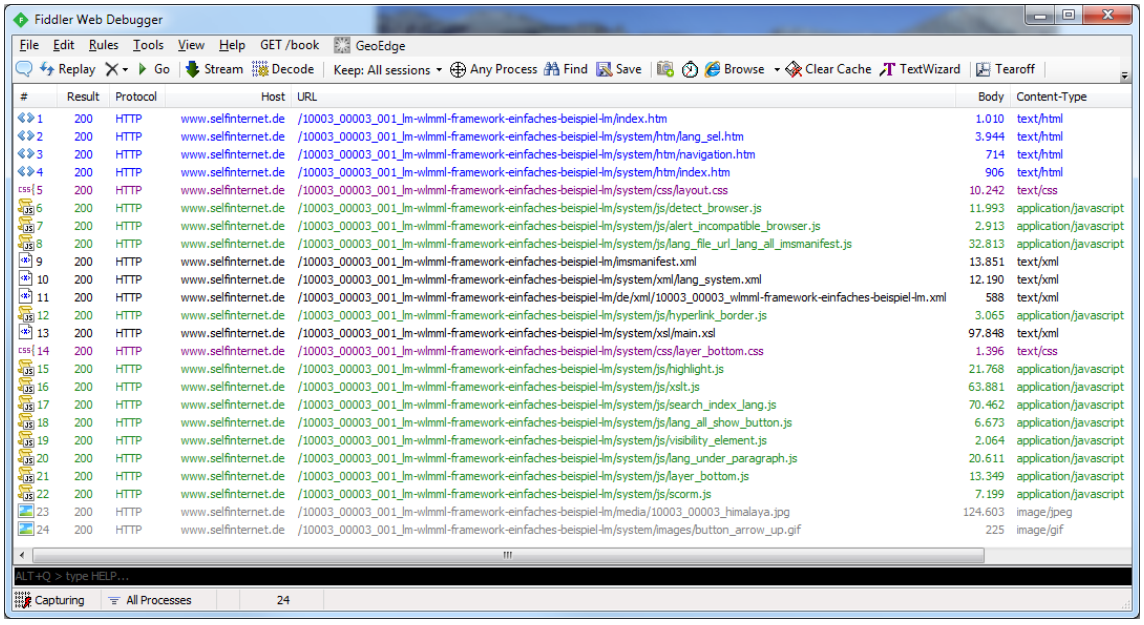


Abbildung 79: Geladene Dateien des WLMML-Frameworks nach Aufruf der Start-HTML-Seite „index.htm“ von einem Web-Server im Internet (FIDDLER 2014)

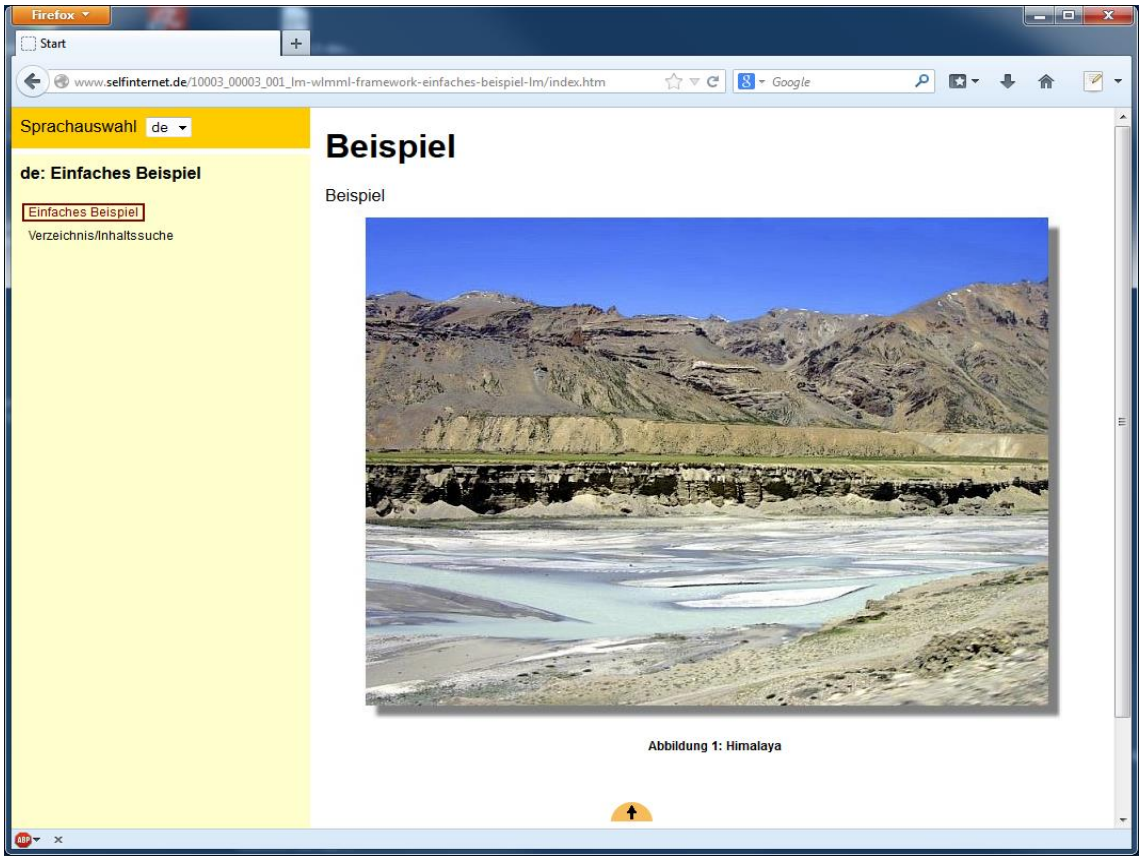


Abbildung 80: Anzeige im Browser („Mozilla Firefox 28.0“) nach Aufruf der Start-HTML-Seite „index.htm“ von einem Web-Server im Internet

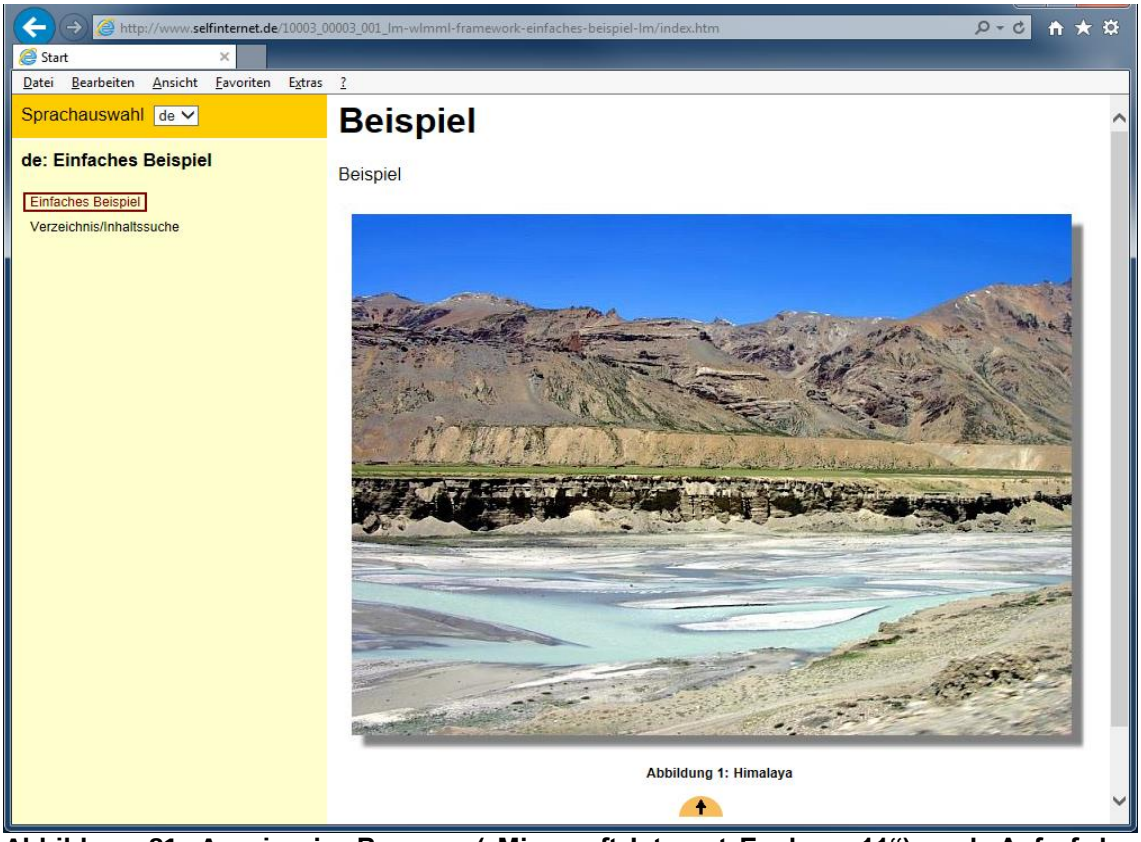


Abbildung 81: Anzeige im Browser („Microsoft Internet Explorer 11“) nach Aufruf der Start-HTML-Seite „index.htm“ von einem Web-Server im Internet

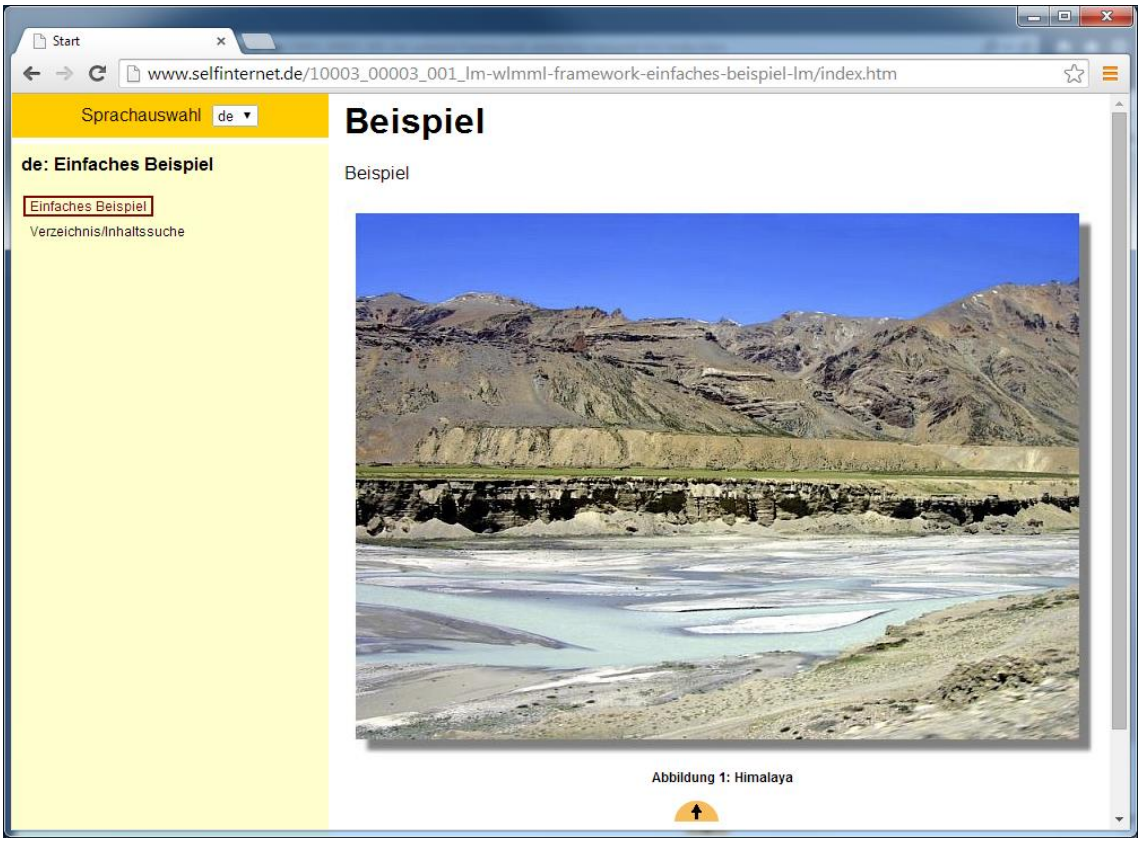


Abbildung 82: Anzeige im Browser („Google Chrome 34“) nach Aufruf der Start-HTML-Seite „index.htm“ von einem Web-Server im Internet

Zur clientseitigen Verarbeitung der WLMML-Dateien wird das WLMML-Framework vor allem mit seinen XSLT- und JavaScript-Dateien herangezogen (siehe Kapitel 3.3.8.1 XSLT, 3.6.1 Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript, vgl. auch Kapitel 3.6.2 Ajax). Aus Flexibilitätsgründen wurde bei der WLMML-Framework-Entwicklung nicht auf bereits vorhandene JavaScript-Frameworks zurückgegriffen (u. a. eigene Entwicklungen möglich, spezifische Anforderungen umsetzbar). Die clientseitige Programmiersprache JavaScript bietet sich besonders an, da serverseitige Programmiersprachen (wie z. B. PHP) im Quelltext von Lerninhalten von LMSs nicht immer unterstützt werden.

Die im WLMML-Framework abgelegten Lerninhalte sind in Standard-Browsern wie z. B. „Mozilla Firefox“, „Google Chrome“ oder „Microsoft Internet Explorer“ lauffähig (siehe Kapitel 4.3.3 Voraussetzungen der Browser-Software). Es ist keine Installation und im Grundsatz auch keine Verbindung zu einem Internet/Intranet-Server nötig (bei voller Funktionstauglichkeit z. B. der Mehrsprachigkeit oder der Volltextsuche in WLMML-Lerninhalten). Der Grund dafür ist (neben dem konsequenten Einsatz internationaler Spezifikationen/Standards), dass nach einem Download der Lerninhalte bei der Verarbeitung der WLMML-Dateien keine Datenübertragung zwischen einem Server und Client (Browser) stattfinden muss (clientseitige Verarbeitung). Die identischen Lerninhalte können damit ohne Anpassungen z. B. online über eine Lernplattform, über einen Web-Server (Abbildung 79, Abbildung 80), einen Datei-Server oder offline (nach dem Download) im Browser betrachtet werden. STREHL et al. 2005 (S. 3) halten fest, dass der Einsatz von WLMML-Lerninhalten unabhängig von der Verwendung eines LMS möglich ist. Sie führen weiter aus, dass WLMML-Lerninhalte eigenständig wie Web-Seiten eingesetzt, als SCORM-Pakete in verschiedene LMS importiert oder auch eigenständig (z. B. lokal) eingesetzt werden können. Ohne ein SCORM-fähiges System stehen die SCORM-spezifischen Funktionalitäten allerdings nicht zur Verfügung.

Im Gegensatz zur beschriebenen Ajax-Technologie (siehe Kapitel 3.6.2 Ajax), bei der eine Serververbindung angenommen wird, ist diese Serveranbindung beim Einsatz des WLMML-Frameworks nicht notwendig. Trotzdem verwendet das WLMML-Framework Aspekte der Ajax-Technologie (z. B. das „XMLHttpRequest“-Objekt, siehe z. B. Kapitel 4.3.2.4.2.2 Zusammenfassung der Lerninhalte in einem Dokument). Auch bei dem System PaKMaS (siehe Kapitel 3.3.6.1.4 „*Passauer Knowledge Management System*) handelt es sich laut SÜSS (2004, S. 128 f) um „*ein webbasiertes Client-Server-System*“, das damit eine Anbindung an einen Server fordert.

In den folgenden Unterkapiteln wird unter anderem weiter auf die clientseitige Verarbeitung eingegangen. Als Beispiel dient das WLMML-Lernmodul „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ (siehe Anhang H). Alle in diesen Kapiteln angesprochenen Dateien sind darin zu finden (siehe Kapitel 4.3.1 Verzeichnisstruktur, Pfadangaben zu den jeweiligen Dateien siehe Abbildung 79 Spalte URL). Grundsätzlich sind alle Dateien im WLMML-Framework mit detaillierten Quelltext-Kommentaren versehen (siehe z. B. Listing 58, Listing 59). Aus den aufgeführten Listings wurden aus Gründen der Übersichtlichkeit in JavaScript-Anweisungen die Aufrufe der „alert()“-Funktionen (siehe Kapitel 3.6 JavaScript) und in allen Dokumenten viele Kommentare entfernt. Alle Funktionalitäten (bis auf die in Kapitel 4.3.2.4.2.3 Suche in verschiedenen Online-Diensten (nur mit Internet-Zugang möglich)) können über den Aufruf der Start-HTML-Datei „index.htm“ online (z. B. über einen Web-Server im Internet) oder offline (z. B. von einer lokalen Festplatte aus) nachvollzogen werden.

4.3.2.4.1 Verarbeitungsvorgang für die Darstellung im Browser

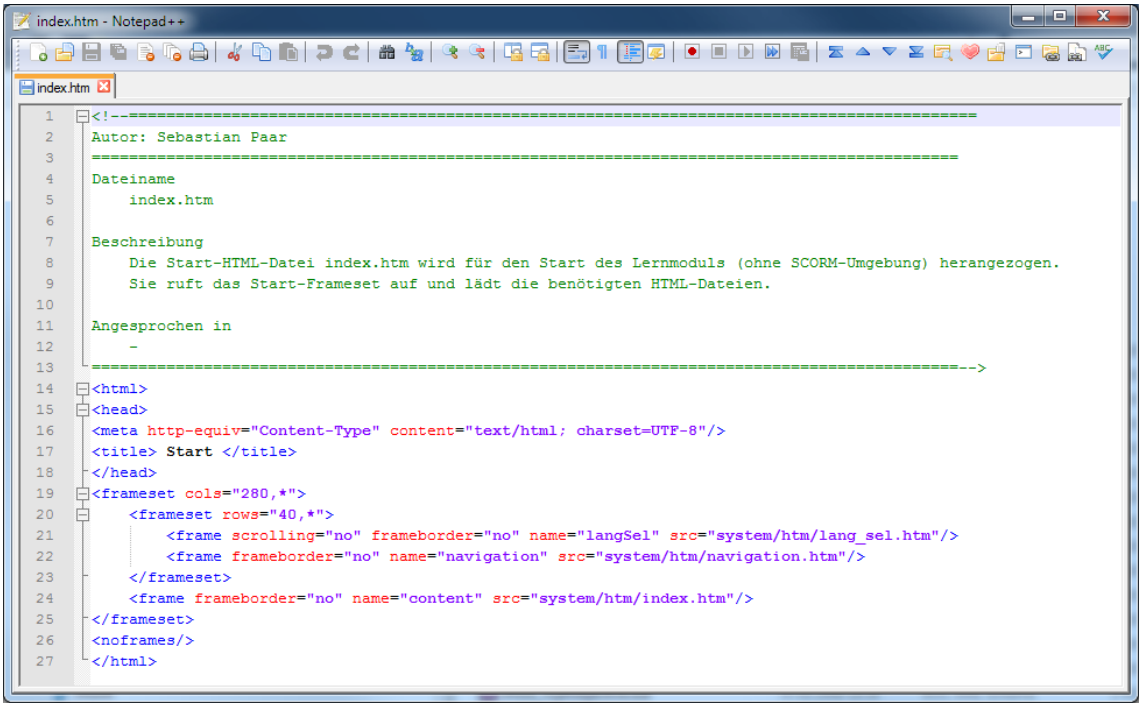
Die Verarbeitung von WLMML-Dateien und die dabei verwendeten Dateien des WLMML-Frameworks sollen im Folgenden beleuchtet werden. Der Schwerpunkt der Betrachtung liegt dabei auf dem Verarbeitungsablauf beim Aufruf der Start-HTML-Datei bzw. einer WLMML-Datei im Browser.

4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-Datei

Das Beispiel-Lernmodul wird über die Start-HTML-Datei „index.htm“ im Root-Verzeichnis des Frameworks aufgerufen. Abbildung 79 zeigt alle Dateien in ihrer Ladereihenfolge und Abbildung 80 das Endergebnis im Browser. Im Folgenden werden alle Dateien der Abbildung 79 erläutert.

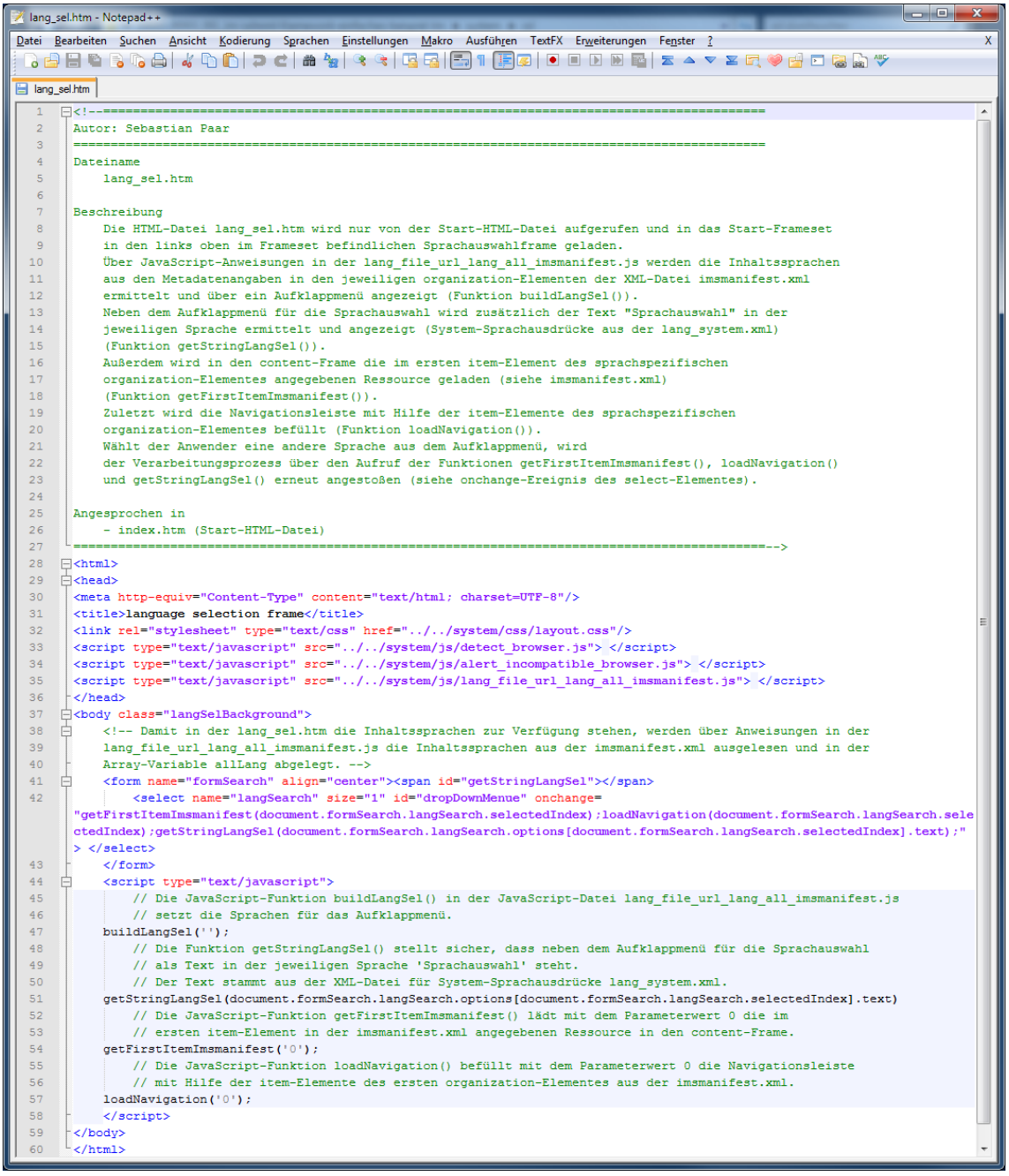
Die Start-HTML-Datei erstellt das Start-Frameset. Wie aus Listing 58 ersichtlich wird, lädt sie dazu in den links oben befindlichen Sprachauswahl-Frame die HTML-Datei „lang_sel.htm“ (Zeile 21, Pfadangabe zur Datei „system/htm/lang_sel.htm“), in den links befindlichen „navigation“-Frame die HTML-Datei „navigation.htm“ (Zeile 22, Pfadangabe zur Datei „system/htm/navigation.htm“) und in den rechts befindlichen „content“-Frame die HTML-Datei „index.htm“ (Zeile 24, Pfadangabe zur Datei „system/htm/navigation.htm“) (siehe auch Abbildung 79 Zeile 1 bis 4). Bei der in den „content“-Frame geladenen HTML-Datei „index.htm“ handelt es sich um eine andere Datei als die gleich benannte Start-HTML-Datei im „Root-Verzeichnis“ des WLMML-Frameworks. Die beiden HTML-Dateien „navigation.htm“ (im Frame mit dem Namen „navigation“) und „index.htm“ (im Frame mit dem Namen „content“) sind nur

Platzhalter. Ausgelöst durch Anweisungen in der HTML-Datei „lang_sel.htm“ werden später an ihre Stelle die Navigationsleiste und die erste Inhaltsseite gesetzt.



Listing 58: Quelltext der HTML-Start-Seite „index.htm“ (inkl. HTML-Kommentare in grüner Farbe, Start-Zeichenfolge „<!--“, Ende-Zeichenfolge „-->“, z. B. Zeile 1 und Zeile 13)

Sobald die HTML-Datei „lang_sel.htm“ in den linken oberen Frame geladen wurde (Listing 59), werden verschiedene Prozesse angestoßen.



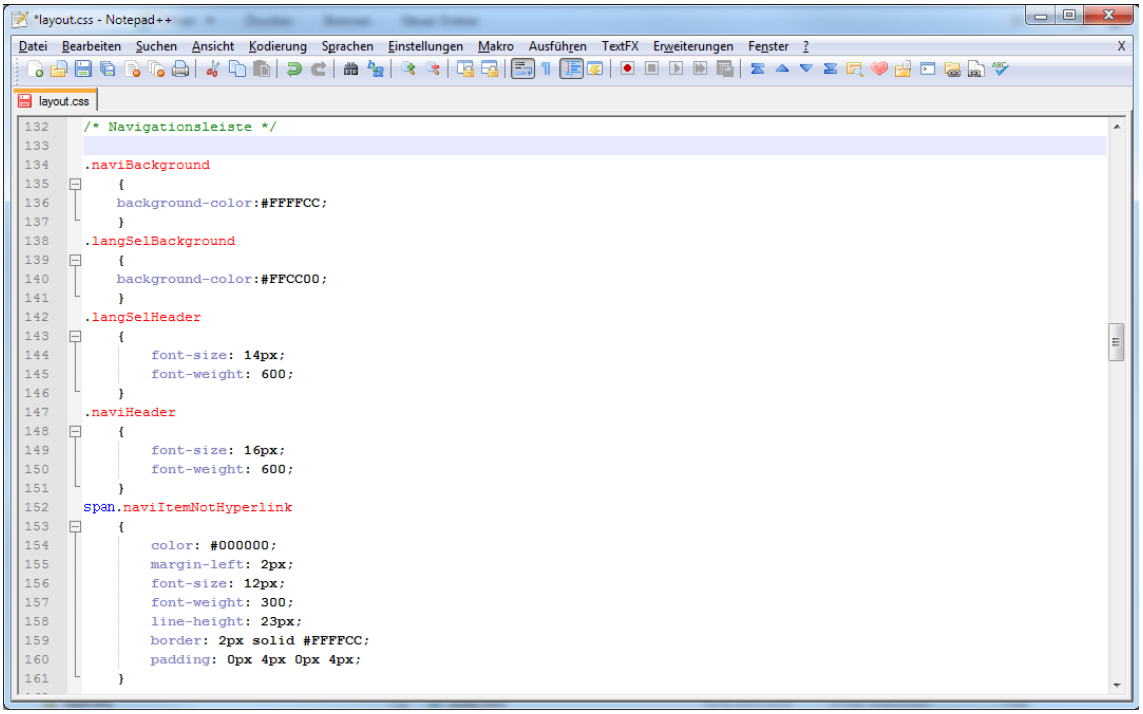
```
<!--============================================================================================
Autor: Sebastian Paar
============================================================================================
Dateiname
    lang_sel.htm

Beschreibung
    Die HTML-Datei lang_sel.htm wird nur von der Start-HTML-Datei aufgerufen und in das Start-Frameset
    in den links oben im Frameset befindlichen Sprachauswahlframe geladen.
    Über JavaScript-Anweisungen in der lang_file_url_lang_all_imsmanifest.js werden die Inhaltssprachen
    aus den Metadatenangaben in den jeweiligen organization-Elementen der XML-Datei imsmanifest.xml
    ermittelt und über ein Aufklappmenü angezeigt (Funktion buildLangSel()).
    Neben dem Aufklappmenü für die Sprachauswahl wird zusätzlich der Text "Sprachauswahl" in der
    jeweiligen Sprache ermittelt und angezeigt (System-Sprachausdrücke aus der lang_system.xml)
    (Funktion getStringLangSel()).
    Außerdem wird in den content-Frame die im ersten item-Element des sprachspezifischen
    organization-Elementes angegebenen Ressource geladen (siehe imsmanifest.xml)
    (Funktion getFirstItemImsmanifest()).
    Zuletzt wird die Navigationsleiste mit Hilfe der item-Elemente des sprachspezifischen
    organization-Elementes befüllt (Funktion loadNavigation()).
    Wählt der Anwender eine andere Sprache aus dem Aufklappmenü, wird
    der Verarbeitungsprozess über den Aufruf der Funktionen getFirstItemImsmanifest(), loadNavigation()
    und getStringLangSel() erneut angestoßen (siehe onchange-Ereignis des select-Elementes).

Angesprochen in
    - index.htm (Start-HTML-Datei)
============================================================================================-->
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
<title>language selection frame</title>
<link rel="stylesheet" type="text/css" href="../../system/css/layout.css"/>
<script type="text/javascript" src="../../system/js/detect_browser.js"> </script>
<script type="text/javascript" src="../../system/js/alert_incompatible_browser.js"> </script>
<script type="text/javascript" src="../../system/js/lang_file_url_lang_all_imsmanifest.js"> </script>
</head>
<body class="langSelBackground">
    <!-- Damit in der lang_sel.htm die Inhaltssprachen zur Verfügung stehen, werden über Anweisungen in der
    lang_file_url_lang_all_imsmanifest.js die Inhaltssprachen aus der imsmanifest.xml ausgelesen und in der
    Array-Variable allLang abgelegt. -->
    <form name="formSearch" align="center"><span id="getStringLangSel"></span>
        <select name="langSearch" size="1" id="dropDownMenue" onchange=
"getFirstItemImsmanifest(document.formSearch.langSearch.selectedIndex);loadNavigation(document.formSearch.langSearch.sele
ctedIndex);getStringLangSel(document.formSearch.langSearch.options[document.formSearch.langSearch.selectedIndex].text);"
> </select>
    </form>
    <script type="text/javascript">
        // Die JavaScript-Funktion buildLangSel() in der JavaScript-Datei lang_file_url_lang_all_imsmanifest.js
        // setzt die Sprachen für das Aufklappmenü.
    buildLangSel('');
        // Die Funktion getStringLangSel() stellt sicher, dass neben dem Aufklappmenü für die Sprachauswahl
        // als Text in der jeweiligen Sprache 'Sprachauswahl' steht.
        // Der Text stammt aus der XML-Datei für System-Sprachausdrücke lang_system.xml.
    getStringLangSel(document.formSearch.langSearch.options[document.formSearch.langSearch.selectedIndex].text)
        // Die JavaScript-Funktion getFirstItemImsmanifest() lädt mit dem Parameterwert 0 die im
        // ersten item-Element in der imsmanifest.xml angegebenen Ressource in den content-Frame.
    getFirstItemImsmanifest('0');
        // Die JavaScript-Funktion loadNavigation() befüllt mit dem Parameterwert 0 die Navigationsleiste
        // mit Hilfe der item-Elemente des ersten organization-Elementes aus der imsmanifest.xml.
    loadNavigation('0');
    </script>
</body>
</html>
```

Listing 59: Quelltext der HTML-Datei „lang_sel.htm“, (inkl. JavaScript-Kommentare in grüner Farbe, Zeichenfolge „//“ am Beginn einer Kommentarzeile, z. B. Zeile 45)

Zuerst wird eine weitere Datei einbezogen, die mit ihren Formatvorlagen für die Formatierung der HTML-Seite verantwortlich ist. Es handelt sich um die zentrale CSS-Datei „layout.css“ (Listing 59 Zeile 32, siehe auch Ladereihenfolge in Abbildung 79 Zeile 5). Listing 60 gibt einen Ausschnitt aus dieser Datei mit Formatvorlagen für die Navigationsleiste wieder. Listing 60 Zeile 138 zeigt die allgemeine CSS-Klasse „.langSelBackground“, die in Listing 59 Zeile 37 als Formatvorlage für die Hintergrundfarbe der HTML-Seite eingesetzt wird (linker oberer Frame).



Listing 60: **Ausschnitt aus der CSS-Datei „layout.css“ mit Formatvorlagen für die Navigationsleiste (inkl. CSS-Kommentar in grüner Farbe, Start-Zeichenfolge „/*“, Ende-Zeichenfolge „*/“, z. B. Zeile 132)**

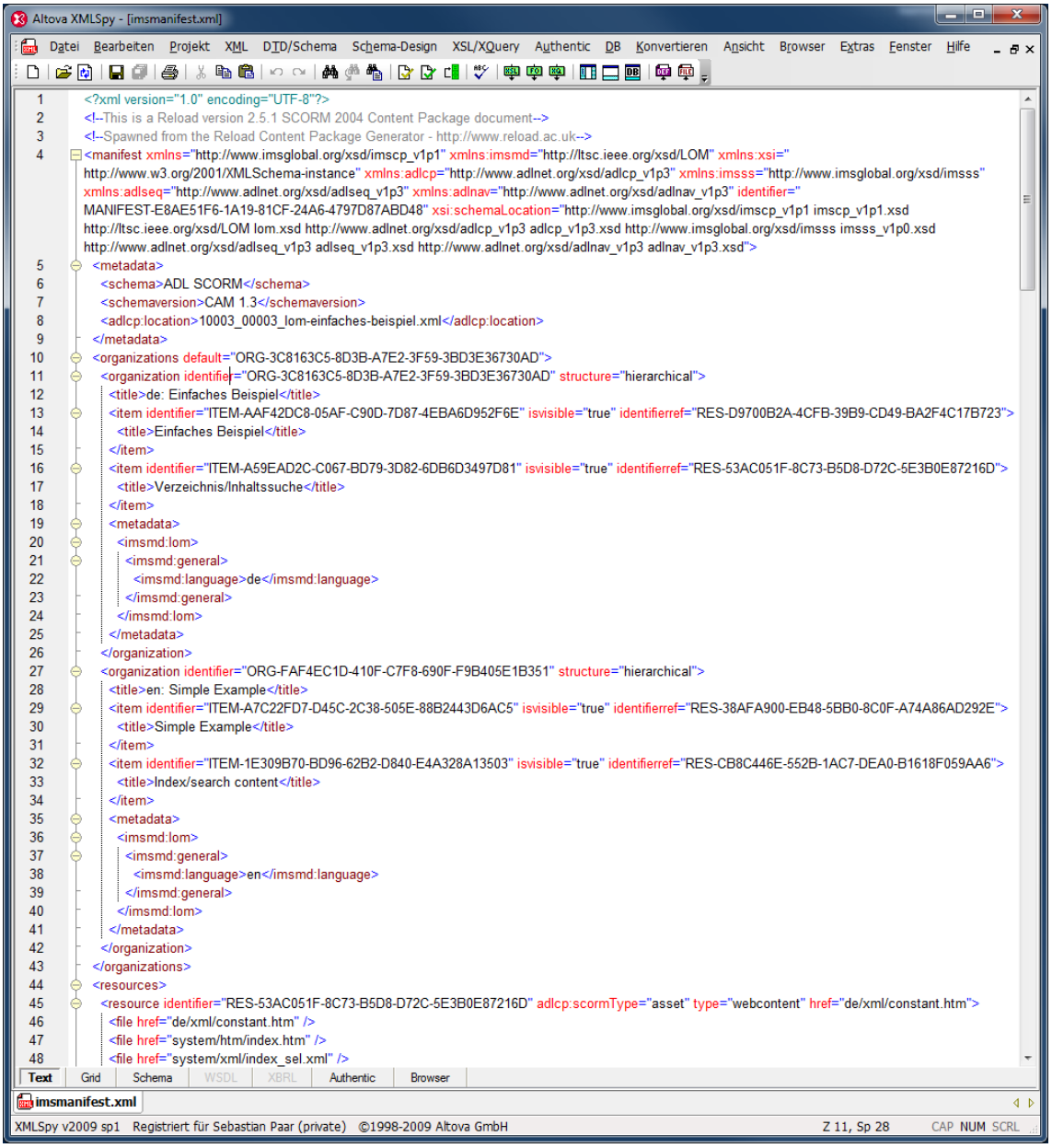
Als nächstes wird die JavaScript-Datei „detect_browser.js“ geladen (Listing 59 Zeile 33, siehe auch Ladereihenfolge in Abbildung 79 Zeile 6). Sie ermittelt den Browser, der die Start-HTML-Seite aufgerufen hat und speichert das Ergebnis in der Variablen „gBrowser“. Durch die Auswertung des Variableninhaltes kann auf die unterschiedlichen Browser differenziert in den späteren Anweisungen reagiert werden. Damit soll sichergestellt werden, dass trotz unterschiedlicher Umsetzung der Verarbeitung der XML-Dateien in den verschiedenen Browser das gewünschte Ergebnis eintritt.

Tritt trotzdem bei der Verarbeitung ein Fehler auf, erscheint eine dreisprachige Meldung (deutsch, englisch, esparanto) mit dem Hinweis am Bildschirm, dass nur die angegebenen Browser die Inhalte darstellen können. Sie wird über eine „alert“-Anweisung in der JavaScript-Datei „alert_incompatible_browser.js“ erzeugt (Listing 59 Zeile 34, siehe auch Ladereihenfolge in Abbildung 79 Zeile 7).

Die JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“ wird anschließend aufgerufen (Listing 59 Zeile 35, siehe auch Ladereihenfolge in Abbildung 79 Zeile 8). Sie stellt unterschiedliche Funktionen zur Verfügung, die unter anderem die verschiedenen Lerninhaltssprachen ermitteln. Dazu wird die XML-Datei

„imsmanifest.xml“ aufgerufen und ausgewertet (siehe auch Ladereihenfolge in Abbildung 79 Zeile 9). Listing 61 zeigt einen Ausschnitt aus dieser Datei (vergleiche auch 3.3.7.1 IMS CP, 3.3.7.3.6 IMS CP und IMS LD in einem XML-Beispiel). Zu Beginn werden im Root-Element „manifest“ (Listing 61 Zeile 4) die Namespaces (3.3.4 XML-Namensräume) und die Schema-Dateien (3.3.3 DTD / W3C XML Schema Definition) für die Validierung dieser XML-Datei referenziert. In Zeile 5 bis 9 (Listing 61) wird auf eine externe Metadaten-Datei (LOM-Datei) hingewiesen. Anschließend folgen die Angaben der Ressourcen (Listing 61 Zeile 44 ff). Welche Inhalte in die Navigationsleiste kommen und welche Datei als erste Inhaltsseite im „content“-Frame erscheint, wird aus Informationen in dieser zentralen XML-Datei „imsmanifest.xml“ ermittelt. Sie listet strukturiert alle WLMML-Dateien SCORM-konform auf (siehe Kapitel 3.8 SCORM). Die XML-Datei „imsmanifest.xml“ stellt innerhalb von SCORM eine Art Inhaltsverzeichnis der Lerneinheit dar (mit Pfadangabe zu den einzelnen Dateien = Ressourcen). Sie wird als SCORM-Grundbaustein im WLMML-Framework wie auch in SCORM-kompatiblen Lernplattformen vorausgesetzt. Im WLMML-Framework spiegelt sich der Inhalt der XML-Datei „imsmanifest.xml“ u. a. als Navigationsstruktur im linken Teil des Browser-fensters wieder (siehe weiter unten) und wird z. B. als Auflistung der in der Volltextsuche (Textsuche in WLMML-Inhaltsdateien) zu durchsuchenden WLMML-Inhaltsdateien benötigt (siehe Kapitel 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten).

Um die verschiedenen Lerninhaltssprachen aus der zentralen XML-Datei „imsmanifest.xml“ zu ermitteln, sind die jeweiligen „organization“-Elemente von Bedeutung. Aus ihren Metadatenangaben (Listing 61 Zeile 22 und 38) werden über Anweisungen in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“ die Sprachen ausgelesen.

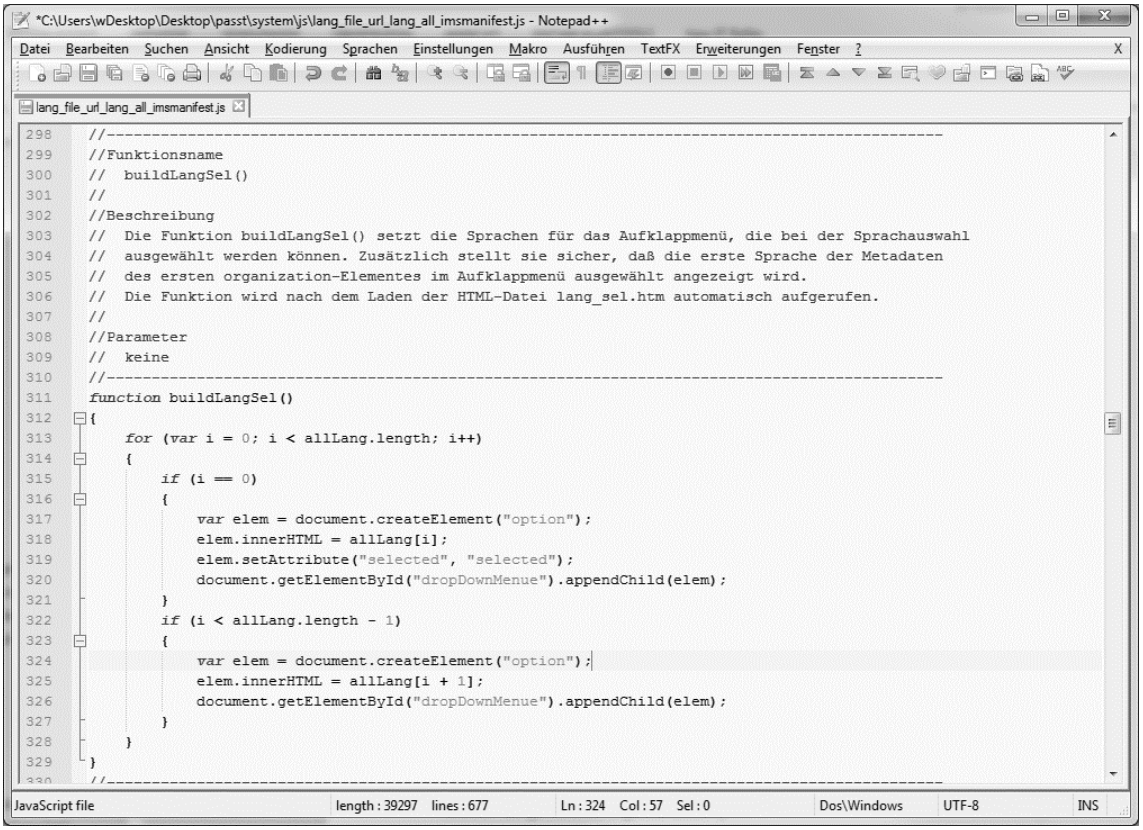


Listing 61: Ausschnitt aus der XML-Datei „imsmanfest.xml“ (inkl. HTML-Kommentare in grauer Farbe, Start-Zeichenfolge „<!--“, Ende-Zeichenfolge „-->“, z. B. Zeile 2)

Dazu wird zuerst die XML-Datei „imsmanifest.xml“ aufgrund der Browserunterschiede über zwei mögliche Varianten in die globale Variable „xmlSource“ geladen (Listing 62 Zeile 104 und 116, siehe auch Kapitel 3.6.1 Clientseitige Verarbeitung von XML-Dateien mit XSLT und JavaScript, 3.6.2 Ajax). Die Variable steht damit mit ihrem Inhalt nicht nur in dieser Funktion sondern in der gesamten JavaScript-Datei zur Verfügung. Anschließend werden über die Anweisungen in der Funktion „searchAllLangImsmanifest()“ die Sprachen aus den „language“-Element-Inhalten ausgelesen (Listing 62 Zeile 138 und 144, unterschiedliche Herangehensweise aufgrund der Browserunterschiede) und in die Array-Variable „allLang“ geschrieben (Listing 62 Zeile 149).

Listing 62: Auslesen der Inhaltssprachen aus den Metadatenangaben in den jeweiligen „organization“-Elementen, Ausschnitt aus der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“

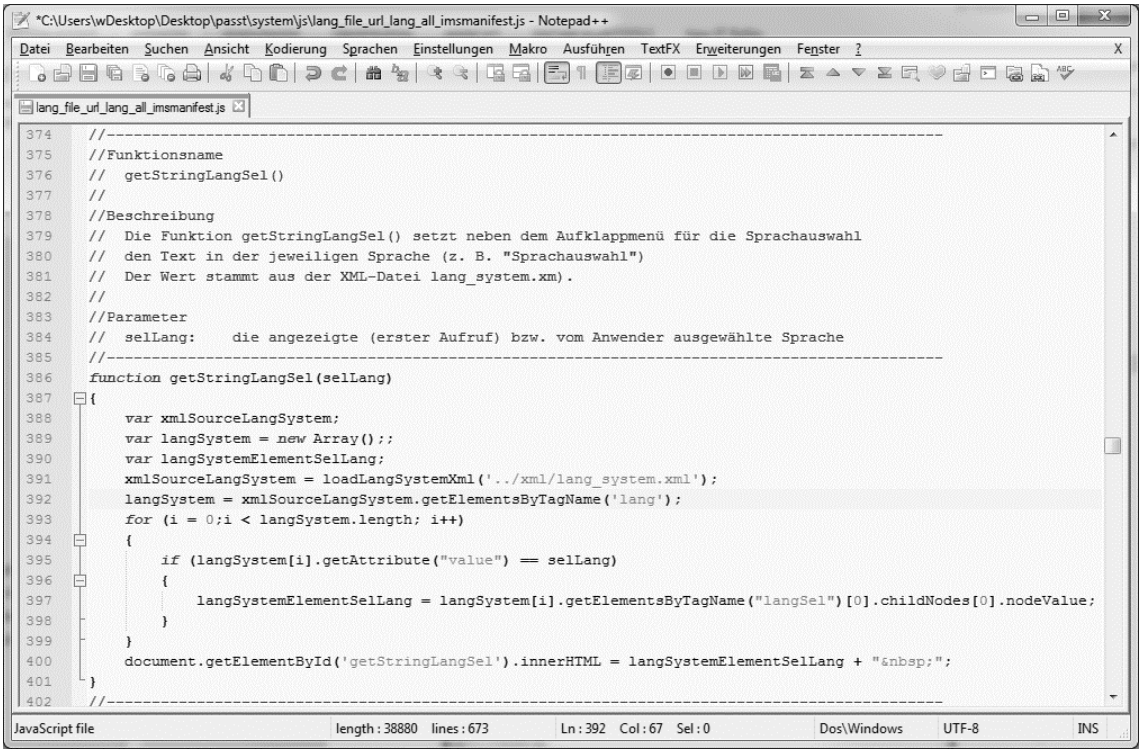
Mit Hilfe der Funktion „buildLangSel()“ (Aufruf der Funktion Listing 59 Zeile 47, Funktion selbst Listing 63) werden die Inhaltssprachen (Inhalt der Array-Variable „allLang“, Listing 63 Zeile 313) für das Aufklappmenü gesetzt (linker oberer Frame). Sie können bei einem Wechsel der Sprache durch den Anwender ausgewählt werden. Zusätzlich stellt die Funktion „buildLangSel()“ sicher, dass die erste Inhaltsprache (erste „if“-Anweisung Listing 63 Zeile 315) aus den Metadaten des ersten „organization“-Elementes (der XML-Datei „imsmanifest.xml“) im Aufklappmenü („dropDownMenue“, Listing 59 Zeile 42) ausgewählt angezeigt wird (Listing 63 Zeile 319).



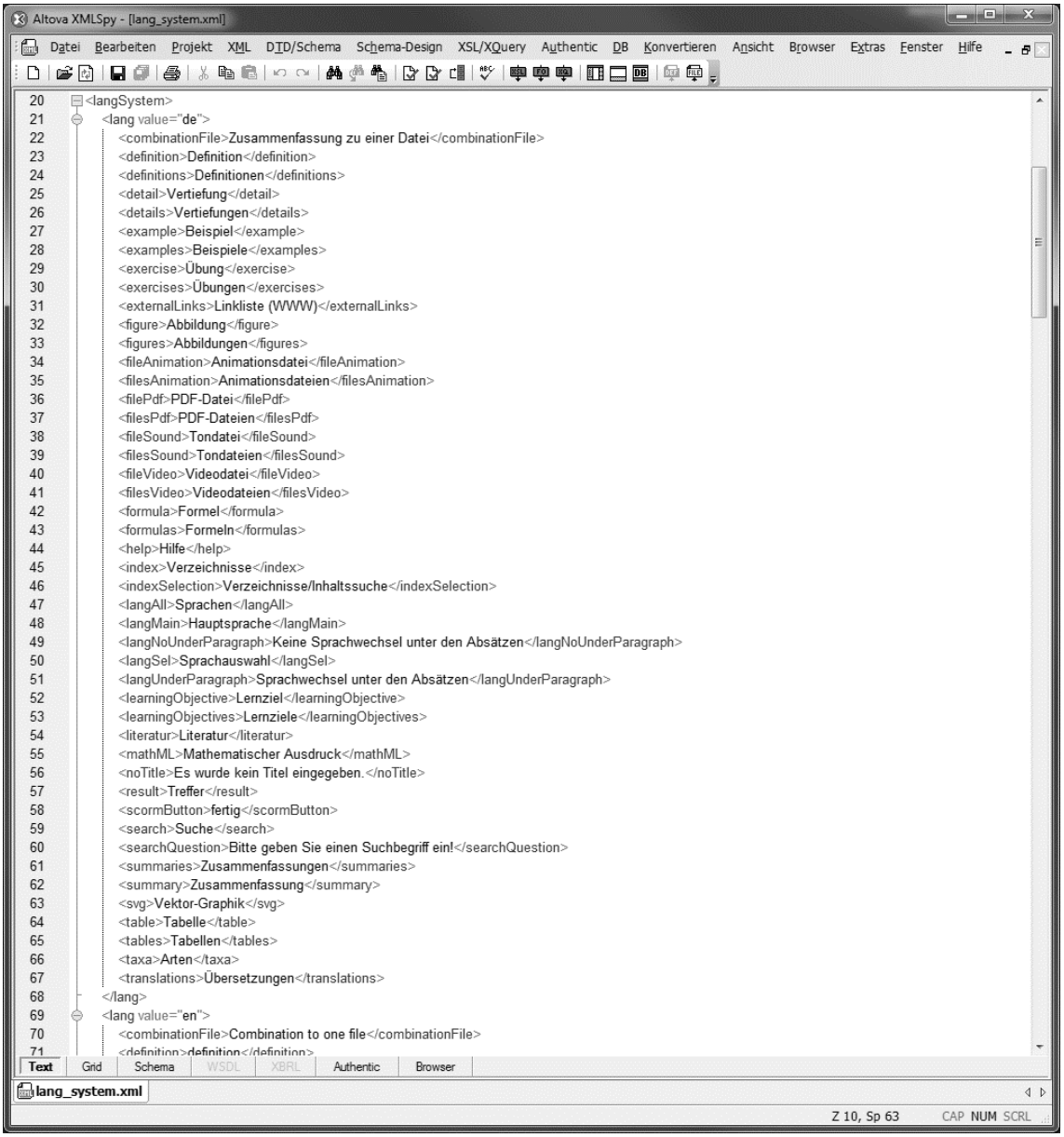
Listing 63: Funktion „buildLangSel()“ (Setzen der Sprachen im Aufklappmenü des linken oberen Frames (Start-Frameset))

Neben dem Aufklappmenü für die Sprachauswahl wird zusätzlich der Text "Sprachauswahl" in der jeweiligen Sprache angezeigt (siehe Abbildung 80 links oben). Damit der Text in der richtigen Sprache erscheint, wird das ausgewählte Sprachkürzel als Parameter (Variable „selLang“) an die Funktion „getStringLangSel()“ übergeben (Listing 64, Zeile 386). Der Parameterwert wird in der HTML-Datei „lang_sel.htm“ (Listing 59) ermittelt und von dort beim Funktionsaufruf der „getStringLangSel()“ entweder in Zeile 42 oder in Zeile 51 an diese Funktion transferiert. Dabei übergibt die Anweisung in Zeile 51 (Listing 59) automatisch nach dem Laden der Datei den ersten in der Funktion „buildLangSel()“ als „ausgewählt“ („selectedIndex“, Listing 63 Zeile 319) gesetzten Sprachkürzelwert. Die Anweisung in Zeile 42 (Listing 59) greift erst bei einem „onchange“-Ereignis (Ausdruck Ereignis, siehe auch Kapitel 3.6 JavaScript) des Aufklappmenüs (Anwender wählt eine andere Sprache über das Aufklappmenü aus) und übergibt dann den vom Anwender ausgewählten Sprachkürzelwert („document.formSearch.langSearch.options-[document.formSearch.langSearch.selectedIndex].text“). Die Funktion „getStringLangSel()“ öffnet mit Hilfe der Funktion „loadLangSystemXml()“ (Quelltext siehe JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“) zuerst die XML-Datei „lang_system.xml“ (Listing 64 Zeile 391, siehe auch Ladereihenfolge in Abbildung 79 Zeile 10). Die XML-Datei „lang_system.xml“ beinhaltet die sprachspezifischen

Systemausdrücke in 6 Sprachen (Listing 65). Die Anweisung in Listing 64 Zeile 392 legt sie in der Array-Variable „langSystem“ ab. Zum Abschluss wird der Textinhalt des Elementes „langSel“ in der als Parameter übergebenen Sprache ausgewählt (Listing 64 Zeile 397) und in das „span“-Element mit dem „id“-Wert „getStringLangSel“ (Listing 59 Zeile 41) in der HTML-Datei „lang_sel.htm“ geschrieben (Zeile 400). Als Ergebnis seht in der jeweiligen Sprache links von dem Sprach-Auswahlmenü der sprachspezifische Text (Abbildung 80 links oben, Text „Sprachauswahl“).



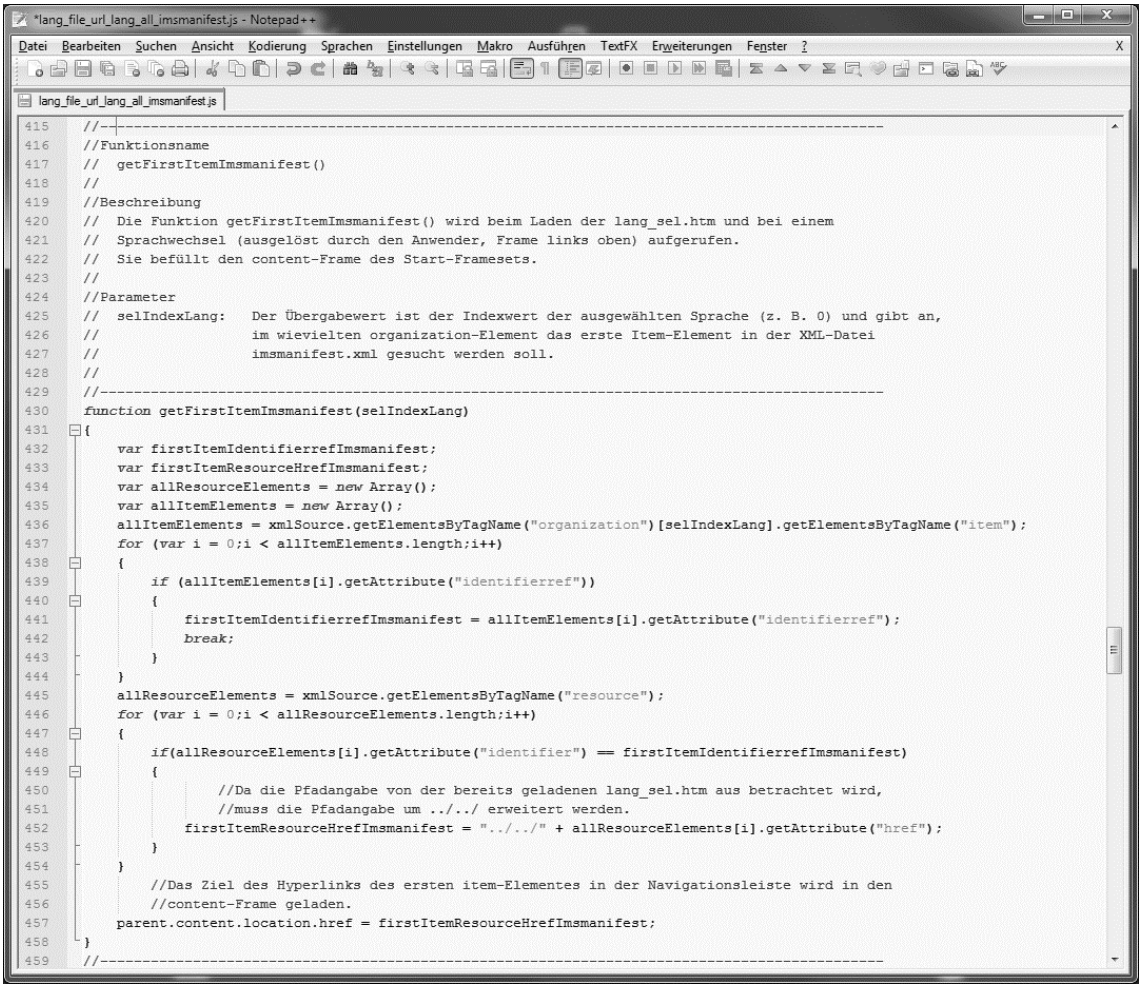
Listing 64: Funktion „getStringLangSel()“ (Holen des sprachspezifischen Systemausdrucks aus der XML-Datei „lang_system.xml“)



Listing 65: Ausschnitt aus der XML-Datei „lang_system.xml“ mit den sprachspezifischen Systemausdrücken in 6 Sprachen

Der vorletzte Funktionsaufruf in der HTML-Datei „lang_sel.htm“ (Listing 59 Zeile 54) gilt der Funktion „getFirstItemImsmanifest()“ (in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“). Sie befüllt den „content“-Frame des Start-Framesets (Listing 66). Als Parameterwert erhält die Funktion den Indexwert der ausgewählten Sprache (z. B. „0“). Er gibt an, im wievielten „organization“-Element (z. B. entspricht der Indexwert „0“ dem ersten „organization“-Element) das erste „item“-Element in der XML-Datei „imsmanifest.xml“ gesucht werden soll. Die Reihenfolge der Sprachen im Aufklappmenü im links oben liegenden Frame spiegelt dabei die Reihenfolge der „organization“-Elemente in der XML-Datei „imsmanifest.xml“ wieder. Aus diesem Grund kann aus dem Index der Sprache auf das „organization“-Element rückgeschlossen werden. In der Funktion wird zunächst das erste „item“-Element mit

einem „identifierref“-Attributwert, das sich in dem über den Parameterwert ausgewählten „organization“-Element befindet, bestimmt (Listing 66 Zeile 436). In diesem Zusammenhang wird auf die globale Variable „xmlSource“ zurückgegriffen, die die XML-Datei „imsmanifest.xml“ beinhaltet (Listing 66 Zeile 436, siehe Befüllung der Variablen in Listing 62 Zeile 104 und 117). Nachdem dieses „item“-Element bestimmt wurde, wird dessen „href“-Attributwert (Zieladresse) im zugeordneten „resource“-Element ausgewählt (Listing 66 Zeile 452) und diese Zieladresse in den „content“-Frame geladen (Listing 66 Zeile 457). Welche Anweisungen beim Laden einer solchen WLMML-Datei ausgeführt werden (siehe Ladereihenfolge in Abbildung 79 Zeile 11), wird im nächsten Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei geschildert.




```
415  //--------------------------------------------------------------------------------------------
416  //Funktionsname
417  //  getFirstItemImsmanifest()
418  //
419  //Beschreibung
420  //  Die Funktion getFirstItemImsmanifest() wird beim Laden der lang_sel.htm und bei einem
421  //  Sprachwechsel (ausgelöst durch den Anwender, Frame links oben) aufgerufen.
422  //  Sie befüllt den content-Frame des Start-Framesets.
423  //
424  //Parameter
425  //  selIndexLang:   Der Übergabewert ist der Indexwert der ausgewählten Sprache (z. B. 0) und gibt an,
426  //                  im wievielten organization-Element das erste Item-Element in der XML-Datei
427  //                  imsmanifest.xml gesucht werden soll.
428  //
429  //--------------------------------------------------------------------------------------------
430  function getFirstItemImsmanifest(selIndexLang)
431  {
432      var firstItemIdentifierrefImsmanifest;
433      var firstItemResourceHrefImsmanifest;
434      var allResourceElements = new Array();
435      var allItemElements = new Array();
436      allItemElements = xmlSource.getElementsByTagName("organization")[selIndexLang].getElementsByTagName("item");
437      for (var i = 0;i < allItemElements.length;i++)
438      {
439          if (allItemElements[i].getAttribute("identifierref"))
440          {
441              firstItemIdentifierrefImsmanifest = allItemElements[i].getAttribute("identifierref");
442              break;
443          }
444      }
445      allResourceElements = xmlSource.getElementsByTagName("resource");
446      for (var i = 0;i < allResourceElements.length;i++)
447      {
448          if(allResourceElements[i].getAttribute("identifier") == firstItemIdentifierrefImsmanifest)
449          {
450              //Da die Pfadangabe von der bereits geladenen lang_sel.htm aus betrachtet wird,
451              //muss die Pfadangabe um ../../ erweitert werden.
452              firstItemResourceHrefImsmanifest = "../../" + allResourceElements[i].getAttribute("href");
453          }
454      }
455          //Das Ziel des Hyperlinks des ersten item-Elementes in der Navigationsleiste wird in den
456          //content-Frame geladen.
457      parent.content.location.href = firstItemResourceHrefImsmanifest;
458  }
459  //--------------------------------------------------------------------------------------------
```

Listing 66: Funktion „getFirstItemImsmanifest()“ (Befüllen des „content“-Frames des Start-Framesets)

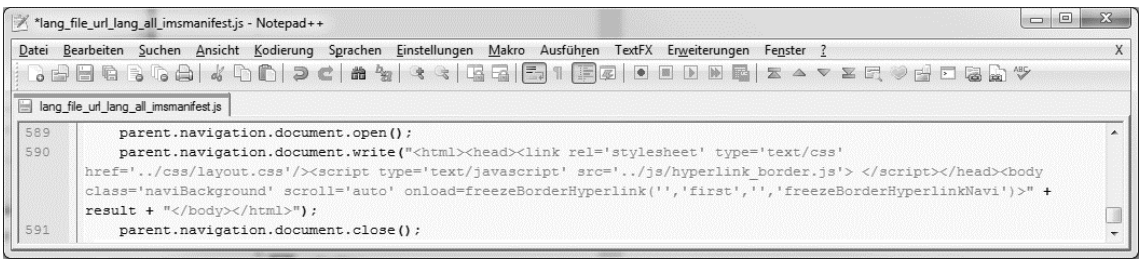
Der letzte Funktionsaufruf in der HTML-Datei „lang_sel.htm“ (Listing 59 Zeile 57) gilt der Funktion „loadNavigation()“ (in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“). Die Funktion befüllt den linken „navigation“-Frame mit der Navigationsleiste (Listing 67). Als Parameterwert erhält die Funktion

den Indexwert der ausgewählten Sprache. Über diesen Indexwert wird (wie auch bei der Funktion „getFirstItemImsmanifest ()“) festgestellt, das wievielte „organization“-Element in der XML-Datei „imsmanifest.xml“ angesprochen werden soll. Der Parameterwert „0“ steht dabei z. B. für das erste „organization“-Element. Auch bei dieser Funktion wird auf die globale Variable „xmlSource“ zurückgegriffen (siehe Erläuterungen zu Funktion („getFirstItemImsmanifest ()“) und in der Variablen „xmlNode“ das über den Parameterwert ausgewählte „organization“-Element (mit Kind-Elementen) gespeichert (Listing 67 Zeile 485). Aus dem Inhalt dieser Variablen werden die Textinhalte der „item“-Elemente ausgelesen und diejenigen „href“-Attributwerte aus den „resource“-Elementen geholt, bei denen der „identifierref“-Attributwert des „item“-Elementes mit dem „identifier“-Attributwertes des „resource“-Elementes übereinstimmt. Aus beidem (Text und Zieladresse) wird anschließend ein Hyperlink zusammengesetzt (Listing 67 Zeile 519). „item“-Elemente, die auf kein „resource“-Element referenzieren, erscheinen nur als Text (Listing 67 Zeile 505). Die Liste dieser „item“-Elemente wird anschließend in den „navigation“-Frame in Form von HTML-Anweisungen geschrieben (Listing 68 Zeile 590). Dabei wird auch die JavaScript-Datei „hyperlink_border.js“ in das HTML-Ergebnis integriert und geladen (siehe Ladereihenfolge in Abbildung 79 Zeile 12). JavaScript-Anweisungen in dieser Datei (Funktion „freezeBorderHyperlink()“, Listing 68 Zeile 590) versehen nach dem ersten Laden automatisch das erste „item“-Element mit einer Rahmenlinie (Beispiel: Abbildung 80, links oben dunkelrot eingerahmter Hyperlink mit dem Text „Einfaches Beispiel“). Wählt der Anwender in der Navigationsleiste einen anderen Hyperlink aus, wird dieser hervorgehoben. Als Überschrift erscheint im „navigation“-Frame automatisch der Textinhalt des ersten „title“-Elementes im ausgewählten „organization“-Element (Listing 67 Zeile 491, Beispiel: Abbildung 80, links oben Text „de: Einfaches Beispiel“, Text in der XML-Datei „imsmanifest.xml“ siehe Listing 61 Zeile 12).

Die Funktion „loadNavigation()“ (inkl. der Funktion „holeKinder()“, Listing 67 Zeile 521) ist in Anlehnung an JavaScript-Anweisungen von MINTERT & KÜHNE (2000, S. 221) entstanden.

```
*lang_file_url_lang_all_imsmanifest.js - Notepad++
Datei Bearbeiten Suchen Ansicht Kodierung Sprachen Einstellungen Makro Ausführen TextFX Erweiterungen Fenster ?
lang_file_url_lang_all_imsmanifest.js
460  //-----------------------------------------------------------------------------------------
461  //Funktionsname
462  //  loadNavigation()
463  //
464  //Beschreibung
465  //  Die Funktion loadNavigation() wird beim Laden der lang_sel.htm und bei einem
466  //  Sprachwechsel (ausgelöst durch den Anwender, Frame links oben) aufgerufen.
467  //  Sie befüllt den navigation-Frame vor allem mit Hilfe der item-Elemente
468  //  des jeweiligen organization-Elementes aus der imsmanifest.xml (Auswahl über den Parameterwert).
469  //
470  //Parameter
471  //  selIndexLang:   Der Übergabewert ist der Indexwert der ausgewählten Sprache (z. B. 0) und gibt an,
472  //                  im wievielten organization-Element das erste Item-Element in der XML-Datei
473  //                  imsmanifest.xml gesucht werden soll.
474  //-----------------------------------------------------------------------------------------
475  function loadNavigation(selIndexLang)
476  {
477      var result;
478      var xmlNode;
479      var allResourceElements = new Array();
480      var itemIdentifierrefImsmanifest;
481      var itemResourceHrefImsmanifest;
482          //Die Variable charBeforeString gibt an, mit wievielen Zeichen (wie weit)
483          //der Navigationspunkt eingerückt werden soll.
484      var charBeforeString = "";
485      xmlNode = xmlSource.getElementsByTagName("organization")[selIndexLang];
486      allResourceElements = xmlSource.getElementsByTagName("resource");
487          //Ab hier inklusive der Funktion holeKinder() wurde in Anhalt an den Quelltext im Buch Addison-Wesley
488          //"Workshop JavaScript" von Stefan Mintert, Christine Kühnel, 2000 Addison-Wesley Verlag, S.221 gearbeitet.
489          //Über dem Inhaltsverzeichnis soll als Überschrift der Textinhalt des ersten title-Elementes
490          //des ausgewählten organization-Elementes angezeigt werden.
491      result = "<p><b>" + xmlNode.getElementsByTagName("title")[0].childNodes[0].nodeValue + "</b></p>";
492      for (var i = 0; i < xmlNode.childNodes.length; i++)
493      {
494          var kind = xmlNode.childNodes[i];
495          if (kind.nodeType == 1)
496          {
497              if(kind.tagName == 'item')
498              {
499                  itemIdentifierrefImsmanifest = kind.getAttribute("identifierref");
500                      //Wenn kein identifierref-Attribut für ein item vorhanden ist,
501                      //handelt es sich um einen um ein item, das auf keine Datei verweist.
502                      //Es wird nur der Textinhalt dieses item-Elementes ausgegeben.
503                  if (!itemIdentifierrefImsmanifest)
504                  {
505                      result += "<span class='naviItemNotHyperlink'>" + kind.getElementsByTagName("title")[0].
childNodes[0].nodeValue + "</span><br/>";
506                  }
507                  else
508                  {
509                      for (var z = 0;z < allResourceElements.length;z++)
510                      {
511                          if(allResourceElements[z].getAttribute("identifier") == itemIdentifierrefImsmanifest)
512                          {
513                              itemResourceHrefImsmanifest = "../../" + allResourceElements[z].getAttribute("href");
514                                  //Wenn der richtige Wert gefunden wurde,
515                                  //soll die Suche abgebrochen werden (Performance-Gewinn).
516                              break;
517                          }
518                      }
519                      result += "<a href='" + itemResourceHrefImsmanifest + "' class='navi'
onclick=freezeBorderHyperlink(this,'notFirstHyperlink','navi','freezeBorderHyperlinkNavi') target='content'>" + kind
.getElementsByTagName("title")[0].childNodes[0].nodeValue + "</a><br/>";
520                  }
521                  holeKinder(kind,charBeforeString);
```

Listing 67: Ausschnitt aus der Funktion „loadNavigation()“, Befüllen des „navigation“-Frame des Start-Framesets



```
*lang_file_url_lang_all_imsmanifest.js - Notepad++
Datei Bearbeiten Suchen Ansicht Kodierung Sprachen Einstellungen Makro Ausführen TextFX Erweiterungen Fenster ?
lang_file_url_lang_all_imsmanifest.js
589      parent.navigation.document.open();
590      parent.navigation.document.write("<html><head><link rel='stylesheet' type='text/css'
href='../css/layout.css'/><script type='text/javascript' src='../js/hyperlink_border.js'> </script></head><body
class='naviBackground' scroll='auto' onload=freezeBorderHyperlink('','first','','freezeBorderHyperlinkNavi')>" +
result + "</body></html>");
591      parent.navigation.document.close();
```

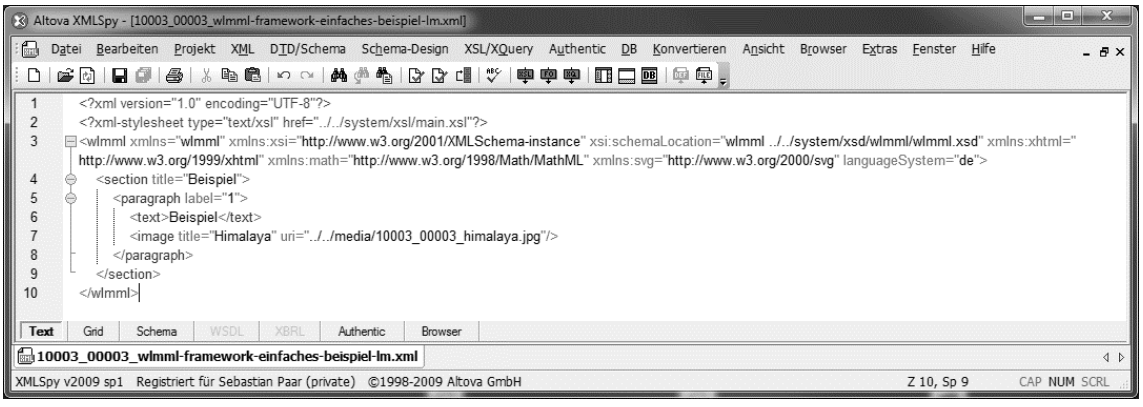
Listing 68: Ausschnitt aus der Funktion „loadNavigation()“, „Schreiben“ des Inhalts („write“-Anweisung) in den „navigation“-Frame des Start-Framesets

Sind der Navigation-Frame (Name „navigation“), der Sprachauswahl-Frame (Name „langSel“) und der Inhalt-Frame (Name „content“) mit den vorgesehenen Inhalten versehen, kann der Anwender mit den Lerninhalten arbeiten. Wenn er über das Aufklappmenü mit den verschiedenen Sprachen (Frame links oben im Start-Frameset) eine andere Sprache auswählt, stößt das „onchange“-Ereignis des Aufklappmenüs („onchange“-Ereignis des „select“-Elementes, Listing 59 Zeile 41) die erneute Abarbeitung der Funktionen „getFirstItemImsmanifest()“, „loadNavigation()“ und „getStringLangSel()“ an. Damit werden sprachspezifisch eine neue Navigationsleiste aufgebaut, die sprachspezifischen Systemausdrücke verwendet und ein neuer Inhalt im „content“-Frame angezeigt.

Nach diesen Verarbeitungsprozessen ist das Start-HTML-Frameset ohne den Inhalt des „content“-Frame fertig zusammengesetzt. Welche Verarbeitungsprozesse beim Laden einer WLMML-Datei in den „content“-Frame ablaufen, zeigt das nächste Kapitel.

4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei

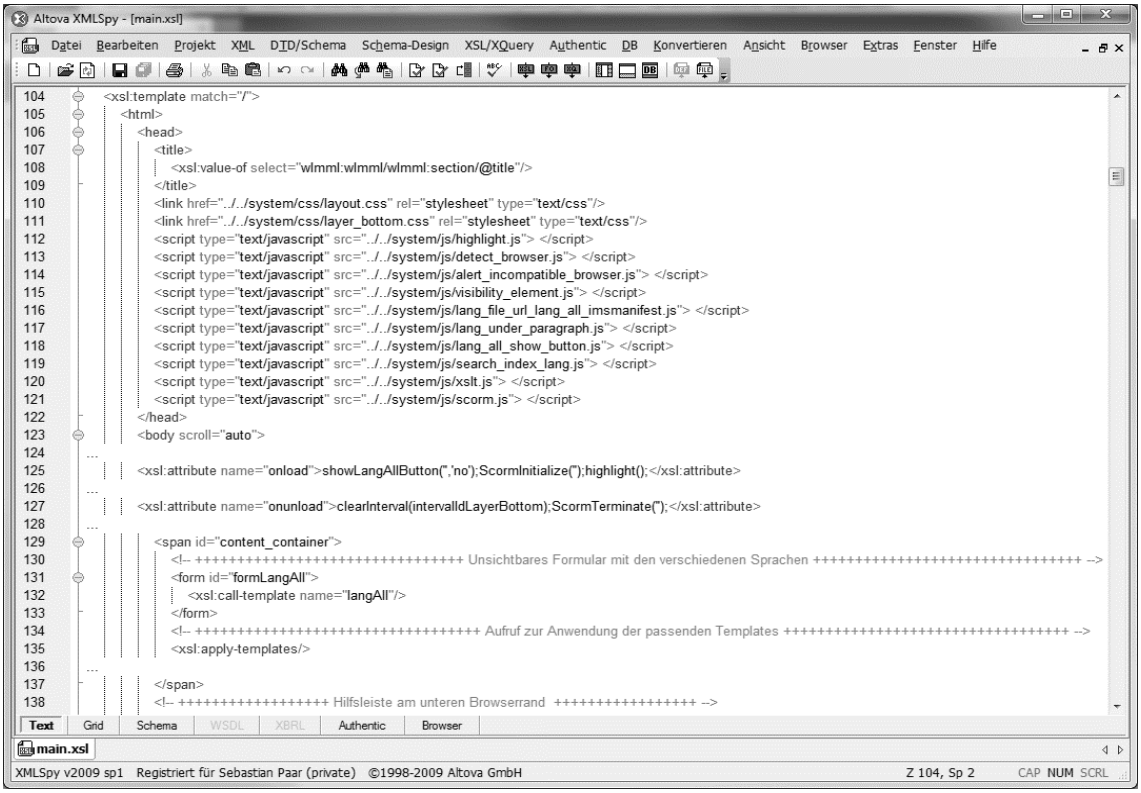
Im Folgenden wird der Ablauf der Verarbeitung beim Laden einer WLMML-Datei in den „content“-Frame oder beim Laden einer WLMML-Datei über eine Navigationsleiste aus einer SCORM-Umgebung heraus beschrieben (siehe Hinweis in vorherigen Kapitel). Als Beispiel dient die WLMML-Datei „10003_00003_wlmml-framework-einfaches-beispiel-lm.xml“ (Listing 69, Ergebnis im Browser siehe Abbildung 80 rechte Seite mit weißem Hintergrund, Ladereihenfolge in Abbildung 79 Zeile 11). Sie referenziert in der zweiten Zeile (Listing 69) die XSLT-Datei „main.xsl“ ein (Ladereihenfolge in Abbildung 79 Zeile 13).



Listing 69: Quelltext der WLMML-Datei „10003_00003_wlmml-framework-einfaches-beispiel-lm.xml“

Diese zentrale XSLT-Datei ist in jeder WLMML-Datei mit E-Learning-Inhalten eingebunden und transformiert WLMML-Inhalte nach HTML. Sie erstellt für die WLMML-

Inhalte ein HTML-Grundgerüst (Listing 70, siehe auch Listing 81) und arbeitet über „template“-Anweisungen die WLMML-Elemente der betroffenen WLMML-Datei ab (siehe Kapitel 3.3.8.1 XSLT). Das Ergebnis, bestehend aus HTML-, CSS- und JavaScript-Anweisungen, wird in das HTML-Grundgerüst eingegliedert. Es entsteht als „Output“ ein HTML-Dokument, das im Browser wie eine Webseite abgearbeitet und angezeigt wird. Ein Ausschnitt des Quelltextes dieser XSLT-Datei zeigt Listing 70. Das „template“ in Zeile 104 greift über den „match“-Attributwert „/“ auf den Wurzelknoten des WLMML-Dokumentes zu und ersetzt ihn durch ein HTML-Grundgerüst (Listing 70 Zeile 105 bis 138, siehe auch Listing 81 Zeile 139 bis 181). In Zeile 108 (Listing 70) wird als Inhalt des „title“-Elementes (HTML-Element) der „title“-Attributwert des ersten „section“-Elementes gesetzt. Anschließend werden die beiden CSS-Dateien „layout.css“ und „layer_bottom.css“ in der Ergebnis-HTML-Datei geladen (falls noch nicht bereits geschehen, siehe Kapitel 4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-Datei, siehe auch Ladereihenfolge in Abbildung 79 Zeile 14). Die Anweisungen in der CSS-Datei „layer_bottom.css“ formatieren die Hilfsleiste am unteren Browserrand (Zeile 138 ff, siehe Listing 81) und fixieren sie dort z. B. während eines „Scroll“-Vorgangs. Die normalerweise unsichtbare Hilfsleiste wird erst über JavaScript-Anweisungen in der JavaScript-Datei „layer_bottom.js“ sichtbar, wenn über einen Mausklick die Lasche am unteren Browserrand ausgewählt wird (Lasche siehe Abbildung 80 rechts unten, siehe auch Kapitel 4.3.2.4.2 Funktionalitäten in der Hilfsleiste am unteren Browserrand).



Listing 70: Ausschnitt aus der zentralen XSLT-Datei „main.xsl“ (erster Teil der Erzeugung eines HTML-Grundgerüstes, zweiter Teil siehe Listing 81)

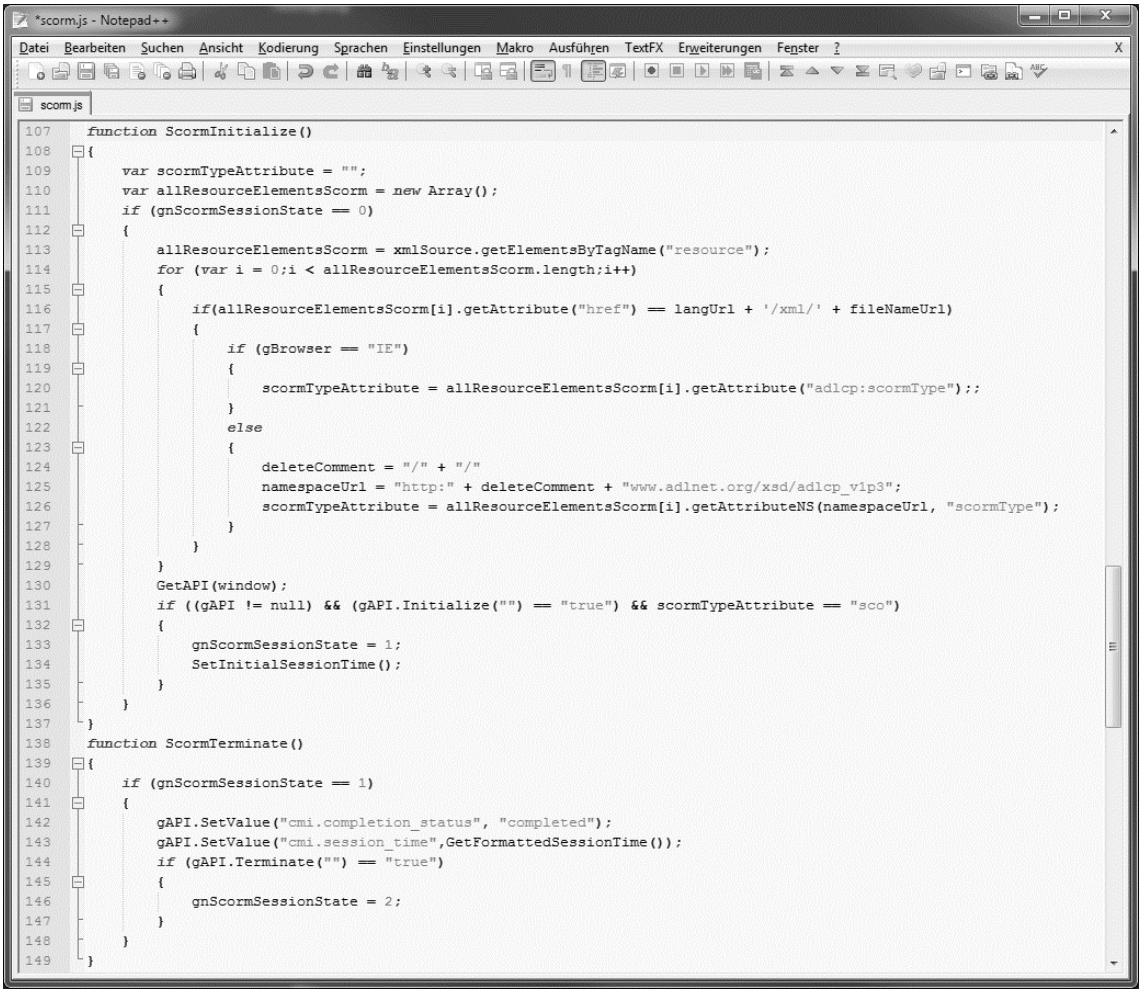
Die folgenden in Listing 70 Zeile 112 bis 121 referenzierten JavaScript-Dateien stellen weitere Funktionalitäten sicher. Die wichtigsten Anweisungen in den Dateien „detect_browser.js“, „alert_incompatible_browser.js“ und „lang_file_url_lang_all_imsmanifest.js“ wurden bereits in dem vorangegangenen Kapitel beschrieben. Die restlichen Dateien werden in der folgenden Aufzählung kurz erläutert.

- „highlight.js“ (Listing 70 Zeile 112, Ladereihenfolge in Abbildung 79 Zeile 15)
  (Hervorheben gesuchten Textes in einer WLMML-Inhaltsdatei, Auswahl eines Absatzes für die Anzeige in einer anderen Sprache, siehe Kapitel 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten, 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen, Teile der Programmierung in Anhalt an das JavaScript-Bookmarklet "Bookmarklet zum Hervorheben von Text" nach WIKIPEDIA-BOOKMARKLETS 2007)
- „visibility_element.js“ (Listing 70 Zeile 115, Ladereihenfolge in Abbildung 79 Zeile 19)
  (Einblenden eines ausgeblendeten Elementes bzw. Ausblenden eines eingeblendeten Elementes, z. B. bei WLMML-Elementen „detail“ und „solution“)
- „lang_under_paragraph.js“ (Listing 70 Zeile 117, Ladereihenfolge in Abbildung 79 Zeile 20)
  (Erstellen sowie Ein- und Ausblenden der verschiedenen Inhaltssprachen unter den Absätzen einer WLMML-Inhaltsdatei, Anzeige der Inhaltssprachen als Hyperlinks, siehe Kapitel 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen)
- „lang_all_show_button.js“ (Listing 70 Zeile 118, Ladereihenfolge in Abbildung 79 Zeile 18)
  (Steuerung, ob der sprachspezifische Text „Sprachwechsel unter den Absätzen“ in der verborgenen Hilfsleiste am unteren Browserrand angezeigt wird, der anklickbare Text erlaubt die verschiedenen Sprachen unter den Absätzen eines WLMML-Inhaltsdatei ein- bzw. auszublenden)
- „search_index_lang.js“ (Listing 70 Zeile 119, Ladereihenfolge in Abbildung 79 Zeile 17)
  (Anzeige einer Literaturstelle in einem eigenen Fenster bei Auswahl eines Literaturverweises im Fließtext, Anzeige der verschiedenen Verzeichnisse, Durchführen der Volltext-Suche in WLMML-Inhaltsdateien, setzen der Sprachen für das Aufklappmenü bei der Volltextsuche, siehe Kapitel 4.3.2.4.3 Automatische Erstellung von Verzeichnissen, 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten)
- „xslt.js“ (Listing 70 Zeile 120, Ladereihenfolge in Abbildung 79 Zeile 16)

(Transformation von WLMML-Dateien mit XSLT- und JavaScript-Anweisungen)

- „scorm.js“ (Listing 70 Zeile 121, Ladereihenfolge in Abbildung 79 Zeile 22) (Sicher stellen von „SCORM 2004“-Funktionalitäten, Inhalt nach OSTYN (2007, S. 35 ff) und OSTYN (2006), siehe auch Kapitel 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM))

Nach der Referenzierung dieser JavaScript-Dateien im „head“-Element werden für das „body“-Element zwei Attribute „onload“ (Listing 70 Zeile 125) und „onunload“ (Listing 70 Zeile 127) gesetzt. Das Attribut „onload“ beinhaltet nach dem Laden des HTML-Dokumentes den Aufruf der JavaScript-Funktionen „showLangAllButton()“ (aus der JavaScript-Datei „lang_all_show_button.js“), „ScormInitialize()“ (aus der JavaScript-Datei „scorm.js“) und „highlight()“ (aus der JavaScript-Datei „highlight.js“). Die Funktion „showLangAllButton()“ mit dem zweiten Parameter-Wert „no“ stellt sicher, dass beim Aufruf des HTML-Ergebnis-Dokumentes der sprachspezifische Text „Sprachwechsel unter den Absätzen“ in der verborgenen Hilfsleiste am unteren Browserrand nicht angezeigt wird. Ist eine „SCORM2004-API“-Instanz vorhanden (Prüfung über Anweisungen in der JavaScript-Datei „scorm.js“), startet die Funktion „ScormInitialize()“ die Kommunikations-Session zwischen Lernobjekt (SCO) und Lernplattform (siehe auch Kapitel 3.8 SCORM). Diese Funktion wird bei jedem Aufruf einer WLMML-Inhaltsdatei aufgerufen. Ihre Anweisungen prüfen zunächst, ob das „resource“-Element der aufgerufenen WLMML-Datei in der Manifest-Datei „imsmanifest.xml“ als „adlcp:scormType“-Attributwert „sco“ aufweist (Listing 71 Zeile 113 bis 129, 131, siehe auch Anhang I). Anschließend stoßen weitere Anweisungen die Suche nach einer „SCORM2004-API“-Instanz an (Listing 71 Zeile 130) und kommunizieren über die gefundene „SCORM2004-API“-Instanz mit dem LMS. Dabei halten sie die Startzeit der Kommunikations-Session fest (Listing 71 Zeile 134).



Listing 71: Ausschnitt aus dem Quelltext der JavaScript-Datei „scorm.js“, Funktionen „ScormInitialize()“ und „ScormTerminate()“, angepasst nach OSTYN (2007, S. 35 ff) und OSTYN (2006), siehe auch JESUKIEWICZ (2009 b, RTE-3-30).

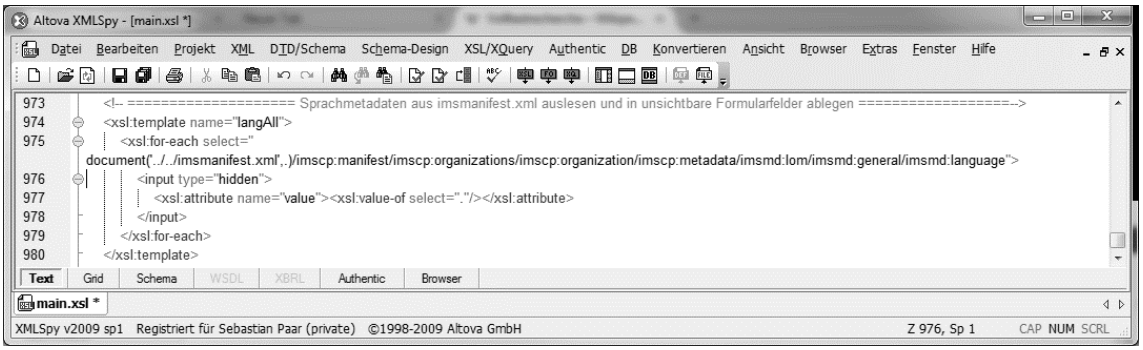
Wenn die aufgerufene Datei verlassen wird, stellen Anweisungen in der Funktion „ScormTerminate()“ (ebenfalls aus der JavaScript-Datei „scorm.js“ des WLMML-Frameworks, siehe auch Kapitel 3.8 SCORM, Terminate(“”)) folgende Punkte als Rückmeldungen an das LMS sicher (siehe auch Kapitel 4.3.2.1.1 Beispiel-LMS von ADL):

- Setzen des Status ("cmi.completion_status") der aufgerufenen und als „SCO“ gekennzeichneten WLMML-Inhaltsdatei auf „completed“ (Lerninhalt wurde „fertig“ betrachtet) (Listing 71 Zeile 142)
- Festhalten der Zeitdauer, wie lange das SCO geöffnet war (Listing 71 Zeile 143)
- Beenden der Kommunikations-Session zwischen LMS und SCO (Listing 71 Zeile 144)

Die letzte Funktion „highlight()“ im „onload“-Attributwert des „body“-Elementes (Listing 70 Zeile 125) überprüft abschließend, ob ein Suchbegriff über die URL übergeben wurde. Nur wenn dies der Fall ist, wird aus der URL der Suchbegriff ermittelt und zur Markierung an die Funktion „highlightWord()“ in der JavaScript-Datei „highlight.js“ übergeben.

Das Attribut „onunload“ beinhaltet nach dem Entladen (Verlassen) des HTML-Dokumentes den Aufruf der beiden JavaScript-Funktionen „clearInterval()“ (aus der JavaScript-Datei „layer_bottom.js“) und „ScormTerminate()“ (aus der JavaScript-Datei „scorm.js“). Die erste Funktion „clearInterval()“ beendet den Timer, der permanent überprüft, ob der Anwender im Fließtext einer WLMML-Datei ein Wort markiert und die Return-Taste gedrückt hat. In diesem Fall würde der markierte Text für die Suche in einem Online-Verzeichnis wie z. B. Wikipedia abgeschickt. Gestartet wurde der Timer über Anweisungen in der JavaScript-Datei „layer_bottom.js“. Die zweite Funktion „ScormTerminate()“ ist bereits weiter oben beschrieben worden.

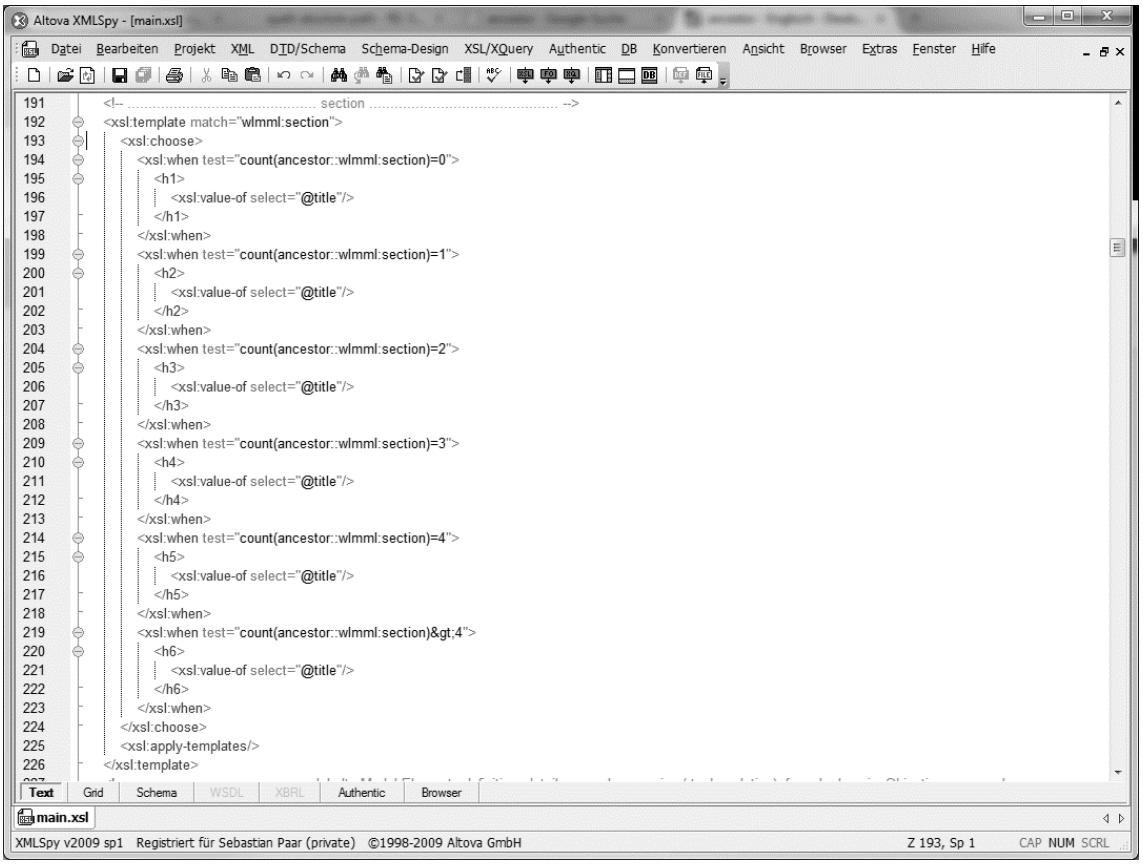
Nach der Belegung von Attributen des „body“-Elementes wird ein „span“-Element mit dem „id“-Attributwert "content_container" geöffnet (Listing 70 Zeile 129). Dieses Element dient als Behälter für alle weiteren Inhalte des Ergebnis-HTML-Dokumentes ohne die Fußzeile (siehe Listing 81). Es wird in der Zeile 137 (Listing 70) geschlossen. Im Anschluss daran lesen XSLT-Anweisungen die Inhaltssprachen aus der Manifest-Datei „imsmanifest.xml“ aus und legen sie in unsichtbare Formularfelder ab (Listing 70 Zeile 131 bis 133). Dazu wird ein „template“ mit dem Namen „langAll“ aufgerufen (Listing 70 Zeile 131). Es öffnet über die XSLT-Funktion „document()“ die Manifest-Datei, greift auf die Sprach-Metadaten der einzelnen „organization“-Elementen zu und legt die Sprachangaben in unsichtbare Formularfelder ab (Listing 72 Zeile 974 ff). Auf sie wird zurückgegriffen, wenn für einen Sprachwechsel unter den Absätzen von WLMML-Inhaltsdateien die Sprachkürzel angezeigt werden sollen.



Listing 72: Template „langAll“ aus der zentralen XSLT-Datei „main.xsl“ (Auslesen der Sprach-Metadaten aus der Manifest-Datei und Ablegen in unsichtbare Formularfelder)

In das bereits erwähnte HTML-Grundgerüst werden neben den einzeln referenzierten JavaScript- und CSS-Dateien die HTML-Ergebnisse der „template“-Anweisungen für die im jeweiligen WLMML-Dokument vorgefundenen WLMML-Elemente eingebunden. Dazu wird die XSLT-Anweisung „apply-templates“ eingesetzt (Listing 70 Zeile 135, siehe auch Kapitel 3.3.8.1 XSLT). Sie ruft die einzelnen „templates“ für die verschiedenen Elemente in der WLMML-Datei auf und gibt deren Verarbeitungsergebnis zurück.

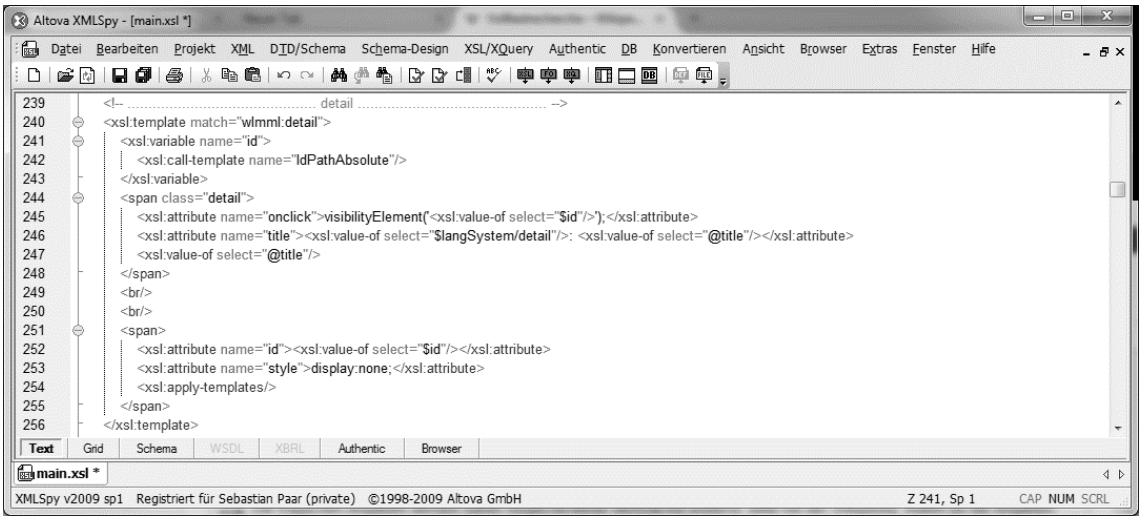
Beinhaltet das WLMML-Dokument z. B. ein „section“-Element, transformiert es das zuständige Template je nach Verschachtelungstiefe in eine HTML-Überschrift („h1“ bis „h6“). Je tiefer ein „section“-Element verschachtelt ist, um so kleiner wird die Überschrift (kleinste Überschrift HTML-Element „h6“) (Listing 73). Dieser Verarbeitungsvorgang in der zentralen XSLT-Datei „main.xsl“ greift für alle WLMML-Elemente.



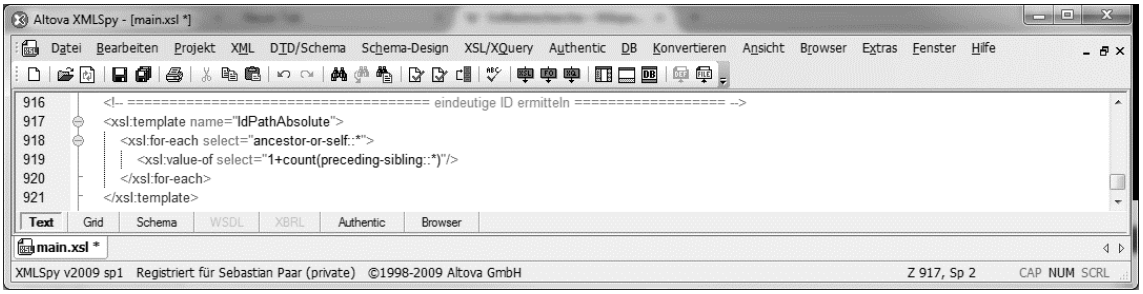
Listing 73: Template für das Element „section“ aus der zentralen XSLT-Datei „main.xsl“ (Transformation in HTML-Elemente „h1“ bis „h6“

Wird z. B. ein „detail“- oder ein „solution“-Element gefunden, ruft das zuständige „detail“-Template (Listing 74 Zeile 240 bis 256) zuerst ein anderes Template mit dem Namen „IdPathAbsolute“ auf (Listing 75). Dieses Template wurde auf der Grundlage

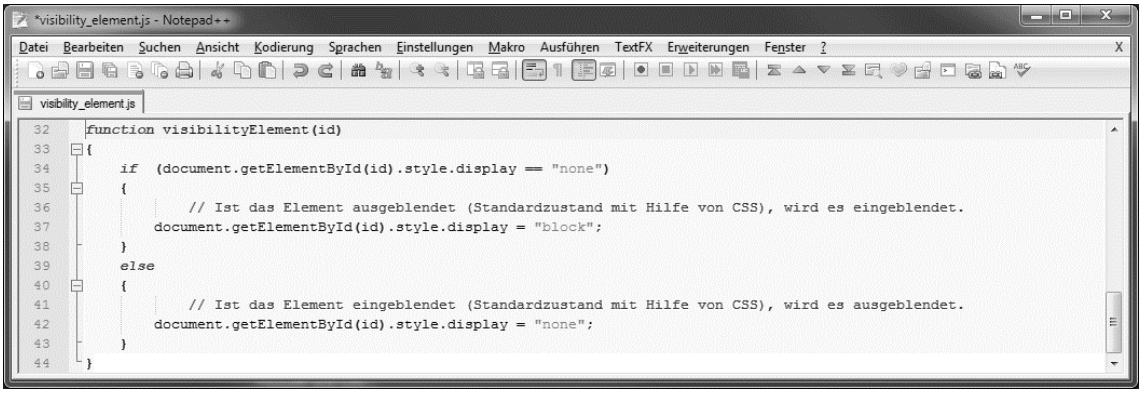
von Anweisungen nach KLOTZ (2003) erstellt und holt einen in diesem WLMML-Dokument eindeutigen Kennzeichner für das jeweilige Element, aus dem heraus es aufgerufen wurde. Der Kennzeichner leitet sich von der absoluten Pfadangabe zu diesem Element ab (Listing 75). Die XSLT-Anweisungen in Listing 74 Zeile 241 bis 243 speichern den Wert des Kennzeichners in der Variablen „id“. Anschließend wird der „title“-Attributwert des Elementes „detail“ in ein „span“-Element geschrieben und dieses „span“-Element mit einem „onclick“-Ereignis versehen (Listing 74 Zeile 244 bis 248). Zusätzlich bekommt das „span“-Element ein „title“-Attribut, das als Attributwert den sprachspezifischen Systemausdruck für das Element „detail“ aus der XML-Datei „lang_system.xml“ erhält. Darüber hinaus wird dahinter mit einem Doppelpunkt getrennt der „title“-Attributwert des „detail“-Elementes (Listing 74 Zeile 246) hinzugefügt. Unter diesem später am Bildschirm sichtbaren „span“-Elementinhalt steht ein zweites „span“-Element mit dem „id“-Attributwert der Kennzeichner-Variablen „id“ (Listing 74 Zeile 251 bis 255). Dieses zweite „span“-Element ist im Standardzustand nicht sichtbar (CSS-Anweisung „display:none;“, Listing 74 Zeile 253) und enthält den Inhalt des „detail“-Elementes (Listing 74 Zeile 254). Wenn der Anwender auf den Textinhalt des ersten immer sichtbaren „span“-Elementes klickt, wird der Wert der Variablen als Parameter an die JavaScript-Funktion „visibilityElement()“ (in der JavaScript-Datei „visibility_element.js“) übergeben. Anweisungen in dieser Funktion blenden dann dieses über den „id“-Wert eindeutig identifizierbare unsichtbare „span“-Element über den Wechsel der CSS-Eigenschaft „display“ ein (Listing 76). Sollte es bereits eingeblendet sein, wird es ausgeblendet.



Listing 74: Template für das Element „detail“ aus der zentralen XSLT-Datei „main.xsl“ (inklusive Aufruf des Templates „IdPathAbsolute“)

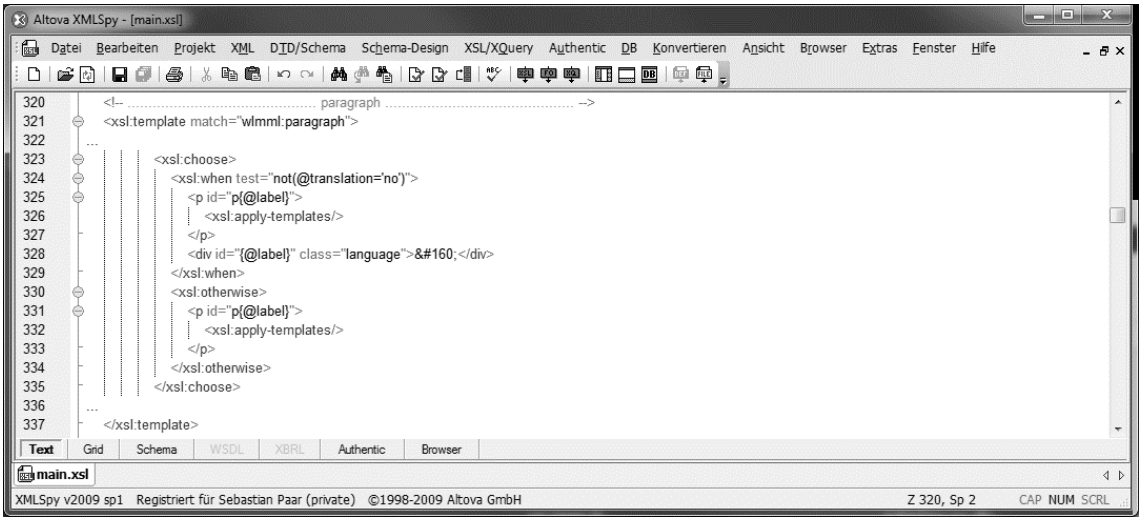


Listing 75: Template „IdPathAbsolute“ aus der zentralen XSLT-Datei „main.xsl“ (Ermittlung eines in diesem Dokument eindeutigen Kennzeichners)



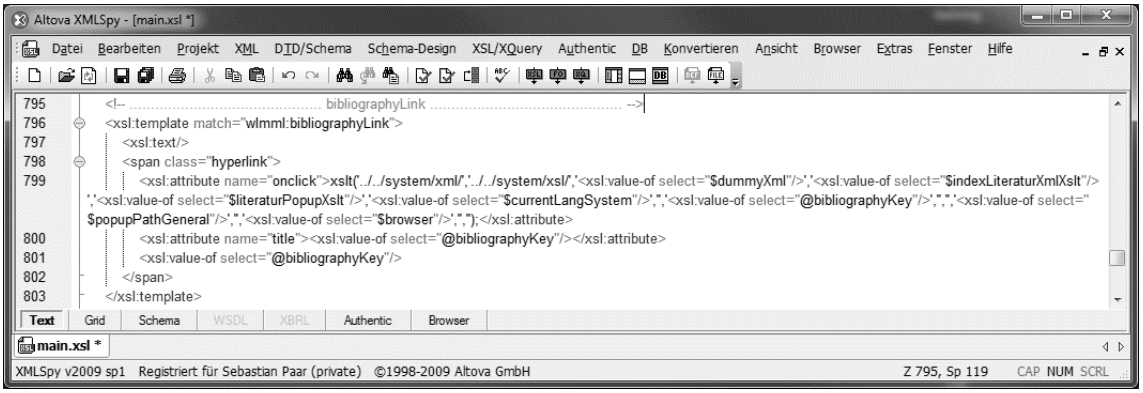
Listing 76: Funktion „visibilityElement()“ aus der JavaScript-Datei „visibility_element.js“ (Ein- bzw. Ausblenden von Elementen)

Auch für die anderen Elemente in Listing 69 wie „paragraph“, „bibliographyLink“, „text“ oder „image“ finden sich Templates in der XSLT-Datei „main.xsl“. Um z. B. über Hyperlinks unter jedem Absatz („paragraph“-Element) den Inhalt des Absatzes in der jeweiligen Sprache auswählen zu können, transformieren Anweisungen im „paragraph“-Template jeden für den Sprachwechsel vorgesehenen Absatz (Listing 77 Zeile 324, „translation“-Attributwert ist ungleich „no“) nach HTML in ein „p“-Element und versehen es mit einem „id“-Attribut (Listing 77 Zeile 325). Darin wird der „label“-Attributwert des „paragraph“-Elementes mit einem führenden Buchstaben „p“ (Abkürzung für „paragraph“) abgelegt. In das darunter angegebene „div“-Element schreiben später Anweisungen der JavaScript-Datei „lang_under_paragraph.js“ die verschiedenen Inhaltssprachen. Sie werden als Hyperlinks angeboten. Diese Hyperlinks führen zu den WLMML-Inhaltsdateien in der ausgewählten Sprache und zeigen den jeweiligen Absatz in einem eigenen Fenster hervorgehoben an (Anweisungen in der „highlight.js“) (siehe auch Kapitel 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen).



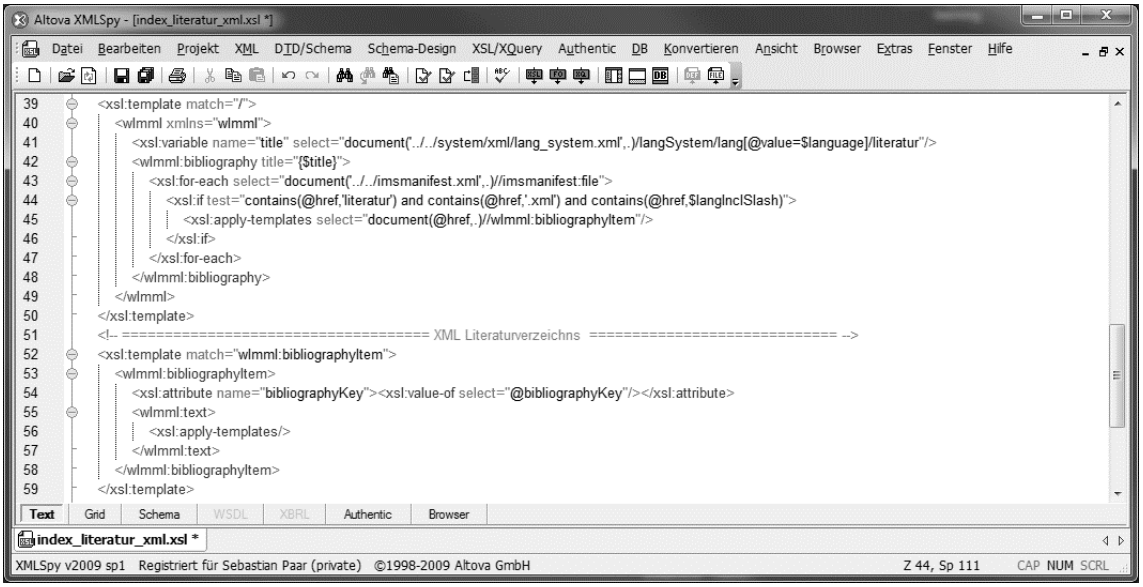
Listing 77: Ausschnitt aus dem Template „paragraph“ aus der zentralen XSLT-Datei „main.xsl“ (u. a. Zuweisung eines „id“-Wertes an Absätze, die bei einem Sprachwechsel in der ausgewählten Sprache angezeigt werden sollen)

Etwas aufwendiger stellt sich die Umsetzung eines Literaturverweises im Fließtext dar. Klickt man auf den Text, der als Kürzel für eine Literaturstelle eingebunden wurde, öffnet sich ein Fenster, in dem die Literaturstelle angezeigt wird. Sie wurde aus den verschiedenen Literatur-WLMML-Dateien herausgesucht. Dazu wird der „bibKey“-Parameterwert mit dem Attributwert „bibkey “des „bibliographyLink“-Elementes belegt und an die JavaScript-Funktion „xslt()“ in der JavaScript-Datei „xslt.js“ übergeben (Listing 78 Zeile 799). Der Wert der Variablen „currentLangSystem“ ist der „languageSystem“-Attributwert des „wlmml“-Elementes der jeweiligen WLMML-Datei (Listing 78 Zeile 799). Bei einem Mausklick auf einen Literaturverweis werden nur die Literaturverzeichnis-Dateien durchsucht, die in dem Sprachordner liegen, dessen Kürzel sich in der Angabe der Systemsprache der WLMML-Datei wiederfindet.



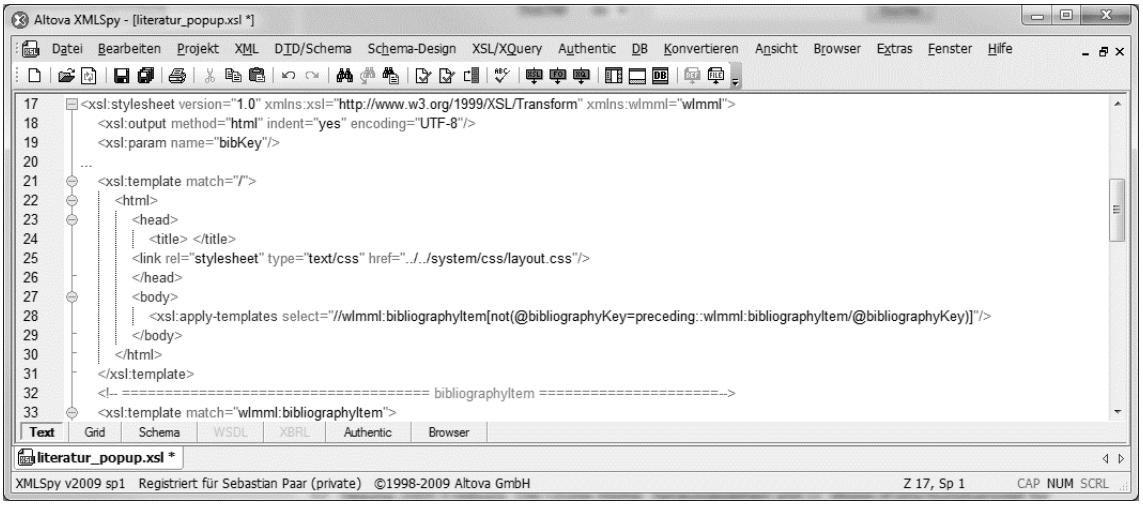
Listing 78: Template für das Element „bibliographyLink“ (Literaturverweis) aus der zentralen XSLT-Datei „main.xsl“ (inklusive Aufruf der JavaScript-Funktion „xslt()“)

Anweisungen in der JavaScript-Funktion „xslt()“ fassen in einer ersten Transformation alle Literaturstellen der verschiedenen Literaturverzeichnis-Dateien in der jeweiligen Sprache zusammen. Dazu wird die XSLT-Datei „index_literatur_xml.xsl“ als Parameter an die Funktion „xslt()“ übergeben (Listing 78 Zeile 799). Anweisungen in dieser XSLT-Datei rufen zunächst alle WLMML-Dateien aus der Manifestdatei „imsmanifest.xml“ auf (Listing 79 Zeile 43), die im Dateinamen den Text „literatur“ aufweisen (Listing 79 Zeile 44). Alle WLMML-Dateien, die ein Literaturverzeichnis darstellen, müssen diesen Text im Dateinamen enthalten. Anschließend sammeln Anweisungen alle in diesen Dateien gefundenen Literaturstellen und legen sie in einem virtuellen Gesamt-Literaturverzeichnis (XML-Dokument) ab (Listing 79 u. a. Zeile 52 ff). Der sprachspezifische Ausdruck für Literatur wird für eine spätere Verwendung (Zusammensetzung eines gesammelten Literaturverzeichnisses im Bereich „Verzeichnisse/Inhaltssuche“) aus der XML-Datei „lang_system.xml“ geholt (Listing 79 Zeile 41) und als „title“-Attributwert des Elementes „bibliography“-Elementes gesetzt (Listing 79 Zeile 42).



Listing 79: Ausschnitt aus der XSLT-Datei „index_literatur_xml.xsl“, Zusammenstellung eines virtuellen Literaturverzeichnisses

Anweisungen in der JavaScript-Funktion „xslt()“ übergeben anschließend dieses erste Transformationsergebnis (XML-Dokument) mit Parametern zur zweiten Transformation an die XSLT-Datei „literatur_popup.xsl“ (Listing 78 Zeile 799).



Listing 80: Ausschnitt aus der XSLT-Datei „literatur_popup.xsl“, Anzeige der gesuchten Literaturstelle

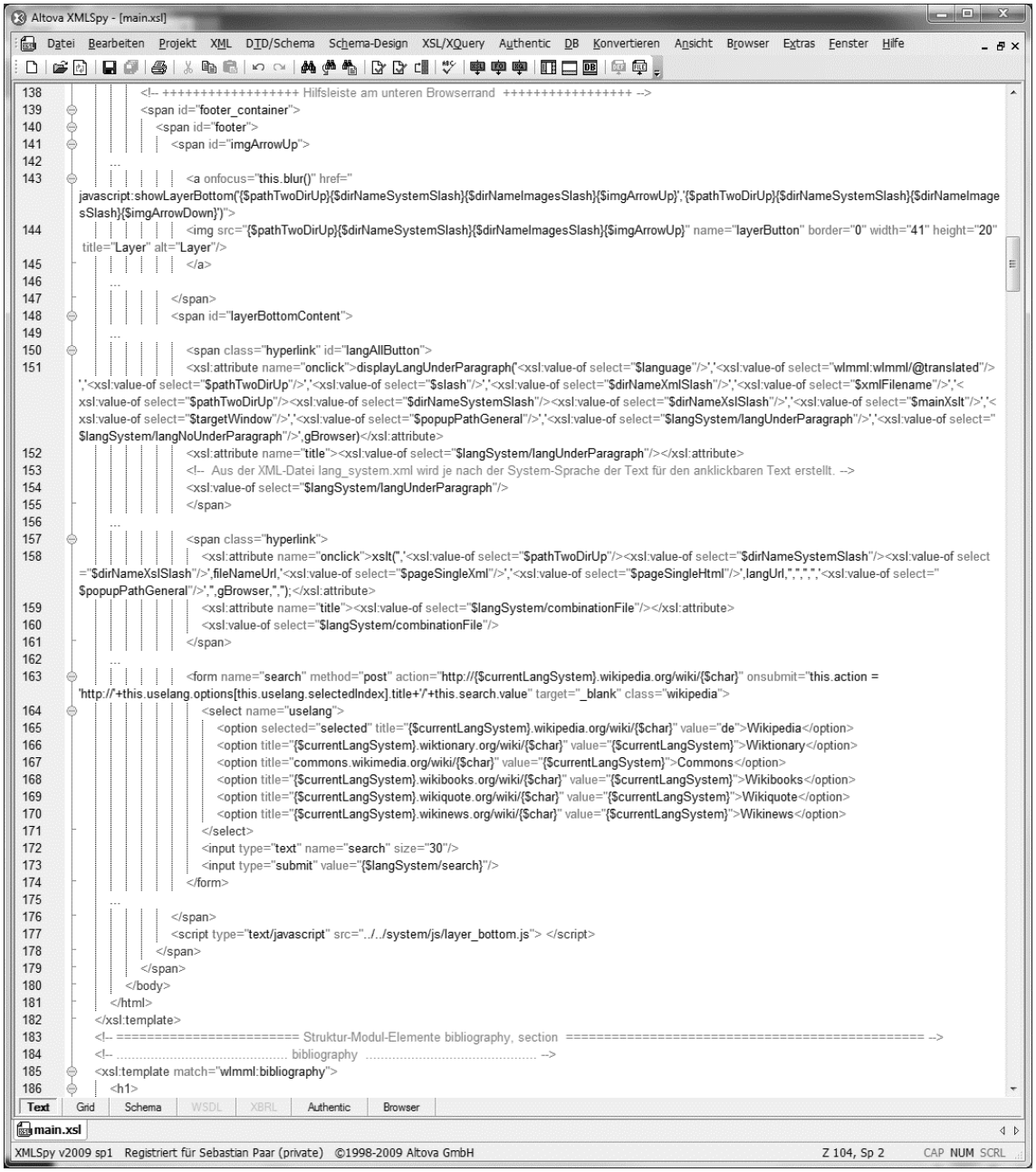
Die XSLT-Befehle dieser Datei setzen eine HTML-Datei zusammen (Listing 80 Zeile 22 ff), suchen mit Hilfe des übergebenen Parameters „bibKey“ (Listing 80 Zeile 19) nach der gesuchten Literaturstelle (ohne doppelte Werte, Listing 80 Zeile 28) und binden die gefundene Literaturstelle in die HTML-Datei ein. Anweisungen in der JavaScript-Funktion „xslt()“ stellen anschließend sicher, dass das zweite Transformationsergebnis (gesuchte Literaturstelle) im Browser in einem neuen Fenster angezeigt wird.

Sind alle Templates der XSLT-Datei „main.xsl“ abgearbeitet und die Ergebnisse in das HTML-Gerüst eingebunden, erzeugen Anweisungen die Hilfsleiste am unteren Browserrand (Listing 81 Zeile 139 bis 179). Als Behälter dient das „span“-Element mit dem „id“-Attributwert „footer_container“ (Listing 81 Zeile 139 bis 179). Er beinhaltet nur ein weiteres „span“-Element mit dem „id“-Attributwert „footer“ (Listing 81 Zeile 140), das sämtliche Inhalte dieser Hilfsleiste bzw. Fußzeile aufnimmt (Listing 81 Zeile 140 bis 178). CSS-Anweisungen in der „layout_bottom.css“ formatieren und fixieren die Fußzeile am unteren Rand des Browsers. Auch während eines „Scroll“-Vorgangs verbleibt sie dort. Die dafür notwendigen CSS-Anweisungen stammen von RIEHLE (2007). Die Hilfsleiste wird nur durch Anklicken der Lasche am unteren Browserrand sichtbar (Abbildung 83 unten) und wieder unsichtbar (Abbildung 80 unten). Für dieses Verhalten sind JavaScript-Befehle in der „layer_bottom.js“ verantwortlich. Die JavaScript-Funktion „showLayerBottom()“ (Listing 81 Zeile 143) schaltet dazu über die CSS-Eigenschaft „display“ den Inhalt des „span“-Elementes mit dem „id“-Attributwert „layerBottomContent“ (Listing 81 Zeile 148) sichtbar und unsichtbar.

In den linken Bereich der Hilfsleiste schreiben Anweisungen in Listing 81 Zeile 154 den sprachspezifischen Systemausdruck für einen Sprachwechsel unter den Absätzen (siehe auch Abbildung 83 unten). Der Text stammt aus der XML-Datei „lang_system.xml“. Wählt der Anwender dieses anklickbare „span“-Element aus („onclick“-Ereignis in Listing 81 Zeile 151), ruft er damit die Funktion „displayLang-UnderParagraph()“ aus der JavaScript-Datei „lang_under_paragraph.js“ mit mehreren Parametern auf (Listing 81 Zeile 151). Diese Funktion stellt sicher, dass die jeweiligen Sprachkürzel in Form von Hyperlinks unter den dafür vorgesehenen Absätzen erscheinen. Wählt der Anwender einen dieser Hyperlinks aus, soll ihm dieser Absatz in der ausgewählten Sprache angezeigt werden (siehe auch Anmerkungen zum „paragraph“-Template weiter oben).

Als weiterer Inhalt der Hilfsleiste wird in Listing 81 Zeile 160 der sprachspezifische Systemausdruck für die Zusammenfassung zu einer Datei gesetzt (siehe auch Abbildung 83 rechts unten). Wenn der Anwender dieses anklickbare „span“-Element auswählt („onclick“-Ereignis in Listing 81 Zeile 158), lassen sich über Anweisungen in der JavaScript-Datei „xslt.js“ die WLMML-Inhaltsdateien zu einem einzigen Dokument zusammenfassen (z. B. für die einfache Druckausgabe, siehe Kapitel 4.3.2.4.2.2 Zusammenfassung der Lerninhalte in einem Dokument). Die Browser „Google Chrome“ und „Apple Safari“ nutzen auch Anweisungen in der „lang_file_url_lang_all_imsmanifest.js“.

Den letzten Bestandteil der Hilfsleiste erstellen die Anweisungen in Listing 81 Zeile 163 bis 174. Sie erzeugen ein Formular mit einem Eingabefeld (Listing 81 Zeile 172, Abbildung 83 unten), das es ermöglicht verschiedene Online-Dienste zu durchsuchen (siehe Kapitel 4.3.2.4.2.3 Suche in verschiedenen Online-Diensten (nur mit Internet-Zugang möglich)). Die XSLT-Variable „currentLangSystem“ übergibt dabei als Standard-Suchsprache den „languageSystem“-Attributwert des „wlmml“-Elementes im abgearbeiteten WLMML-Dokument. Die sprachspezifische Beschriftung des Sende-„Buttons“ wird wieder aus Inhalten der XML-Datei für die Systemausdrücke bezogen („lang_system.xml“) erstellt (Listing 81 Zeile 173).



Listing 81: Ausschnitt aus der zentralen XSLT-Datei „main.xsl“ (zweiter Teil der Erzeugung eines HTML-Grundgerüstes)

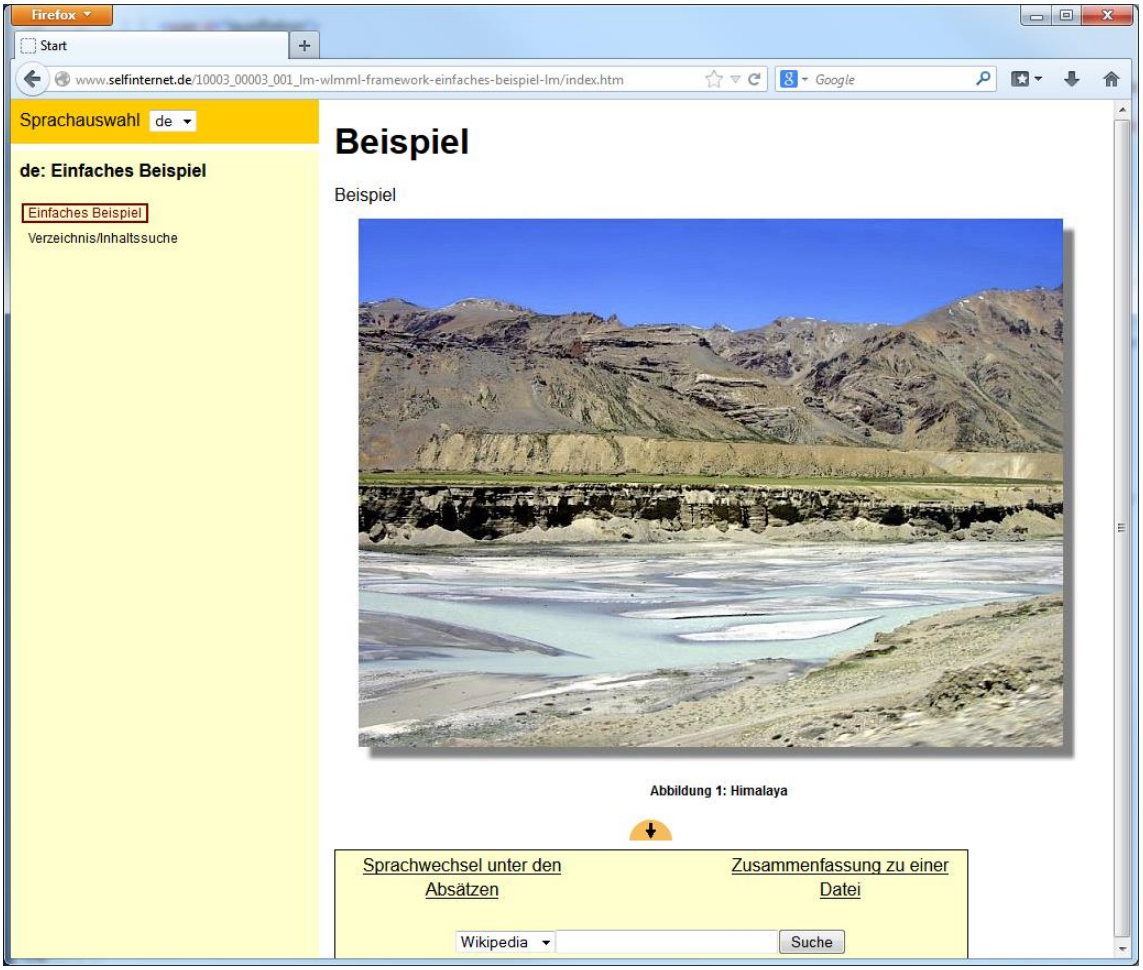


Abbildung 83: Durch Anklicken der Lasche sichtbar gewordene Hilfsleiste am unteren Browserrand, Darstellung im Browser „Mozilla Firefox 28.0“

In Listing 81 Zeile 177 wird zum Abschluss die JavaScript-Datei „layer_bottom.js“ geladen (Ladereihenfolge in Abbildung 79 Zeile 21). Sie stellt wie bereits besprochen Funktionalitäten der am unteren Browserrand verborgenen Hilfsleiste sicher (siehe Kapitel 4.3.2.4.2 Funktionalitäten in der Hilfsleiste am unteren Browserrand).

Gemäß Abbildung 79 Zeile 23 und 24 fehlen in der Ladreihenfolge nur noch die beiden Bilddateien „10003_00003_himalaya.jpg“ (Listing 69 Zeile 7, großes Bild im „content“-Frame, siehe Abbildung 80 rechts) und „button_arrow_up.gif“ (Lasche der Hilfsleiste am unteren Browserrand, siehe Abbildung 80 rechts unten). Wenn Sie dem Browser zur Verfügung stehen, ist die WLMML-Datei fertig geladen und kann im Browser betrachtet werden (Abbildung 80, Inhalt des „content“-Frame, rechte Seite).

4.3.2.4.2 Funktionalitäten in der Hilfsleiste am unteren Browserrand

Wie in Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei beschrieben, wird die Hilfsleiste über Anweisungen in der zentralen XSLT-Datei „main.xsl“ erstellt (siehe vor allem Listing 81). Die verborgene Hilfsleiste wird mit einer Lasche am unteren Rand des Browserfensters angezeigt (Abbildung 84).

Abbildung 84: Lasche am unteren Browserrand

Wird die Lasche angeklickt, erscheint die Hilfsleiste (Abbildung 85 roter Rahmen). Wird sie nochmals angeklickt, verschwindet die Hilfsleiste wieder.

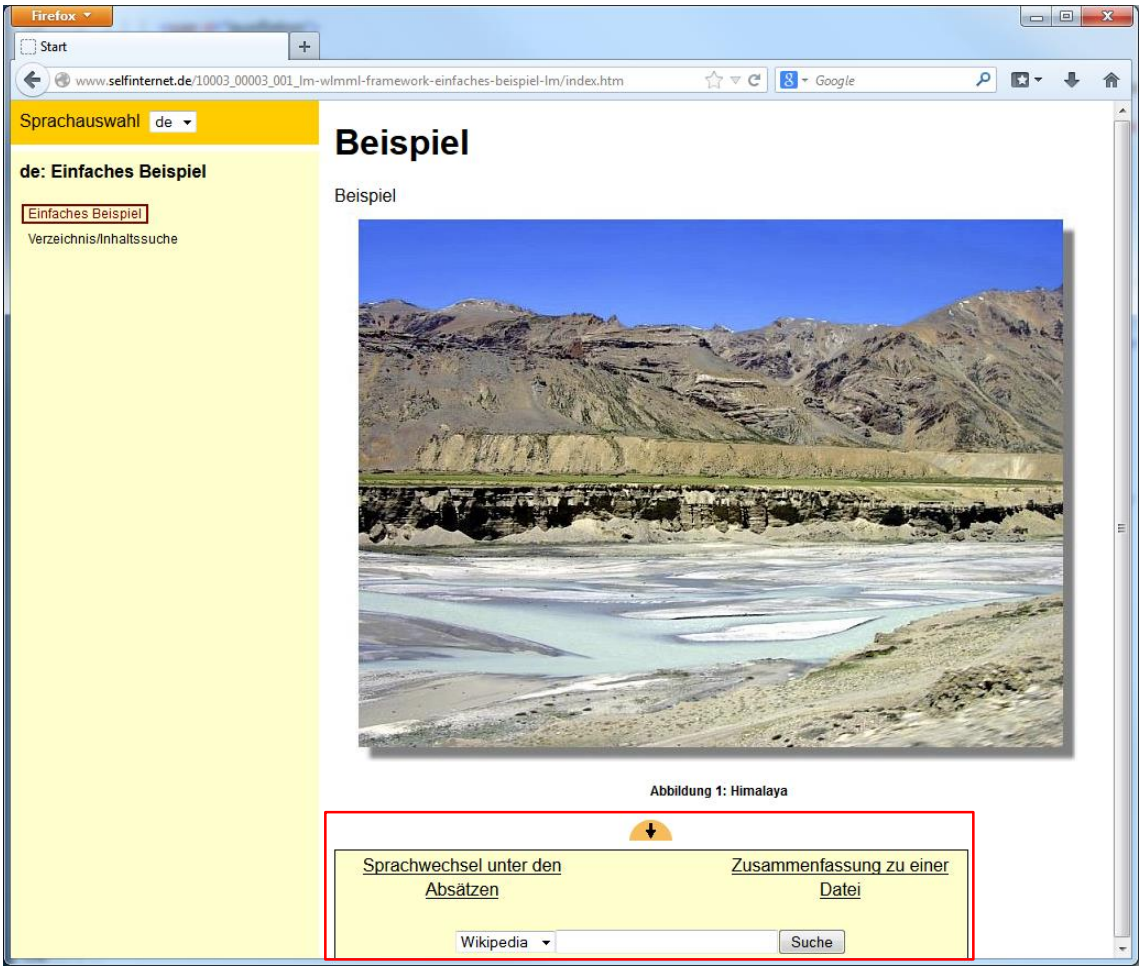


Abbildung 85: Durch Anklicken der Lasche sichtbar gewordene Hilfsleiste (roter Rahmen) am unteren Browserrand, Darstellung im Browser „Mozilla Firefox 28.0“

Die Hilfsleiste stellt verschiedene Funktionalitäten zur Verfügung:

- Anzeige (Funktionalität) von anklickbaren Sprachkürzel-Hyperlinks unter Absätzen
- Zusammenfassung der WLMML-Dateien zu einer einzigen im Browser sichtbaren und druckbaren Seite

- Wikipedia/Wiktionary/Commons/Wikibooks/Wikiquote/Wikinews-Suche (online) in der jeweiligen Sprache, in der der Lerninhalt momentan angezeigt wird

4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen

Die Hilfsleiste ermöglicht es über den sprachspezifischen Systemausdruck „Sprachwechsel unter den Absätzen“ unter Absätzen im Fließtext Sprachkürzel-Hyperlinks ein- bzw. anschließend wieder auszublenden (Abbildung 86, Abbildung 87 unten, siehe auch QUEDNAU et al. 2007, S. 4 f).

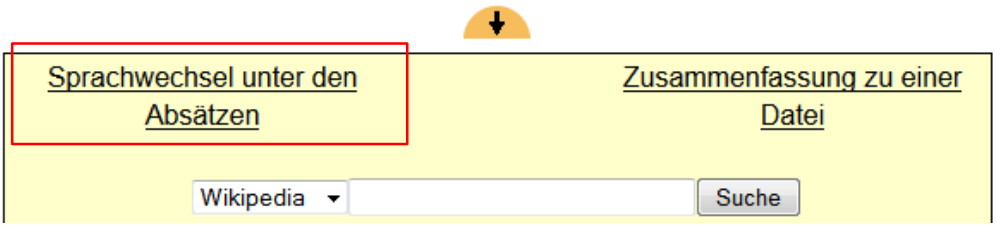


Abbildung 86: Hilfsleiste am unteren Browserrand mit der Funktionalität „Sprachwechsel unter den Absätzen“ (roter Rahmen)

Die anklickbaren Sprachkürzel-Hyperlinks werden unter jedem vom Autor der Lerneinheit dafür vorgesehenen Absatz angezeigt (siehe Kapitel 4.3.2.3.3 Sprachangaben in WLMML-Dateien, 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei (Thema Element „paragraph“)) und geben die Kürzel der weiteren Sprachen wieder, in denen der Absatz übersetzt vorliegt (Abbildung 87). Klickt man auf einen dieser Hyperlinks, öffnet sich ein Fenster, in dem der Inhalt des jeweiligen Absatzes in der angeklickten Sprache hervorgehoben angezeigt wird (Abbildung 88).

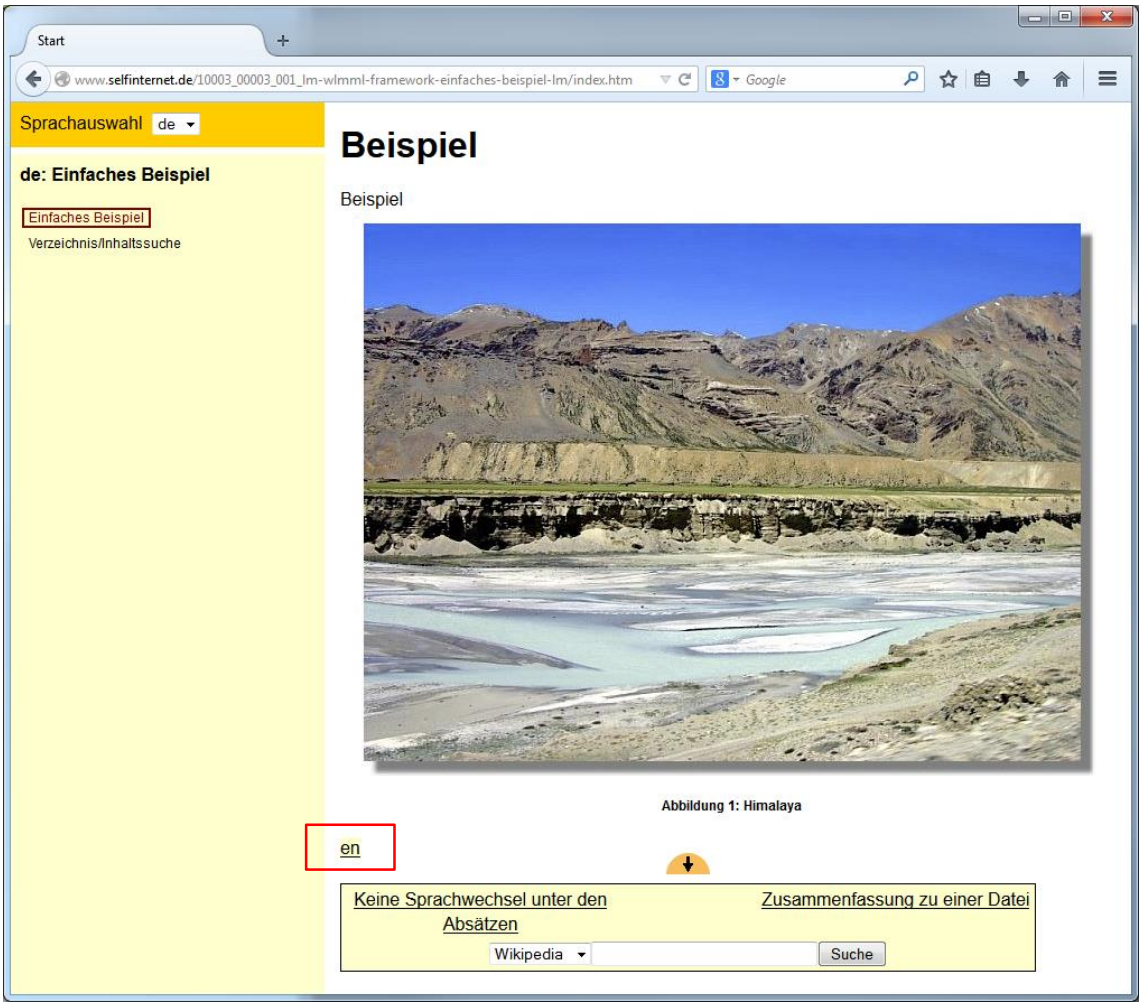


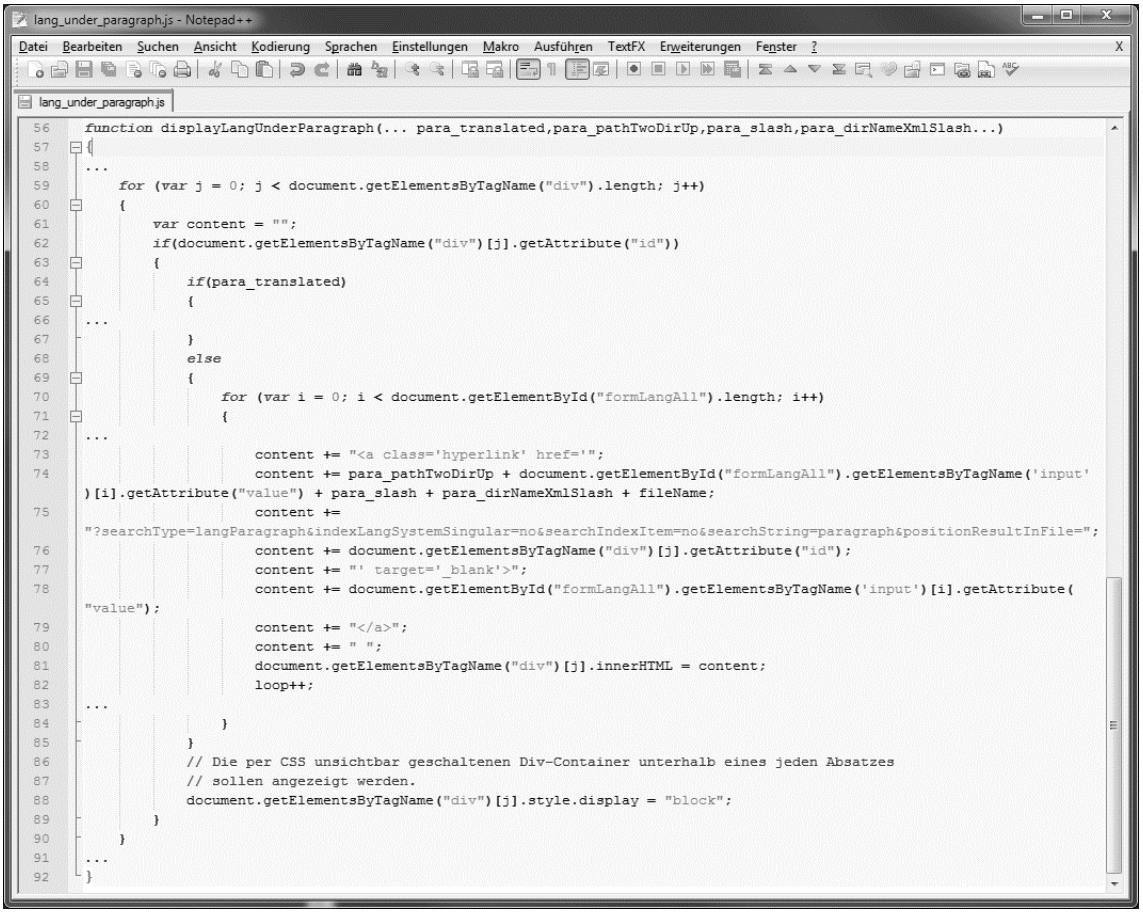
Abbildung 87: „Sprachwechsel unter den Absätzen“ in der Hilfsleiste wurde ausgewählt, Sprachen unter jedem dafür vorgesehenen Absatz werden angezeigt (roter Rahmen), Darstellung im Browser „Mozilla Firefox 29.0.1“

Abbildung 88: Sprachkürzel „en“ wurde ausgewählt, Absatz in dieser Sprache wird hervorgehoben in einem eigenen Fenster angezeigt, Darstellung im Browser „Mozilla Firefox 29.0.1“

Für die Ausgabe der Sprachkürzel unter den dafür vorgesehenen Absätzen („paragraph“-Elemente) sind Anweisungen der Funktion „displayLang-UnderParagraph()“ verantwortlich (in der JavaScript-Datei „lang_under-_paragraph.js“). Wählt der Anwender den sprachspezifischen Systemausdruck „Sprachwechsel unter den Absätzen“ aus, startet er damit diese Funktion und übergibt ihr mehrere Parameter (Listing 82 Zeile 56). Zu Beginn des Listing 82 in Zeile 59 werden alle „div“-Elemente des HTML-Dokumentes durchlaufen. Ein „div“-Element mit einem Attribut „id“ wird von Anweisungen in der XSLT-Datei „main.xslt“ nur dann unter ein „paragraph“-Element gesetzt, wenn der „translation“-Attributwert des Absatzes nicht „no“ beinhaltet (siehe Listing 77 Zeile 324 bis 329). Enthält er nicht den Wert „no“, werden unter dem Absatz die verschiedenen Sprachkürzel in Form von Hyperlinks angezeigt (Abbildung 87, Anzeige über Beeinflussung der CSS-Eigenschaft „display“, Listing 82 Zeile 88). Ansonsten bekommt der Absatz keine Sprachkürzel zugewiesen. Dies ist dann der Fall, wenn der Autor des Lerninhaltes ein „paragraph“-Element mit dem „translation“-Attributwert „no“ versieht und er diesen

Absatz damit nicht für eine Übersetzung vorsieht (siehe auch Kapitel 4.2.3.5 Attribute für das Element „paragraph“).

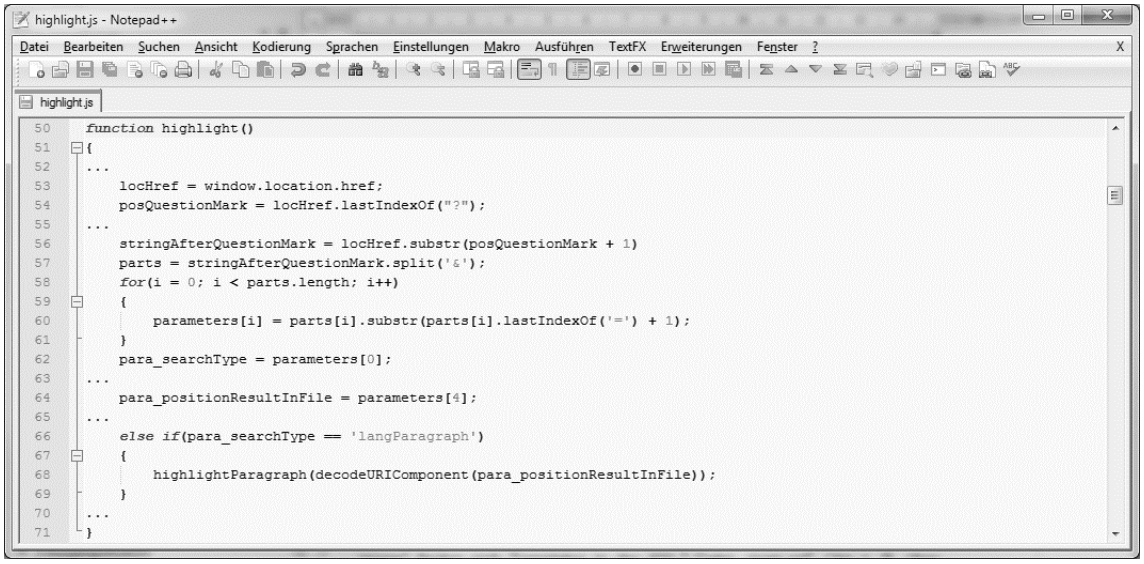
Bei dem angesprochenen Aufruf der Funktion „displayLangUnderParagraph()“ werden verschiedene Parameter übergeben. Einer davon ist „para_translated“. Wenn dieser Parameterwert einen Wert aufweist (Listing 82 Zeile 64), werden die verschiedenen Sprachen unter jedem Absatz aus dem „translated“-Attributwert des „wlmml“-Elementes (siehe auch Kapitel 4.2.3.6 Attribute für das Element „wlmml“) extrahiert. Wurde das „translated“-Attribut nicht genutzt, kommen die Sprachen aus den Metadaten der jeweiligen „organization“-Elemente in der Manifest-Datei „imsmanifest.xml“ (Ergebnis aus Anweisungen in der XSLT-Datei „main.xsl“, dort Ablage der Sprachen in das unsichtbare Formular mit der „id“ „formLangAll“, Listing 82 Zeile 70, siehe auch Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei). Damit werden sozusagen die Standardsprachen unter den Absätzen angeboten. Die Hyperlinks über die sich die Sprachen auswählen lassen, setzen Anweisungen in Listing 82 Zeile 73 bis 79 zusammen. Diese Hyperlinks zu den WLMML-Inhaltsdateien in den jeweiligen Sprachen weisen URL-Anhänge auf (Inhalt u. a. „id“-Attributwert des „p“-Elementes), die für die Anzeige des Absatzes in der gewählten Zielsprache ausgewertet werden. Es wird der Absatz in der jeweiligen sprachspezifischen WLMML-Datei in einem neuen Fenster hervorgehoben angezeigt, der mit dem übergebenen „label“-Parameterwert und der ausgewählten Sprache übereinstimmt (Abbildung 88).



```
56  function displayLangUnderParagraph(... para_translated,para_pathTwoDirUp,para_slash,para_dirNameXmlSlash...)
57  {
58  ...
59      for (var j = 0; j < document.getElementsByTagName("div").length; j++)
60      {
61          var content = "";
62          if(document.getElementsByTagName("div")[j].getAttribute("id"))
63          {
64              if(para_translated)
65              {
66  ...
67              }
68              else
69              {
70                  for (var i = 0; i < document.getElementById("formLangAll").length; i++)
71                  {
72  ...
73                      content += "<a class='hyperlink' href='";
74                      content += para_pathTwoDirUp + document.getElementById("formLangAll").getElementsByTagName('input'
    )[i].getAttribute("value") + para_slash + para_dirNameXmlSlash + fileName;
75                      content +=
    "?searchType=langParagraph&indexLangSystemSingular=no&searchIndexItem=no&searchString=paragraph&positionResultInFile=";
76                      content += document.getElementsByTagName("div")[j].getAttribute("id");
77                      content += "' target='_blank'>";
78                      content += document.getElementById("formLangAll").getElementsByTagName('input')[i].getAttribute(
    "value");
79                      content += "</a>";
80                      content += " ";
81                      document.getElementsByTagName("div")[j].innerHTML = content;
82                      loop++;
83  ...
84                  }
85              }
86              // Die per CSS unsichtbar geschaltenen Div-Container unterhalb eines jeden Absatzes
87              // sollen angezeigt werden.
88              document.getElementsByTagName("div")[j].style.display = "block";
89          }
90      }
91  ...
92  }
```

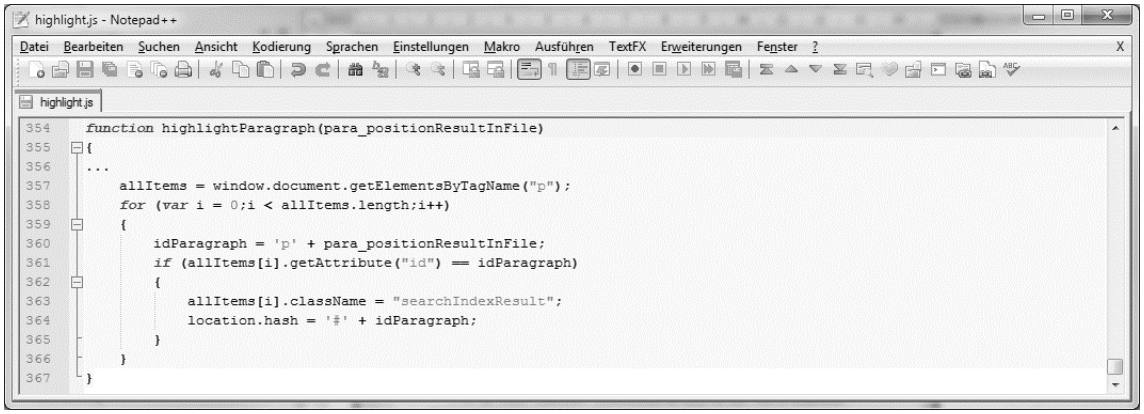
Listing 82: Ausschnitt aus der Funktion „displayLangUnderParagraph()“ aus der JavaScript-Datei „lang_under_paragraph.js“ (Anzeige der Sprachkürzel unter dafür vorgesehene Absätze („paragraph“-Elemente))

Anweisungen in der JavaScript-Datei „highlight.js“ übernehmen die Suche und anschließende Hervorhebung des betroffenen Absatzes. Diese Datei ist in jeder WLMML-Inhaltsdatei über die XSLT-Datei „main.xsl“ eingebunden. Wählt der Anwender einen Hyperlink in Form eines Sprachkürzels aus, lädt er damit eine WLMML-Datei in dieser Sprache aus und ruft bei diesem Vorgang („onload“-Ereignis, siehe Listing 70 Zeile 125) automatisch die Funktion „highlight()“ auf (Funktion in der JavaScript-Datei „highlight.js“). Sie nimmt die per URL übertragenen Parameter entgegen (Listing 83 Zeile 53 bis 64), wertet sie aus (Listing 83 Zeile 66) und stößt die weitere Verarbeitung über die Funktion „highlightParagraph()“ an (Listing 83 Zeile 68).



Listing 83: Ausschnitt aus der Funktion „highlight()“ aus der JavaScript-Datei „highlight.js“ (Extrahieren der Parameter aus den URL-Anhängen, Anstoß der Suche und Hervorhebung des jeweiligen Absatzes)

Die Funktion „highlightParagraph()“ übernimmt die Suche und Hervorhebung des jeweiligen Absatzes (Listing 84). Zuerst werden alle „p“-Elemente des Dokumentes gesammelt (Listing 84 Zeile 357) und anschließend überprüft, welches davon einen „id“-Attributwert aufweist, der mit dem per Parameter übergebenen „id“-Attributwert (z. B. „p110“) übereinstimmt (Listing 84 Zeile 361). Ist dies der Fall, hebt eine CSS-Anweisung (Zuweisung der CSS-Klasse „searchIndexResult“) das jeweilige Element hervor (Listing 84 Zeile 363). Abschließend wird der gefundene Absatz noch automatisch im Browserfenster aufgesucht, um dem Anwender den Einsatz der Bildlaufleiste zu ersparen (Listing 84 Zeile 364).



Listing 84: Ausschnitt aus der Funktion „highlightParagraph()“ aus der JavaScript-Datei „highlight.js“ (Suche und Hervorhebung des jeweiligen Absatzes)

4.3.2.4.2.2 Zusammenfassung der Lerninhalte in einem Dokument

Die Hilfsleiste am unteren Browserrand stellt eine Möglichkeit zur Verfügung alle zu einer Sprache gehörenden WLMML-Dateien eines Lernpaketes zu einer einzigen im

Browser sichtbaren Seite zusammenzufassen (Abbildung 89). Der Anwender kann sich damit schnell alle WLMML-Inhalte anzeigen lassen und diese Seite auch ausdrucken.

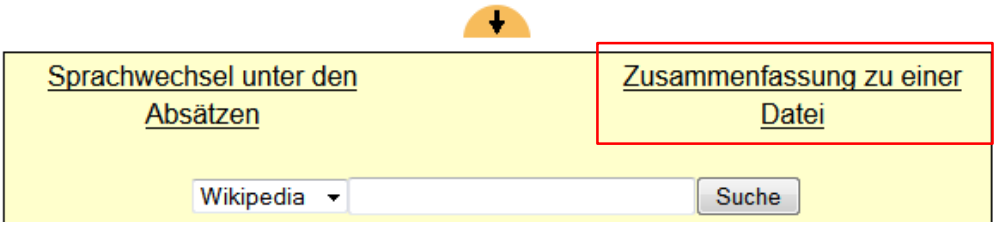
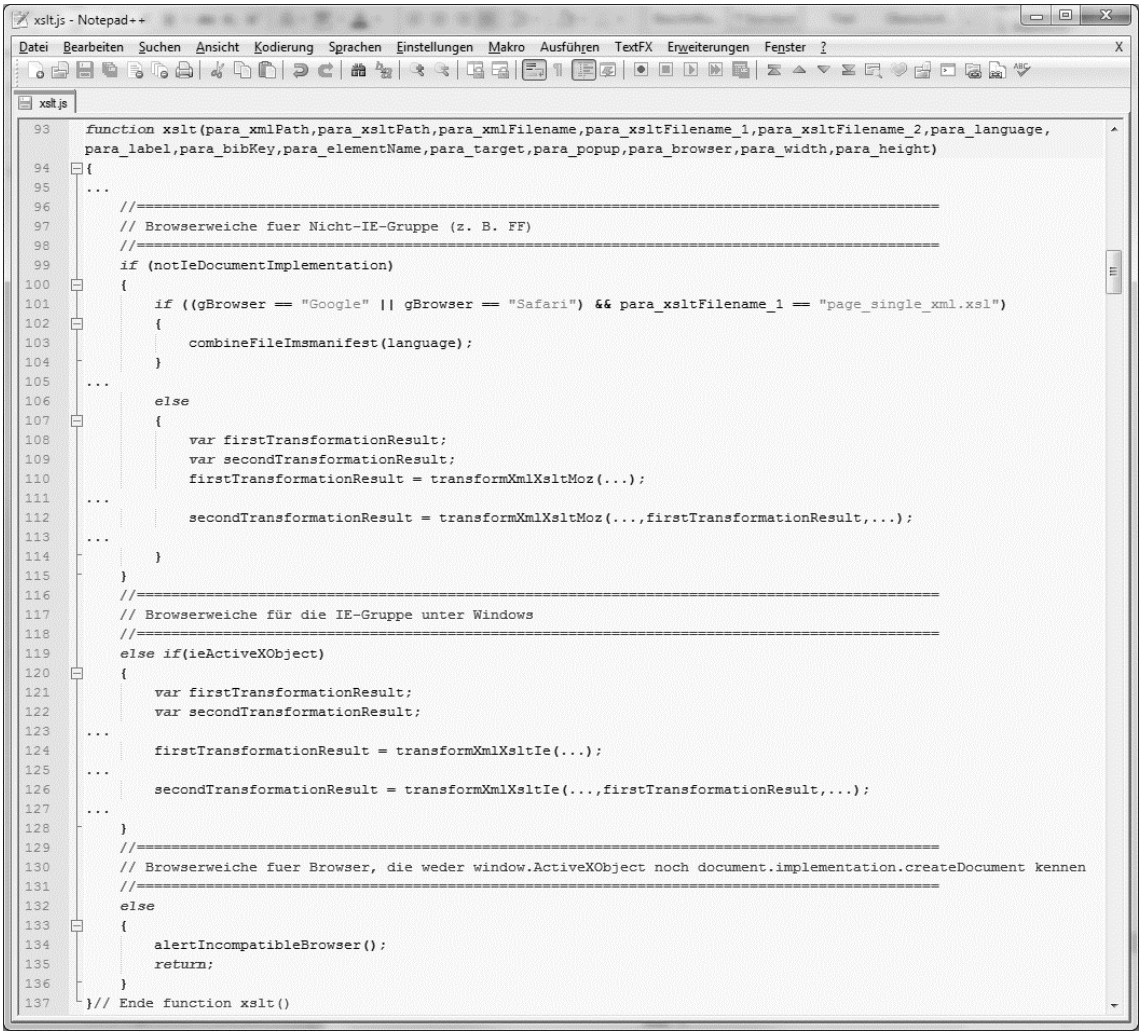


Abbildung 89: Hilfsleiste am unteren Browserrand mit der Funktionalität „Zusammenfassung zu einer Datei“ (roter Rahmen)

Klickt der Anwender auf den sprachspezifischen Ausdruck „Zusammenfassung zu einer Datei“ (Abbildung 89), ruft er die Funktion „xslt()“ in der JavaScript-Datei „xslt.js“ auf. Er übergibt als Parameter unter anderem den WLMML-Dateinamen, die XSLT-Dateien „page_single_xml.xsl“ und „page_single_html.xsl“, die Sprache, sowie den Browsertyp (siehe Listing 81 Zeile 154). Die Funktion „xslt()“ nimmt die Parameter entgegen (Listing 85 Zeile 93), unterscheidet in der Verarbeitung drei Browsergruppen „Google Chrome und Apple Safari“, „Mozilla Firefox“ und „Microsoft Internet Explorer“ (Listing 85 Zeile 99, 101, 119) und stößt je nach Browsergruppe zwei XSLT-Transformationen an (Listing 85 Zeile 110 und 112 für „Mozilla Firefox“, Zeile 124 und 126 für „Microsoft Internet Explorer“). Für die Browser „Google Chrome“ und „Apple Safari“ wird die Funktion „combineFileImsmanifest()“ in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“ aufgerufen. Sie stellt ebenfalls alle WLMML-Inhaltsdateien des Lernpaketes in der ausgewählten Sprache in einem Browserfenster zusammen (allerdings findet nur eine XSLT-Transformation mit der „page_single_html.xsl“ statt).

Am Beispiel des „Mozilla Firefox“ werden nachfolgend die beiden XSLT-Transformationen erläutert (ähnlicher Vorgang bei „Microsoft Internet Explorer“). Listing 85 Zeile 110 ruft die Funktion „transformXmlXsltMoz()“ auf und stößt die erste Transformation an.

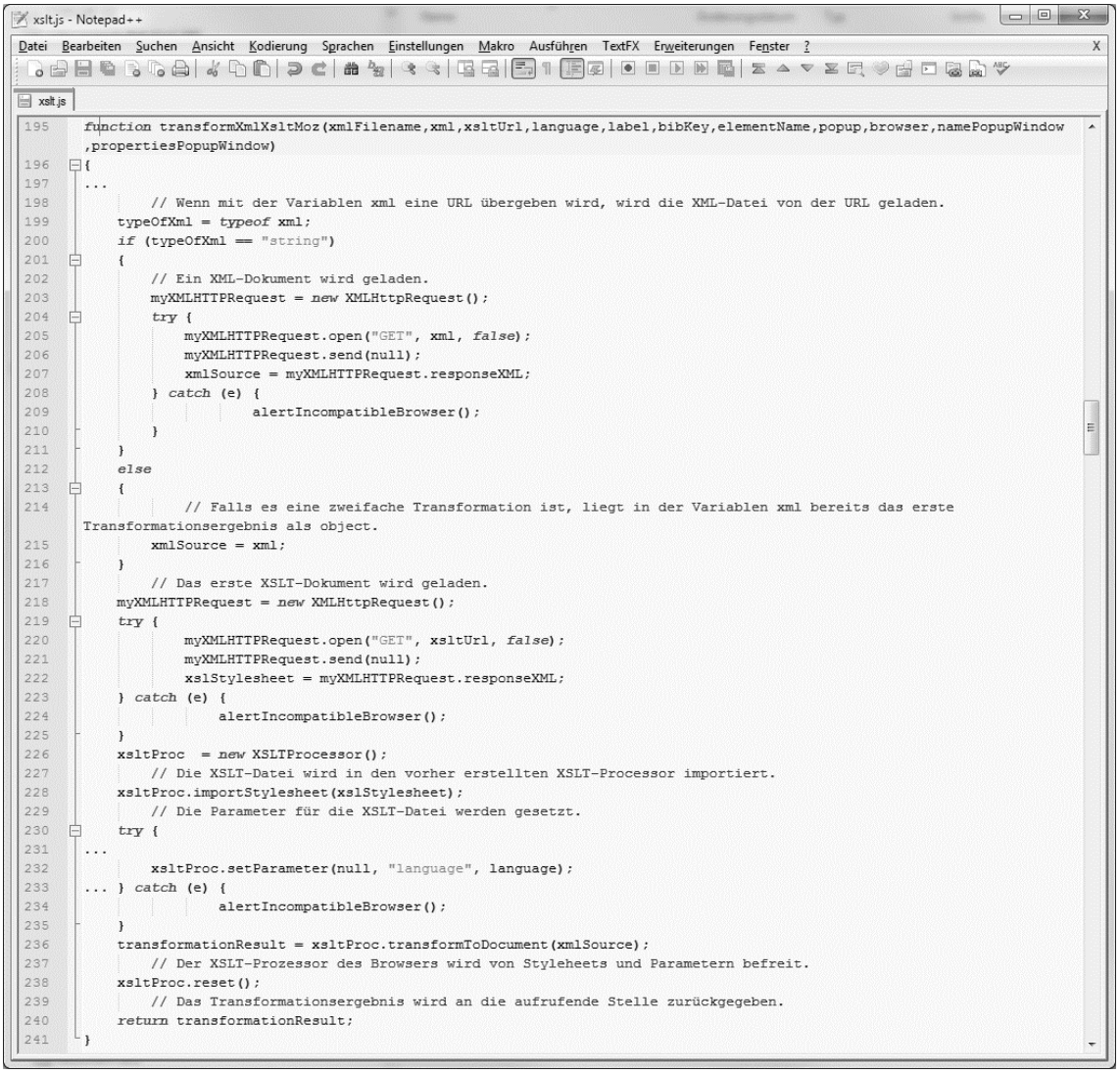


```
93  function xslt(para_xmlPath,para_xsltPath,para_xmlFilename,para_xsltFilename_1,para_xsltFilename_2,para_language,
    para_label,para_bibKey,para_elementName,para_target,para_popup,para_browser,para_width,para_height)
94  {
95  ...
96      //============================================================================================
97      // Browserweiche fuer Nicht-IE-Gruppe (z. B. FF)
98      //============================================================================================
99      if (notIeDocumentImplementation)
100     {
101         if ((gBrowser == "Google" || gBrowser == "Safari") && para_xsltFilename_1 == "page_single_xml.xsl")
102         {
103             combineFileImsmanifest(language);
104         }
105 ...
106         else
107         {
108             var firstTransformationResult;
109             var secondTransformationResult;
110             firstTransformationResult = transformXmlXsltMoz(...);
111 ...
112             secondTransformationResult = transformXmlXsltMoz(...,firstTransformationResult,...);
113 ...
114         }
115     }
116     //============================================================================================
117     // Browserweiche für die IE-Gruppe unter Windows
118     //============================================================================================
119     else if(ieActiveXObject)
120     {
121         var firstTransformationResult;
122         var secondTransformationResult;
123 ...
124         firstTransformationResult = transformXmlXsltIe(...);
125 ...
126         secondTransformationResult = transformXmlXsltIe(...,firstTransformationResult,...);
127 ...
128     }
129     //============================================================================================
130     // Browserweiche fuer Browser, die weder window.ActiveXObject noch document.implementation.createDocument kennen
131     //============================================================================================
132     else
133     {
134         alertIncompatibleBrowser();
135         return;
136     }
137 }// Ende function xslt()
```

Listing 85: Ausschnitt aus der Funktion „xslt()“ aus der JavaScript-Datei „xslt.xsl“ (Browserweiche und Anstoß der Transformationen)

Die Funktion „transformXmlXsltMoz()“ (Listing 86 Zeile 195) prüft zuerst, ob der übergebene Parameterwert der Variablen „xml“ vom Typ „string“ ist (Listing 86 Zeile 199. Ist dies der Fall, wird davon ausgegangen, dass es sich um eine URL einer XML-Datei handelt und die entsprechende XML-Datei geladen (Listing 86 Zeile 200 bis 211). Im Fehlerfall gibt die Funktion „alertIncompatibleBrowser()“ eine dreisprachige Meldung am Bildschirm aus (Listing 86 Zeile 208, 209). Das Ergebnis nimmt die Variable „xmlSource“ auf. Ist der Parameterwert der Variablen „xml“ nicht vom Typ „string“, wird davon ausgegangen, dass in der Variablen „xml“ bereits ein Transformationsergebnis vorliegt und der Inhalt ebenfalls in die Variable „xmlSource“ abgelegt (Listing 86 Zeile 215). Anschließend laden die Anweisungen in Listing 86 Zeile 218 bis 225 die erste XSLT-Datei (hier: „page_single_xml.xsl“) und geben im Fehlerfall ebenfalls eine Meldung aus. Danach werden die folgenden Aktionen durchgeführt (vergleiche auch Kapitel 3.6.2 Ajax):

- Erstellung einer XSLT-Prozessor-Instanz (Listing 86 Zeile 226)
- Import der XSLT-Datei (Listing 86 Zeile 228)
- Übergabe der Parameter (Listing 86 Zeile 232)
- Anzeige einer Meldung im Fehlerfall (Listing 86 Zeile 234)
- Transformation mit den XML-Daten (Listing 86 Zeile 236)



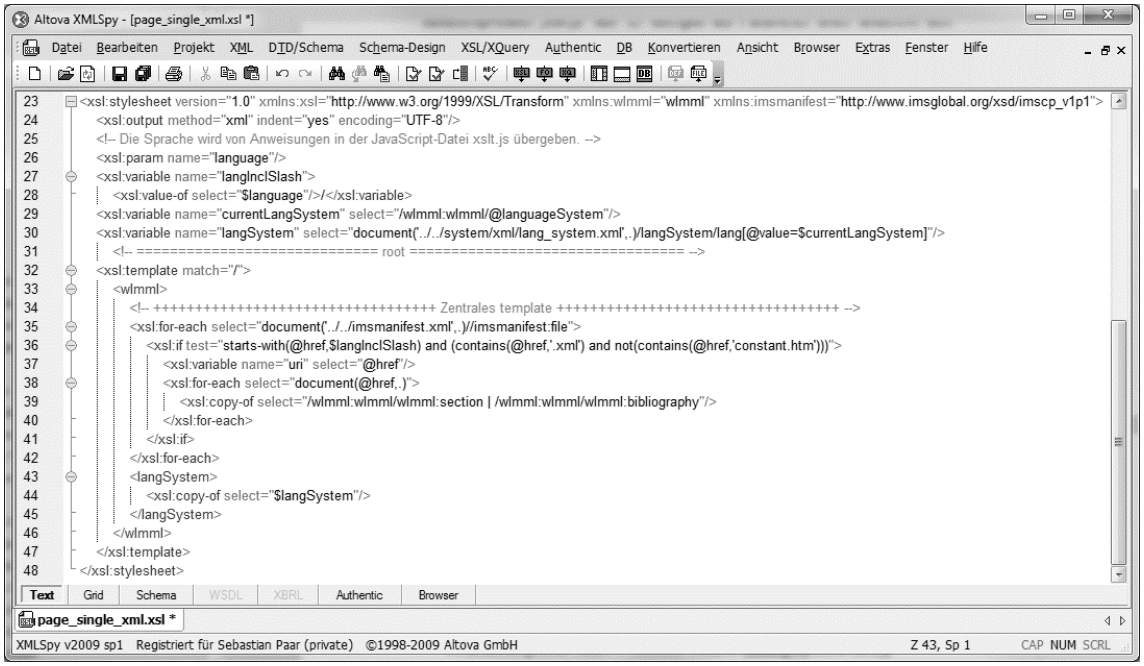

```
195  function transformXmlXsltMoz(xmlFilename,xml,xsltUrl,language,label,bibKey,elementName,popup,browser,namePopupWindow
     ,propertiesPopupWindow)
196  {
197  ...
198          // Wenn mit der Variablen xml eine URL übergeben wird, wird die XML-Datei von der URL geladen.
199      typeOfXml = typeof xml;
200      if (typeOfXml == "string")
201      {
202          // Ein XML-Dokument wird geladen.
203          myXMLHTTPRequest = new XMLHttpRequest();
204          try {
205              myXMLHTTPRequest.open("GET", xml, false);
206              myXMLHTTPRequest.send(null);
207              xmlSource = myXMLHTTPRequest.responseXML;
208          } catch (e) {
209                  alertIncompatibleBrowser();
210          }
211      }
212      else
213      {
214          // Falls es eine zweifache Transformation ist, liegt in der Variablen xml bereits das erste
     Transformationsergebnis als object.
215          xmlSource = xml;
216      }
217          // Das erste XSLT-Dokument wird geladen.
218      myXMLHTTPRequest = new XMLHttpRequest();
219      try {
220              myXMLHTTPRequest.open("GET", xsltUrl, false);
221              myXMLHTTPRequest.send(null);
222              xslStylesheet = myXMLHTTPRequest.responseXML;
223      } catch (e) {
224              alertIncompatibleBrowser();
225      }
226      xsltProc  = new XSLTProcessor();
227          // Die XSLT-Datei wird in den vorher erstellten XSLT-Processor importiert.
228      xsltProc.importStylesheet(xslStylesheet);
229          // Die Parameter für die XSLT-Datei werden gesetzt.
230      try {
231  ...
232          xsltProc.setParameter(null, "language", language);
233  ... } catch (e) {
234              alertIncompatibleBrowser();
235      }
236      transformationResult = xsltProc.transformToDocument(xmlSource);
237          // Der XSLT-Prozessor des Browsers wird von Styleheets und Parametern befreit.
238      xsltProc.reset();
239          // Das Transformationsergebnis wird an die aufrufende Stelle zurückgegeben.
240      return transformationResult;
241  }
```

Listing 86: Ausschnitt aus der Funktion „transformXmlXsltMoz()“ aus der JavaScript-Datei „xslt.js“ (Transformation mit Hilfe des XSLT-Prozessors im Browser)

Für die erste Transformation ist der Inhalt der übergebenen WLMML-Datei (Parameterwert der Variablen „xml“) so gut wie nicht von Bedeutung, da er von der XSLT-Datei „page_single_xml.xsl“ in der Transformation überschrieben wird (Listing 87). Nur der „languageSystem“-Attributwert des „wlmml“-Elementes wird in der XSLT-Variablen „currentLangSystem“ abgelegt (Listing 87 Zeile 29). Er ist notwendig, um die sprachspezifischen Systemausdrücke aus der XML-Datei „lang_system.xml“ holen zu können (Listing 87 Zeile 30). Die Ausdrücke werden in das erste

Transformationsergebnis kopiert (Listing 87 Zeile 44) und für die nächste Transformation zur Verfügung gestellt.

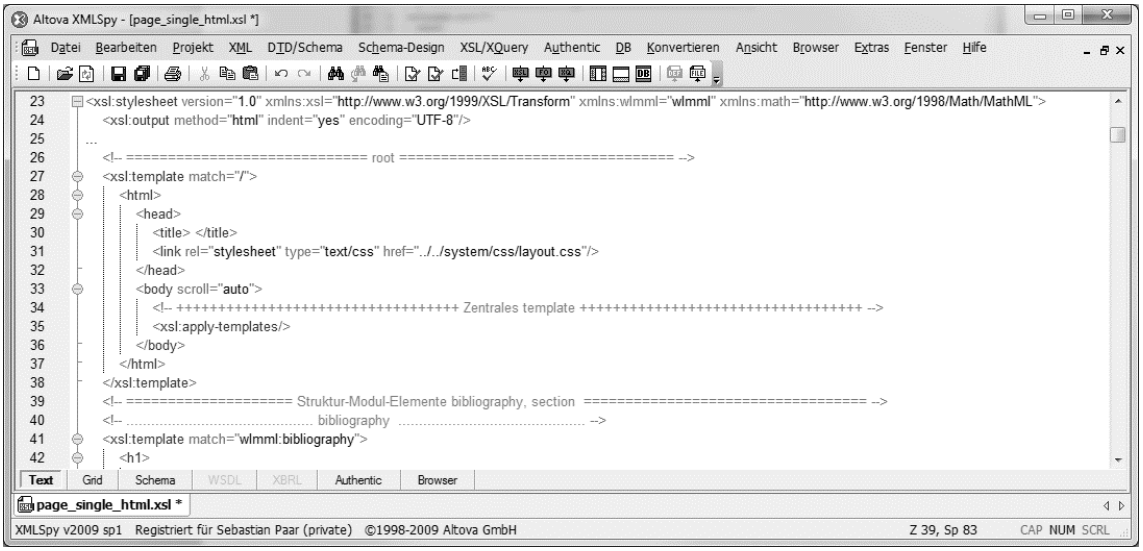
Listing 87 zeigt in den Zeilen 35 bis 36 wie alle „file“-Elemente in den Ressourcen der Manifest-Datei gesucht (Zeile 35, XSLT-Funktion „document()“) und nur die sprachspezifischen XML-Dateien (ohne die „constant.htm“) zur weiteren Verarbeitung ausgewählt werden sollen (Zeile 36, XSLT-Parameter „language“, Übergabe siehe Listing 86 Zeile 232). Die Anweisung in der Zeile 38 (Listing 87) öffnet alle ausgewählten Dateien, deren Inhalte dann die Anweisung in Zeile 39 (Listing 87) in das Transformationsergebnis kopiert. Damit stehen die Inhalte aller betroffenen WLMML-Dateien für die nächste Transformation zur Verfügung.



Listing 87: Ausschnitt aus der XSLT-Datei „page_single_xml.xsl“ (erste Transformation)

Zum Abschluss der Verarbeitung in Listing 86 wird der XSLT-Prozessor des Browsers von Styleheets und Parametern befreit (Listing 86 Zeile 238) und das Transformationsergebnis an die aufrufende Stelle zurückgegeben (Listing 86 Zeile 240). Die aufrufende Stelle wiederum löst die zweite Transformation aus und führt sie wie die erste mit Hilfe der Anweisungen in der Funktion „transformXmlXsltMoz()“ durch. Allerdings setzt sie hierbei nicht die XSLT-Datei „page_single_xml.xsl“ sondern „page_single_html.xsl“ ein. Diese zweite XSLT-Datei erstellt ein HTML-Grundgerüst (Listing 88 Zeile 28 bis 37), arbeitet über „template“-Anweisungen die WLMML-Elemente im ersten Transformationsergebnis ab (Listing 88 Zeile 35, 41) und gliedert das Ergebnis in das Grundgerüst ein. Sie ähnelt damit der zentralen

XSLT-Datei „main.xsl“ (siehe Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei).



Listing 88: Ausschnitt aus der XSLT-Datei „page_single_html.xsl“ (zweite Transformation)

Als Endergebnis entsteht ein einzelnes im Browser darstellbares HTML-Dokument, das die Inhalte aller WLMML-Dateien der jeweiligen Sprache aufweist.

4.3.2.4.2.3 Suche in verschiedenen Online-Diensten (nur mit Internet-Zugang möglich)

In der Hilfsleiste befindet sich außerdem ein Formular zur Suche in verschiedenen Online-Diensten (Abbildung 90). Über sein Eingabefeld kann der Suchbegriff eingegeben und über die Auswahl des Such-Buttons oder über die Betätigung der Enter-Taste die Suche gestartet werden. Die verschiedenen Online-Dienste lassen sich über das Aufklappmenü auswählen. Da sie in verschiedenen Sprachen weltweit angeboten werden, wird über die XSLT-Variable „currentLangSystem“ die Standard-Suchsprache festgelegt (Listing 89 Zeile 169). Die Variable bezieht ihren Inhalt aus dem „languageSystem“-Attributwert des „wlmml“-Elementes im abgearbeiteten WLMML-Dokument. Startet der Anwender die Suche, wird die Online-Informationsquelle in der Standard-Suchsprache in einem neuen Browser-Fenster geöffnet (Listing 89 Zeile 161 bis 166) und durchsucht. Das Suchergebnis erscheint in der für den Online-Dienst üblichen Weise.

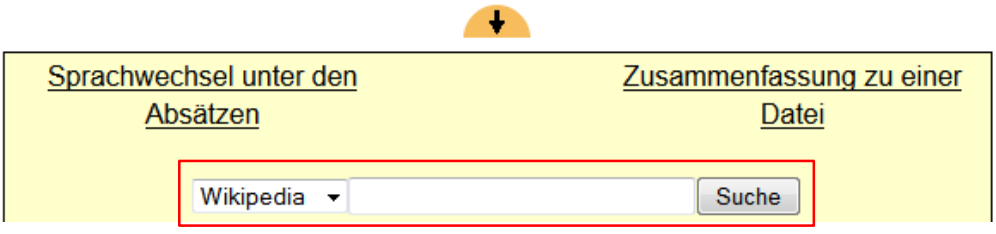
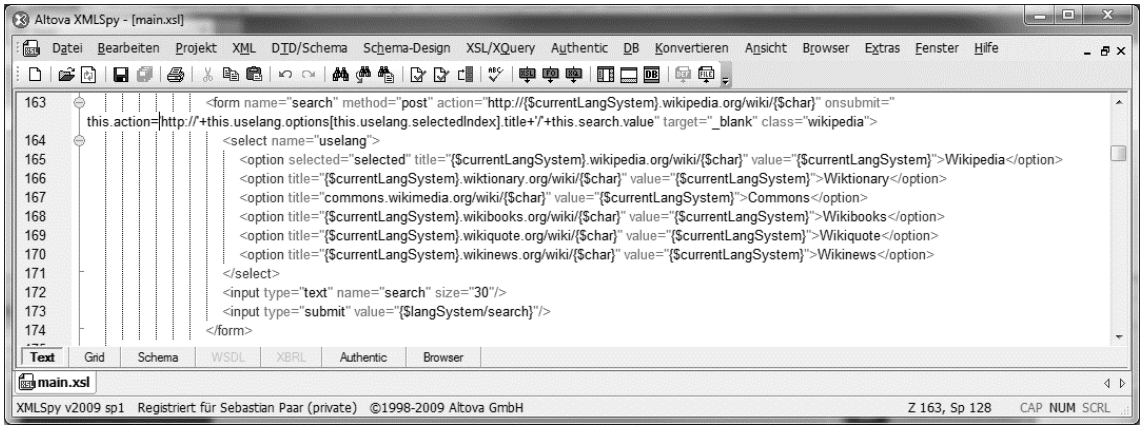


Abbildung 90: Hilfsleiste am unteren Browserrand mit der Funktionalität der Suche in verschiedenen Online-Diensten (roter Rahmen)



Listing 89: Ausschnitt aus der zentralen XSLT-Datei „main.xsl“ (Erstellung des Formulars zur Suche in verschiedenen Online-Diensten

Zur Erleichterung des Suchvorganges muss der Suchbegriff nicht in das Eingabefeld eingegeben werden, sondern es reicht aus das Wort (bzw. Text) im Fließtext zu markieren. Der Text wird vollautomatisch in das Suchfeld eingetragen und lässt sich anschließend über Betätigung der Enter-Taste an den Online-Dienst verschicken. Diese Funktonalität ermöglichen die beiden JavaScript-Funktionen „searchOnlineHihglightedString()“ und „TasteGedrueckt()“ in der JavaScript-Datei „layer_bottom.js“. Anweisungen in der Funktion „searchOnlineHihglightedString()“ halten fest, ob eine Taste gedrückt wurde und Text markiert war. War dies der Fall, wird an die Funktion „TasteGedrueckt()“ übergeben. Wenn sie feststellt, dass die Enter-Taste gedrückt wurde, verschickt sie den Suchbegriff über das Formular an den jeweiligen vom Anwender ausgewählten Online-Dienst.

Der Quelltext der JavaScript-Funktion „TasteGedrueckt()“ in der Datei „layer_bottom.js“ stammt von SELFHTML-EVENT (2007) und wurde angepasst. Weitere Hinweise und Anregungen zur Suche in Wikipedia kamen von WIKIPEDIA-BOOKMARKLETS (2007) (siehe Quelltext der JavaScript-Datei „layer_bottom.js“) und WIKTIONARY-SUCHFELD (2014) (siehe Quelltext der XSLT-Datei „main.xsl“).

4.3.2.4.3 Automatische Erstellung von Verzeichnissen

Nach einem Aufruf des Hyperlinks „Verzeichnis/Inhaltssuche“ in der Navigationsleiste (bzw. beim Laden über eine Navigationsleiste aus einer SCORM-Umgebung heraus) wird eine automatische Zusammenstellung von verschiedenen Verzeichnissen und eine Volltext-Suche angeboten (Abbildung 91, siehe auch QUEDNAU et al. 2007, S. 6). Wählt der Anwender z. B. den Verzeichnis-Typ „Abbildungen“ aus, erscheint das Abbildungs-Verzeichnis im darunter liegenden Ergebnisbereich (Abbildung 92, rechts unten). Dieser Vorgang soll in diesem Kapitel am Beispiel eines

Abbildungsverzeichnisses näher beleuchtet werden. Die automatische Zusammenstellung der anderen Verzeichnisse verläuft ähnlich.

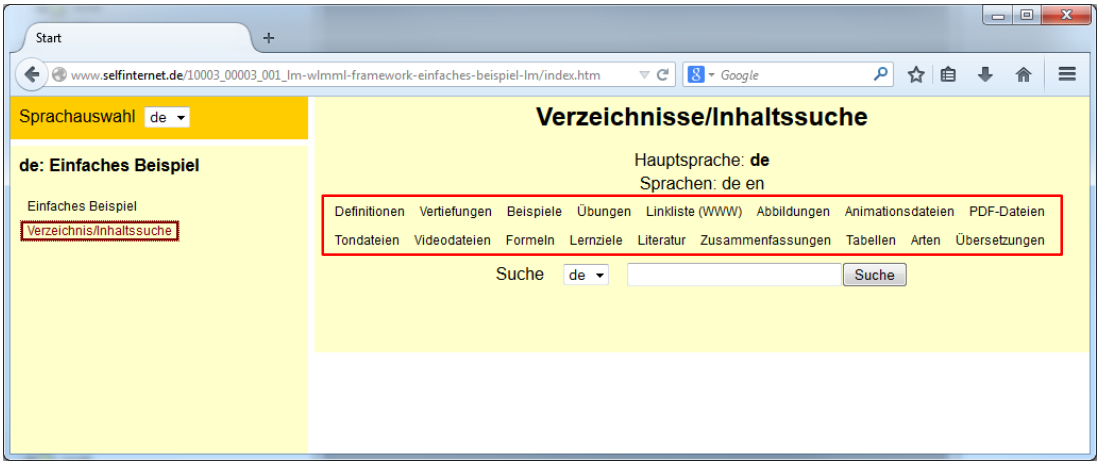
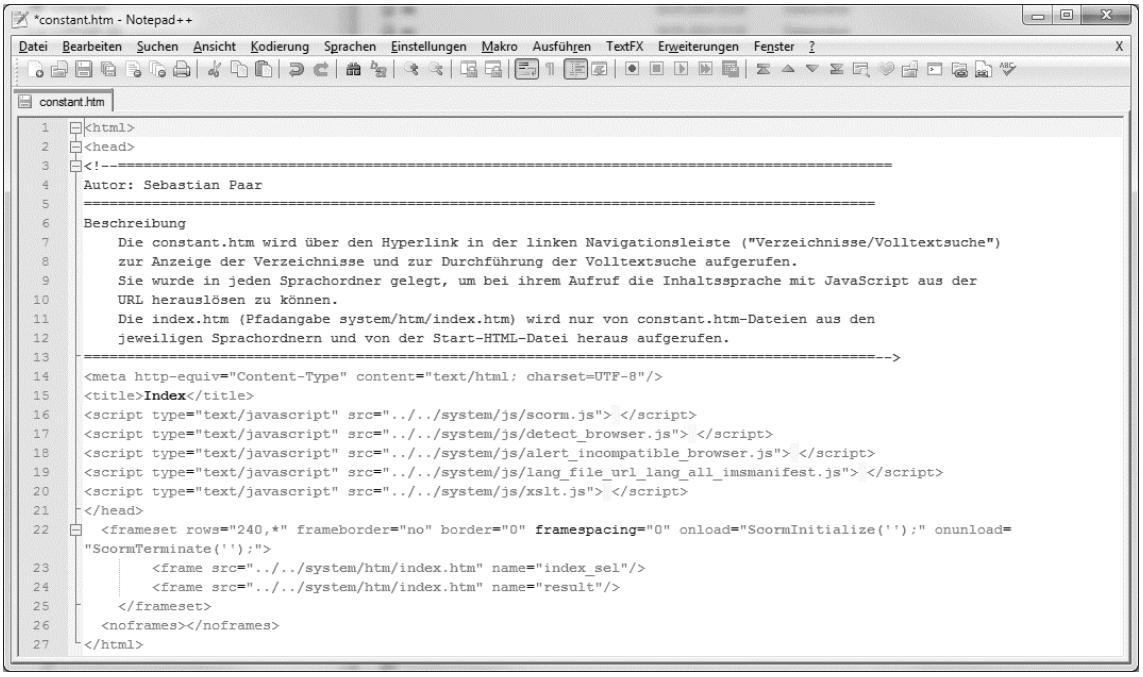


Abbildung 91: Aufruf des Hyperlinks „Verzeichnis/Inhaltssuche“ in der Navigationsleiste (Laden der HTML-Datei „constant.htm“, rechts Verzeichnisse in roten Rahmen, Darstellung im Browser „Mozilla Firefox 29.0.1“)

Abbildung 92: Erstelltes Abbildungsverzeichnis nach Auswahl des Verzeichnis-Typs „Abbildungen“ („online“-Aufruf von einem Web-Server, siehe Adressleiste, Darstellung im Browser „Mozilla Firefox 29.0.1“)

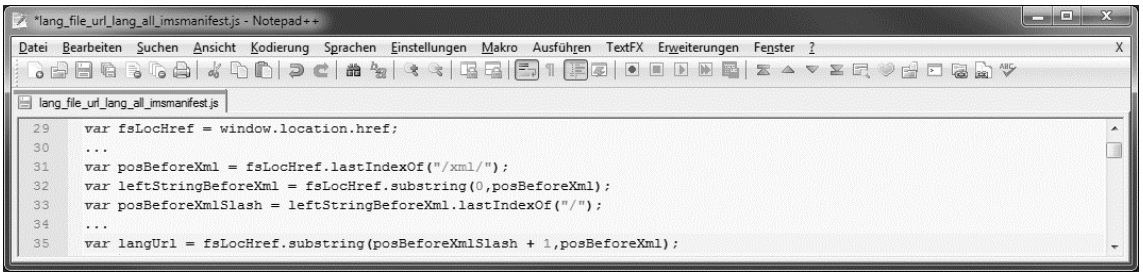
Zunächst wird über die Auswahl des Hyperlinks „Verzeichnis/Inhaltssuche“ die HTML-Datei „constant.htm“ aus dem jeweiligen Sprachordner geladen (Pfadangabe z. B. „de/xml/constant.htm“). Im „head“-Bereich dieser HTML-Datei werden verschiedene JavaScript-Dateien geladen (Listing 90 Zeile 16 bis 20). Die auch für diesen Anwendungsbereich wichtigen Anweisungen in den JavaScript-Dateien „scorm.js“ (siehe auch Funktionen „ScormInitialize()“ und „ScormTerminate()“, Listing 90 Zeile 22), „detect_browser.js“, „alert_incompatible_browser.js“ und „xslt.js“ wurden bereits in den Kapiteln 4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-

Datei und 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei beschrieben.



Listing 90: Quelltext der HTML-Datei „constant.htm“

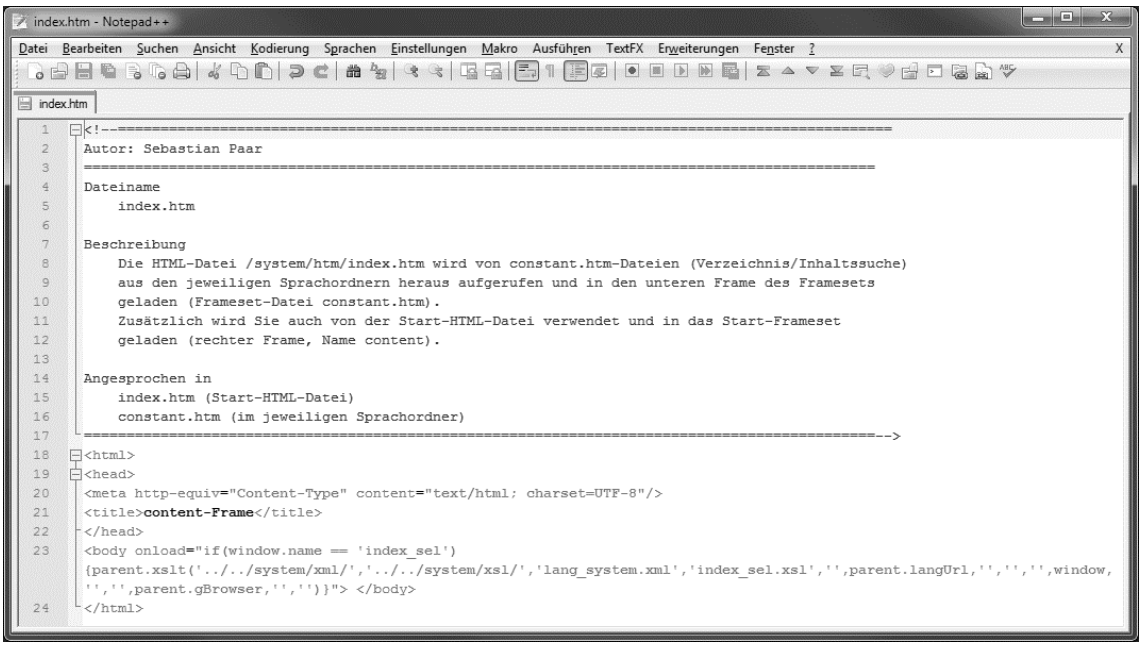
Für die Erstellung der Verzeichnisse muss die Inhaltssprache bekannt sein (auch z. B. für die Anzeige der sprachspezifischen Verzeichnis-Ausdrücke). Befehle in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“ (Listing 91) lesen dafür aus der URL (Listing 91 Zeile 29) der HTML-Datei „constant.htm“ die Inhaltssprache aus und stellen sie über die globale Variable „langUrl“ (Listing 91 Zeile 35) für die weitere Verwendung zur Verfügung. Die JavaScript-Datei „lang_file_url_lang_all_-imsmanifest.js“ wird neben den bereits erwähnten JavaScript-Dateien beim Aufruf der HTML-Datei „constant.htm“ ebenfalls mit geladen (Listing 90 Zeile 19). Da sich in jedem Sprachordner eine HTML-Datei „constant.htm“ befindet (siehe Kapitel 4.3.1 Verzeichnisstruktur), kann bei ihrem Aufruf aus ihrer URL jederzeit die jeweilige Inhaltssprache ermittelt werden.



Listing 91: Ausschnitt aus der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“ (Auslesen der Inhaltssprache aus der URL)

Nach dem „head“-Bereich in dieser HTML-Datei wird ein Frameset aus zwei Frames zusammengesetzt (Listing 90 Zeile 22 bis 25). In beide Frames wird die HTML-Datei „index.htm“ geladen (Pfadangabe zur Datei „system/htm/index.htm“, Listing 90 Zeile 23 und Zeile 24). Der obere Frame, der die Verzeichnisse und die Inhaltssuche aufnehmen soll, trägt den Namen „index_sel“ (Listing 90 Zeile 23, siehe auch Abbildung 92 rechts oben). Der untere Frame erhält den Namen „result“ und soll das Ergebnis der Verzeichnisauswahl bzw. der Inhaltssuche aufnehmen (Listing 90 Zeile 24, siehe auch Abbildung 92 rechts unten).

Die HTML-Datei „index.htm“ (Pfadangabe zur Datei „system/htm/index.htm“) weist keinen Textinhalt auf. Sie dient als Platzhalter für die eigentlichen Inhalte. Nur wenn sie in den Frame mit dem Namen „index_sel“ geladen wird (Listing 92 Zeile 23), ruft sie die JavaScript-Funktion „xslt()“ (in der JavaScript-Datei „xslt.js“) auf (Listing 92 Zeile 23, „onload“-Ereignis im „body“-Element). Als Parameter werden unter anderem die XML-Datei „lang_system.xml“, die XSLT-Datei „index_sel.xsl“ und die Inhaltssprache (Inhalt der Variablen „langUrl“) übergeben. Befehle in dieser JavaScript-Funktion „xslt()“ laden die beiden Dateien und führen eine XSLT-Transformation (inklusive der Inhaltssprache als Parameter) durch (siehe auch Kapitel 4.3.2.4.2.2 Zusammenfassung der Lerninhalte in einem Dokument).

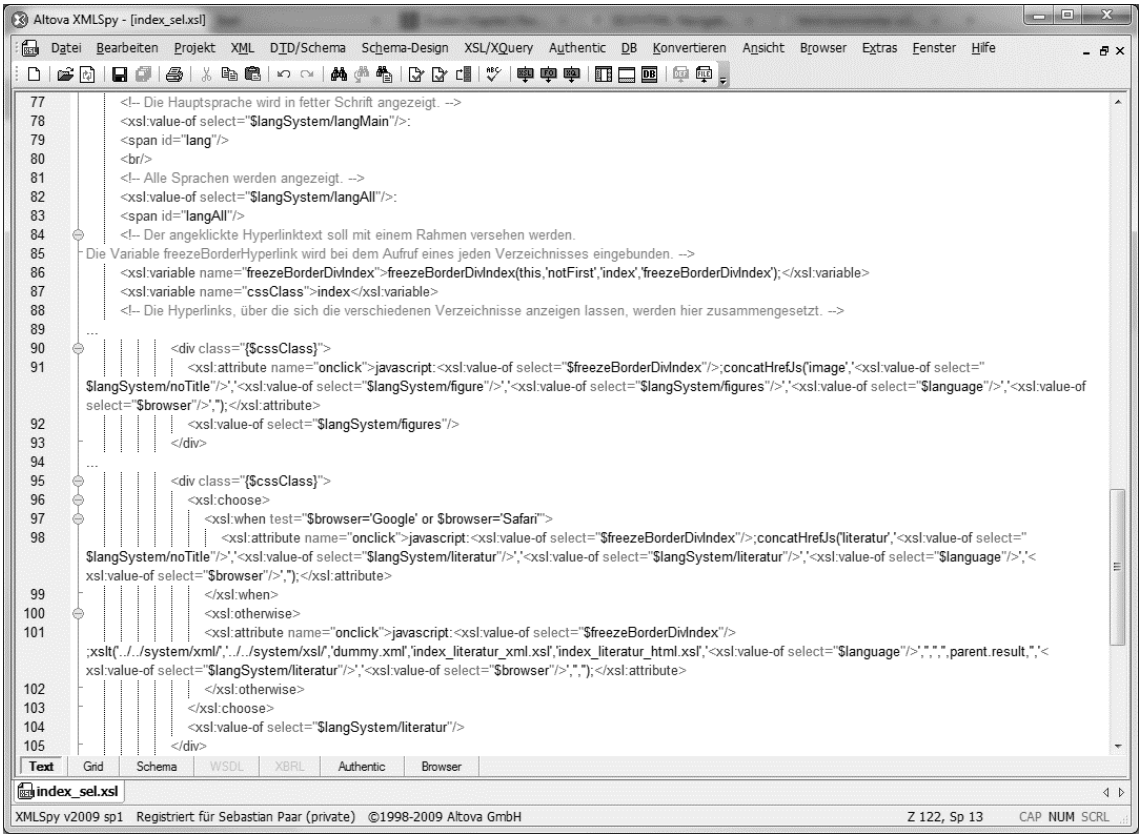


Listing 92: Quelltext der HTML-Seite „index.htm“ (Pfadangabe zur Datei „system/htm/index.htm“), Aufruf der JavaScript-Funktion „xslt()“ (wenn die HTML-Datei in den Frame mit dem Namen „index_sel“ geladen wird)

Die Anweisungen in der XSLT-Datei „index_sel.xsl“ spielen dabei eine entscheidende Rolle (Listing 93). Sie leiten die Anzeige der verschiedenen

Verzeichnisse (und der Volltext-Suche, siehe Kapitel 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten) ein (Ergebnis siehe Abbildung 92 Frame rechts oben). Dazu erzeugen sie ein HTML-Dokument, laden darüber verschiedene JavaScript-Dateien (u. a. „hyperlink_border.js“, „search_index_lang.js“), rufen JavaScript-Funktionen auf und binden in das HTML-Dokument das Verarbeitungsergebnis ihrer XSLT-Templates ein (Listing 93, siehe auch Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei). Die systemspezifischen Sprachausdrücke werden aus der XML-Datei „lang_system.xml“ geholt. Sie spiegeln sich z. B. in den Textstellen „Verzeichnisse/Inhaltssuche“, „Hauptsprache“ (Listing 93 Zeile 78), „Sprachen“ (Listing 93 Zeile 82) oder den sprachspezifischen Verzeichnisnamen (z. B. „Abbildungen“ Listing 93 Zeile 92 oder „Literatur“ Listing 93 Zeile 104) wieder (siehe auch Abbildung 92 Frame rechts oben). Zusätzlich setzen Anweisungen in der XSLT-Datei „index_sel.xsl“ mit Hilfe der Anweisungen in der JavaScript-Datei „search_index_lang.js“ und „hyperlink_border.js“ folgende Inhalte bzw. Markierung (siehe z. B. Ergebnis in Abbildung 92 Frame rechts oben):

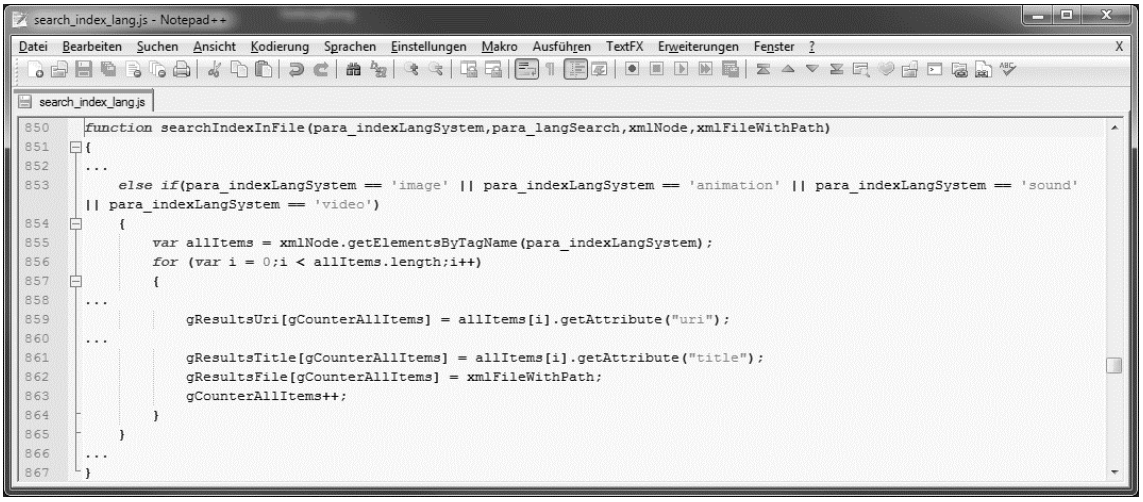
- die derzeitige Inhaltssprache (Hauptsprache) in den oberen Frame der Verzeichnisübersicht (übernommen aus der Variablen „langUrl“ in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“ (siehe Listing 91 Zeile 35), geschrieben werden sie in das „span“-HTML-Element mit dem „id“-Attributwert „lang“ (Listing 93 Zeile 79) durch Anweisungen in der JavaScript-Funktion „langMain()“ in der JavaScript-Datei „search_index_lang.js“)
- die Sprachen aus den Metadaten der jeweiligen „organization“-Elemente in der Manifest-Datei „imsmanifest.xml“ (übernommen aus der Array-Variablen „allLang“ in der JavaScript-Datei „lang_file_url_lang_all_imsmanifest.js“, geschrieben werden sie in das „span“-HTML-Element mit dem „id“-Attributwert „langAll“ (Listing 93 Zeile 83) durch Anweisungen in der JavaScript-Funktion „langAll()“ in der JavaScript-Datei „search_index_lang.js“)
- einen Rahmen um das aktuell ausgewählte „div“-HTML-Element des jeweiligen Verzeichnisses (Anweisungen in der JavaScript-Funktion „freezeBorderDivIndex()“ in der JavaScript-Datei „hyperlink_border.js“, Listing 93, Deklaration der XSLT-Variablen „freezeBorderDivIndex“ in Zeile 86, Verwendung der XSLT-Variablen in Zeilen 91, 98 und 101)



Listing 93: Ausschnitt aus der XSLT-Datei „index_sel.xsl“ (v. a. Einleitung der Anzeige der verschiedenen Verzeichnisse)

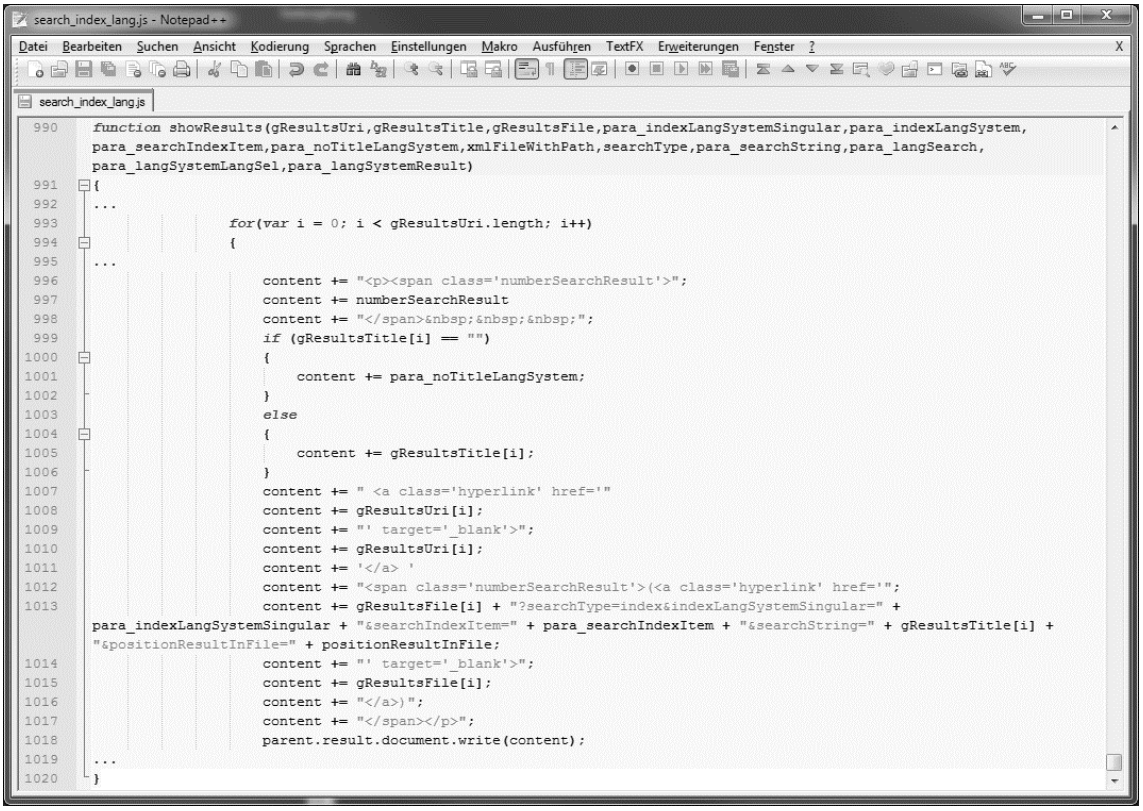
Wählt der Anwender nach der Anzeige der sprachspezifischen Verzeichnisausdrücke im Browser den Ausdruck für „Abbildungen“ aus (Listing 93 Zeile 91, „onclick“-Ereignis des „div“-HTML-Elementes), wird zunächst die JavaScript-Funktion „freezeBorderDivIndex()“ aufgerufen. Deren Anweisungen versehen den sprachspezifischen Ausdruck „Abbildungen“ mit einem Rahmen. Anschließend erfolgt der Aufruf der JavaScript-Funktion „concatHrefJs()“ (in der JavaScript-Datei „search_index_lang.js“) (Listing 93 Zeile 91). Befehle in dieser Funktion veranlassen das Öffnen aller sprachspezifischen XML-Dateien (Pfadangaben aufgeführt in der Manifest-Datei „imsmanifest.xml“) und übergeben den Inhalt dieser XML-Dateien einzeln zur Durchsuchung nach dem ausgewählten Elementnamen (Wert des Parameters „para_indexLangSystem“, Listing 94 Zeile 850) an die JavaScript-Funktion „searchIndexInFile()“. Alle Elemente in einer XML-Datei, die in ihrem Elementnamen (hier „image“) mit dem übergebenen Wert des Parameters „para_-indexLangSystem“ übereinstimmen (Treffer), werden zunächst in einer Array-Variablen „allItems“ abgelegt (Listing 94 Zeile 855). Anschließend übergeben Anweisungen den „uri“- und „title“-Attributwert dieser Elemente, sowie den Dateinamen inklusive Pfadangabe zur späteren Ausgabe am Bildschirm an die globalen Array-Variablen „gResultsUri“, „gResultsTitle“ und „gResultsFile“, Listing 94 Zeile 859, 861 und 862). Handelt es sich bei dem übergebenen Wert des Parameters

„para_indexLangSystem“ um „definition“, „detail“, „example“, „exercise“, „formula“, „learningObjective“, „summary“, „taxon“, „translation“ oder „table“, wird aus der JavaScript-Funktion „searchIndexInFile()“ heraus die Funktion „searchIndexWithinNode()“ aufgerufen. Ihre Anweisungen durchsuchen rekursiv alle Knoten nach dem Parameterwert und stellen das Ergebnis ebenfalls zur späteren Ausgabe am Bildschirm in den globalen Array-Variablen „gResultsTitle“ und „gResultsFile“ zur Verfügung. Der Quelltext dieser JavaScript-Funktion wurde in Anlehnung an MINTERT & KÜHNEL (2000, S. 221) erstellt.



Listing 94: Ausschnitt aus der Funktion „searchIndexInFile()“ aus der JavaScript-Datei „search_index_lang.js“ (Durchsuchen der XML-Dateien nach dem übergebenen Elementnamen z. B. „image“)

Zur Anzeige des Gesamtergebnisses übergibt ein Befehl in der JavaScript-Funktion „concatHrefJs()“ die Treffer an die JavaScript-Funktion „showResults()“ (Listing 95). Befehle dieser Funktion listen die Treffer im Frame mit dem Namen „result“ auf (Listing 95 Zeile 1018). Als erstes wird der sprachspezifische Elementname mit der Anzahl der Treffer als Überschrift wiedergegeben. Anschließend erscheinen die durchnummerierten Treffer (Listing 95 Zeile 997) z. B. bei der Suche nach Abbildungen in Form eines Abbildungsverzeichnisses mit dem „title“-Attributwert des gefundenen Elementes (Listing 95 Zeile 1005), dem Hyperlink zum gefundenen Element (Listing 95 Zeile 1007 bis 1011) und dem Hyperlink zur XML-Datei (Listing 95 Zeile 1012 bis 1016), in der sich die Abbildung befindet (Gesamtergebnis siehe Abbildung 92 rechts unten). Wurde kein „title“-Attributwert für den jeweiligen Treffer angegeben (Listing 95 Zeile 999), wird ein sprachspezifischer Standardwert aus der XML-Datei „lang_system.xml“ ausgegeben (Listing 95 Zeile 1001).



search_index_lang.js - Notepad++

Datei Bearbeiten Suchen Ansicht Kodierung Sprachen Einstellungen Makro Ausführen TextFX Erweiterungen Fenster ?

search_index_lang.js

```
990  function showResults(gResultsUri,gResultsTitle,gResultsFile,para_indexLangSystemSingular,para_indexLangSystem,
     para_searchIndexItem,para_noTitleLangSystem,xmlFileWithPath,searchType,para_searchString,para_langSearch,
     para_langSystemLangSel,para_langSystemResult)
991  {
992  ...
993                  for(var i = 0; i < gResultsUri.length; i++)
994                  {
995  ...
996                      content += "<p><span class='numberSearchResult'>";
997                      content += numberSearchResult
998                      content += "</span>   ";
999                      if (gResultsTitle[i] == "")
1000                     {
1001                         content += para_noTitleLangSystem;
1002                     }
1003                     else
1004                     {
1005                         content += gResultsTitle[i];
1006                     }
1007                     content += " <a class='hyperlink' href='"
1008                     content += gResultsUri[i];
1009                     content += "' target='_blank'>";
1010                     content += gResultsUri[i];
1011                     content += '</a> '
1012                     content += "<span class='numberSearchResult'>(<a class='hyperlink' href='";
1013                     content += gResultsFile[i] + "?searchType=index&indexLangSystemSingular=" +
     para_indexLangSystemSingular + "&searchIndexItem=" + para_searchIndexItem + "&searchString=" + gResultsTitle[i] +
     "&positionResultInFile=" + positionResultInFile;
1014                     content += "' target='_blank'>";
1015                     content += gResultsFile[i];
1016                     content += "</a>)";
1017                     content += "</span></p>";
1018                     parent.result.document.write(content);
1019 ...
1020 }
```

Listing 95: Ausschnitt aus der Funktion „showResults()“ aus der JavaScript-Datei „search_index_lang.js“ (Anzeige des Ergebnisses, z. B. Abbildungsverzeichnis)

Wählt der Anwender den Hyperlink zur XML-Datei aus, in dem sich die Abbildung (bzw. das gesuchte Element) befindet, heben Anweisungen in der JavaScript-Datei „highlight.js“ mit Hilfe der per URL übergebenen Parameter (Listing 95 Zeile 1013) das Element hervor (vergleiche 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen, siehe z. B. Abbildung 93).

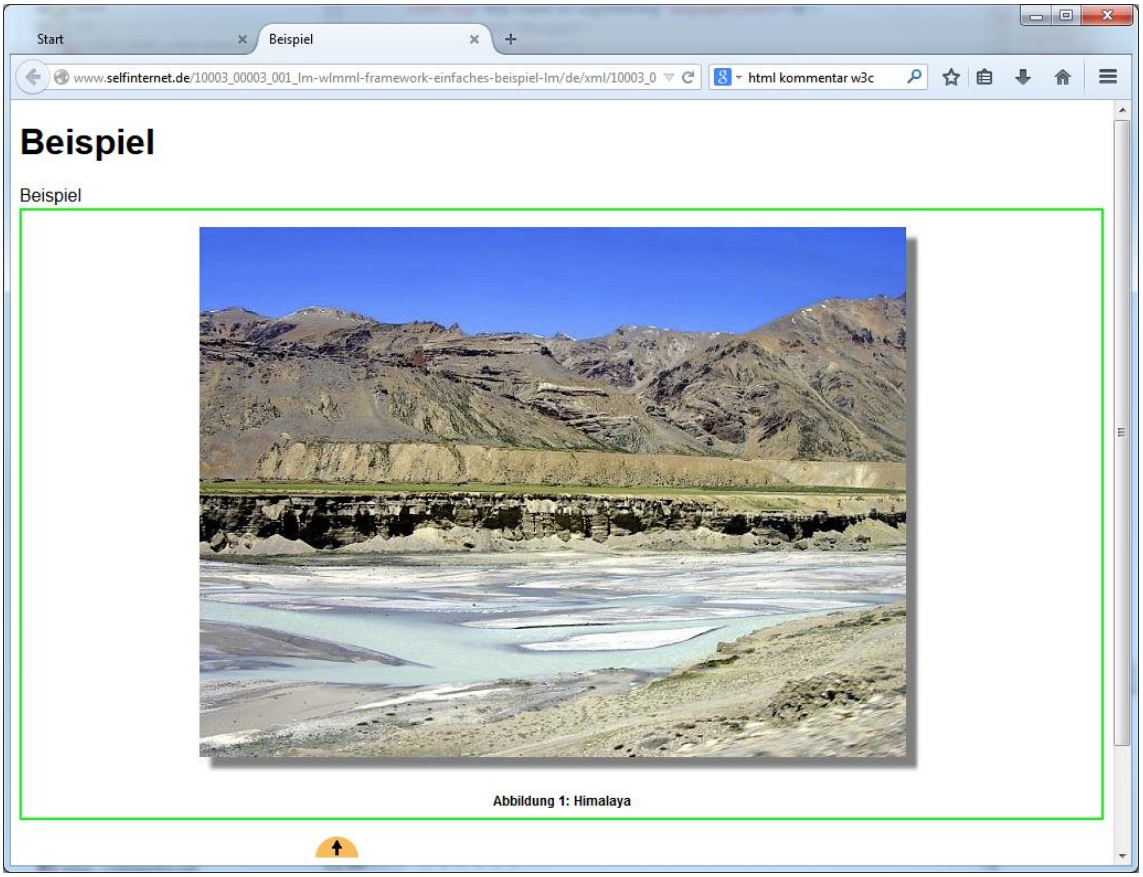
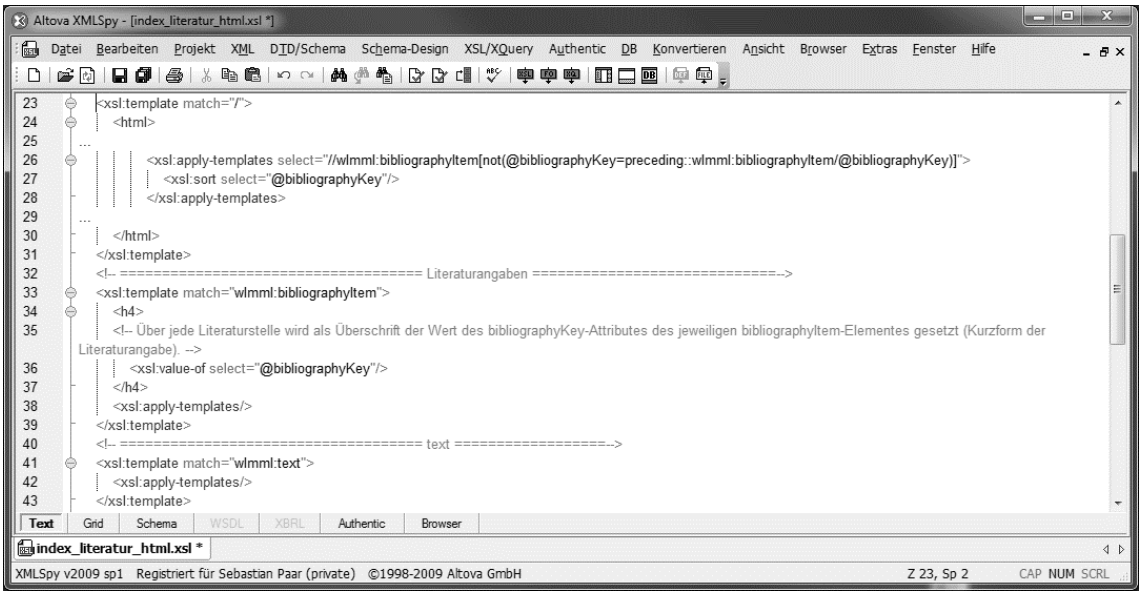


Abbildung 93: Aus einem Abbildungsverzeichnis heraus ausgewählte Trefferdatei (hervorgehobene „Abbildung“, „online“-Aufruf von einem Web-Server, siehe Adressleiste, Browser „Mozilla Firefox 29.0.1“)

Wählt der Anwender nach der Anzeige der sprachspezifischen Verzeichnisausdrücke im Browser den Ausdruck für „Literatur“ aus, wird für die Browser „Google Chrome“ und „Apple Safari“ die JavaScript-Funktion „concatHrefJs()“ (in der JavaScript-Datei „search_index_lang.js“) bzw. für andere Browser die JavaScript-Funktion „xslt()“ (in der JavaScript-Datei „xslt.js“) aufgerufen. Beide Funktionen setzen aus allen sprachspezifischen Literaturverzeichnisdateien das virtuelle Gesamt-Literaturverzeichnis zusammen. Dabei werden alle sprachspezifischen XML-Dateien (WLMML-Dateien) als Literaturverzeichnisdateien berücksichtigt, deren „href“-Attributwerte (in „file“-Elementen) in den Ressourcen der Manifestdatei „imsmanifest.xml“ den Text „literatur“ beinhalten. Die JavaScript-Funktion „concatHrefJs()“ arbeitet in diesem Zusammenhang mit einer einfachen Transformation mit der XSLT-Datei „index_literatur_html.xsl“, die JavaScript-Funktion „xslt()“ mit einer „zweifachen“ XSLT-Transformation mit den XSLT-Dateien „index_literatur_xml.xsl“ und „index_literatur_html.xsl“ (vergleiche Kapitel 4.3.2.4.1.2 Verarbeitungsvorgang bei Aufruf einer WLMML-Datei, siehe Thema Literaturverweis am Endes des Kapitels). Bei der „zweifachen“ XSLT-Transformation listet das Ergebnis der ersten Transformation alle Literaturstellen auf (Anweisungen in der

XSLT-Datei „index_literatur_xml.xsl“), die zweite Transformation erzeugt aus dem Ergebnis der ersten Transformation das virtuelle Gesamt-Literaturverzeichnis in HTML (Anweisungen in der XSLT-Datei „index_literatur_html.xsl“). Listing 96 zeigt einen Ausschnitt aus der XSLT-Datei „index_literatur_html.xsl“. Mit der XPath-Angabe "//wlmml:bibliographyItem[not(@bibliographyKey=preceding::wlmml:-bibliographyItem/@bibliographyKey)]" wird sichergestellt, dass keine doppelten Literaturstellen in das generierte Literaturverzeichnis aufgenommen werden (Listing 96 Zeile 26). Die XSLT-Anweisung in Zeile 27 (Listing 96) sortiert die Ergebnisse. Die beiden Templates in Zeile 33 und 41 stellen die einzelnen Literaturstellen mit ihrem Text-Inhalt zusammen.



Listing 96: Ausschnitt aus der XSLT-Datei „index_literatur_html.xsl“ (Abbildung des virtuellen Gesamt-Literaturverzeichnisses in HTML)

4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten

Die Volltext-Suche soll es ermöglichen einen bestimmten Text in einer Vielzahl von WLMML-Dateien zu finden (siehe auch WIKIPEDIA-VOLLTEXTRECHERCHE 2013). Nach dem Aufruf des Hyperlinks mit dem Text „Verzeichnis/Inhaltssuche“ in der Navigationsleiste (bzw. über eine Navi-gationsleiste aus einer SCORM-Umgebung heraus) wird ein Formular für eine Volltext-Suche angeboten (Abbildung 94). Trägt der Anwender einen Suchbegriff ein und drückt auf die Schaltfläche mit dem sprachspezifischen Ausdruck „Suche“ bzw. betätigt die „Enter“-Taste, erscheint das Suchergebnis im darunter liegenden Ergebnisbereich (im Frame „result“, Abbildung 95 rechts unten, siehe auch QUEDNAU et al. 2007, S. 6). Dieser Vorgang soll in diesem Kapitel näher beleuchtet werden. Die Verarbeitung verläuft sehr ähnlich wie bei der automatischen Erstellung von Verzeichnissen (siehe Kapitel 4.3.2.4.3 Automatische Erstellung von Verzeichnissen).

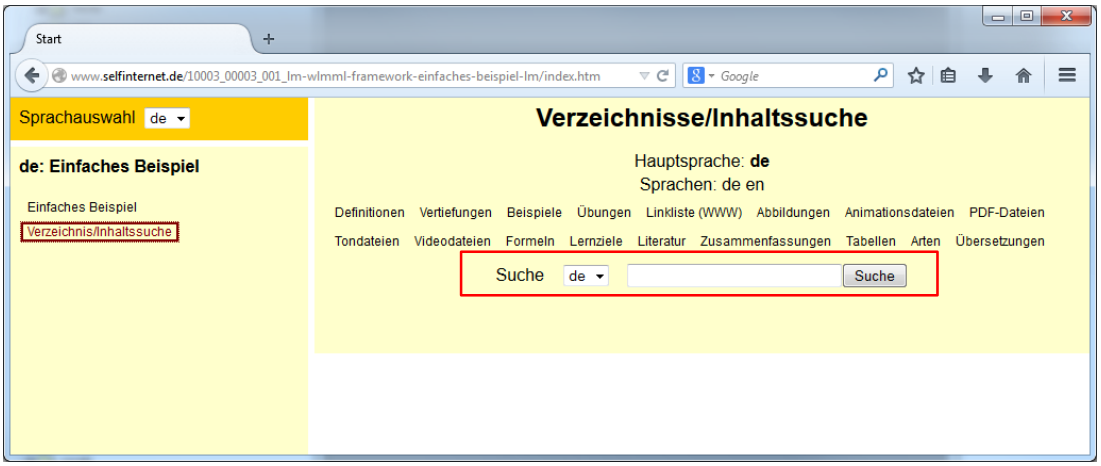


Abbildung 94: Aufruf des Hyperlinks „Verzeichnis/Inhaltssuche“ in der linken Navigationsleiste (Laden der HTML-Datei „constant.htm“, rechts Formular für Volltext-Suche in roten Rahmen, Darstellung im Browser „Mozilla Firefox 29.0.1“)

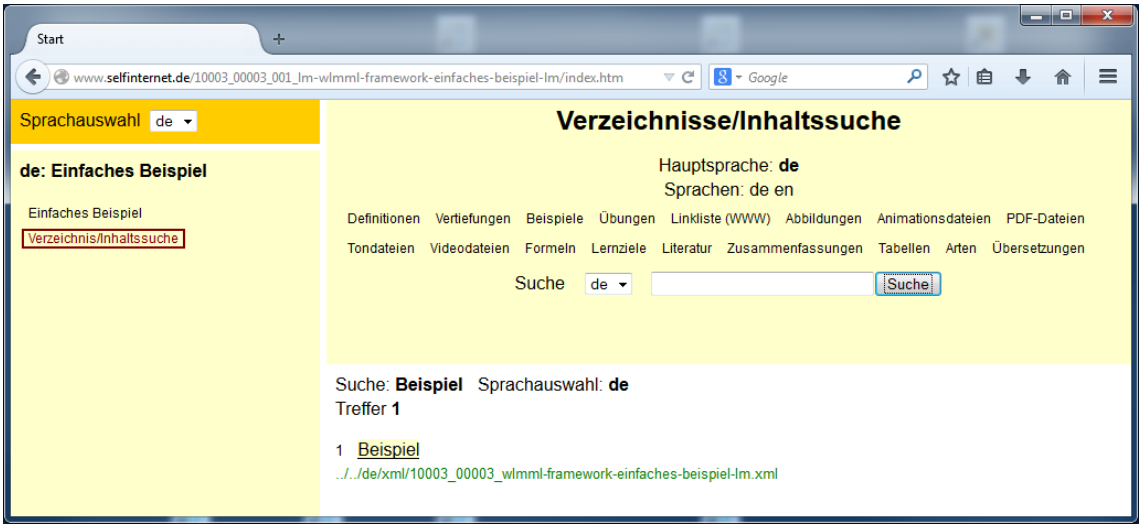
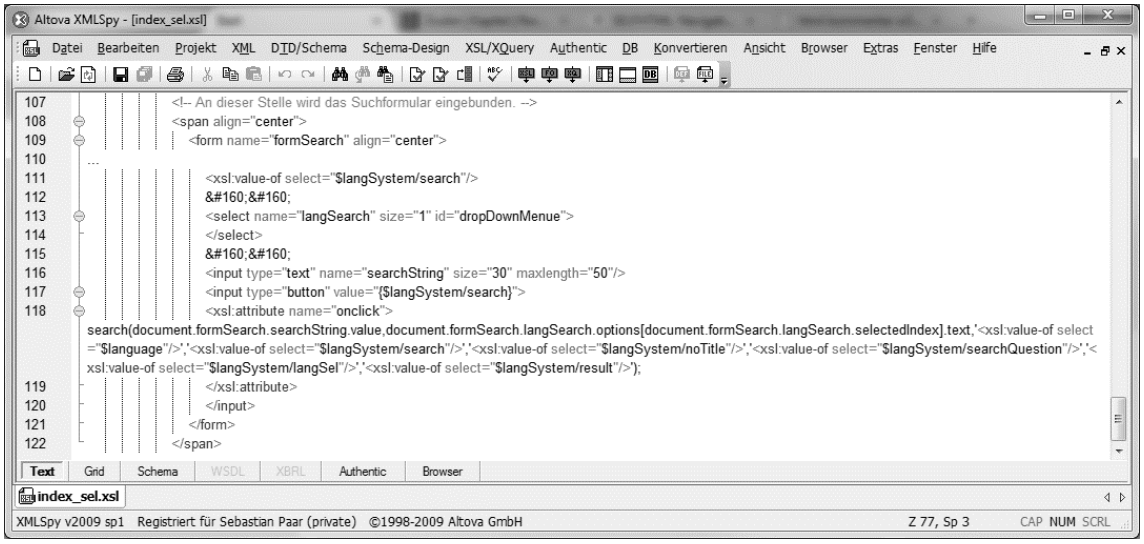


Abbildung 95: Suchergebnis einer Volltextsuche („online“-Aufruf von einem Web-Server, siehe Adressleiste, Darstellung im Browser „Mozilla Firefox 29.0.1“)

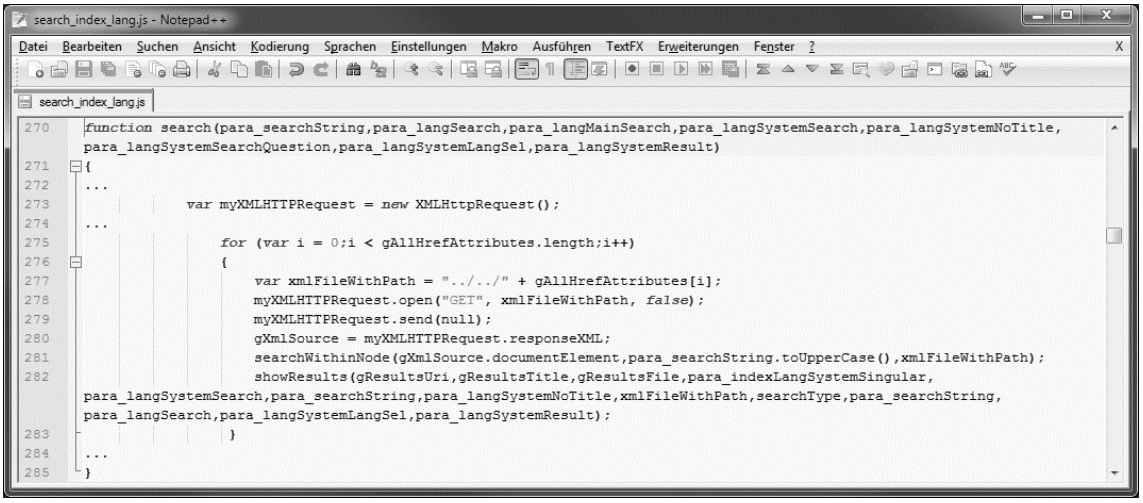
Der Ablauf ist bis einschließlich des Ladens der HTML-Datei „constant.htm“ (Frameset) aus dem jeweiligen Sprachordner völlig identisch (Ergebnis siehe Abbildung 94). Dabei binden Anweisungen in der XSLT-Datei „index_sel.xsl“ das Formular (Listing 97 Zeile 109) für die Volltextsuche ein (Listing 97). Sprachspezifische Systemausdrücke werden aus der XML-Datei „lang_system.xml“ geholt (Listing 97 Zeile 111, 117 und 118). Die Sprachen für das Aufklappmenü (Listing 97 Zeile 113 und 114), die bei der Volltextsuche ausgewählt werden können, werden aus der Array-Variablen „allLang“ (siehe JavaScript-Datei „lang_file_url-_lang_all_imsmanifest.js“) übernommen und über Anweisungen in der JavaScript-Funktion „buildLang()“ (in der JavaScript-Datei „search_index_lang.js“) als Werte des

Aufklappmenüs gesetzt. Das Eingabefeld für den Suchbegriff trägt den Namen „searchString“ (Listing 97 Zeile 116).



Listing 97: Ausschnitt aus der XSLT-Datei „index_sel.xsl“ (Einbinden des Formulars zur Volltextsuche)

Erst bei Auslösen der Suche sind gewisse Unterschiede zur automatischen Erstellung von Verzeichnissen feststellbar. Startet der Anwender die Suche über ein Drücken auf die Schaltfläche mit dem sprachspezifischen Ausdruck „Suche“ bzw. Betätigung der „Enter“-Taste, ruft er die JavaScript-Funktion „search()“ in der JavaScript-Datei „search_index_lang.js“ mit mehreren Parametern auf (Betätigung der Schaltfläche „Suche“: Listing 97 Zeile 118, „onclick“-Ereignis des HTML-Elementes „button“). Einer dieser Parameter ist der Suchbegriff („document.formSearch.searchString.value“, Listing 97 Zeile 118), ein anderer die Suchsprache. Befehle in dieser JavaScript-Funktion „search()“ veranlassen das Öffnen aller sprachspezifischen XML-Dateien (Listing 98 Zeile 278). Die in Listing 98 (Zeile 278 bis 280) aufgeführte Variante dieses Vorgangs geht über das „XMLHttpRequest“-Objekt und findet hier bei allen Browsern außer dem „Microsoft Internet Explorer“ Einsatz (siehe auch Kapitel 4.3.2.4.1.1 Verarbeitungsvorgang bei Aufruf der Start-HTML-Datei). Die in der Manifest-Datei „imsmanifest.xml“ aufgeführten Pfadangaben sind in der globalen Array-Variablen „gAllHrefAttributes“ abgelegt (Listing 98 Zeile 275). Nach dem Öffnen wird der Inhalt jeder XML-Datei einzeln als Parameter (Parameter „gXmlSource.documentElement“) zur Durchsuchung nach dem Suchbegriff (Parameter „para_searchString“) an die JavaScript-Funktion „searchWithinNode()“ übergeben (Listing 98 Zeile 281). Ihre Anweisungen durchsuchen rekursiv alle Knoten nach dem Suchbegriff und stellen die Treffer zur späteren Ausgabe am Bildschirm in den globalen Array-Variablen „gResultsTitle“ und „gResultsFile“ zur Verfügung. Der Quelltext dieser JavaScript-Funktion wurde in Anlehnung an NÁIRON (2007) erstellt.



Listing 98: Ausschnitt aus der Funktion „search()“ aus der JavaScript-Datei „search_index_lang.js“ (Öffnen aller sprachspezifischen XML-Dateien, Aufruf der Funktion „searchWithinNode()“ zum Durchsuchen, Aufruf der Funktion „showResults()“ zur Anzeige des Gesamtergebnisses)

Das Suchergebnis wird wieder sehr ähnlich wie bei der automatischen Erstellung von Verzeichnissen über Befehle in der JavaScript-Funktion „showResults()“(Listing 98 Zeile 282) im Frame mit dem Namen „result“ aufgeführt (siehe auch Abbildung 95 rechts unten). In den ersten beiden Zeilen des Suchergebnisses werden folgende Angaben gemacht (siehe Abbildung 95 rechts unten):

- sprachspezifischer Ausdruck für „Suche“ (aus der XML-Datei „lang_system.xml“)
- Suchbegriff
- sprachspezifischer Ausdruck für „Sprachauswahl“ (aus der XML-Datei „lang_system.xml“)
- Suchsprache
- sprachspezifischer Ausdruck für „Treffer“ (aus der XML-Datei „lang_system.xml“)
- Anzahl der Treffer

Anschließend erscheinen die durchnummerierten Treffer in Listenform mit dem „title“-Attributwert des „section“-Elementes, in dem sie gefunden wurden. Dieser Attributwert wird in Form eines Hyperlinks zur WLMML-Datei angeboten, in der der Suchbegriff auftaucht. Über die Parameter, die der URL zur jeweiligen WLMML-Datei beigefügt sind, heben Anweisungen in der JavaScript-Datei „highlight.js“ den Suchbegriff hervor (vergleiche 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen, Ergebnis siehe Abbildung 96 hervorgehobener Suchbegriff „Beispiel“). Wurde kein „title“-Attributwert für das „section“-Element (in dem sich der Treffer befindet) angegeben, wird ein sprachspezifischer Standardwert aus der XML-Datei

„lang_system.xml“ ausgegeben. Abschließend steht in der zweiten Zeile eines Treffers die Pfadangabe zur WLMML-Datei, in der der Suchbegriff auftaucht.

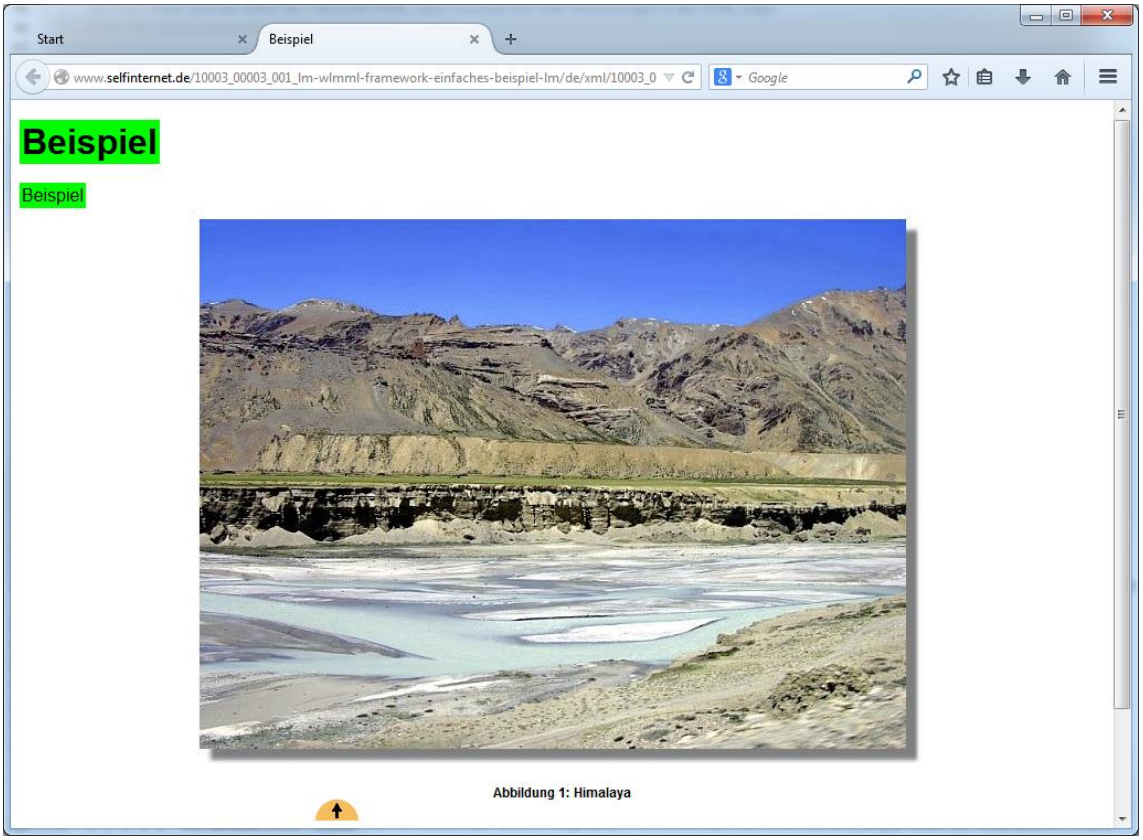


Abbildung 96: Ausgewählte Trefferdatei einer Volltext-Suche (hervorgehobener Suchbegriff „Beispiel“ in einem eigenen Browser-Fenster, „online“-Aufruf von einem Web-Server, siehe Adressleiste, Browser „Mozilla Firefox 29.0.1“)

Welche Voraussetzungen der Browser-Software erfüllt sein müssen, damit die WLMML-Funktionalitäten genutzt werden können, zeigt das nächste Kapitel.

4.3.3 Voraussetzungen der Browser-Software

Im Folgenden werden Anforderungen und Voreinstellungen zur korrekten Anzeige der WLMML-Lerninhalte im Browser vorgestellt.

Die folgenden Browser zeigen WLMML-Lerninhalte korrekt an (alphabetische Reihenfolge):

- „Apple Safari“ (Versionen mit Anfangsziffern 4 bis 5)
- „Google Chrome“ (Versionen mit Anfangsziffern 3 bis 35)
- „Microsoft Internet Explorer“ (Versionen mit Anfangsziffern 6 bis 11)
- „Mozilla Firefox“ (Versionen mit Anfangsziffern 4 bis 30)
- „Opera“ (Versionen mit Anfangsziffern 10 bis 12)

Die ersten vier aufgeführten führen den Browsermarkt in den Regionen Deutschland, Europa und weltweit deutlich an. Ihre Anteile (inkl. „Opera“ mit 3 %) liegen z. B. im Januar 2014 in allen Regionen bei über 85 % (STATCOUNTER 2014, http://gs.statcounter.com/, siehe Kapitel 3.10 Marktanteile Browser).

Bei weiteren Browser-Produkten werden WLMML-Inhalte m ebenfalls korrekt wiedergegeben, da sie oft auf „Trident“-, „Gecko“- oder „WebKit“-Technologien („*Layout engines*“) beruhen (WIKIPEDIA-BROWSER-EN 2014). Zu „Trident“ wird z. B. der Browser „Microsoft Internet Explorer“, zu „Gecko“ z. B. der Browser „Mozilla Firefox“ und zu „WebKit“ werden z. B. die Browser „Google Chrome“ oder „Apple Safari“ gezählt (WIKIPEDIA-BROWSER-EN 2014). Auch auf einem Tablet-PC mit dem Betriebssystem „Android“ (Version 4.0.3) und dem „Android Browser“ (Version 15.1.0.5) bzw. „Microsoft Windows 8.1“ und dem Browser „Microsoft Internet Explorer 11“ konnten WLMML-Inhalte erfolgreich getestet werden. Insgesamt ist der Browser von entscheidender Bedeutung und kaum die Hardware bzw. das Betriebssystem.

Um die Lerninhalte online nutzen zu können, müssen folgende Voreinstellungen bei der Browser-Software gegeben sein:

- Aktivierung von JavaScript
- Deaktivierung bzw. Einstellung des Popup-Blockers
- „Microsoft Internet Explorer“ (Aktivierung von „ActiveX“-Steuerelementen und Plugins)
    - Erlaubnis für die Anzeige von Flash-Animationen („ActiveX“-Steuerelemente und Plugins ausführen)
    - geblockte Inhalte zulassen („ActiveX“-Steuerelemente ausführen, die für Scripting sicher sind)

Für eine offline-Nutzung lokal von der Festplatte müssen aufgrund des eingeschränkten Zugriffs auf lokale Dateien bei einigen Browsern zusätzlich folgende Einstellungen vorgenommen werden:

- „Google Chrome“ ab Version 5
  Starten des Browsers mit dem Parameter „--allow-file-access-from-files“ z. B. über eine Verknüpfung (bei lokalen Aufruf von WLMML-Lernobjekten)
- „Microsoft Internet Explorer“
  Ausführung aktiver Inhalte auf dem Computer zulassen (erweiterte Einstellungen bei Internetoptionen)
- „Opera“ ab Version 11

Akzeptieren und Speichern der „Allow File XMLHttpRequest“-Option in den Konfigurationseinstellungen des Browsers („opera:config#UserPrefs|AllowFileXMLHttpRequest“)

Sollen Flash-/SVG-/MathML-Inhalte angezeigt werden, müssen die entsprechenden Plugins vorhanden sein bzw. die Browser diese Inhalte nativ unterstützen. „Apple Safari“, „Google Chrome“, „Microsoft Internet Explorer“, „Mozilla Firefox“ und „Opera“ sind dazu in immer größeren Umfang in der Lage.

Bei Bedarf muss Java für den Browser installiert bzw. erlaubt werden (siehe Browserhilfe bzw. angezeigte Browserhinweise).

4.4 Lerntheoretischer und didaktischer Ansatz

WLMML (inkl. Framework) bietet unterschiedliche lerntheoretische und didaktische Ansätze (siehe Kapitel 3.1 Lerntheoretische Grundlagen, vergleiche auch WEITL & HÖCK 2003, WEITL et al. 2002).

Über die XML-Sprache WLMML ist es möglich E-Learning-Inhalten XML-Elemente zuzuordnen und sie damit zu strukturieren (siehe Kapitel 4.2 WLMML, siehe auch Kapitel 3.2.3 Modellierung von E-Learning-Inhalten). Zusätzlich werden E-Learning-Inhalte im Zuge der Lernmodulerstellung (siehe Kapitel 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM)) in kleinere Bausteine aufgeteilt und anschließend in einer bestimmten Lernreihenfolge angeboten. Diese Punkte weisen Merkmale der behavioristischen Lerntheorie auf (siehe Kapitel 3.1.1.1 Behaviorismus, siehe auch Tabelle 2).

Mehr Funktionalitäten von WLMML und ihres Frameworks deuten allerdings auf die kognitivistische Lerntheorie hin (insbesondere der computernahe Ansatz). Vor allem die Möglichkeit z. B. Problemstellungen über Übungen, Aufgabenstellungen und Aufgabenlösungen mit WLMML-Elementen abzubilden (siehe Kapitel 4.2 WLMML), stützen diese Ansicht (siehe Kapitel 3.1.1.2 Kognitivismus, Tabelle 1, Tabelle 2, Tabelle 3). Das WLMML-Element „exercise“ (Übung) mit seinen Kind-Elementen „task“ (Aufgabenstellung) und „solution“ (Aufgabenlösung) können dies umsetzen (siehe Kapitel 4.2.2.3 Inhalts-Modul-Elemente). Aber auch die Darbietung von WLMML-Inhalten, bei denen die Präsentation der Inhalte im Vordergrund stehen (Exposition), kann der kognitivistischen Lerntheorie (siehe Kapitel 3.1.1.4 Zusammenfassung, Tabelle 2) zugeordnet werden.

Letztlich sind auch konstruktivistische Ansätze in WLMML (inkl. Framework) erkennbar (vergleiche auch allgemeine Aussage zu Hypertext von HOLZINGER 2000, S. 189). So ermöglichen z. B. die Funktionalitäten des WLMML-Frameworks sich Inhalte in anderen Sprachen anzeigen, sich die Lerninhalte in einem Dokument zusammenfassen, verschiedene Online-Dienste durchsuchen, automatisch Verzeichnisse erstellen und den Lerninhalt durchsuchen zu lassen (siehe Kapitel 4.3.2.3 Mehrsprachigkeit, 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen, 4.3.2.4.2.2 Zusammenfassung der Lerninhalte in einem Dokument, 4.3.2.4.2.3 Suche in verschiedenen Online-Diensten (nur mit Internet-Zugang möglich), 4.3.2.4.3 Automatische Erstellung von Verzeichnissen, 4.3.2.4.4 Volltextsuche in WLMML-Lerninhalten) eine stärkere vom Lernenden steuerbare Lernaktivität (Exploration, siehe Kapitel 3.1.1.4 Zusammenfassung, Tabelle 2).

In den verschiedenen Lerntheorien lassen sich unterschiedliche didaktische Methoden einsetzen (siehe Kapitel 3.1.1.4 Zusammenfassung, 3.1.2 Didaktik, Tabelle 2). Als Grundlage dafür sind z. B. Lernziele und Zusammenfassungen in WLMML abgebildet (WLMML-Elemente „learningObjective“ und „summary“, siehe Kapitel 4.2.2.3 Inhalts-Modul-Elemente, 3.1.2 Didaktik). In WLMML (inkl. Framework) können expositorische, aber auch explorative und problemorientierte Methoden Einsatz finden (siehe Kapitel 3.1.2 Didaktik, Tabelle 4). Expositorische Ansätze zeigen sich z. B. anhand der Möglichkeit Lernprozesse durch Beispiele und Übungen (WLMML-Elemente „example“ und „exercise“, siehe Kapitel 4.2.2.3 Inhalts-Modul-Elemente) zu aktivieren (kognitivistischer Ansatz). Exploratives Lernen betont die Selbststeuerung des Lernenden. Alleine durch den Hypertext-Charakter von WLMML-Lerninhalten und durch die bereits oben benannten Funktionalitäten - wie z. B. sich Inhalte in anderen Sprachen anzeigen zu lassen (siehe Kapitel 4.3.2.3 Mehrsprachigkeit, 4.3.2.4.2.1 Anzeige von Sprachkürzel-Hyperlinks unter Absätzen) - bieten sich explorative Methoden an (siehe Kapitel 3.1.2 Didaktik). Auch problemorientierte Methoden lassen sich über das Lernen mit Fällen durch Beispiele und Übungen (WLMML-Elemente „example“ und „exercise“, siehe Kapitel 4.2.2.3 Inhalts-Modul-Elemente) einsetzen.

Zusätzlich könnten allgemein didaktische Ansätze von SCORM wie z. B. „Sequencing and Navigation (SN)“ eingesetzt werden (3.8 SCORM). Sie würden allerdings vom inhaltsgeprägten WLMML-Framework nicht berücksichtigt und müssten z. B. von einem SCORM-fähigen LMS umgesetzt werden.

Allgemein setzt WLMML (inkl. Framework) seinen Fokus nicht auf eine didaktische Ausrichtung sondern auf die inhaltliche Abbildung von E-Learning-Material. Die Inhalte können dadurch in didaktische Szenarien (siehe Kapitel 3.1.2 Didaktik) wie z. B. unter IMS LD als reine Inhalte integriert werden (siehe Kapitel 3.3.7.3.2 IMS LD-Lerneinheit). Außerdem können WLMML-E-Learning-Inhalte natürlich auch im „Blended Learning“ (Präsenzlehre und E-Learning) (siehe Kapitel 3.1.2 Didaktik) verwendet werden.

5 Resümee und Ausblick

Das Resümee soll Inhalte dieser Arbeit aus unterschiedlichen Perspektiven beleuchten und an verschiedenen Stellen einen Ausblick in die Zukunft bieten.

5.1 Standardisierung

Die häufig an E-Learning-Inhalte gestellten Anforderungen wie Plattform-Unabhängigkeit, Kompatibilität, Zukunftssicherheit, Austauschbarkeit und Nachhaltigkeit lassen den Einsatz standardisierter und weit verbreiteter Auszeichnungs-/Programmiersprachen mehr als sinnvoll erscheinen (siehe Kapitel 2 Motivation, 2.1 Standardisierung, 3.2.1 Standardisierungsbemühungen). Allerdings unterliegen auch Standards bzw. Normen einem Wandel (siehe Kapitel 3.2.1 Standardisierungsbemühungen, im Folgenden wird der internationale Begriff Standard verwendet). Dieser Wandel erfordert einen kontinuierlichen Entwicklungsprozess mit der Festlegung, ob die E-Learning-Inhalte an die veränderten Standards angepasst werden sollen. Eine entscheidende Bedeutung kommt dabei der Kompatibilität (insbesondere Abwärtskompatibilität) zu den Software-Programmen für die Erstellung und Verarbeitung der E-Learning-Inhalte zu. Ist sie nicht gegeben, erscheint ihre Anpassung an die neuen Standards kaum sinnvoll. Wenn z. B. die Browser-Software die E-Learning-Inhalte nach der Umstellung nicht darstellen kann, nutzt der Einsatz neuester Standards wenig. Dabei ist nicht nur der Einsatz neuer Standards sondern auch die Anpassung an eine neue Version eines Standards von Bedeutung (z. B. Entwicklung von HTML 5). Ohne entsprechende Vorbereitung und Testläufe gerät nach der Anpassung an neue Standards die Erstellung und Nutzung von E-Learning-Inhalten aus software-technischer Sicht zu einem Vabanquespiel. Daneben ergibt sich mitunter die Problematik, dass sie nicht mehr weiter entwickelt werden und sich ihr Einsatz als nicht mehr zukunftssicher erweisen mag.

Dies trifft für den XML-Standard nicht zu. Nach wie vor kann XML als das weltweit führende Standard-Speicherformat angesehen werden (siehe Kapitel 3.3 XML). Dabei bietet es z. B. den Vorteil, dass textbasierte XML-Inhalte wie WLMML-Dokumente auch in andere XML-Sprachen oder HTML-Versionen transformiert werden können (z. B. über XSLT). Von Nachteil ist jedoch der höhere Aufwand für die Auszeichnung der Lerninhalte mit XML-Elementen. Der Einsatz der dafür angebotenen Software-Produkte (z. B. „Altova XML-Spy“, „Eclipse“) ist für die Autoren von E-Learning-Inhalten außerdem mit einer höheren Einarbeitungszeit verbunden. „Microsoft Word“- oder „Microsoft PowerPoint“-Dateien lassen sich

dagegen meist einfacher erstellen. Andererseits eröffnet die semantische Auszeichnung über XML-Elemente mehr Möglichkeiten im Umgang mit E-Learning-Inhalten wie z. B. eine automatisierte Erstellung unterschiedlichster Verzeichnisse, Wiederverwendung bzw. Austausch von Lernobjekten oder eine spezifische optische Darstellung (siehe Kapitel 3.4 HTML, 4.3 WLMML-Framework). Die XML-Sprache WLMML wurde für diesen Zweck entwickelt (siehe Kapitel 4.2 WLMML). Zwar existierten zum Zeitpunkt der Erstellung dieser Arbeit bereits unterschiedliche XML-Sprachen zur Auszeichnung von E-Learning-Inhalten bzw. waren im Entstehen (siehe Kapitel 3.3.6 XML-Anwendungen zur Auszeichnung von E-Learning-Inhalten), jedoch erschien die Modellierung der neuen XML-Sprache WLMML für den Bereich der „Life Sciences“ u. a. aus Gründen der Flexibilität notwendig (siehe Kapitel 4.2 WLMML). WLMML beruht auf LMML und nutzt damit ihre Vorteile, bietet andererseits die Freiheiten einer eigenständigen Entwicklung. Als zusätzlicher positiver Ansatz könnte WLMML mit IMS LD kombiniert werden (siehe Kapitel 3.3.7.3 IMS LD). Der pädagogische Überbau würde dabei von IMS LD gestellt, die inhaltliche Abbildung der E-Learning-Inhalte durch WLMML.

Interessanterweise etabliert sich derzeit im „World Wide Web“ nicht die XML-Sprache XHTML sondern HTML 5 als Nachfolger von HTML 4 (siehe Kapitel 3.3.5.3 XHTML, 3.4 HTML). Eine entscheidende Rolle mag dabei die bevorzugte Umsetzung von HTML 5 in den neuen Browser-Generationen spielen. Insgesamt erscheinen jedoch die „XML-Technologies“ (XML, XML Namespaces, XML Schema, XSLT, XPath, XQuery) als relativ zukunftssicher. Ein ähnlicher Eindruck ergibt sich bei CSS (siehe Kapitel 3.5 CSS) und JavaScript (siehe Kapitel 3.6 JavaScript).

Anders stellt sich die Umsetzung (bzw. Entwicklung) von LOM im Bereich des E-Learning dar (siehe auch Kapitel 3.7 IEEE LOM). Ob sie Bestand hat, bleibt abzuwarten. U. a. erschwert der mitunter erhebliche Zeit- und Kostenaufwand für die Erhebung von Metadaten ihren Einsatz (siehe auch QUEDNAU & STREHL 2007, S. 5). Eine größere Akzeptanz dieser bei Autoren mitunter ungeliebten Aufgabe könnte durch den Einsatz anwenderfreundlicher LOM-Editoren erreicht werden. Neben der Angabe von Metadaten kommt der Auswertung dieser Daten entscheidender Bedeutung zu. Inwieweit E-Learning-Software (z. B. LMS) in Zukunft LOM-Daten auswertet, ist schwierig abzuschätzen. Trotzdem darf die Bedeutung von Metadaten für die Wiederverwendung von E-Learning-Inhalten nicht unterschätzt werden (siehe auch JUNGMANN 2012, S. 179 ff, KERRES 2012, S. 448).

Die Unterstützung von SCORM (mit IMS CP) in etablierten LMSs zeigt, dass dieser Standard eine weite Verbreitung gefunden hat (siehe auch Kapitel 3.8 SCORM). Damit scheint auch seine Zukunftssicherheit gegeben. Gleiches gilt für den Einsatz von LMSs im E-Learning-Umfeld (siehe auch Kapitel 3.9 LMS, ROTERING-STEINBERG & RICHTER 2011, S. 7 f). Allerdings werden wohl die Anforderungen des „E-Learning 2.0“ (WIKIPEDIA-E-LEARNING-EN 2014) an die LMSs einen höheren Stellenwert einnehmen. Besonders die Umsetzung sozialer Aspekte im Lernprozess gewinnen an Bedeutung. Der relativ neue lerntheoretische Ansatz des „Konnektivismus“, die Entwicklung von PLE und die allgemeine Hochkonjunktur sozialer Netzwerke stärken diese Hypothese.

Insgesamt bleibt für einen erfolgreichen Einsatz von WLMML-Lerninhalten die Abwärtskompatibilität zu den im WLMML-Kontext eingesetzten Standards in den neuen Browser-Generationen entscheidend. Zudem stellt das Angebot benutzerfreundlicher Software zur Erstellung und Darstellung von WLMML-E-Learning-Inhalten einen zentralen Faktor dar.

5.2 Mehrsprachigkeit

Den hohen Stellenwert der Mehrsprachigkeit hebt z. B. die Europäische Kommission mit ihrer Rahmenstrategie für Mehrsprachigkeit hervor. In dieser Strategie wird u. a. die Unterstützung der Erstellung und Verbreitung „*multilingualer europäischer Inhalte und europäischen Wissens*“ angesprochen und in diesem Zusammenhang der Ausdruck E-Learning genannt (EUROPÄISCHE-KOMMISSION 2005, III.3 Mehrsprachigkeit und die Informationsgesellschaft). Auch STACEY (2007, Users) betont die Bedeutung der Mehrsprachigkeit und beschreibt darüber hinaus, dass Lernende in ihrer Muttersprache effektiver lernen und ihre Lernbereitschaft steigt.

Die XML-Sprache WLMML und das WLMML-Framework sind auf die Erstellung und Nutzung „multilingualer“ Inhalte ausgerichtet (siehe Kapitel 4.3.2.3 Mehrsprachigkeit). Zwar müssen die Inhalte nicht in mehreren Sprachen abgebildet werden, aber wenn dies umgesetzt wird, ist der Aufwand für die Übersetzung und Anpassungen der Inhalte zeit- und damit kostenintensiv (siehe auch SCLATER 2011, S. 187). Besonders z. B. später kommende Anpassungen in Lerninhalten einer Sprache ziehen eine Veränderung der Inhalte in allen anderen Sprachen nach sich. Daneben spielt die Qualität der Übersetzung für die Qualität der E-Learning-Inhalte eine entscheidende Rolle (siehe auch SCLATER 2011, S. 187).

Insgesamt ist auf Grund der weiter zunehmenden Globalisierung eher ein steigender Bedarf an mehrsprachigen Lerninhalten zu erwarten. Zumal nach INTERNET-WORLD-STATS (2011) nur ca. einem Viertel der Internet-Nutzer Englisch als „erste“ Sprache zugeordnet wird. Drei Viertel finden sich dagegen in der Statistik bei anderen Sprachen wieder. Zusätzlich weisen diese drei Viertel der Internet-Nutzer besonders hohe Zuwachsraten auf (INTERNET-WORLD-STATS-LANGUAGES 2011).

5.3 Clientseitige Verarbeitung

Eine clientseitige Verarbeitung von E-Learning-Inhalten mit XML-basierten Auszeichnungssprachen und JavaScript bietet Vorteile, weist aber auch Nachteile auf.

Der größte Vorteil besteht darin, dass zur Nutzung von WLMML-Lerninhalten kein Internetzugang vorhanden sein muss. Die E-Learning-Inhalte können z. B. nach ihrem Download aus dem Internet von der Festplatte, einem USB-Stick oder einer CD-ROM offline ohne Internetzugang nahezu in ihrer vollen Funktionalität genutzt werden. Die zentrale Rolle für die Verarbeitung und Darstellung übernimmt dabei der Browser (Client-Software) (siehe Kapitel 4.3.2.4 Clientseitige Verarbeitung). Damit bleibt die Nutzung der Lerninhalte z. B. unabhängig von Weiterentwicklungen serverseitiger Programmiersprachen, die bei einer serverseitigen Abarbeitung der E-Learning-Inhalte maßgeblich wären. Insbesondere JavaScript erweist sich bei einem clientseitigen Einsatz mit Browser-Software als sehr konkurrenzfähige Programmiersprache (siehe auch Kapitel 3.6 JavaScript). So beschreibt z. B. STEYER 2014 (S. 144) wie mobile Webanwendungen im Browser über den „*HTML5-Anwendungs-Cach*“ und „*IndexedDB*“ „*offline-fähig*“ gemacht werden können. Aufgrund dieser neuesten Entwicklungen erscheint auch die clientseitige „offline“-Verarbeitung zukunftsorientiert (siehe auch MINTERT 2012, S. 41 ff). Zumal serverseitige Programmiersprachen (wie z. B. PHP) im Quelltext von E-Learning-Inhalten von LMSs nicht immer unterstützt werden. Daneben weisen zusätzlich CRANE & PASCARELLO (2006, S. 489 f) auf die Vorteile clientseitiger Verarbeitung von XML-Dateien mit Hilfe von Anweisungen in XSLT-, und JavaScript-Dateien hin.

Steht ein Internetzugang zur Verfügung, lassen sich die gleichen WLMML-Lerninhalte ohne Anpassungen auch online auf einem Web-Server zur Verfügung stellen. In diesem Fall werden die WLMML-Inhalte online vom Web-Server geladen und genauso wie bei der Variante ohne Internetzugang clientseitig vom Browser mit

Hilfe der Dateien im WLMML-Framework verarbeitet. Allerdings können dann alle Funktionalitäten wie z. B. die Suche in verschiedenen Online-Diensten (siehe Kapitel 4.3.2.4.2.3 Suche in verschiedenen Online-Diensten (nur mit Internet-Zugang möglich)) oder allgemein Hyperlinks zu anderen Informationsquellen im Internet genutzt werden.

Der Vorteil einer clientseitigen Verarbeitung ist aber gleichzeitig auch ihr Nachteil. Änderungen in der clientseitigen Software (Browser) z. B. bezüglich XML-, XSLT- und JavaScript-Unterstützung können die Nutzung der WLMML-Inhalte stark beeinträchtigen bzw. unmöglich machen. Bei einem Update eines Browsers sind Tests notwendig, ob die WLMML-Inhalte noch in ihrer vollen Funktionalität genutzt werden können. Anders ausgedrückt hätte eine serverseitige Verarbeitung den Vorteil, dass die Schwierigkeiten durch die unterschiedlichen Implementierungen der Verarbeitung von XML mit XSLT und JavaScript im Browser umgangen werden könnten. Die zukünftige Entwicklung der unterschiedlichen Browser wird zeigen, inwieweit sie z. B. „XML-Technologies“ (XML, XML Namespaces, XML Schema, XSLT, XPath, XQuery, siehe Kapitel 3.3 XML) oder JavaScript (siehe Kapitel 3.6 JavaScript) umsetzen werden. Die weltweite Verbreitung dieser Technologien lässt allerdings vermuten, dass ihre Implementierung sehr wahrscheinlich bleibt.

Sollte es trotzdem notwendig erscheinen auf eine serverseitige Verarbeitung der WLMML-Inhalte umzustellen, lassen sich WLMML-Inhalte z. B. über XSLT nach HTML oder andere XML-Sprachen transformieren und als eigenständige Dateien zur Nutzung anbieten. Auch das WLMML-Framework mit den XSLT- und JavaScript-Dateien könnte durch eine serverseitige Programmierung (z. B. PHP) nachgebildet und online mit serverseitiger Unterstützung zur Verfügung gestellt werden.

Insgesamt bieten E-Learning-Inhalte auf der Basis von WLMML den Vorteil, dass sie mit den meisten Hard- und Software-Produkten offline und online genutzt werden können. Auch unter Betriebssystemen wie z. B. „Microsoft Windows 8.1“, „Android 4.0.3“ oder „Ubuntu Desktop 12.10“ konnten auf Tablet-PCs, Laptops und PCs mit Standard-Browsern WLMML-Inhalte erfolgreich aufgerufen werden. Entscheidend ist dabei, welcher Browser auf dem jeweiligen Gerät eingesetzt wird (siehe Kapitel 4.3.3 Voraussetzungen der Browser-Software).

5.4 Trends

Im Folgenden werden mögliche Entwicklungen im Bereich des E-Learning beleuchtet.

Von MEEKER (2012, S. 50) wird bei ihrer Präsentation vor Studenten der „Stanford University“ bereits der derzeitige Stand der Dinge im Bereich der Ausbildung mit folgenden Ausdrücken beschrieben: „*Interactive / Online / Accessible by Anyone Anywhere Anytime*“. Aufgrund der noch weiter fortschreitenden Durchdringung des Alltags durch das Internet liegt die Vermutung nahe, dass damit E-Learning auch in Zukunft eine bedeutende Rolle spielen wird. Dabei finden nicht nur Laptops und herkömmliche Desktop-PCs, sondern zunehmend auch andere Endgeräte wie Netbooks, Tablet-PCs oder Smartphones mit ihren vielfältigen „Apps“ Einsatz (siehe Kapitel 3.2 E-Learning). In diesem Zusammenhang tauchen die Ausdrücke „Mobiles Lernen“ oder „Ubiquitäres Lernen“ auf und spiegeln dabei den Gedanken wieder, dass vermehrt als Lernunterstützung mobile und allgegenwärtige Computer-technologien genutzt werden (SPECHT et al. 2013, 1. Definitionen, siehe auch ZHAO et al. 2010, STOLLER-SCHAI 2010, GREEN 2013, S. 3 ff). Die stetig steigende Anzahl mobiler Endgeräte macht dies mehr als wahrscheinlich. Auch Themen wie „Microblogging“ und „Microlearning“ spielen in diesem Kontext eine Rolle (STOLLER-SCHAI 2010, S. 3 f, ROBES 2009, BUCHEM et al. 2013, siehe auch Kapitel 3.1.1.1 Behaviorismus). BUCHEM et al. (2013, 1. Einführung) halten dabei fest, dass „Blogs“ und „Microblogs“ „*neben Wikis und Podcasts zu den meist genutzten Social-Media- bzw. Web-2.0-Diensten*“ gehören (zu „Social Media“ siehe auch CAVAZZA 2014). Als „Microblogs“ bezeichnen sie „*kleinere Blogformate mit Einträgen von circa 140 bis 250 Zeichen*“. Blog steht in diesem Zusammenhang für die Kurzform von Weblog, wobei Weblog ein Kunstwort aus ,,Web‘‘ und ,,log" (Abkürzung für Logbuch, Tagebuch) ist (WIKIPEDIA-BLOG 2014).

E-Learning mag damit in Zukunft mobiler, selbstbestimmter, eigenverantwortlicher, interaktiver (z. B. auch „Game-based Learning“, siehe dazu LE et al. 2013, JOHNSON et al. 2014, S. 57, MEEKER 2012, S. 51) und vernetzter werden (siehe auch siehe auch Kapitel 3.2.2 Übersicht Methoden und Medien, 3.1.1.4 Zusammenfassung). Besonders das vernetzte Lernen z. B. über soziale Netzwerke wie z. B. „Facebook“, „Google+“, „XING“, „Twitter“ oder über Medienplattformen wie das Video-Portal „YouTube“ gewinnt immer mehr an Bedeutung (siehe auch Kapitel 3.1.1.4 Zusammenfassung). PLEs (siehe Kapitel 3.9 LMS) tragen dieser Entwicklung bereits Rechnung. Auch der Ausdruck „E-Learning 2.0“ (WIKIPEDIA-E-LEARNING-EN 2014) - sozusagen E-Learning mit „Web 2.0“-Technologien (z. B. Wikis, Weblogs,

Podcasts, soziale Netzwerke, Medienplattformen) - ist in diesem Zusammenhang zu sehen (KERRES 2012, S. 454 ff, KERRES 2006, EHLERS 2008, u. a. S. 11, HAUG et al. 2012, 2.3.2 Web 2.0, REDECKER et al. 2010). Über die Stärkung des sozialen Aspektes und eines aktiven, kollaborativen Umgangs mit neuen Medien entsteht ein „*Mitmach-Web*“ (EBNER et al. 2013). Auch JOHNSON et al. (2014, S. 9) heben die steigende Bedeutung des sozialen Aspektes als Trend im E-Learning hervor. Sie führen im „*NMC Horizon Report: 2014 Higher Education Edition*“ als „*Schlüsseltrends, die den Einsatz von Technologien im Hochschulbereich befördern*“ folgende Themen auf JOHNSON et al. (2014, S. 1):

- „*zunehmende Verbreitung sozialer Medien*“
- „*Integration von Online-, Blended- und kollaborativem Lernen*“
- „*Zunahme von datengetriebenem Lernen*“ (Assessment und Analyse von Datenspuren Online-Lernender zur Verbesserung des Lehrangebotes, siehe auch „Learning Analytics“ JOHNSON et al. 2014, S. 51)
- „*Paradigmenwechsel von Studierenden als Konsumenten hin zu Studierenden als Machern*“

Zur Identifikation der „*strategischen, organisatorischen und praktischen Implikationen*“ wurde der Referenzrahmen für innovative Lernumgebungen von BOCCONI et al. (2012, S. 3) herangezogen (Abbildung 97). Er beschreibt acht „*key dimensions*“ und 28 „*reference parameters*“, die auf „*Creative Classrooms*“ Einfluss nehmen.

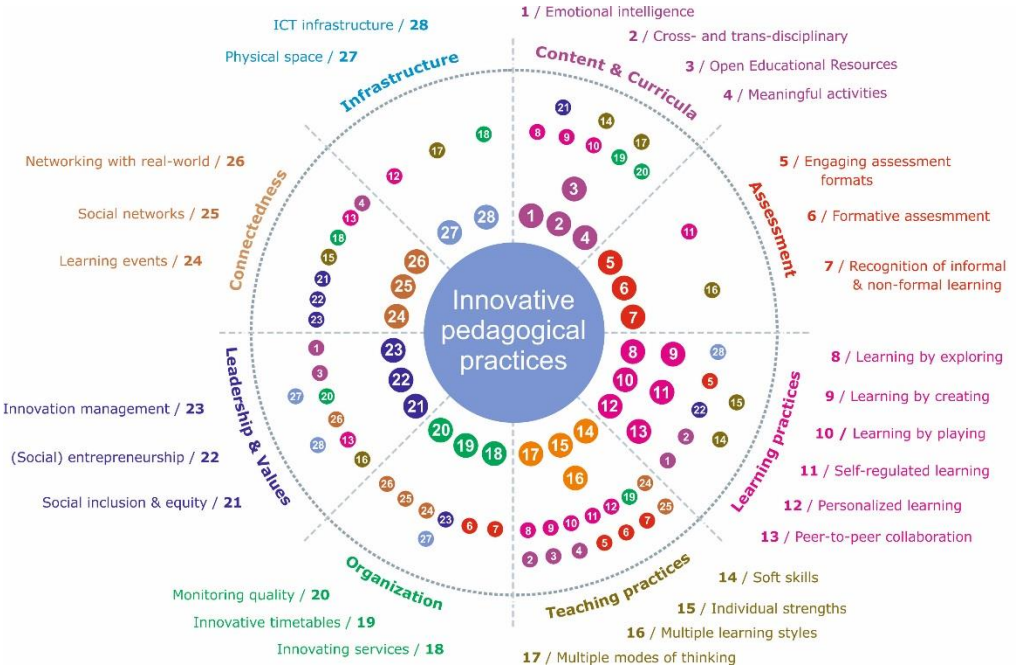


Abbildung 97: Referenzrahmen für eine innovative Lernumgebung, acht „*key dimensions*“ und 28 „*reference parameters*“, “*Creative Classrooms key dimensions and building blocks*” (BOCCONI et al. 2012, S. 3, zu “Kreative Klassenzimmer” siehe EU (2012, S. 5 ff) im Rahmen von „ET 2020“ (EU 2009))

Zusätzlich beschreiben JOHNSON et al. (2014, S. 1 ff) „*Besondere Herausforderungen, die den Einsatz von Technologien im Hochschulbereich behindern*“ und „*Wichtige lehr-/lerntechnologische Entwicklungen im Hochschulbereich*“. Als eine „*lehr-/lerntechnologische Entwicklung*“ nennen sie z. B. „*Flipped Classroom*“ (JOHNSON et al. 2014, S. 48). Dieser Ansatz – auch bezeichnet als „*umgedrehter Unterricht*“ – wird von WIKIPEDIA-FC 2014 beschrieben als „*eine Unterrichtsmethode des integrierten Lernens, in der die Hausaufgaben und die Stoffvermittlung insofern vertauscht werden, als die Lerninhalte zu Hause von den Schülern erarbeitet werden und die Anwendung in der Schule geschieht*“ (siehe auch LOVISCACH 2013, S. 105). Für das Erarbeiten der Lerninhalte zu Hause könnten z. B. auch mit WLMML-beschriebene Lerninhalte als Wissensgrundlage für den anschließend folgenden Unterricht herangezogen werden. Gleiches gilt z. B. für das Bestreben bei nicht konsekutiven Studiengängen die Vorkenntnisse der Studienbewerber auf ein gewünschtes Niveau zu bringen (siehe Kapitel 2 Motivation).

Als weiterer Trend mag E-Learning vermehrt unter dem Aspekt der Qualität betrachtet werden (siehe z. B. EHLERS 2013, siehe auch Kapitel 3.2.1.2 Standardisierung de jure <-> de facto, 4.3.2.1 Modularisierung und Wiederverwendung (inklusive SCORM)). Auf dieses Thema geht auch die EU (EU 2009) in einem ihrer vier strategischen Ziele für die europäische Zusammenarbeit auf dem Gebiet der allgemeinen und beruflichen Bildung bis 2020 ein. Sie benennt das Ziel wie folgt: „*Verbesserung der Qualität und Effizienz der allgemeinen und beruflichen Bildung*“. Daneben werden die drei weiteren Ziele „*Verwirklichung von lebenslangem Lernen und Mobilität*“, „*Förderung der Gerechtigkeit, des sozialen Zusammenhalts und des aktiven Bürgersinns*“ und die "*Förderung von Innovation und Kreativität*“ aufgeführt. Insgesamt erscheint es naheliegend, dass bei einer weiteren Verbreitung von E-Learning-Angeboten das Thema Qualitätssicherung an Stellenwert gewinnen wird.

Darüber hinaus dürfte das Urheberrecht an E-Learning-Inhalten eine immer wichtigere Rolle spielen (siehe z. B. HANSEN & SEEHAGEN-MARX 2013). Die Idee des freien Lernens mit offenen Bildungsressourcen („Open Educational Resources“, OER) drückt den Wunsch aus „*Lehrmaterialien nach den eigenen Bedürfnissen aus unterschiedlichen Quellen auswählen, benutzen, kombinieren und weiterverwerten zu können*“ (ETEACHING-OER 2014). ETEACHING-OER (2014) gibt dazu folgende Definition für den Ausdruck OER an: „*Im Allgemeinen wird darunter digitalisiertes Lehr-/Lernmaterial verstanden, das im Internet zur freien Verfügung steht.*“ Ähnlich

zielen Ausdrücke wie „OpenAccess“ oder „OpenContent“ auf die „*freie Verfügbarkeit wissenschaftlicher Informationen - in der Regel Publikationen – im Internet ab*“ (ETEACHING-OA 2012). Die benannten Ansätze (OER, „OpenAccess“, „OpenContent“) spiegeln die Bedenken vor rechtlichen Konsequenzen bei der Verwendung urheberrechtlich geschützter Materialien wieder. Vor diesem Hintergrund können auch „Massive Open Online Courses“ (MOOCs) gesehen werden (YOUSEF et al. 2014). WIKIPEDIA-MOOC (2014, siehe auch ETEACHING-MOOC 2014) definiert MOOC wie folgt: „*A massive open online course (MOOC; /muːk/) is an online course aimed at unlimited participation and open access via the web. In addition to traditional course materials such as videos, readings, and problem sets, MOOCs provide interactive user forums that help build a community for students, professors, and teaching assistants (TAs).*“ Prototypisch für diese derzeit stark steigende Anzahl von offenen Online-Kursen war nach LOVISCACH (2013, S. 106, siehe auch ETEACHING-MOOC 2014) der von George Siemens und Stephen Downes im Jahr 2008 abgehaltene Kurs „*Connectivism and Connected Knowledge*“ (zu Konnektivismus siehe auch Kapitel 3.1.1.4 Zusammenfassung). Insbesondere vor dem Hintergrund der zunehmenden Verbreitung sozialer Medien mag damit der lerntheoretische Ansatz des Konnektivismus an Bedeutung gewinnen.

Auch Themen wie „Content Management“, Wiederverwendung oder finanzielle Aspekte werden in Zukunft im E-Learning-Bereich - besonders für Organisationen, die Lerninhalte anbieten - eine wichtige Rolle spielen (FERRER & ALFONSO 2011, S. 197 ff). Auf XML-Sprachen basierende E-Learning-Inhalte (wie z. B. WLMML-Dokumente) könnten dazu ihren Beitrag leisten.

Anhang

Anhang A Quelltext eines IMS LD-XML-Dokumentes („imsmanifest.xml“) mit IMS CP- und IMS LD-Elementen, erstellt mit der Software „RELOAD“)

```
<?xml version="1.0" encoding="UTF-8"?>
<!--This is a Reload version 2.5.6 Learning Design document-->
<!--Spawned from the Reload Learning Design Generator –
http://www.reload.ac.uk-->
<manifest xmlns=http://www.imsglobal.org/xsd/imscp_v1p1
xmlns:imsmd="http://www.imsglobal.org/xsd/imsmd_v1p2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:imsld="http://www.imsglobal.org/xsd/imsld_v1p0"
identifier="MANIFEST-1"
 xsi:schemaLocation="http://www.imsglobal.org/xsd/imscp_v1p1
 imscp_v1p1.xsd http://www.imsglobal.org/xsd/imsmd_v1p2
 imsmd_v1p2p4.xsd http://www.imsglobal.org/xsd/imsld_v1p0
 IMS_LD_Level_A.xsd">
  <organizations>
    <imsld:learning-design identifier="LD-1" level="A" uri="">
      <imsld:title>wast Test</imsld:title>
      <imsld:components>
        <imsld:roles identifier="LD-2">
          <imsld:learner identifier="LD-3">
            <imsld:title>Learner</imsld:title>
          </imsld:learner>
        </imsld:roles>
        <imsld:environments>
          <imsld:environment identifier="LD-4">
            <imsld:title>New Environment</imsld:title>
            <imsld:learning-object identifier="LD-5"
                   type="knowledge-object">
              <imsld:title>New Learning Object</imsld:title>
             <imsld:item identifierref="RES-1" />
```

```
            </imsld:learning-object>
          </imsld:environment>
        </imsld:environments>
      </imsld:components>
      <imsld:method>
        <imsld:play identifier="LD-6">
          <imsld:title>Play</imsld:title>
          <imsld:act identifier="LD-7">
            <imsld:title>Act</imsld:title>
            <imsld:role-part identifier="LD-8">
              <imsld:title>Role Part</imsld:title>
              <imsld:role-ref ref="LD-3" />
            </imsld:role-part>
          </imsld:act>
        </imsld:play>
      </imsld:method>
    </imsld:learning-design>
  </organizations>
  <resources>
    <resource identifier="RES-1" type="webcontent" href="test.html">
      <file href="test.html" />
    </resource>
  </resources>
</manifest>
```

Anhang B Ausschnitt aus dem Quelltext einer IEEE LOM-XML-Datei, eingeschränkt auf die General Gruppe (nach DUVAL 2003 b)

```
<?xml version="1.0" ?>
<lom xmlns="http://ltsc.ieee.org/xsd/LOMv1p0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ltsc.ieee.org/xsd/LOMv1p0 lom.xsd
http://ltsc.ieee.org/xsd/LOMv1p0/custom adlmd_vocabv1p0.xsd">
<general>
<identifier>
<catalog>Sample Catalog 1</catalog>
<entry>Sample Entry 1</entry>
</identifier>
<identifier>
<catalog>Sample Catalog 2</catalog>
<entry>Sample Entry 2</entry>
</identifier>
<title>
<string language="en">Sample Title</string>
<string language="en">Sample Sub-Title</string>
</title>
<language>en</language>
<language>fr</language>
<description>
<string language="en">Sample Description 1</string>
<string language="fr">Sample Description 1 in French</string>
</description>
<description>
<string language="en">Sample Description 2</string>
</description>
<keyword>
<string language="en">Key1</string>
<string language="fr">Key1 in French</string>
</keyword>
<keyword>
```

```
<string language="en">Key2</string>
<string language="fr-CA">Key2 in French Canadian</string>
</keyword>
<coverage>
<string language="en">16th century France</string>
</coverage>
<coverage>
<string language="en">17th century France</string>
</coverage>
<!--
 loose vocabulary XSD used, using a different source and value other than
that defined in LOM
-->
<aggregationLevel>
<source>ADLv1.3</source>
<value>asset</value>
</aggregationLevel>
</general>

…

</lom>
```

Anhang C **WLMML-XSD-Datei (siehe beigelegte CD-ROM)**

Anhang D **WLMML-Schema-Dokumentation im HTML-Format und als „Microsoft Word Dokument“ („.docx“-Datei) (siehe beigelegte CD-ROM)**

Anhang E **Anleitung für die Erstellung von WLMML-Dateien (siehe beigelegte CD-ROM)**

Anhang F **WLMML-Framework (siehe beigelegte CD-ROM)**

Anhang G **Muster-WLMML-Lernmodul „10003_00002_001_lm-wlmml-framework-muster-lm“ (siehe beigelegte CD-ROM)**

Anhang H **Einfaches zweisprachiges Beispiel eines WLMML-Lernmoduls „10003_00003_001_lm-wlmml-framework-einfaches-beispiel-lm“ (siehe beigelegte CD-ROM)**

Anhang I **Anleitung für die Erstellung eines mehrsprachigen „SCORM2004“-kompatiblen WLMML-Lernmoduls (siehe beigelegte CD-ROM)**

Anhang J **Anleitung für die Erstellung webbasierter, interaktiver Übungen mit dem Autorenwerkzeug „Hot Potatoes“ (siehe beigelegte CD-ROM)**

Literaturverzeichnis

ADAM, A. (2002): SVG Scalable Vector Graphics, Das Praxisbuch, Professional Series, Franzis' Verlag GmbH, 2002, ISBN: 3-7723-6190-0

ADOLPHE, K. (2012): Intelligent eLearning with XML, Kim Adolphe, President and CEO, Gemini Performance Solutions Inc., Calgary, Alberta, Canada http://www.gemini.com/about-us/xml_whitepaper.html, Abruf am 2012-04-28

ADLNET-RELOAD (2009): ADL SCORM 2004 RELOAD Editor Version 1.1 (Reload Content Editor), Version 1.1 (November 20, 2009), 2009, http://www.adlnet.gov/wp-content/uploads/2011/07/ADL_SCORM_2004_RELOAD_Editor_1_1.zip, Abruf am 2014-01-15

ADLNET-SCORM (2014): SCORM, CAPABILITY INFORMATION, Advanced Distributed Learning, http://www.adlnet.gov/scorm/, Abruf am 2014-01-05

ADLNET-SCORM-SPECIFICATION (2012): SCORM 2004 4th Edition Specification, Technology: SCORM, Last Updated: January 17, SCORM 2004 4th Edition .zip of all of the SCORM books, ADL Advanced Distributed Learning, 2012, http://www.adlnet.gov/resources/SCORM-2004-4th-Edition-Specification?type=technical_documentation, Abruf am 2014-01-07

ADLNET-SCORM-SRTE (2009): ADL Sample Run-Time Environment, SCORM-conformant LMS Software, SCORM 2004 4th Edition, ADL Advanced Distributed Learning, 2009, http://www.adlnet.gov/wp-content/uploads/2011/07/SCORM.2004.4ED.SRTE_.v1.1.1.zip, Abruf am 2014-06-08

ANDERSON, R., BIRBECK, M., DIDIER, M., KAY, M., LIVINGSTONE, S., LOESGEN, B., MOHR, S., OZU, N., PEAT, B., PINNOCK, J., STARK, P., WILLIAMS, K. (2000): XML professionell (1. Auflage). Bonn: MITP-Verlag, 2000, ISBN: 3-8266-0633-7

ARIADNE (2013): ARIADNE Foundation, Alliance of Remote Instructional Authoring and Distribution Networks for Europe, http://www.ariadne-eu.org/, Abruf am 2013-08-15

ARIADNE-METADATA (2014): Metadata, Technologies, ARIADNE Foundation, http://www.ariadne-eu.org/content/technologies, Abruf am 2014-01-04

ARNOLD, P. (2001): Didaktik und Methodik telematischen Lehrens und Lernens: Lernräume, Lernszenarien, Lernmedien; State of the art und Handreichung, Waxmann Verlag GmbH, 2001, ISBN 3-8309-1107-6

ARNOLD, P., LARS K., THILLOSEN, A., ZIMMER, G. (2011): Handbuch E-Learning: Lehren und Lernen mit digitalen Medien, 2. Auflage,W. Bertelsmann Verlag GmbH &Co. KG, Bielefeld, 2011, ISBN: 978-3-7639-4888-8

ARTACHO, M. R. (2004): PALO language Home Page, LTCS Group -- UNED University, http://sensei.lsi.uned.es/palo/, Abruf am 2013-08-19

AUBOURG, J., SONG, J., STEEN, H. R. M. (2012): World Wide Web Consortium (W3C): XMLHttpRequest, W3C Working Draft 6 December 2012, http://www.w3.org/TR/2012/WD-XMLHttpRequest-20121206/, Abruf am 2014-01-03

BABU, S. C. (2001): e-Learning Standards, last modified 8. August, 2001, http://www.cdacindia.com/html/pdf/Session6.1.pdf, Abruf am 2003-07-12

BAILEY, E. (2012): Metadata acronyms – MLR or LRMI?, e-learning standards, Metadata and Standards, NSW Department of Education and Communities, May 31, 2012, http://elearningstandards.wordpress.com/tag/mlr/, Abruf am 2014-01-04

BARKER, P., CAMPBELL, L. M., ROBERTS, A., SMYTHE, C. (2006): IMS Meta-data Best Practice Guide for IEEE 1484.12.1-2002 Standard for Learning Object Metadata, Version 1.3 Final Specification, IMS Global Learning Consortium, http://www.imsglobal.org/metadata/mdv1p3/imsmd_bestv1p3.html, Abruf am 2013-08-23

BATES, A. W. T. (2011): Content Management and E-Learning: A Strategic Perspective, in: Ferrer, N., F., Alfonso, J. M. (2011): Content Management for E-Learning, Springer Verlag, 2011, ISBN 978-1-4419-6958-3

BAUMGARTNER, P. (2006): E-Learning Szenarien: Vorarbeiten zu einer didaktischen Taxonomie. In: E-Learning - alltagstaugliche Innovation? Seiler-Schiedt, E./Kälin, S/. Sengstag, C., Münster: Waxmann Verlag, 2006, S. 238-247, http://www.peter.Baumgartner.name/schriften/article-de/szenarien-taxonomie.pdf,
Abruf am 2012-08-28

BAUMGARTNER, P., HÄFELE H., MAIER-HÄFELE, K. (2002 a): E-Learning Praxishandbuch, Auswahl von Lernplattformen, Marktübersicht – Funktionen – Fachbegriffe, Studien Verlag, Innsbruck, 2002, ISBN: 3-7065-1771-X

BAUMGARTNER, P., HÄFELE H., MAIER-HÄFELE, K. (2002 b): E-Learning Standards aus didaktischer Perspektive, in: Campus 2002: Die virtuelle Hochschule in der Konsolidierungsphase, Gudrun Bachmann, Odette Haefeli, Michael Kindt (Hrsg.), Medien der Wissenschaft; Band 18, Gesellschaft für Medien in der Wissenschaft e. V., Waxmann Verlag GmbH, Münster, 2002, ISBN: 3-8309-6191-X

BAUMGARTNER, P., LASKE, S., WELTE, H. (2000): Handlungsstrategien von LehrerInnen - ein heuristisches Modell. In: Impulse für die Wirtschaftspädagogik. Festschrift zum 65. Geburtstag von Prof.Dr. Rolf Dubs. C. Metzger, H. Seitz und F. Eberle. St. Gallen, Verlag des schweizerischen kaufmännischen Verbandes (SKV), 2000, S. 247-266, http://www.peter.baumgartner.name/schriften/article-de/handlungsstrategien-von-lehrerinnen-ein-heuristisches-modell (Artikel und Tabelle), Abruf am 2012-08-31

BAUMGARTNER, P., PAYR, S. (1997): Erfinden lernen. In: Konstruktivismus und Kognitionswissenschaft. Kulturelle Wurzeln und Ergebnisse. Zu Ehren Heinz von Foersters. K. H. Müller und F. Stadler. Wien-New York, Springer, 1997, S. 89-106, http://www.peter.baumgartner.name/material/article/erfinden_lernen.pdf, Abruf am 2012-08-19

BAUMGARTNER, P., PAYR, S. (1999): Lernen mit Software, 2.Aufl. Innsbruck, StudienVerlag, 1999, ISBN: 3-7065-1444-3

BEHME, H.; MINTERT, S. (2000): XML in der Praxis, 2. Auflage, Addison-Wesley Verlag, 2000, ISBN 3-8273-1636-7

BERGMANN, O., BORMANN, C. (2005): AJAX, Frische Ansätze für das Web-Design, TEIA AG – Internet Akademie und Lehrbuch Verlag, Berlin, 2005, ISBN: 3-935539-26-6

BERGLUND, A. (2006): World Wide Web Consortium (W3C): Extensible Stylesheet Language (XSL) Version 1.1, W3C Recommendation 05 December 2006, http://www.w3.org/TR/xsl11/, Abruf am 2013-08-25.

BERGLUND, A., BOAG, S., CHAMBERLIN, D., FERNANDEZ, M. F., KAY, M., ROBIE, J. SIMEON, J. (2010): World Wide Web Consortium (W3C): XML Path Language (XPath) 2.0 (Second Edition), W3C Recommendation 14 December 2010, http://www.w3.org/TR/2010/REC-xpath20-20101214/, Abruf am 2013-08-25

BERJON, R., FAULKNER, S., LEITHEAD, T., NAVARA, E. D., O'CONNOR, E., PFEIFFER, S., HICKSON, I. (2013): World Wide Web Consortium (W3C): HTML5, A vocabulary and associated APIs for HTML and XHTML, W3C Candidate Recommendation 6 August 2013, http://www.w3.org/TR/2013/CR-html5-20130806/, Abruf am 2013-10-13

BERKING, P. (2011): Choosing Authoring Tools, ADL (Advanced Distributed Learning), May 11, 2011, Version 4.8.3, http://www.adlnet.gov/wp-content/uploads/2011/07/Choosing-Authoring-Tools.pdf, Abruf am 2011-09-07

BERUFSBILDUNGSBERICHT 2002: Berufsbildungsbericht 2002, Bundesministerium für Bildung und Forschung Bildungsbrief des Fachverlag Deutscher Wirtschaftsdienst, 5/2002, http://www.bmbf.de/pub/bbb2002.pdf, Abruf am 2008-03-16

BINSTOCK, C., PETERSON, D., SMITH, M., WOODING, M., DIX, C., GALTENBERG, C. (2003): The XML Schema Complete Reference, Addison Wesley, 2003, ISBN: 0-672-32374-5

BIRBECK, M., GYLLING, M., MCCARRON, S., PEMBERTON, S. (2010): World Wide Web Consortium (W3C): XHTML™ 2.0, W3C Working Group Note 16 December 2010, http://www.w3.org/TR/2010/NOTE-xhtml2-20101216/, Abruf am 2013-04-16

BLEISCH, S., FISLER, J. (2005): eLesson Markup Language eLML – eine XML basierte Applikation für die beschreibende Auszeichnung von nachhaltigen und flexiblen e-Learning Inhalten, 3. E-Learning Fachtagung Informatik, DeLFI, 2005, Rostock, ISBN: 3-88579-395-4

BLUMSTENGEL, A. (1998): Entwicklung hypermedialer Lernsysteme, wvb Wissenschaftlicher Verlag Berlin 1998, zugl.: Paderborn, Univ., Diss., 1998, ISBN: 3-932089-13-8

BMBF (2008): Lebenslanges Lernen, Bundesministerium für Bildung und Forschung, http://web.archive.org/web/20080822031749/http://www.bmbf.de/de/411.php, Abruf am 2008-08-22

BMBF (2010 a): Lernen im Lebenslauf, Bundesministerium für Bildung und Forschung, 12.07.2010, http://www.bmbf.de/de/lebenslangeslernen.php, Abruf am 2011-08-15

BMBF (2010 b): Programm für Lebenslanges Lernen: das neue europäische Bildungsprogramm, Bundesministerium für Bildung und Forschung, 12.11.2010, http://www.bmbf.de/de/919.php, Abruf am 2011-08-15

BOCCONI, S., KAMPYLIS, P., PUNIE, Y. (2012): Innovating Teaching and Learning Practices: Key Elements for Developing Creative Classrooms in Europe, eLearning Papers, ISSN: 1887-1542, elearningeuropa.info, P.A.U. Education, S.L., Barcelona (Spain), 2012, http://www.openeducationeuropa.eu/en/download/file/fid/25386, Abruf am 2014-08-11

BODE, A., DESEL, J., RATHMAYER, S., WESSNER, M. (2003): DeLFI 2003: Die 1. E-Learning Fachtagung Informatik. Tagung der Fachgruppe E-Learning der Gesellschaft für Informatik e.V. (GI), 16. – 18. September 2003 in Garching, Köllen Druck+Verlang GmbH, Bonn, 2003, ISBN 3-88579-366-0

BOLOGNA (1999): The Bologna Declaration of 19 June 1999, Joint declaration of the European Ministers of Education, „The European Higher Education Area“, 1999, http://www.bologna-berlin2003.de/pdf/bologna_declaration.pdf , Abruf am 2011-09-10

BONGERS, F. (2004): XSLT 2.0 („Das umfassende Handbuch“), Galileo Press GmbH, Bonn, 2004, ISBN: 3-89842-361-1

BONEU, J., M. (2011): Survey on Learning Content Management Systems, , in: Ferrer, N., F., Alfonso, J. M. (2011): Content Management for E-Learning, Springer Verlag, 2011, ISBN 978-1-4419-6958-3

BÖR, A. (2003): Lernplattformen und Standards, Arbeitskreis Multimediapraxis 25. April 2003, Technische Universität München, Lehrstuhl für Kommunikationsnetze Prof. Dr.-Ing J. Eberspächer

BOS, B., CELIK, T., HICKSON, I., LIE, H. W. (2011):World Wide Web Consortium (W3C): Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification, W3C Recommendation 07 June 2011, http://www.w3.org/TR/2011/REC-CSS2-20110607/, Abruf am 2013-10-17

BOS, B., LIE, H. W., LILLEY, C., JACOBS, I. (1998):World Wide Web Consortium (W3C): Cascading Style Sheets, level 2, CSS2 Specification, W3C Recommendation 12-May-1998 (revised 11 April 2008), http://www.w3.org/TR/2008/REC-CSS2-20080411/, Abruf am 2013-10-17

BOURRET, R. (2009): XML Namespaces FAQ, Last updated January, 2009, http://www.rpbourret.com/xml/NamespacesFAQ.htm, Abruf am 2011-09-10

BRADLEY, N. (2004): The XML schema companion, Addison-Wesley, 2004, ISBN: 0-321-13617-9

BRAY, T. (1999): XML Namespaces by Example, January 19, 1999, http://www.xml.com/pub/a/1999/01/namespaces.html, Abruf am 2011-09-10

BRAY, T., HOLLANDER, D., LAYMAN, A. (1999): Namespaces in XML, World Wide Web Consortium 14-January-1999, Recommendation, http://www.w3.org/TR/REC-xml-names, Abruf am 2005-01-01

BRAY, T., HOLLANDER, D., LAYMAN, A., TOBIN, R. (2004): Namespaces in XML 1.1, W3C Recommendation, 04 February 2004, http://www.w3.org/TR/2004/REC-xml-names11-20040204, Abruf am 2005-01-01

BRAY, T., HOLLANDER, D., LAYMAN, A., TOBIN, R., THOMPSON, H. S. (2009): World Wide Web Consortium (W3C): Namespaces in XML 1.0 (Third Edition), W3C Recommendation 8 December 2009, http://www.w3.org/TR/xml-names/, Abruf am 2013-04-14

BRAY, T., PAOLI, J., SPERBERG-MCQUEEN, C. M., MALER, E., YERGEAU, F. (2008): World Wide Web Consortium (W3C): Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation 26 November 2008, http://www.w3.org/TR/2008/REC-xml-20081126/, Abruf am 2012-04-28

BRITAIN, S. (2011): On the Relationship Between Pedagogical Design and Content Management in eLearning, in: Ferrer, N., F., Alfonso, J. M. (2011): Content Management for E-Learning, Springer Verlag, 2011, ISBN 978-1-4419-6958-3

BRUGGER, R. (2002): Bewertung von Lernplattformen - Swiss Virtual Campus, Handbuch E-Learning (Kapitel 5.1.3), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 1. Erg.-Lfg. Januar 2002, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2002, ISBN: 978-3-87156-298-3

BUCHEM, I., EBNER, M., SCHÖN, S., APPELT, R., KAISER, S. (2013): Blogging und Microblogging, Anwendungsmöglichkeiten im Bildungskontext . In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version August 2013, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/128/99, Abruf am 2014-08-12

BUNSCHKOWSKI, M., RÖSER, M., TAVANGARIAN, D., VOIGT D. (2004): Eine differenzierte Metadatennotation XML-basierter E-Learning-Ressourcen für ein zuverlässiges Metadatenmanagement, Lehrstuhl Rechnerarchitektur, Institut für Informatik, Universität Rostock, in: in: Band Deutsche e-Learning Fachtagung Informatik (DeLFI), 2004, http://subs.emis.de/LNI/Proceedings/Proceedings52/GI.-.Proceedings.52-13.pdf, Abruf am 2013-08-19

CAGLE, K., CORNING, M., DIAMOND, J., DUYNSTEE, T., GUDMUNDSSON, O. G., MASON, M., PINNOCK, J., SPENCER, P., TANG, J., WATT, A. (2001): Professional XSL. Wrox Press, 2001, ISBN: 1-861003-57-9

CARLISLE, D., ION, P. (2010): World Wide Web Consortium (W3C): XML Entity Definitions for Characters, W3C Recommendation 01 April 2010, http://www.w3.org/TR/2010/REC-xml-entity-names-20100401/, Abruf am 2013-04-16

CARLISLE, D., ION, P., MINER, R. (2010): World Wide Web Consortium (W3C): Mathematical Markup Language (MathML) Version 3.0, W3C Recommendation 21 October 2010, http://www.w3.org/TR/2010/REC-MathML3-20101021/, Abruf am 2013-04-07

CASE, P., DYCK, M., HOLSTEGE, M., AMER-YAHIA, S., BOTEV, C., BUXTON, S., DOERRE, J., MELTON, J., RYS, M., SHANMUGASUNDARAM, J. (2011): World Wide Web Consortium, XQuery and XPath Full Text 1.0, W3C Recommendation 17 March 2011, http://www.w3.org/TR/2011/REC-xpath-full-text-10-20110317/, Abruf am 2012-06-03

CAVAZZA, F. (2014): Social Media Landscape 2014, http://www.fredcavazza.net/2014/05/22/social-media-landscape-2014/, Abruf am 2014-08-12

CEDEFOP (2002): eLearning und Ausbildung in Europa, Umfrage zum Einsatz von Learning zur beruflichen Aus- und Weiterbildung in der Europäischen Union, Cedefop Reference series; 25, Amt für amtliche Veröffentlichungen der Europäischen Gemeinschaften, 2002, http://www.cedefop.europa.eu/EN/Files/3021_de_short.pdf, Abruf am 2011-10-31

CEDEFOP (2008): Terminology of European education and training policy A selection of 100 key terms, European Centre for the Development of Vocational Training (Cedefop), Luxembourg: Office for Official Publications of the European Communities, 2008, ISBN 92-896-0472-7, http://www.cedefop.europa.eu/en/Files/4064_EN.PDF, Abruf am 2011-09-07

CEN/ISSS-WS/LT (2000): CEN/ISSS WS/LT WORKSHOP CWA 14040 AGREEMENT Oktober 2000, A Standardization Work Programme for “Learning and Training Technologies & Educational Multimedia Software”, 2000, http://web.archive.org/web/20030523070054/http://www.cenorm.be/isss/cwa_download_area/cwa14040.pdf, Abruf am 2014-11-30

CEN-WS-LT-EML (2013): CEN WS-LT Learning Technology Standards Observatory, Educational Modelling Languages – Overview, http://www.cen-ltso.net/(X(1)S(q5iaps45qzsfbe45v5oy0255))/main.aspx?put=196&AspxAutoDetectCookieSupport=1, Abruf am 2013-07-06

CHIRU, C., PACURARU, R., ISBASOIU, E. (2009): XML in elearning, 5th International Scientific Conference, eLearning and Software for Education, Bucharest, 2009, https://adlunap.ro/else2009/papers/1074.1.Chiru.pdf, Abruf am 2014-11-30

CLARK, J. (1999): World Wide Web Consortium (W3C): XSL Transformations (XSLT), Version 1.0, W3C Recommendation 16 November 1999, http://www.w3.org/TR/1999/REC-xslt-19991116.html, Abruf am 2013-08-11.

CLARK, J., DEROSE, S. (1999): World Wide Web Consortium (W3C): XML Path Language (XPath), Version 1.0, W3C Recommendation 16 November 1999, http://www.w3.org/TR/1999/REC-xpath-19991116/, Abruf am 2013-08-25

CLARK, R. E. (1983): Reconsidering Research on Learning from Media, Review of Educational Research. Winter, 1983, Vol. 53, No. 4, Pp. 445-459, http://www.c3l.uni-oldenburg.de/cde/media/readings/clark83.pdf, Abruf am 2012-09-02

CLIX (2005): imc Advanced Learning Solutions, Learning Management System Software http://www.im-c.de/homepage/index.htm, Abruf am 2005-04-17

COENEN, O. (2002): E-Learning-Architektur für universitäre Lehr- und Lernprozesse, 2. Auflage, Josef Eul Verlag GmbH, Lohmar, Köln, 2002, ISBN: 3-89012-834-X

CRANE, D., PASCARELLO, E. (2006): Ajax IN ACTION, Manning Publications Co. Greenwich, 2006, ISBN: 1-932394-61-3

CZAPUTA, C. (2008): Didaktische Szenarien und IMS Learning Design, Analyse zur Einschätzung einer Koppelung Didaktischer Szenarien nach Baumgartner mit der Spezifikation IMS Learning Design, Master Thesis, Department für Interaktive Medien und Bildungstechnologien, Donau-Universität Krems, 2008, http://www.peter-baumgartner.at/weblog/stuff/mtczaputa_final.pdf, Abruf am 13-08-20

DAHLSTRÖM, E., DENGLER, P., GRASSO, A., LILLEY, C., MCCORMACK, C., SCHEPERS, D., WATT, J., FERRAIOLO, J., JUN, F., JACKSON, D. (2011): World Wide Web Consortium (W3C): Scalable Vector Graphics (SVG) 1.1 (Second Edition), W3C Recommendation 16 August 2011, http://www.w3.org/TR/2011/REC-SVG11-20110816/, Abruf am 2013-04-07

DAY, D., PRIESTLEY, M., SCHELL, D. (2001): Introduction to the Darwin Information Typing Architecture, DITA, published 01 Mar 2001, IBM Corporation, http://www.ibm.com/developerworks/xml/library/x-dita1/?ca=dgr-wikiaDita1, Abruf am 2013-08-19

DEIBLER, N., DOSCH-HAWORTH, P., NORWOOD, A., KRYSTAL, S. (2008): ADL Guidelines for creating reusable Content with SCORM 2004, Version 1.0 for public comment, ADL Advanced Distributed Learning, July 31, 2008, , http://www.adlnet.gov/wp-content/uploads/2011/07/ADL_Guidelines_Creating_Reusable_Content.pdf, Abruf am 2014-01-09

DEITEL, H., M., DEITEL, P. J., NIETO, T., R., LIN, T., M., SADHU, P. (2001): XML How To Program, 2001 by Prentice-Hall, Inc., Upper Saddle River, New Jersey 07458, 2001, ISBN: 0-13-028417-3

DELFMANN, P., KAROW, M. (2008): Informationsmodellierung: Modelle, Sprachen und Methoden, ERCIS (European Research Center for Information Systems), Westfälische Wilhelms-Universität Münster, 2008, http://udoo.uni-muenster.de/modules/downloads/getDownload.php5?id=24&ei=RTxTUKy2CMbMswaLuIDQDg&usg=AFQjCNHuqSglMc5xQrrab4cce_Lkj7xDZA, Abruf am 2012-09-14

DEROSE, S., MALER, E., ORCHARD, D., WALSH, N. (2010): World Wide Web Consortium: XML Linking Language (XLink) Version 1.1, W3C Recommendation 06 May 2010, http://www.w3.org/TR/2010/REC-xlink11-20100506/, Abruf am 2012-06-02.

DEUTSCHE RECHTSCHREIBUNG (2006): Regeln und Wörterverzeichnis: Entsprechend den Empfehlungen des Rats für deutsche Rechtschreibung, München und Mannheim - Februar 2006, http://rechtschreibrat.ids-mannheim.de/download/regeln2006.pdf, Abruf am 2012-08-25

DIN 1422 (1983): Veröffentlichungen aus Wissenschaft, Technik, Wirtschaft und Verwaltung. Gestaltung von Manuskripten und Typoskripten, in: Präsentationstechnik für Dissertationen und wissenschaftliche Arbeiten: DIN-Normen, Beuth, 2000, ISBN: 3-410-14816-7

DIN 1505 (1984):Titelangaben von Dokumenten, DIN 1505-2: Titelangaben von Dokumenten: Zitierregeln (Ausgabe: Januar 1984), DIN 1505-3: Titelangaben von Dokumenten: Verzeichnisse zitierter Dokumente (Literaturverzeichnisse), in: Präsentationstechnik für Dissertationen und wissenschaftliche Arbeiten: DIN-Normen, Beuth, 2000, ISBN: 3-410-14816-7

DOCBOOK (2013): DocBook.org, Offical home page for DocBook, http://www.docbook.org/, Abruf am 2013-08-19

DODDS, P., THROPP, S. E. (2006): SCORM 2004, 3th Edition, Sharable Content Object Reference Model, Overview, VERSION 1.0, ADL Advanced Distributed Learning, OCTOBER 20, 2006, in: http://www.adlnet.gov/wp-content/uploads/2011/07/SCORM.2004.3ED.DocSuite.zip, Abruf am 2014-01-05

DÖRING, K. W. (2001): Weiterbildung: Computerlernen ist Mythos, Der Tagesspiegel, 17.02.2001, http://www.tagesspiegel.de/zeitung/weiterbildung-computerlernen-ist-mythos/203662.html, Abruf am 2012-09-02

DUBLIN CORE (2013): Dublin Core Metadata Initiative, http://www.dublincore.org/, Abruf am 2013-08-15

DUDEN (2012): Schreibweise E-Learning http://www.duden.de/rechtschreibung/E_Learning, Abruf am 2012-08-13

DUMBILL, E. (2001): From the Editor, xml.com, The Guide to W3C XML Schema, A Collection of Articles from XML.com, 2001, O'Reilly

DÜRST, M., FREYTAG, A. (2013): World Wide Web Consortium (W3C): Unicode in XML and other Markup Languages, Unicode Technical Report #20, W3C Working Group Note 24 January 2013, http://www.w3.org/TR/2013/NOTE-unicode-xml-20130124/#Unicode, Abruf am 2013-07-05

DUVAL, E. (2002 a): Learning Technology Standardization: „Too Many? Too Few?", Universität Leuven, Johann Wolfgang Goethe-Universität – Frankfurt/Main, Begleitveranstaltung zum Förderprogramm Neue Medien in der Bildung, Workshop am 10./11. April 2002, https://web.archive.org/web/20040511203433/http://www.httc.de/nmb/images/duval.pdf , Abruf am 2014-01-05

DUVAL, E. (2002 b): LOM: Learning Object Metadata, Test plan and examples, http://www.cs.kuleuven.ac.be/~erikd/LOM/20030110/LOMExamples.zip, Abruf am 2003-07-12

EBNER, A., SCHÖN, S., NAGLER, W. (2013): Einführung, Das Themenfeld "Lernen und Lehren mit Technologien". In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version August 2013, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/109/78, Abruf am 2014-08-12

ECMA-262 (2011): Standard ECMA-262, ECMAScript Language Specification, Edition 5.1, June 2011, http://www.ecma-international.org/publications/standards/Ecma-262.htm, Abruf am 2013-10-19

EHLERS, U. (2008): Qualität im E-Learning 2.0, Handbuch E-Learning (Kapitel 4.29), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 23. Erg.-Lfg. Januar 2008, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2008, ISBN: 978-3-87156-298-3

EHLERS, U. (2013): Qualitätssicherung im E-Learning, Veränderungen durch derzeitige Technologien und Konzepte. In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version August 2013, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/146/105, Abruf am 2014-08-12

EHLERS, U., PAWLOWSKI, J. M. (2006): Handbook on Quality and Standardization in E-Learning, Springer, 2006, ISBN: 3540327878

EIBL, C. J. (2006): Forschungsschwerpunkte der Informatik im Bereich E-Learning, Fachgruppe Didaktik der Informatik und E-Learning, Universität Siegen, 2006, http://www.die.informatik.uni-siegen.de/tl_files/pdf/lehre/Eibl_ELE_Expose.pdf,
Abruf am 2013-07-06

EISENBERG, D., J. (2002): SVG Essentials, Producing Scalable Vector Graphics with XML, O'Reilly, 2002, ISBN: 0-596-00223-8

ELML (2013): eLML - eLesson Markup Language, http://www.elml.org/, Abruf am 2013-08-19

ELOQ (2014): Analyse von E-Learning-Standards, ELoQ, E-Learningbasierte Logistik Qualifizierung, Technische Universität Dortmund, http://www.projekt-eloq.de/analyse-von-e-learning-standards, Abruf am 2014-01-08

ESAE (2005): European Society of Association Education, What is Lifelong Learning? The view from the European Commission, European Society of Association Education, 2005, http://web.archive.org/web/20120313143603/http://www.esae.org/articles/2007_08_005.pdf, Abruf am 2011-08-15

ETEACHING-CLIX (2012): CLIX-Steckbrief, e-teaching.org, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2012, http://www.e-teaching.org/technik/produkte/clixsteckbrief,
Abruf am 2014-06-09

ETEACHING-ILIAS (2012): ILIAS-Steckbrief, e-teaching.org, letzte Änderung: 22.09.2012, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2012, http://www.e-teaching.org/technik/produkte/iliassteckbrief, Abruf am 2014-06-09

ETEACHING-MOOC (2014): Moodle-Steckbrief, e-teaching.org, letzte Änderung: 19.03.2014, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2014, http://www.e-teaching.org/lehrszenarien/mooc/index_html, Abruf am 2014-08-12

ETEACHING-MOODLE (2013): Moodle-Steckbrief, e-teaching.org, letzte Änderung: 01.03.2013, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2013, http://www.e-teaching.org/technik/produkte/moodlesteckbrief, Abruf am 2014-06-09

ETEACHING-LMS (2012): Lernmanagement-Systeme (LMS), e-teaching.org, letzte Änderung: 23.08.2012, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2012, http://www.e-teaching.org/technik/distribution/lernmanagementsysteme/index_html,
Abruf am 2014-01-11

ETEACHING-OA (2012): Open Access und Open Content, e-teaching.org, letzte Änderung: 29.11.2012, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2012, http://www.e-teaching.org/projekt/rechte/recht_open_access/index_html, Abruf am 2014-08-12

ETEACHING-OER (2014): Open Educational Resources, e-teaching.org, letzte Änderung: 08.01.2014, Projektleitung: Prof. Dr. Dr. Friedrich W. Hesse, Portal Projekt des Leibniz-Instituts für Wissensmedien (IWM) Tübingen, 2014, http://www.e-teaching.org/didaktik/recherche/oer/index_html, Abruf am 2014-08-12

EU (2006): Beschluss Nr. 1720/2006/EG des europäischen Parlaments und des Rates vom 15. November 2006 über ein Aktionsprogramm im Bereich des lebenslangen Lernens, Amtsblatt der Europäischen Union L 327, 2006, http://www.bmbf.de/pubRD/Aktionspgr_LLL_Amtsbaltt.pdf, Abruf am 2014-11-30

EU (2009): Schlussfolgerungen des Rates vom 12. Mai 2009 zu einem strategischen Rahmen für die europäische Zusammenarbeit auf dem Gebiet der allgemeinen und beruflichen Bildung („ET 2020“), 2009/C 119/02, Der Rat der Europäischen Union, Amtsblatt der Europäischen Union, 2009, http://eur-lex.europa.eu/legal-content/DE/ALL/;ELX_SESSIONID=242HTyPDvKshbmxxzYG1nQysPy9P4L5vP1pqgxrtXkJJSf3Yzy9T!2061545948?uri=CELEX:52009XG0528(01), Abruf am 2014-08-11

EU (2011): The Lifelong Learning Programme: education and training opportunities for all, http://www.bmbf.de/en/919.php, Abruf am 2014-11-30

EU (2012): Allgemeiner Leitfaden, Aufforderung zur Einreichung von Vorschlägen – EACEA/20/2012 im Rahmen des Programms für lebenslanges Lernen (Exekutivagentur Bildung, Audiovisuelles und Kultur), Umsetzung der strategischen Ziele für die europäische Zusammenarbeit auf dem Gebiet der allgemeinen und beruflichen Bildung (ET 2020) (Zusammenarbeit der Akteure, Experimente und Innovation), Lebenslanges Lernen: Eurydice und Unterstützung politischer Maßnahmen, 2012, http://eacea.ec.europa.eu/llp/funding/2012/documents/call_et/de_ecet_call_2012_final.pdf, Abruf am 2014-08-11

EUROPÄISCHE-KOMMISSION (2005): Mitteilung der Kommission an den Rat, das Europäische Parlament, den Europäischen Wirtschafts- und Sozialausschuss und den Ausschuss der Regionen - Eine neue Rahmenstrategie für Mehrsprachigkeit /* KOM/2005/0596 endg. */, 2005, http://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:52005DC0596, Abruf am 2014-07-19

EUROPÄISCHE-KOMMISSION (2009): Bericht der Europäischen Kommission an den Rat, das Europäische Parlament, den Europäischen Wirtschafts- und Sozialausschuss und den Ausschuss der Regionen - Abschlussbericht über die Durchführung und die Auswirkungen der zweiten Phase (2000-2006) des Aktionsprogramms der Gemeinschaft in den Bereichen der allgemeinen (Sokrates) und beruflichen Bildung (Leonardo da Vinci) und des Mehrjahresprogramms (2004-2006) für die wirksame Integration von Informations- und Kommunikationstechnologien (IKT) in die Systeme der allgemeinen und beruflichen Bildung in Europa (eLearning) /* KOM/2009/0159 endg. */, 52009DC0159, 2009, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:52009DC0159:DE:NOT, Abruf am 2011-09-06

FALLSIDE, D., WALMSLEY, P (2004): XML Schema Part 0: Primer Second Edition, W3C Recommendation 28 October 2004, http://www.w3.org/TR/xmlschema-0/, dt. Übersetzung unter http://www.edition-w3c.de/TR/2001/REC-xmlschema-0-20010502/, Abruf am 2005-01-01

FLANAGAN, D. (2006): JavaScript: The Definitive Guide, Fifth Edition, O'Reilly, 2006, ISBN-13: 978-0-596-10199-2

FERRER, N., F., ALFONSO, J. M. (2011): Content Management for E-Learning, Springer Verlag, 2011, ISBN 978-1-4419-6958-3

FIDDLER (2014): Fiddler Web Debugger (v2.4.8.0), fiddler@telerik.com, http://www.telerik.com/fiddler, Abruf am 2014-04.26

FISLER, J., BLEISCH, S. (2012): eLML-eLesson Markup Language, 2012, http://www.elml.org/website/en/text/website.pdf, Abruf am 2013-08-19

FLORIAN, A. (2012): Einführung des Learning Managementsystems ILIAS an der Universität der Bundeswehr München, Dr. Alexander Florian, in: e-teaching.org, Stand: 04.02.2012, http://www.e-teaching.org/etresources/media/pdf/langtext_2012_florian-alexander_einfuehrung-des-learning-managementsystems-ilias%20.pdf, Abruf am 2014-01-11

FORBRIG, P. (2007): Objektorientierte Softwareentwicklung mit UML, 3., völlig neu überarbeitete und erweiterte Auflage, Buchreihe: Lehrbücher zur Informatik, herausgegeben von Prof. Dr. Michael Lutz und Prof. Dr. Christian Märtin, Carl Hanser Verlag München, 2007, ISBN: 3-446-40572-0

FREITAG, B. (2002 a): LMML - Eine XML-Sprachfamilie für eLearning Content, In: Informatik bewegt: Informatik 2002 - 32. Jahrestagung der Gesellschaft für Informatik e.v. (GI), 30. September - 3.Oktober 2002 in Dortmund. P-19, 349-353 (2002), GI, Gesellschaft für Informatik, Bonn, 2002, ISBN 3–88579–348–2, http://subs.emis.de/LNI/Proceedings/Proceedings19/GI-Proceedings.19-52.pdf, Abruf am 2012-09-13

FROMMER, A. (2010): Modellierung, Einführung in die Informatik und Programmierung (Informatik 1), Fachgruppe Mathematik und Informatik, Fachbereich Mathematik und Naturwissenschaften, Bergische Universität Wuppertal, Wintersemester 2009/2010, http://www-ai.math.uni-wuppertal.de/SciComp/teaching/ws09/info1/handouts/slidesA280-306_4.pdf, Abruf am 2012-09-15

FREITAG, B. (2002 b): Was ist Informationsmanagement?, Der Lehrstuhl für Informationsmanagement der Universität Passau, Prof. Dr. Burkhard Freitag, 2002, http://web.archive.org/web/20030403041120/http://www.im.uni-passau.de/Ueber_den_IM_Lehrstuhl.pdf, Abruf am 2012-09-13

FREITAG, B., SÜSS, C., DZIARSTEK, C. (2002): Adaptation und Wiederverwendung von XML-basiertem eLearning-Content, In: Informatik bewegt: Informatik 2002 - 32. Jahrestagung der Gesellschaft für Informatik e.v. (GI), 30. September - 3.Oktober 2002 in Dortmund. P-19, 354-358, 2002, ISBN 3–88579–348–2, http://subs.emis.de/LNI/Proceedings/Proceedings19/GI-Proceedings.19-53.pdf,
Abruf am 2012-09-13

GAGARINOV, A., IVERSON, M. (2005): Transforming Word Documents into the XSL-FO Format, February 2005, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/odc_wd2003_ta/html/OfficeWordWordMLtoXSL-FO.asp, Abruf am 2006-11-13

GAMPERL, J. (2007): AJAX, Grundlagen, Frameworks, APIs, 2. aktualisierte und erweiterte Auflage, Galileo Press, Bonn,2007, ISBN: 978-3-89842-857-6

GAO, S. S., SPERBERG-MCQUEEN, C. M., THOMPSON, H. S. (2012): World Wide Web Consortium (W3C): W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures, W3C Recommendation 5 April 2012, http://www.w3.org/TR/xmlschema11-1/,
Abruf am 2013-02-17

GARRETT, J. J. (2005): Ajax: A New Approach to Web Applications, February 18, 2005, http://web.archive.org/web/20080702075113/http://www.adaptivepath.com/ideas/essays/archives/000385.php, Abruf am 2013-12-08

GEEB, F. (2003): Das XML/XSLT-Seminar – Einführung für Studium und Beruf, Business Village GmbH, 2003, Göttingen, ISBN 3-934424-13-9

GEIGER, K. (2005): Planung und Erstellung eines Online-Kurses zur Vorbereitung auf das Fach Holztechnologie im Masterstudiengang Nachhaltiges Ressourcenmanagement, Diplomarbeit, Technische Universität München, Holzforschung München, München, 2005, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/holztechnologie.pdf,
Abruf am 2011-08-14

GERSDORF, R. (2003): Topic Maps zur Strukturierung von eLearning Inhalten, Lehrstuhl für Wirtschaftsinformatik, TU Dresden, 2003 ,http://www.community-of-knowledge.de/fileadmin/user_upload/attachments/topicmap_f53.pdf, Abruf am 2012-11-11

GLAHN, C. (2001): Wie Bildungsprozesse standardisiert beschrieben werden können. Konzepte, Perspektiven und Grenzen von IMS Learning Design, 5. Business Meeting Innsbruck, 29.11.2002, http://web.archive.org/web/20050527141428/http://www.learninglab.de/elan/kb3/pratical/lom/docs/abstractbildungsprozessestandard-christian-glahn.pdf, Abruf am 2013-08-20

GOLDFARB, C. F., PRESCOD, P. (2002): XML Handbook, The Chares F. Goldfarb Definitive XML Series, fourth Edition, 2002, ISBN: 0-13-065198-2

GOODMAN, D. (1998): Dynamic HTML, The Definitive Reference, A Comprehensive Resource for HTML, CSS, DOM, & JavaScript, O'Reilly, 1998, ISBN: 1-56592-494-0

GOODMAN, D. (2001): JavaScript Bible, 4th Edition, Hungry Minds, Inc., 2001, ISBN: 0-7645-3342-8

GREEN, K. C. (2013): The 24th National Survey of Computing and Information Technology in US Higher Education, The Campus Computing Projekt, Campus Computing, 17 October 2013, http://www.campuscomputing.net/sites/www.campuscomputing.net/files/CampusComputing2013_1.pdf , Abruf am 2014-01-11

GRIES, V., LUCKE, U., TAVANGARIAN, D. (2009): Werkzeuge zur Spezialisierung von XML-Sprachen für die vereinfachte, didaktisch unterstützte Erstellung von eLearning-Inhalten, in: DeLFI2009 – Die 7. E-Learning Fachtagung, Informatik der Gesellschaft für Informatik e.V., Lernen im Digitalen Zeitalter, 2009, http://www.e-learning2009.de/media/GI_P153.pdf, Abruf am 2013-07-06

GRINTSCH, K. (2011): Exposition und Exploration, Leben und Lernen im Web 2.0, 31. Juli 2011, http://www.didamedia.de/2011/07/exposition-und-exploration/, Abruf am 2012-08-23

GROSSO, P., WALSH, N. (2000): XSL Concepts and Practical Use, XML Europe 2000, Paris, France, Version 1.5, 2000, http://nwalsh.com/docs/tutorials/xsl/xsl/slides.html, Abruf am 2004-01-19

GRÖHBIEL, U., SCHIEFNER, M. (2006): Die E-Learning-Landkarte - eine Entscheidungshilfe für den E-Learning-Einsatz in der betrieblichen Weiterbildung, Handbuch E-Learning (Kapitel 3.11), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 17. Erg.-Lfg. Januar 2006, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2006, ISBN: 978-3-87156-298-3

GRÖHBIEL, U., SCHIEFNER, M. (2007): Die E-Learning-Entscheidungsmatrix, Handbuch E-Learning (Kapitel 3.15), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 19. Erg.-Lfg. Januar 2007, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2007, ISBN: 978-3-87156-298-3

GUNNER, T. B. (2008): Adaptivität im E-Learning, Benutzer- und Inhaltsmodellierung mit IMS LIP & IEEE LOM, Diplomica Verlag GmbH, Hamburg, 2008, ISBN: 978-3-8366-5715-0

HÄFELE, H. (2002): E-Learning Standards, betrachtet aus didaktischer Perspektive, http://www.qualifizierung.com/download/index.php?subcat=3&PHPSESSID=7cdd0d0eb9a479a1943d35524f89bd31, 2002, Abruf am 2003-06-17

HÄFELE, H., MAIER-HÄFELE, K. (2008): 101 e-Learning Seminarmethoden – Methoden und Strategien für die Online- und Blended-Learning-Seminarpraxis, 3. Überarb. Auflage, managerSeminare Verlags GmbH, 2008, ISBN 978-3-936075-07-6

HAMBACH, S. (2004): Vorgehensmodelle für die Entwicklung von E-Learning-Angeboten, Abteilung Multimediale Kommunikation Fraunhofer-Institut für Graphische Datenverarbeitung, Institutsteil Rostock, in: Band Deutsche e-Learning Fachtagung Informatik (DeLFI) 2004, http://subs.emis.de/LNI/Proceedings/Proceedings52/GI.-.Proceedings.52-28.pdf, Abruf am 2012-11-11

HAMEL, J. H., RYAN-JONES, D. (2002): Designing Instruction with Learning Objects, International Journal of Educational Technology, 2002, http://education.illinois.edu/ijet/v3n1/hamel/, Abruf am 2012-10-28

HANSEN, J., SEEHAGEN-MARX, H. (2013): Urheberrecht & Co. in der Hochschullehre, Rechtliche Aspekte des Technologieeinsatzes beim Lehren und Lernen. In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version August 2013, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/141/120, Abruf am 2014-08-12

VAN HARMELEN, M. (2008): Design trajectories: four experiments in PLE implementation, Independent Consultant and University of Manchester, UK, Zeitschrift: Interactive Learning Environments, Band: 16, 35-46, Verlag: Routledge, 2008, http://elgg.jiscemerge.org.uk/mark/files/-1/190/four_ple_impl_revised_and_formatted.doc, Abruf am 2014-01-11

HARNISCH, C. (2013): JavaScript everywhere, Technology Scout, in: Entwickler Magazin 5.2013, S. 48 - 52, www.entwickler-magazin.de, 2013

HAROLD, E. R. (2004): XML Das mitp-Standardwerk zur professionellen Programmierung mit XML, Sonderauflage des Titels „Die XML Bibel“, copyright 2004 by mitp-Verlag/Bonn, 2. Auflage 2004, ISBN: 3-8266-0915-8

HAROLD, E. R., MEANS, W. S. (2001): XML IN A NUTSHELL, A Desktop Quick Reference, O'REILLY, 2001, ISBN: 0-596-00058-8

HATTIE, J. A. C. (2009): Visible learning: A synthesis of over 800 meta-analyses relating to achievement, Routledge, Taylor & Francis Group, London and New York, 2009, ISBN: 0-415-47617-8

HAUG, S., SCHMIDT, M., THILLOSEN, A., WEDEKIND, J. (2012): e-teaching.org eBook, Kapitel HTML, Wedekind, J. (Hrsg.), 18.12.2012, eBook Download unter http://www.e-teaching.org/materialien/ebook/download/, Abruf am 2013-10-17

HEARN, M. (2005): Using the Mozilla Javascript interface to XSL Transformations, December 21, 2005, https://developer.mozilla.org/en-US/docs/Using_the_Mozilla_JavaScript_interface_to_XSL_Transformations, Abruf am 2013-12-26

HERMANS, H., KOPER, R., LOEFFEN, A., MANDERFELD, J., RUSMAN, M. (2000): Edubox-EML Reference Manual (Beta Version). Educational Technology Expertise Center, Open University of Netherlands, 2000, http://eml.ou.nl/eml/, Abruf am 2001-12-09

HODGINS, W. (2005): Definition Learning Object, WG12: Learning Object Metadata, IEEE Learning Technology Standards Committee, 2005, http://web.archive.org/web/20140211075748/http://ltsc.ieee.org/wg12/, Abruf am 2014-07-19

HODGINS, W., DUVAL, E. (2002): IEEE 1484.12.1-2002, Draft Standard for Learning Object Metadata, IEEE Learning Technology Standards Committee (LTSC), 15 July 2002, http://web.archive.org/web/20140213160551/http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf, Abruf am 2014-07-19

HODGINS, W., DUVAL, E., LEWIS, S. (2004): IEEE P1484.12.3/D2 Draft Standard for - Extensible Markup Language (XML) Schema Definition Language Binding for Learning Object Metadata, IEEE Learning Technology Standards Committee (LTSC), 2004, http://web.archive.org/web/20120127103908/http://ltsc.ieee.org/wg12/files/IEEE_1484_12_03_d2.pdf, Abruf am 2014-07-19

HOLZINGER, A. (2000): Basiswissen Multimedia, Band 2: Lernen, Kognitive Grundlagen mulitmedialer Informationssysteme, Vogel Buchverlag, Würzburg, 2000, ISBN: 3-8023-1857-9

HOLZNER, S. (2001): Insider XML. München: Markt + Technik Verlag. 1213 S., 2001, ISBN: 3-8272-6090-6

HORNUNG-PRÄHAUSER, V., LUCKMANN, M., KALZ, M. (Hrsg.) (2008): Selbsorganisiertes Lernen im Internet – Einblick in die Landschaft der webbasierten Bildungsinnovationen, Studienverlag Ges. m. b. h. H., Erlerstraße 10, A-6020 Innsbruck, 2008, ISBN 978-3-7065-4641-6

HORS, A. L., HEGARET, P. L., WOOD, L., NICOL, G., ROBIE, J., CHAMPION, M., BYRNE, S. (2004): Document Object Model (DOM) Level 3 Core Specification, Version 1.0, W3C Recommendation 07 April 2004, http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/DOM3-Core.html, Abruf am 2013-11-17

IFIS (2002): Institut für Informationssysteme und Softwaretechnik Universität Passau, Director: Prof. Dr. Burkhard Freitag, Business Manager: Dr. Ulrich Zukowski, 2002, http://web.archive.org/web/20080615045214/http://www.lmml.de/, Abruf am 2014-11-30

ILIAS (2003): ILIAS open source, Universität zu Köln, http://www.ilias.uni-koeln.de/, Abruf am 2003-03-28

ILIAS (2005): ILIAS, Open Source Lernplattform, http://www.ilias.de/ios/index.html, Abruf am 2005-04-17

ILIAS (2014): ILIAS, Open Source e-Learning, Open Source Lernplattform, http://www.ilias.de/, Abruf am 2014-01-16

IMS (2013): IMS Global Learning Consortium, http://www.imsglobal.org/, Abruf am 2013-08-15

IMS-CC-FAQ (2014): Common Cartridge Frequently Asked Questions, IMS Global Learning Consortium, http://www.imsglobal.org/cc/ccfaqs.html, Abruf am 2014-01-12

IMS-CP (2013): IMS Content Packaging Specification, IMS Global Learning Consortium, http://www.imsglobal.org/content/packaging/, Abruf am 2013-08-21

IMS-LD (2003): IMS Learning Design Specification, IMS Global Learning Consortium, Version 1.0 Final Specification, 2003, http://www.imsglobal.org/learningdesign/index.cfm, Abruf am 2013-08-20

IMS-MD (2013): IMS Learning Resource Meta-data Specification, Version 1.3 - Final Specification, IMS Global Learning Consortium, http://www.imsglobal.org/metadata/, Abruf am 2013-08-23

IMS-QTI (2013): IMS Question & Test Interoperability Specification, IMS Global Learning Consortium, http://www.imsglobal.org/question/, Abruf am 2013-08-24

INTERNET-WORLD-STATS (2011): Internet World Stats, Usage and Population Statistics, Miniwatts Marketing Group. All rights reserved worldwide. Page updated on Oct 6, 2011, http://www.internetworldstats.com/stats.htm, Abruf am 2011-11-20

INTERNET-WORLD-STATS-LANGUAGES (2011): Top Ten Languages Used in the Web (Number of Internet Users by Language), Internet World Stats, Miniwatts Marketing Group. All rights reserved worldwide, 2011, http://www.internetworldstats.com/stats7.htm, Abruf am 2014-07-27

ISSING, L. (2002): Instruktions-Design für Multimedia, In: Information und Lernen mit Multimedia und Internet, Lehrbuch für Studium und Praxis, 3., vollständig überarbeitete Auflage 2002, Herausgeber: Prof. Dr. Ludwig J. Issing, freie Universität Berlin und Prof. Dr. Paul Klimsa, Technische Universität Ilmenau, Verlagsgruppe Beltz, Psychologie Verlags Union, Weinheim, 2002, ISBN: 3-621-27449-9

ISSING, L. J., KLIMSA, P. (2002): Information und Lernen mit Multimedia und Internet, Lehrbuch für Studium und Praxis, 3., vollständig überarbeitete Auflage 2002, Herausgeber: Prof. Dr. Ludwig J. Issing, freie Universität Berlin und Prof. Dr. Paul Klimsa, Technische Universität Ilmenau, Verlagsgruppe Beltz, Psychologie Verlags Union, Weinheim, 2002, ISBN: 3-621-27449-9

ISO-639-1 (2002): ISO (International Organization for Standardization), ISO 639-1:2002, Codes for the representation of names of languages -- Part 1: Alpha-2 code, 2002, http://www.iso.org/iso/catalogue_detail?csnumber=22109, Abruf am 2014-03-09

ISO-8879 (1986): ISO (International Organization for Standardization), ISO 8879:1986, Information processing -- Text and Office Systems -- Standard Generalized Markup Language (SGML), 1986, http://www.iso.org/iso/catalogue_detail.htm?csnumber=16387, Abruf am 2013-01-15

ISO-12785 (2009): ISO/IEC 12785-1:2009: Information technology -- Learning, education, and training -- Content packaging -- Part 1: Information model
ISO/IEC 12785-2:2011: Information technology -- Learning, education, and training -- Content packaging -- Part 2: XML binding
ISO/IEC TR 12785-3:2012: Information technology -- Learning, education, and training -- Content packaging -- Part 3: Best practice and implementation guide, 2009, http://www.iso.org/iso/home/search.htm?qt=12785&published=on&active_tab=standards&sort_by=rel, Abruf am 2013-08-21

ISO-13250 (2003): ISO/IEC 13250:2003, Information technology -- SGML applications -- Topic maps, 2003, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=38068, Abruf am 2012-11-11

ISO-16262 (2011): ISO/IEC 16262:2011, Information technology -- Programming languages, their environments and system software interfaces -- ECMAScript language specification, 2011, http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=55755, Abruf am 2013-01-15

ISO-19505 (2012): ISO/IEC 19505:2012, Information technology -- Object Management Group Unified Modeling Language (OMG UML), 2012, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=32624, Abruf am 2013-02-10

ISO-19796-1 (2005): ISO/IEC 19796-1:2005, Information technology -- Learning, education and training -- Quality management, assurance and metrics -- Part 1: General approac, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=33934, deutsch: DIN EN ISO/IEC 19796-1:2009-08 (D) Informationstechnik - Lernen, Ausbilden und Weiterbilden - Qualitätsmanagement, - sicherung und -metriken - Teil 1: Allgemeiner Ansatz (ISO/IEC 19796-1:2005); Deutsche Fassung EN ISO/IEC 19796-1:2009,): ISO/IEC 19796-1 aus dem Jahr 2005, http://www.beuth.de/cmd%3Bjsessionid=2654F64378F7829CA3CB7BA36BB264F8.4?workflowname=infoInstantdownload&docname=1529377&contextid=beuth&servicerefname=beuth&ixos=toc, Abruf am 2012-08-31

ISO-22537 (2006): ISO/IEC 22537:2006: Information technology -- ECMAScript for XML (E4X) specification, 2006, http://www.iso.org/iso/catalogue_detail.htm?csnumber=41002, Abruf am 2013-11-17

ISO-29163 (2009): ISO/IEC TR 29163-1:2009, Information technology -- Sharable Content Object Reference Model (SCORM®) 2004 3rd Edition -- Part 1: Overview Version 1.1, 2009, https://www.iso.org/obp/ui/#iso:std:iso-iec:tr:29163:-1:ed-1:v1:en, Abruf am 2014-01-08

ISO-29990 (2010): ISO 29990:2010 Learning services for non-formal education and training -- Basic requirements for service providers, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=53392 , deutsch: DIN ISO 29990:2010-12 (D) Lerndienstleistungen für die Aus- und Weiterbildung - Grundlegende Anforderungen an Dienstleister (ISO 29990:2010), 2010, http://www.beuth.de/de/norm/din-iso-29990/135409271?SearchID=433621376, Abruf am 2012-08-31

JAHNKE, T. (2006): Was ist Didaktik?, Vorlesung WS 06/07, Seminar 7.11.06, Einführung in die Mathematikdidaktik, Prof. Dr. T. Jahnke, Lehrstuhl für Didaktik der Mathematik, Universität Potsdam, 2006, www.math.uni-potsdam.de/prof/o_didaktik/aa/Veran/Archiv/Was_ist_Didaktik.ppt, Abruf am 2012-08-12

JAKOBS, K. (2000): Standardisation Processes in It: Impact, Problems and Benefits of User Participation, Series: Professional Computing, Springer, 2000, ISBN 978-3-528-05689-6, siehe auch online unter Google books: http://books.google.de/books?id=u7dZfS9ZHaYC&pg=PA9&hl=de&source=gbs_toc_r&cad=4#v=onepage&q&f=false, Abruf am 2012-09-02

JELITTO, M. (2012): Links zu "Evaluation von Lernplattformen", Linksammlung (erstellt im Rahmen des BMBF-geförderten Projektes MMISS), 2012, http://www.evaluieren.de/infos/links/plattfor.htm, Abruf am 2014-01-11

JESUKIEWICZ, P. (2009 a): SCORM 2004, 4th Edition, Content Aggregation Model (CAM), Version 1.1, Advanced Distributed Learning, August 14, 2009, ADL in: http://www.adlnet.gov/wp-content/uploads/2011/07/SCORM_2004_4ED_v1_1_Doc_Suite.zip, Abruf am 2014-01-05

JESUKIEWICZ, P. (2009 b): SCORM 2004, 4th Edition, Run-Time Environment (RTE), Version 1.1, ADL Advanced Distributed Learning, August 14, 2009, in: http://www.adlnet.gov/wp-content/uploads/2011/07/SCORM_2004_4ED_v1_1_Doc_Suite.zip, Abruf am 2014-01-05

JESUKIEWICZ, P. (2009 c): SCORM 2004, 4th Edition, Sequencing and Navigation (SN), Version 1.1, ADL Advanced Distributed Learning, August 14, 2009, in: http://www.adlnet.gov/wp-content/uploads/2011/07/SCORM_2004_4ED_v1_1_Doc_Suite.zip, Abruf am 2014-01-05

JISC (2008): JISC Study of Shared Services in UK Further and Higher Education, Report 2: The software currently in use for administrative systems in UK FE and HE, Undertaken on behalf of the JISC, Duke & Jordan Ltd with AlphaPlus Ltd, Mary Auckland, Chris Cartledge,Simon Marsden and Bob Powell, April 2008, http://www.jisc.ac.uk/media/documents/programmes/jos/sharedservicesreport2.pdf, Abruf am 2014-01-10

JOHNSON, L., ADAMS BECKER, S., ESTRADA, V., FREEMAN, A. (2014): NMC Horizon Report: 2014 Higher Education Edition. Deutsche Ausgabe (Übersetzung: Helga Bechmann, Multimedia Kontor Hamburg). Austin, Texas: The New Media Consortium, 2014, ISBN 978-0-9897335-7-1, siehe auch online unter http://www.mmkh.de/fileadmin/dokumente/Publikationen/2014-Horizon-Report-HE_German.pdf, Abruf am 2014-08-11

JUNGMANN, B. (2012): Wiederverwendung von Contents im E-Learning: Entwicklung eines interdisziplinären Konzepts, AV Akademikerverlag, 2012, ISBN: 978-3-639-39489-4

JUNGMANN, B., WIRTH, K., PETZOLD, O., KLAUSER, F., SCHOOP, E. (2004): Didaktische Funktionen und deren Umsetzung in DTD´s: Ein interdisziplinäres Regelwerk für die Ausgestaltung netzbasierter Lernangebote, 2004, http://www.bow.uni-osnabrueck.de/ReRe_6_1.pdf, Abruf am 2014-01-08

KAISER, S. (2003): Mündliche Auskunft auf einem Vortrag im Medienzentrum der Technischen Universität München am 25.04.2003, Siglinde Kaiser, DIN Deutsches Institut für Normung e. V., Referat Entwicklungsbegleitende Normung, Burggrafenstr. 6, 10787 Berlin

KALZ, M., SCHÖN, S., LINDER, M., ROTH, D., BAUMGARTNER, P. (2011): Systeme im Einsatz, Lernmanagement, Kompetenzmanagement und PLE. In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version vom 26. April 2011, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/39/66, Abruf am 2014-01-09

KAY, M. (2001): XSLT, 2nd Edition, Programmer's Reference, Wrox Press, 2001, ISBN: 1-861005-06-7

KAY, M. (2007): World Wide Web Consortium (W3C): XSL Transformations (XSLT) Version 2.0, W3C Recommendation 23 January 2007, http://www.w3.org/TR/2007/REC-xslt20-20070123/, Abruf am 2012-06-03

KECHER, C. (2006): UML 2.0, Das umfassende Handbuch, Galileo Press, Bonn, 2006, ISBN: 978-3-89842-738-8

KERRES, M. (2012): Mediendidaktik, Konzeption und Entwicklung mediengestützter Lernangebote von Prof. Dr. Michael Kerres, 3. vollständig überarbeitete Auflage, Oldenbourg Verlag München, 2012, ISBN 978-3-486-27207-9

KERRES, M. (2006): Potenziale von Web 2.0 nutzen, in: Handbuch E-Learning (Kapitel 4.0), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 35. Erg.-Lfg. Januar 2011, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2006, ISBN: 978-3-87156-298-3

KERRES, M., JECHLE, T. (2001): Didaktische Konzeption des Tele-Lernens, In: L. J. Issing und P. Klimsa (Hrsg.), Information und Lernen mit Multimedia, 2. Auflage, Weinheim: Beltz, 2001, siehe auch http://mediendidaktik.uni-duisburg-essen.de/system/files/kerres-jechle4issing_0.pdf, Abruf am 2012-08-28

KIEDROWSKI, J., BROMBERGER, N. (2006): Open Source Learning Management Systeme: Von der Randerscheinung zur akzeptierten Lösung in Wirtschaft, Verwaltung und Hochschule, in: Praxisbuch E-Learning: Ein Reader des Kölner Expertennetzwerkes cel_C, Hrsg.: Stefan Ludwigs, Ulrike Timmler, Martin Tilke, W. Bertelsmann Verlag, Bielefeld, 2006, ISBN: 3-7639-3416-2

KIM, L. (2003): XMLSPY Handbook, Wiley Publishing, Inc., 2003, ISBN: 0-7645-4964-2

KLEBL, M. (2004): Lehrprozesse planen, Lernprozesse strukturieren mit IMS Learning Design, Lehrstuhl für Arbeitswissenschaft und Betriebspädagogik, Philosophisch-Pädagogische Fakultät, Katholische Universität Eichstätt-Ingolstadt, 2004, http://www.elive-ld.com/elive/content/e56/e806/Lernprozessemodellieren_ger.PDF, Abruf am 2013-01-15

KLIMSA, P. (1993): Neue Medien und Weiterbildung: Anwendung und Nutzung in Lernprozessen der Weiterbildung, Deutscher Studien Verlag, 1993, ISBN: 3892714320

KLOTZ, L. (2003): XSLT & XPATH absolute path, Stylus Studio Developer Network, 2003, http://www.stylusstudio.com/xmldev/200309/post70030.html, Abruf am 2005-04-15

KNEBEL, N. (2002): OUNL EML and standards, 2002, http://learningnetworks.org/forums/showthread.php?s=&threadid=26, Abruf am 2003-06-20

KÖHLER, T (2010): E-Learning als Ko-Konstruktion von Wissen: sind die Prinzipien der E-Learning-Didaktik (schon) übertragbar?, Keynote - Prof. Dr. Thomas Köhler, 2. Symposium E-Learning an der TU Dresden, 09. – 10.03.2010, Aufzeichnung des Vortrages mit Ton und Folien: http://proel.wcms-file3.tu-dresden.de/Keynote_Koehler/Keynote_Koehler.html, Abruf am 2012-08-16

KOPER, R. (2000): From change to renewal: Educational technology foundations of electronic learning Environments, Educational Technology Expertise Center, Open University of Netherlands, 2000, http://web.archive.org/web/20060204183427/http://eml.ou.nl/introduction/docs/koper-inaugural-address.pdf, Abruf am 2013-08-20

KOPER, R. (2001): Modeling units of study from a pedagogical perspective, Open University of Netherlands, 2001, http://web.archive.org/web/20050514215136/http://eml.ou.nl/introduction/docs/ped-metamodel.pdf, Abruf am 2002-01-20

KOPER, R. (2002): Educational Modelling Language: adding instructional design to existing specifications, Open University of the Netherlands, Otec, Workshop: Standardisierung im eLearning, Johann Wolfgang Goethe-Universität, Frankfurt/Main, 10./11. April 2002, http://web.archive.org/web/20040511155852/http://www.httc.de/nmb/images/koper_folien.pdf, Abruf am 2013-08-20

KOPER, R. (2003): Combining re-usable learning resources and services to pedagogical purposeful units of learning. In A. Littlejohn (Ed.), Reusing Online Resources: A Sustainable Approach to eLearning (pp. 46-59). London: Kogan Page, 2003, http://dspace.ou.nl/bitstream/1820/39/2/Combining-preprint.pdf, Abruf am 2012-09-16

KOPER, R. (2013): Questions and Tests, About Question, Test, Assessment Tools & Standards, http://portal.ou.nl/en/web/rkp/wiki/-/wiki/Questions+and+Tests/Start, Abruf am 2013-08-24

KOPER, R., OLIVIER, B, ANDERSON, T. (2003 a): IMS Learning Design Information Model, Version 1.0, IMS Global Learning Consortium, http://www.imsproject.org/learningdesign/ldv1p0/imsld_infov1p0.html, Abruf am 2013-08-20

KOPER, R., OLIVIER, B, ANDERSON, T. (2003 b): IMS Learning Design Best Practice and Implementation Guide, Version 1.0, IMS Global Learning Consortium, http://www.imsproject.org/learningdesign/ldv1p0/imsld_bestv1p0.html, Abruf am 2003-06-18

KOPER, R., OLIVIER, B, THOR, A., NORTON, M. (2003 c): IMS Learning Design XML Binding, Version 1.0, IMS Global Learning Consortium, http://www.imsproject.org/learningdesign/ldv1p0/imsld_bindv1p0.html, Abruf am 2003-06-19

KOPKA, C., SCHMEDDING, D., SCHRÖDER, J. (2004): Der Unified Process im Grundstudium, Didaktische Konzeption, Einsatz von Lernmodulen und Erfahrungen, in: Band Deutsche e-Learning Fachtagung Informatik (DeLFI) 2004, Lehrstuhl für Software-Technologie, Fachbereich Informatik der Universität Dortmund, 2004, http://subs.emis.de/LNI/Proceedings/Proceedings52/GI.-.Proceedings.52-12.pdf, Abruf am 2012-11-11

KORNELSEN, L., VOIGT, D. (2005): Dokumentation zum WWR Build Tool v2.9, Stylesheets für die Modultransformation, Lehrstuhl Rechnerarchitektur, Universität Rostock , Stand 12.12.2005, http://www.ml-3.org/ML3/1.2/docu/transformation_authoring/build_tool_documentation_german.pdf, Abruf am 2013-08-16

KORNELSEN, L., LUCKE, U., TAVANGARIAN, D., WALDHAUER, M., OSSIPOVA, N. (2004): Strategien und Werkzeuge zur Erstellung multimedialer Lehr- und Lernmaterialien auf Basis von XML, in: Engels, G., Seehusen, S. (Hrsg.): DeLFI 2004, Tagungsband der 2. e-Learning Fachtagung Informatik, 6. - 8. September 2004, Paderborn, Germany, Gesellschaft für Informatik, 2004, ISBN: 3-88579-381-4

KRAAN, W., LAY, S., GORISSEN, P. (2012 a): IMS Question & Test Interoperability Overview, Version: 2.1 Final, IMS Global Learning Consortium, Date Issued: 31 August 2012, http://www.imsglobal.org/question/qtiv2p1/imsqti_oviewv2p1.html, Abruf am 2013-08-24

KRAAN, W., LAY, S., GORISSEN, P. (2012 b): IMS Question & Test Interoperability Assessment Test, Section and Item Information Model, Version: 2.1 Final, IMS Global Learning Consortium, Date Issued: 31 August 2012, http://www.imsglobal.org/question/qtiv2p1/imsqti_infov2p1.html, Abruf am 2013-08-24

KRAAN, W., LAY, S., GORISSEN, P. (2012 c): IMS Question & Test Interoperability Implementation Guide, Version: 2.1 Final, IMS Global Learning Consortium, Date Issued: 31 August 2012, http://www.imsglobal.org/question/qtiv2p1/imsqti_implv2p1.html, Abruf am 2013-08-24

KRAAN, W., LAY, S., GORISSEN, P. (2012 d): IMS Question & Test Interoperability Implementation Guide, Version: 2.1 Final, IMS Global Learning Consortium, Date Issued: 31 August 2012, XML-Beispieldatei choice.xml, http://www.imsglobal.org/question/qtiv2p1/examples/items/choice.xml, Abruf am 2013-08-24

KRAAN, W., LAY, S., GORISSEN, P. (2012 e): IMS Question & Test Interoperability Integration Guide, Version: 2.1 Final, IMS Global Learning Consortium, Date Issued: 31 August 2012, http://web.archive.org/web/20131123193506/http://www.imsglobal.org/question/qtiv2p1/imsqti_intgv2p1.html, Abruf am 2014-11-30

LAUR-ERNST, U. (2002): eLearning – eine Bedingung für lebenslanges Lernen, Berufsbildung für eine globale Gesellschaft, Perspektiven im 21. Jahrhundert, BiBB, Dokumentation 4. BIBB-Fachkongress, 2002, http://www.bibb.de/redaktion/fachkongress2002/cd-rom/PDF/04_P_01A.pdf, Abruf am 2011-09-06

LE, S., WEBER, P., EBNER, M. (2013): Game-Based Learning, Spielend Lernen? In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version August 2013, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/120/102, Abruf am 2014-08-12

LEISEGANG, C., MINTERT, S. (2006): XSLT im webbrowser, Es kann nur zwei geben, Zeitschrift iX 10/2006, Heise Zeitschriften Verlag, 2006

LEISEGANG, C., MINTERT, S., SPANNEBERG, B. (2007): Äpfel und Birnen, fünf clientseitige Ajax-Frameworks, Die Nullnummern, fünf serverseitige Ajax-Frameworks in: iX SPECIAL 1/2007, Web 2.0 – das Kompendium, S. 30 – 41, Heise Zeitschriften Verlag, 2007

LIAM, R. E. Q. (2010 a): XML ESSENTIALS, Liam R. E. Quin, liam@w3.org, http://www.w3.org/standards/xml/core, Abruf am 2012-04-28

LIAM, R. E. Q. (2010 b): SCHEMA, Liam R. E. Quin, liam@w3.org, http://www.w3.org/standards/xml/schema, Abruf am 2012-04-28

LIAM, R. E. Q. (2010 c): PUBLISHING, Liam R. E. Quin, liam@w3.org, http://www.w3.org/standards/xml/publishing, Abruf am 2013-08-25

LIAM, R. E. Q. (2010 d): TRANSFORMATION, Liam R. E. Quin, liam@w3.org, http://www.w3.org/standards/xml/publishing, Abruf am 2013-08-25

LIE, H. W., BOS, B. (1996):World Wide Web Consortium (W3C): Cascading Style Sheets, level 1, W3C Recommendation 17 Dec 1996, revised 11 Apr 2008, http://www.w3.org/TR/2008/REC-CSS1-20080411/, Abruf am 2013-10-17

LIE, H. W., BOS, B. (1997): Cascading Style Sheets: Layouts für das Web-Publishing, Addison Wesley Longman GmbH, 1997, ISBN: 3-8273-1257-4

LOBIN, H. (2000): Informationsmodellierung in XML und SGML, Springer Berlin Heidelberg; Auflage: 1. Aufl. 2000, ISBN: 3-54065-356-2

LOBIN, H., STÜHRENBERG, M., REHM, G. (2003): eLearning und offene Standards: zum Einsatz XML-strukturierter Lernobjekte, 2003, http://www.uni-giessen.de/~g91062/pdf/Lobin-et-al-SDV-2003.pdf, Abruf am 2012-09-13

LÖSER, A., GRUNE, C., HOFFMANN, M. (2003): Didaktisches Modell, Taxonomie von Lernobjekten und Auswahl von Metadaten für ein Online Curriculum. TU-Berlin, Germany, Technical Report 2003/14, http://www.relearn.de/wp-content/uploads/2008/09/loeser-grune-hoffman_didaktisches-modell.pdf,
Abruf am 2014-04-10

LOVISCACH, J. (2013): Stanford per Steckdose, neue Wege zum Lehren und Lernen im Internet. In: iX Kompakt 2/2013 – Beruf und Karriere, S. 105 - 107, 2013, Zeitschrift Heise-Verlag

LUKESCH, H. (2007): Lernen und Lehren: Lehren/Lehrmethoden, Prof. Dr. Helmut Lukesch, Institut für Experimentelle Psychologie, Universität Regensburg, 2007, http://www-app.uni-regensburg.de/Fakultaeten/PPS/Psychologie/Lukesch/downloads/Lehre/Lukesch/vl01lehrmethoden.pdf, Abruf am 2012-08-19

MARESCH, G. (2008): Blended-Learning-Didaktik, Österreichisches Zentrum für Begabtenförderung und Begabungsforschung (Hrsg.), Studienverlag Ges. m. b. h. H., Erlerstraße 10, A-6020 Innsbruck, 2008, ISBN 978-3-7065-4550-1

MARTINEZ-ORTIZ, I., MORENO-GER, P. SIERRA, J. L., FERNANDEZ-MANJON, B. (2007): Educational Modeling Languages, A Conceptual Introduction and a High-Level Classification, in: Fernandez-Manjon, B., Sanchez-Perez, J. M., Gomez-Pulido, J. A., Vega-Rodriguez, M. A., Bravo-Rodriguez, J. (2007): Computer and Education, E-Learning, From Theory to Practice, Springer-Verlag, 2007, ISBN: 978-1-4020-4913-2

MATTSON, L. (2014): IMS Global Common Cartridge Profile: Implementation, Version 1.3 Final Specification, IMS Global Learning Consortium, 30 May 2014, http://www.imsglobal.org/cc/ccv1p3/imscc_Implementation-v1p3.html,
Abruf am 2014-11-30

MCCARRON, S., AUSTIN, D., PERUVEMBA, S., MCCARRON, S., ISHIKAWA, M., BIRBECK, M. (2010): World Wide Web Consortium (W3C): XHTML™ Modularization 1.1 - Second Edition, W3C Recommendation 29 July 2010, http://www.w3.org/TR/2010/REC-xhtml-modularization-20100729/, Abruf am 2013-04-07

MDN-E4X (2013): MDN, Mozilla Developer Network, Dokumentation, E4X, https://developer.mozilla.org/de/docs/E4X, Abruf am 2013-11-17

MEEKER, M. (2012): INTERNET TRENDS @ STANFORD – BASES KICK OFF, KPCB Kleiner Perkins Caufield Byers, 12/3/2012, http://www.businessinsider.com/mary-meeker-2012-internet-trends-year-end-update-2012-12?op=1, Abruf am 2014-07-27

MEIER, R. (2006): Praxis E-Learning - Grundlagen, Didaktik, Rahmenanalyse, Medienauswahl, Qualifizierungskonzept, Betreuungskonzept, Einführungsstrategie, Erfolgssicherung, GABAL Verlag GmbH, Offenbach, 2006, ISBN 3-89749-595-3

MEISSNER, K., RÖTTGER, S., WEHNER, F. (2001) Dynamische Visualisierung modularer, XML-basierter Kursdokumente, Technische Universität Dresden, in: Proceedings of '11. Arbeitstreffen der GI-Fachgruppe 1.1.5/7.0.1 Intelligente Lehr-/Lernsysteme', Dortmund, Germany, 11/2001

MICHEL, T. (1999): XML kompakt: eine praktische Einführung. Wien: Hanser-Verlag. 240 S., 1999, ISBN: 978-3-4462-1302-9

MICROSOFT-CREATEPROCESSOR-METHOD (2013): createProcessor Method, Microsoft Developer Network, http://msdn.microsoft.com/en-us/library/ms753809(v=vs.85).aspx, Abruf am 2013-12-26

MICROSOFT-WORD-XML-MARKUP (2014): Microsoft, Hilfe und Support, „Benutzerdefiniertes XML-Markup wird entfernt, wenn Sie ein Dokument in Word 2010 öffnen“, http://support.microsoft.com/kb/2445060/de, Abruf am 2014-04-14

MINTERT, S., KÜHNEL, C. (2000): Workshop JavaScript, Addison-Wesley Verlag, 2000, ISBN: 3-8273-1718-5

MINTERT, S. (2012): Datenbanken: Im Browser Daten lokal verwalten, in: iX Kompakt 2/2012 – Webdesign, Heise-Verlag, 2012

ML3 (2005): <ML>³ Multidimensional LearningObjects and Modular Lectures Markup Language, Universität Rostock, http://www.ml-3.org/, Last Update: December 12th 2005, Abruf am 2013-08-16

MONNET, C., LINKER, W. J. (2011): Blende(n)d lernen mit Aktivierendem Lernen/Accelerated Learning (AL), Handbuch E-Learning (Kapitel 4.41), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 37. Erg.-Lfg. Januar 2011, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2011, ISBN: 978-3-87156-298-3

MONTANDON, C. (2004): Standardisierung im e-Learning, Eine empirische Untersuchung an Schweizer Hochschulen, Institut für Wirtschaftsinformatik der Universität Bern, Arbeitsbericht Nr. 161, 2004, http://www.iwi.unibe.ch/content/publikationen/arbeitsberichte/2004/e6050/e6133/e7162/e7164/e7170/AB161.pdf, Abruf am 2011-11-01

MOODLE (2014): Moodle, Welcome to the Moodle community!, https://moodle.org/, Abruf am 2014-01-17

MSDN-ASYNC (2014): async Property, Microsoft Developer Network, http://msdn.microsoft.com/en-us/library/ms761398(v=vs.85).aspx, Abruf am 2014-01-03

MSDN-DOM-REFERENCE (2014): DOM Reference, Übersicht, Microsoft Developer Network, http://msdn.microsoft.com/en-us/library/ms764730(v=vs.85).aspx, Abruf am 2014-01-03

MSDN-MSXML3 (2000): MSDN Magazine, The XML Files: MSXML 3.0 Supports XPath 1.0, XSLT 1.0, XDR, and SAX2, Aaron Skonnard, http://msdn.microsoft.com/en-us/magazine/cc302348.aspx, Abruf am 2013-12-27

MSDN-XMLHTTPREQUEST (2014): XMLHttpRequest object, Internet Explorer Dev Center, Build date: 11/17/2013, http://msdn.microsoft.com/en-us/library/ie/ms535874(v=vs.85).aspx, Abruf am 2014-01-02

MÜNZ, S. (2003): DHTML, Dynamisches HTML – Bringen Sie unkompliziert und professionell Schwung in Ihre Webseiten, Franzis‘ Verlag GmbH, 2003, ISBN: 3-7723-6695-3

MÜNZ, S., GULL, C. (2012): HTML 5 Handbuch, 2. Aktualisierte und erweiterte Auflage, Franzis Verlag GmbH, 2012, ISBN: 978-3-645-60151-1

MÜNZ, S., KESSLER, A. (2001): Cascading Style Sheets, Internet intern, Data Becker GmbH & Co. KG, 2001, ISBN: 3-8158-2102-9

NÁIRON, J. C. G. (2007): Manipulating XML using JavaScript, El Micox Codes (Codes, Javascript, AJAX, Webstandards, CSS, DOM, XML , PHP, ASP, HTTP, etc.), Micox, 2007, http://elmicoxcodes.blogspot.gr/2007/02/manipulating-xml-using-javascript.html, Abruf am 2014-05-30

NIEGEMANN, H. M., HESSEL, S., HOCHSCHEID-MAUEL, D., ASLANSKI, K., DEIMANN, M., KREUZBERGER, G. (2004): Kompendium E-Learning, Springer-Verlag Berlin Heidelberg, 2004, ISBN 1439-3107

NÖBAUER, M (2005): JavaScript Grundlagen Fachbegriffe, http://www.exine.de/clientseitig/js_work_fachbegriffe.htm, Abruf am 2005-04-15

OASIS (2001): Oasis and Robin Cover: The SGML/XML Web Page. SGML and XML News. http://www.oasis-open.org/cover/sgmlnew.html, Abruf am 2001-12-09

OLAT (2014): OLAT (Online Learning and Training), Open Source LMS, Universität Zürich, http://www.olat.org/, Abruf am 2014-01-16

OLIVIER, B., TATTERSALL, C. (2005): The Learning Design Specification, in: Koper R., Tattersall C.: Learning Design: A Handbook on Modelling and Delivering Networked Education and Training, Berlin: Springer Verlag, 2005, ISBN: 3-540-22814-4

ONESTAT (2006): Global usage share Mozilla Firefox has increased according to OneStat.com, 09.06.2006, http://www.onestat.com/html/aboutus_pressbox44-mozilla-firefox-has-slightly-increased.html, Abruf am 2006-11-13

OSTYN, C (2006): ISO 8601 duration and time stamps in SCORM 2004, August 2006, http://www.ostyn.com/standards/scorm/samples/ISOTimeForSCORM.htm#timestampfmt (ostynscormtime.js), Abruf am 2014-06-14

OSTYN, C (2007): In the Eye of the SCORM, An introduction to SCORM 2004 for Content Developers, Update 0.9-8.8– March 2007, http://www.ostyn.com/standards/docs/Eye_Of_The_SCORM_draft.pdf, Abruf am 2014-01-07

OSTYN, C (2010): Sample SCORM related scripts, Generic reusable script for SCORM 2004 conformant SCOs, ostyn2004sco.js, http://www.ostyn.com/standards/scorm/samples/ostyn2004sco.js, Abruf am 2010-01-05

PAAR, S. (2002): Die Strukturierung von Lehreinheiten mit XML-basierten Modellierungssprachen, Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 14. Tagung 2002 Tharandt, Die Grüne Reihe, Tagungsband herausgegeben von M. Köhl (TU Dresden) und H.-D.Quednau (TU München), 2002, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/paar.html, Abruf am 2011-09-11

PAAR, S. (2003): Standardisierungsbemühungen im Bereich E-Learning am Beispiel von IMS LD und IEEE LOM, Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 15. Tagung 2003 Freiburg, Die Grüne Reihe, Tagungsband herausgegeben von U. Wunn (Forschungsanstalt für Waldökologie und Forstwirtschaft Rheinland-Pfalz in Trippstadt) und H.-D.Quednau (TU München), 2003, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/paar2.html, Abruf am 2011-09-11

PAAR, S. (2004): Der Einsatz von XSLT im Bereich E-Learning am Beispiel von IMS LD und LMML, Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 16. Tagung 2004 Freising, Die Grüne Reihe, Tagungsband herausgegeben von U. Wunn (Forschungsanstalt für Waldökologie und Forstwirtschaft Rheinland-Pfalz in Trippstadt) und H.-D.Quednau (TU München), 2004, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/paar3.html, Abruf am 2011-09-11

PAAR, S., STREHL, O., QUEDNAU, H. D. (2006): Clientseitige Verarbeitung von XML (WLMML)-Dateien mit XSLT und JavaScript, Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 18. Tagung in Trippstadt 2006, Die Grüne Reihe, Tagungsband herausgegeben von A. Degenhard und U. Wunn 2006 (Forschungsanstalt für Waldökologie und Forstwirtschaft Rheinland-Pfalz in Trippstadt), 2006, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/Trippstadt_Paar.html, Abruf am 2011-09-11

PARTL (2002): Universität für Bodenkultur: Zentraler Informatik Dienst ZID, http://www.boku.ac.at/htmleinf/xmlkurz.html#applikationen, Abruf am 2002-01-21.

PAWLOWSKI, J. M. (2001): Das Essener-Lern-Modell (ELM): Ein Vorgehensmodell zur Entwicklung computerunterstützter Lernumgebungen, Dissertation, Fachbereich Wirtschaftswissenschaften der Universität Essen, 2001, http://deposit.d-nb.de/cgi-bin/dokserv?idn=963514113&dok_var=d1&dok_ext=pdf&filename=963514113.pdf, Abruf am 2003-06-20

PAWLOWSKI, J. M. (2002): Modellierung didaktischer Konzepte mit dem Essener-Lern-Modell. Standardisierung im e-Learning, Workshop. 10./11.04.2002, Frankfurt/Main, 2002, http://beta1.wi-inf.uni-essen.de/hh/bib-pdf-pub/4858.pdf, Abruf am 2003-06-20

PEMBERTON, S., AUSTIN, D., AXELSSON, J., CELIK, T., DOMINIAK, D., ELENBAAS, H., EPPERSON, B., ISHIKAWA, M., MATSUI, S., MCCARRON, S., NAVARRO, A., PERUVEMBA, S., RELYEA, R., SCHNITZENBAUMER, S., STARK, P. (2002): World Wide Web Consortium (W3C): XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition), A Reformulation of HTML 4 in XML 1.0, W3C Recommendation 26 January 2000, revised 1 August 2002, http://www.w3.org/TR/xhtml1/, Abruf am 2013-10-13

PFEIFFER, H. (2006): AJAX, Modernes Webscripting, Open Source Reihe, Data Becker Verlag, 2006, ISBN: 978-3-8158-2801-4

PHILLIPS, L. A. (2000): Special Edition Using XML, copyright August 2000 by Que, 2000, ISBN: 0-7897-1996-7

PIETERS, S. (2013): World Wide Web Consortium (W3C): Differences from HTML4, W3C Working Draft 28 May 2013, http://www.w3.org/TR/2013/WD-html5-diff-20130528/, Abruf am 2013-10-17

PIOTROWSKI, M., RÖSNER, D. (2005): Integration von E-Assessment und Content-Management, Otto-von-Guericke-Universität Magdeburg, Institut für Wissens- und Sprachverarbeitung, 3. E-Learning Fachtagung Informatik, DeLFI 2005, Rostock, 2005, ISBN: 3-88579-395-4

PLASSMANN, A., SCHMITT, G. (2007): Psychologie Online Lernen, Lern-Psychologie. Essen: Universität Duisburg-Essen, Campus Essen, Universität Duisburg-Essen, Campus Essen, Fachbereich Bildungswissenschaften, Rechts- und Verwaltungswissenschaften, Prof. Dr. Günter Schmitt & Ansgar A. Plassmann, 2007, http://www.uni-due.de/edit/lp/common/lernen.htm, Abruf am 2012-08-19

PLEGER, G (2001): mündliche Auskunft, Treffen vom 11.10.2001, Medienzentrum des Landes Tirol, Innsbruck.

POHL, C. (1999): Methodik und Realisation von Systemen zur effizienten Wissensvermittlung durch Hypermedia, Europäische Hochschulschriften: Reihe 5, Volks- und Betriebswirtschaft; Bd. 2428, zugleich: Würzburg, Univ., Dissertation 1998, Peter Lang GmbH, Europäischer Verlag der Wissenschaften, Frankfurt am Main 1999, ISBN: 3-631-33919-4

PONGRATZ, J. (2002): Learning Technology-Standards: Ein Überblick, 2002, http://wwwbode.cs.tum.edu/Lehrstuhl/Lehre/Seminare/SS2002/eLs-Vortrag-HansPongratz.pdf, Abruf am 2003-06-26

PUPPE, F. (2003): Intelligente Tutorsysteme, Lehrstuhl für Informatik VI, Universität Würzburg, http://www-info6.informatik.uni-wuerzburg.de/teach/ss-2003/its/uebungen/ITS-2.2.pdf , Abruf am 2003-06-20

QUEDNAU, H. D. (2010): Baza statistika metodaro (Einführung in die Statistik - in Esperanto, deutsch und ukrainisch), modifiziert: Sonntag, 25. Juli 2010 23:25:57, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/baza_metodaro.html, Abruf am 2014-06-09

QUEDNAU, H. D., PAAR, S. (2007): Umsetzung mehrsprachiger Lerninhalte auf der Basis von LMML, XSLT und JavaScript, H.-D. Quednau und S. Paar, Fachgebiet Biometrie und Angewandte Informatik der Technischen Universität München, 2007, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/banjaluka_1.pdf, Abruf am 2007-12-21

QUEDNAU, H. D., PAAR, S., STREHL, O. (2007): PLURLINGVAJ VIRTUALAJ KURSOJ GENERITAJ PER LA XML-LINGVO „WLMML“, „Das Erstellen mehrsprachiger virtueller Kurse mit der XML-Sprache „WLMML“, In: Asociaţia internaţională de ştiinţe din România(2007): Comunicarea din perspektivă transdisciplinară / Transfaka komunikado, 19-25, 2007, ISBN: 978-973-1753-08-9, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/sibiu_2007.pdf, Abruf am 2011-09-11

QUEDNAU, H.D., STRECKFUSS, M (2000): Multimedia-Lehr- und Lernsoftware für eine Forstwissenschaftliche Fakultät, Lehrbereich für Forstliche Biometrie und Angewandte Informatik, Forstwissenschaftliche Fakultät der LMU München (DE) und 5. Sektion der Internationalen Akademie der Wissenschaften San Marino (AIS), erschienen in grkg/Humankybernetik 41, Heft 2, S. 47-55, Juni 2000, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/berlin98.html, Abruf am 2011-09-10

QUEDNAU, H. D., STREHL, O. (2007): Die Bedeutung standardisierter Metadaten bei der Erstellung von digitalen Lernobjekten, H.-D. Quednau und O. Strehl, Fachgebiet Biometrie und Angewandte Informatik der Technischen Universität München, 2007, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/banjaluka_2.pdf,
Abruf am 2014-06-15

RADULESCU, V. A. (2005): Entwicklung des Online-Kurses "Großraubtiere in Europa" unter Anwendung aktueller Werkzeuge und Spezifikationen des E-Learning, Diplomarbeit, Technische Universität München, Wissenschaftszentrum für Ernährung, Landnutzung und Umwelt, Studienfakultät für Forstwissenschaft und Ressourcenmanagement, Lehreinheit Biometrie und Angewandte Informatik, Lehreinheit Wildbiologie und Wildtiermanagement, Freising, 2005, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/grossraubtiere.pdf, Abruf am 2011-08-14

RAGGETT, D., HORS, A. L., JACOBS, I. (1999): World Wide Web Consortium (W3C): HTML 4.01 Specification, W3C Recommendation 24 December 1999, http://www.w3.org/TR/1999/REC-html401-19991224/, Abruf am 2013-10-13

RAHTZ, S., CARLISLE, D. (2013): World Wide Web Consortium (W3C): Entity declarations, master file detailing all Unicode characters with names in various entity sets and applications, TeX equivalents and other data, http://www.w3.org/2003/entities/2007xml/unicode.xml, Abruf am 2013-04-16

RATHMAYER, S. (2005): elecTUM im Aufbau, Corporate eLearning - Unternehmens-News, CHECK.point eLearning, 2005, http://www.checkpoint-elearning.de/article/1647.html, Abruf am 2014-01-16

RAWLINGS, A., VAN ROSMALEN, P., KOPER, R., RODRÍGUEZ-ARTACHO, M., LEFRERE, P. (2002): CEN/ISSS WS/LT Learning Technologies Workshop, "Survey of Educational Modelling Languages (EMLs)", Version 1, September 19st 2002, http://www.eife-l.org/publications/standards/elearning-standard/cenissslt/emlsurvey, Abruf am 2013-07-06

REDECKER, C., ALA-MUTKA, K., PUNIE, Y. (2010): Learning 2.0 - The Impact of Social Media on Learning in Europe, POLICY BRIEF, JRC Technical Notes, European Commission, Institute for Prospective Technological Studies, JRC 56958, Luxembourg: Office for Official Publications of the European Communities, European Communities, 2010, http://ftp.jrc.es/EURdoc/JRC56958.pdf, Abruf am 2012-08-16

RECOURSE (2010 a): ReCourse Learning Design Editor, free, open source, cross-platform and extensible, developed as part of the European-funded TENCompetence project, 2010, http://tencompetence-project.bolton.ac.uk/ldauthor/index.html, Abruf am 2013-08-21

RECOURSE (2010 b): TENCompetence Learning Design Services for Coppercore Service Integration, Runtime System (Player) für ReCourse Learning Design Editor-"units of learning", 2010, http://tencompetence-project.bolton.ac.uk/ldruntime/index.html, Abruf am 2013-08-21

RELOAD (2008): Reusable eLearning Object Authoring & Delivery, RELOAD Editor, SCORM Player, Learning Design Editor, Learning Design Player, 23 July 2008: The RELOAD web-site is undergoing an overhaul, 2008, http://www.reload.ac.uk/, Abruf am 2013-08-21

REY, G. D. (2009): E-Learning: Theorien, Gestaltungsempfehlungen und Forschung, Verlag Hans Huber, Hogrefe AG, Bern, 2009, ISBN 978-3-456-84743-6

RIEHLE, D. (2007): Footer aller Art - feststehende Elemente realisieren, Dennis Riehle, E-Mail:selfhtml@riehle-web.com, Homepage-URL: http://tutorial.riehle-web.com/, 2007, in: SELFHTML, HTML-Dateien selbst erstellen, Internetauftritt von Stefan Münz aus dem Jahr 2004, nach 2004 von einer Redaktion weiterentwickelt, 2007, http://aktuell.de.selfhtml.org/artikel/css/footer/, Abruf am 2014-06-10

RINN, U., BETT, K., WEDEKIND, J., ZENTEL, P., MEISTER, D. M., HESSE, F. W. (2003): Virtuelle Lehre an deutschen Hochschulen im Verbund, Teil I, Eine empirische Untersuchung der Projektkonzeptionen von Vorhaben zur Förderung des Einsatzes Neuer Medien in der Hochschullehre im Förderprogramm „Neue Medien in der Bildung“, Online-Publikation des Projektes kevih – Konzepte und Elemente virtueller Hochschule, Institut für Wissensmedien, Tübingen, 2003, ISBN 3-9809055-0-0, http://www.uni-paderborn.de/fileadmin/mw/Meister/Virtuelle_HSLehre_Teil1.pdf, Abruf am 2012-09-13

ROBES, J. (2009): Microlearning und Microtraining: Flexible Kurzformate in der Weiterbildung, Handbuch E-Learning (Kapitel 4.36), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 30. Erg.-Lfg. Oktober 2009, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2009, ISBN: 978-3-87156-298-3

RÖPNACH, R., AUERT (2011): Nachhaltige Gestaltung von Lernprozessen, Handbuch E-Learning (Kapitel 3.20), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 35. Erg.-Lfg. Januar 2011, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2011, ISBN: 978-3-87156-298-3

ROTERING-STEINBERG, S., RICHTER, T. (2011): E-Learning/Blended Learning in der Personalentwicklung, Handbuch E-Learning (Kapitel 6.14), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 35. Erg.-Lfg. Januar 2011, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2011, ISBN: 978-3-87156-298-3

ROTHFUSS, G., RIED, C. (2001): Content Management mit XML: Grundlagen und Anwendungen, Xpert.press, Springer-Verlag Berlin Heidelberg 2001, ISBN: 3-540-66594-3

RWTHAACHEN LOM-EDITOR (2005): IEEE LOM Editor, LOM Editor is the Learning Object Metadata editor complying with the IEEE Learning Object Metadata (LOM) Standard, RWTH Aachen, PD Dr. Ralf Klamma, AOR, Mohamed Amine Chatti, December 2005: LOM Editor 1.0 released, 2005, http://dbis.rwth-aachen.de/cms/projects/LOMEditor, Abruf am 2014-01-05

SALL, K. B. (2003): XML and Namespaces, May 30, 2003, http://www.informit.com/articles/article.aspx?p=31837&seqNum=1, Abruf am 2011-09-10

SANCHEZ-ALONSO, S., LOPEZ, M. G., FROSCH-WILKE, D. (2011): E-Learning Standards for Content Management, in: Ferrer, N., F., Alfonso, J. M. (2011): Content Management for E-Learning, Springer Verlag, 2011, ISBN 978-1-4419-6958-3

SCHAFFERT, S., KALZ, M. (2009): Persönliche Lernumgebungen: Grundlagen, Möglichkeiten und Herausforderungen eines neuen Konzepts, Handbuch E-Learning (Kapitel 5.16), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 27. Erg.-Lfg. Januar 2009, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2009, ISBN: 978-3-87156-298-3

SCHARNBECK, B. (2005): Einführung des Learning Management Systems CLIX Campus an der Fakultät Forstwissenschaft und Ressourcenmanagement der Technischen Universität München, Masterarbeit, Technische Universität München, Studiengang Master Landnutzung Fakultät Wissenschaftszentrum Weihenstephan, Fachgebiet Biometrie und Angewandte Informatik, Prof. Dr. Hans-Dietrich Quednau, 2005

SCHNEIDER, A. (2005):IMS Learning Design als Grundlage für die Gestaltung von e-Learning-Systemen, Diplomarbeit, Lehrstuhl für Entwicklung betrieblicher Informationssysteme, Fachbereich Wirtschaftswissenschaften, Johann Wolfgang Goethe-Universität, Frankfurt am Main, Datum der Veröffentlichung (online):04.08.2005, http://publikationen.ub.uni-frankfurt.de/files/4153/diplomarbeit_ims_ld.pdf, Abruf am 2013-08-23

SCHNEIDER, D. K. (2011 a): Learning Object Metadata Standard. (2011, February 14). EduTech Wiki, A resource kit for educational technology teaching, practice and research. Retrieved 16:18, January 5, 2014 from http://edutechwiki.unige.ch/mediawiki/index.php?title=Learning_Object_Metadata_Standard&oldid=30803, Abruf am 2014-01-05

SCHNEIDER, D. K. (2011 b): Educational modeling language. (2011, July 5). EduTech Wiki, A resource kit for educational technology teaching, practice and research. Retrieved 20:31, July 6, 2013 from http://edutechwiki.unige.ch/mediawiki/index.php?title=Educational_modeling_language&oldid=34037, Abruf am 2013-07-06

SCHNEIDER, J., YU, R., DYER, J. (2005): Standard ECMA-357, ECMAScript for XML (E4X) Specification, 2nd edition (December 2005), http://www.ecma-international.org/publications/standards/Ecma-357.htm, Abruf am 2006-09-14

SCHRAITLE, T. (2004): DocBook-XML, Medienneutrales und plattformunabhängiges Publizieren, SUSE PRESS, 2004, ISBN: 3-89990-078-2

SCHREYER, M. (2002): met@BiM - Ein semantisches Datenmodell für Baustoff-Informationen im World Wide Web, Anwendungen für Beton mit rezyklierter Gesteinskörnung, Doktorarbeit, Fakultät Bauingenieur- und Vermessungswesen der Universität Stuttgart, Institut für Werkstoffe im Bauwesen der Universität Stuttgart 2002, http://www.b-i-m.de/Public/IWB/dissertation_mschreyer.pdf, Abruf am 2012-04-28

SCHRÖDER, H.. (2002): Lernen – Lehren – Unterricht: lernpsychologische und didaktische Grundlagen, 2. Auflage, München, Wien, Oldenbourg, 2002, ISBN: 3-486-25973-3

SCHULMEISTER, R. (2003): Lernplattformen für das virtuelle Lernen: Evaluation und Didaktik, Oldenbourg Wissenschaftsverlag GmbH, München, 2003, ISBN: 3-486-27250-0

SCHULMEISTER, R. (2004): Statement 01/04 01/04 -Thema Open Source Software,2004, Handbuch E-Learning online unter www.global-learning.de, http://web.archive.org/web/20040504211902/http://www.global-learning.de/g-learn/cgi-bin/gl_userpage.cgi?StructuredContent=m07031604, Abruf am 2012-09-16

SCHULMEISTER, R. (2006): eLearning: Einsichten und Aussichten, Oldenbourg Verlag München Wien, 2006, ISBN: 3-486-58003-5

SCHUMANN, A. (2008): Marktüberblick E-Learning-Systeme, ISMAS Institut für strategische Marktanalysen, Handbuch E-Learning (Kapitel 2.5), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 24. Erg.-Lfg. Januar 2008, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2008, ISBN: 978-3-87156-298-3

SCHUMANN, M. (2009): E-Learning, Universität Potsdam, http://www.uni-potsdam.de/db/wiki/elearning/index.php/E-Learning, 2009, Abruf am 2011-09-09

SCHULZ, W. (1965): Unterricht – Analyse und Planung, In: Heimann/Otto/Schulz. Unterricht – Analyse und Planung. Hannover, 1965

SCHÜSSLER, I. (2003): Ermöglichungsdidaktik – eine didaktische Theorie?, In: Ermöglichungsdidaktik: Erwachsenenpädagogische Grundlagen und Erfahrungen, Grundlagen der Berufs- und Erwachsenenbildung, Band 35, Hrsg.: Rolf Arnold, Ingeborg Schüßler, Schneider Verlag Hohengehren, 2003, ISBN: 978-3-89676-717-2

SCLATER, N. (2011): Open Educational Resources: Motivations, Logistics and Sustainability, in: Ferrer, N., F., Alfonso, J. M. (2011): Content Management for E-Learning, Springer Verlag, 2011, ISBN 978-1-4419-6958-3

SEEBOERGER-WEICHSELBAUM, M. (2004): Das Einsteigerseminar XML, bhv, copyright 2004 by Verlag moderne Industrie Buch AG & CO. KG Landsberg, 4. überarbeitete Auflage, 2004, ISBN: 3-8266-7285-2

SELFHTML (2007): SELFHTML, HTML-Dateien selbst erstellen, Internetauftritt von Stefan Münz aus dem Jahr 2004, nach 2004 von einer Redaktion weiterentwickelt, Version 8.1.2 vom 01.03.2007, http://de.selfhtml.org/, Abruf am 2014-01-26

SELFHTML-EVENT (2007): SELFHTML: event: Allgemeines zur Verwendung, Internetauftritt von Stefan Münz aus dem Jahr 2004, nach 2004 von einer Redaktion weiterentwickelt, http://de.selfhtml.org/inter/sprache.htm, Abruf am 2013-07-05

SELFHTML-KODIERUNGEN (2007): SELFHTML: Bits, Bytes und Zeichen, Internetauftritt von Stefan Münz, 2007, http://de.selfhtml.org/inter/sprache.htm, Abruf am 2013-07-05

SIEBERT, H. (2003): Konstruktivistische Leitlinien einer Ermöglichungsdidaktik, In: Ermöglichungsdidaktik: Erwachsenenpädagogische Grundlagen und Erfahrungen, Grundlagen der Berufs- und Erwachsenenbildung, Band 35, Hrsg.: Rolf Arnold, Ingeborg Schüßler, Schneider Verlag Hohengehren, 2003, ISBN: 978-3-89676-717-2

SIEMENS, G. (2004): Connectivism: A Learning Theory for the Digital Age, elearnspace: everything elearning, December 12, 2004, George Siemens, http://www.elearnspace.org/Articles/connectivism.htm, Abruf am 2012-08-23

SKULSCHUS, M., WIEDERSTEIN, M. (2004): XML Schema, Galileo Press GmbH, Bonn, 2004, ISBN: 3-89842-472-5

SKULSCHUS, M., WIEDERSTEIN, M. (2005): XSL-FO für PDF und Druck, mitp-Verlag/Bonn, 2005, ISBN: 3-8266-1531-X

SMYTHE, C., JACKL, A. (2004 a): IMS Content Packaging Information Model, Version 1.1.4, IMS Global Learning Consortium, 04 October 2004, http://www.imsglobal.org/content/packaging/cpv1p1p4/imscp_infov1p1p4.html, Abruf am 2013-08-21

SMYTHE, C., JACKL, A. (2004 b): IMS Content Packaging XML Binding, Version 1.1.4, IMS Global Learning Consortium, 04 October 2004, http://www.imsglobal.org/content/packaging/cpv1p1p4/imscp_bindv1p1p4.html, Abruf am 2013-08-21

SMYTHE, C., JACKL, A. (2004 c): IMS Content Packaging Best Practice and Implementation Guide, Version 1.1.4, IMS Global Learning Consortium, 04 October 2004, http://www.imsglobal.org/content/packaging/cpv1p1p4/imscp_bestv1p1p4.html, Abruf am 2013-08-21

SMYTHE, C., JACKL, A., KRAAN, W. (2004): IMS Content Packaging Summary of Changes, Version 1.1.4 Final Specification, IMS Global Learning Consortium, 04 October 2004, http://www.imsglobal.org/content/packaging/cpv1p1p4/imscp_sumcv1p1p4.html, Abruf am 2013-08-21

SMYTHE, C., TOWLE, B. (2006): Guidelines for Using the IMS LRM to IEEE LOM 1.0 Transform, Version 1.0, IMS Global Learning Consortium, 31 August 2006, http://www.imsglobal.org/metadata/mdv1p3/imsmd_transformv1p0.html, Abruf am 2013-08-23

SOUTH, J. B., MONSON, D. W. (2000): A university-wide system for creating, capturing, and delivering learning objects. In D. A. Wiley (Ed.), The Instructional Use of Learning Objects: Online Version, 2000, http://www.reusability.org/read/, Abruf am 2012-09-16

SPANNEBERG, B., MINTERT, S. (2007): Nadeln im Heu, clientseitige Ajax- und RIA-Tools, in: iX SPECIAL 1/2007, Web 2.0 – das Kompendium, S. 8 - 13, Heise Zeitschriften Verlag, 2007

SPECHT, M., EBNER, M., LÖCKER, C. (2013): Mobiles und ubiquitäres Lernen, Technologien und didaktische Aspekte. In: M. Ebner, S. Schön (Hrsg.): Lehrbuch für Lernen und Lehren mit Technologie, L3T, Version August 2013, http://l3t.tugraz.at/index.php/LehrbuchEbner10/article/download/113/97, Abruf am 2014-08-11

STACEY, P. (2007): Open educational resources in a global context, First Monday, Volume 12, Number 4 — 2 April 2007, http://www.firstmonday.org/ojs/index.php/fm/article/view/1769/1649, Abruf 2014-08-03

STANGL, W. (2012): eLearning, E-Learning, Blended Learning, Werner Stangl Arbeitsblätter, Werner Stangl, Assistenzprofessor am Institut für Psychologie und Pädagogik an der Sozial- und Wirtschaftswissenschaftlichen Fakultät der Johannes-Kepler-Universität Linz, http://www.stangl-taller.at/ARBEITSBLAETTER/LERNEN/Elearning.shtml, Abruf 2012-08-31

STANGL, W. (2014): Begriffsdefinition Didaktik, Online-Lexikon: Psychologische Begriffsbestimmungen, Werner Stangl, Assistenzprofessor am Institut für Psychologie und Pädagogik an der Sozial- und Wirtschaftswissenschaftlichen Fakultät der Johannes-Kepler-Universität Linz, http://psychologie.stangl.eu/definition/Didaktik.shtml, Abruf am 2014-11-30

STATCOUNTER (2014): StatCounter, GlobalStats, http://gs.statcounter.com/, Abruf am 2014-01-13

STATCOUNTER-FAQ (2014): Frequently Asked Questions, StatCounter, GlobalStats, http://gs.statcounter.com/faq, Abruf am 2014-01-13

STEIN, M (2002): workshop XML, Addison-Wesley Verlag, Imprint der Pearson Education Deutschland GmbH, München/Germany, 2002, ISBN: 3-8273-1867-X

STEYER, M. (2014): Allzeit bereit, offline-fähige Browser-Anwendungen mit JavaScript, in: iX Developer 1/2014 – JavaScript heute, Zeitschrift, Heise-Verlag, 2014

STEYER, R. (2007): jetzt lerne ich AJAX, Ihr einfacher Einstieg in Web 2.0, Markt + Technik Verlag, 2007, ISBN: 978-3-8272-4225-9

STIERAND, B. (2006): Alleskönner unter sich, aktuelle AJAX-Frameworks im Vergleich, in: AJAXspecial, Technolgien, Tools, Tutorials für Web Professionals, AJAX im Einsatz, S. 52 - 60, Zeitschrift, entwickler-magazin.de/ajaxspecial, 2006

STOLLER-SCHAI, D. (2010): Mobiles Lernen – die Lernform des Homo Mobilis, Handbuch E-Learning (Kapitel 4.39), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 32. Erg.-Lfg. April 2010, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2010, ISBN: 978-3-87156-298-3

STREHL, O., QUEDNAU, H. D., PAAR, S. (2005): Konzept einer standard-konformen E-Learning-Modul-Bibliothek, Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 17. Tagung 2005 Freiburg, Die Grüne Reihe, herausgegeben von U. Wunn (Forschungsanstalt für Waldökologie und Forstwirtschaft Rheinland-Pfalz in Trippstadt), 2005, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/Freiburg_092005.html, Abruf am 2011-09-11

STREHL, O., QUEDNAU, H. D., PAAR, S., MUEHLBAUER, M. (2007): Metadaten forstlicher Lernobjekte, Deutscher Verband Forstlicher Forschungsanstalten, Sektion Forstliche Biometrie und Informatik, 18. Tagung in Trippstadt 2006, Die Grüne Reihe, Tagungsband herausgegeben von A. Degenhard und U.Wunn 2007 (Forschungsanstalt für Waldökologie und Forstwirtschaft Rheinland-Pfalz in Trippstadt), 2007, http://www.forst.tu-muenchen.de/EXT/PUBL/quednau/Trippstadt_Strehl.html, Abruf am 2011-09-11

STÜHRENBERG, M. (2004): Erfahrungen und Trends zum Einsatz von XML Learning Objects aus dem BMBF-Projekt MiLCA, in: Engels, G., Seehusen, S. (Hrsg.): DeLFI 2004, Tagungsband der 2. e-Learning Fachtagung Informatik, 6. - 8. September 2004, Paderborn, Germany, Gesellschaft für Informatik, 2004, ISBN: 3-88579-381-4

STÜHRENBERG, M. (2013): MiLCA - Medienintensive Lehrinhalte in der Computerlinguisitk-Ausbildung, http://www.maik-stuehrenberg.de/arbeit/projekte/milca/index.html, Abruf am 2013-08-19

SUHR, R., SUHR R. (1993): Software Engineering: Technik und Methodik, R. Oldenbourg Verlag GmbH, München, 1993, ISBN: 3-486-21647-3

SÜSS, C. (2000): Adaptive Knowledge Management: A Meta-Modelling Approach and its Binding to XML. In: H.-J. Klein (Ed.), 12.GI-Workshop Grundlagen von Datenbanken, Plön, TR 2005, Christian-Albrechts-Universität Kiel, Germany, 2000

SÜSS, C. (2001): LMML Document Type Definitions (DTDs). http://daisy.fmi.uni-passau.de/pakmas/LM2L/, Abruf am 2001-11-21

SÜSS, C. (2004): Eine Architektur für die Wiederverwendung und Adaption von eLearning-Inhalten, VVB LAUFERSWEILER VERLAG, edition scientifique, Wettenberg, 2004, ISBN: 3-89687-476-4

SÜSS, C., FREITAG, B. (2000 a): Entwicklung und Nutzung von Teachware: Das Passauer Knowledge Management System (PaKMaS), In: Rudolf Kammerl (Hrsg.), Computerunterstütztes Lernen, München, Oldenbourg, 2000

SÜSS, C., FREITAG, B. (2000 b): Datenbanken und Informationssysteme: Passauer Knowledge Management System (PaKMaS), Multimedia Informationssysteme für Ausbildung und Wissensmanagement, http://web.archive.org/web/20020221183803/http://daisy.fmi.uni-passau.de/pakmas/pakmas.html, Abruf am 2013-08-09

SÜSS, C., FREITAG, B. (2001 a): Passauer Knowledge Management System PaKMaS, IFIS-Report 2001/02, IFIS - Institut für Informationssysteme und Softwaretechnik. Universität Passau, 2001

SÜSS, C., FREITAG, B. (2001 b): Learning Material Markup Language –LMML. IFIS-Report 2001/03, IFIS - Institut für Informationssysteme und Softwaretechnik. Universität Passau, 2001

SÜSS, C., FREITAG, B., BRÖSSLER, P. (1999): Metamodeling for web-based teachware managment. In P.P. Chen, D.W. Embley, J. Kouloumdijan, S.W. Liddle, and J.F. Roddick, editors, Advances in Conceptual Modeling. ER'99 Workshop on the World-Wide Web and Conceptual Modeling, volume 1727 of LNCS, pages 360–373, Paris, France, Springer-Verlag, November 1999. http://reference.kfupm.edu.sa/content/m/e/metamodeling_for_web_based_teachware_man_110776.pdf, Abruf am 2012-09-14

TEEGE, G., BREITLING, P. (2002): Targeteam: Adaptierbare Lehrinhalte auf Basis von XML und XSLT, Workshop „E-Learning Content auf Basis von XML“, GI-Jahrestagung, Dortmund, 2002, http://subs.emis.de/LNI/Proceedings/Proceedings19/GI-Proceedings.19-55.pdf, Abruf am 2013-08-19

TERHART, E. (2005): Lehr-Lern-Methoden: Eine Einführung in Probleme der methodischen Organisation von Lehren und Lernen (Grundlagentexte Pädagogik), 4. Auflage, 2005, Beltz Juventa, ISBN: 3779903555

THOMPSON, H., BEECH, D., MALONEY, M., MENDELSOHN, N. (2004): XML Schema Part 1: Structures Second Edition, W3C Recommendation 28 October 2004, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/, Abruf am 2005-01-01

THROPP, S. (2004): The Impact of the Standardization Process on SCORM 2004, Advanced Distributed Learning (ADL), July 22, 2004, http://www.adlnet.gov/scorm/articles/8.cfm, Abruf am 2006-11-16

TÖNNSEN, K. (2007): Potentialerweiterung webbasierter und hypermedialer Lernsysteme durch Integration technischer Experimente und Realobjekte, Dissertation, Universität Flensburg, Institut für Technik und Didaktik, 2007, http://www.zhb-flensburg.de/dissert/toennsen/Dissertation_Toennsen.pdf, Abruf am 2012-08-19

TORSTEN, H., KARTEN, M., NIEHOFF, J. (2003): Plattformübergreifende Publikation rekombinierbarer Lernobjekte auf Basis von XML, Fakultät Ingenieurwissenschaften und Informatik, Fachhochschule Osnabrück, in: Bode, A., Desel, J., Rathmayer, S., Wessner, M. (2003): DeLFI 2003: Die 1. E-Learning Fachtagung Informatik. Tagung der Fachgruppe E-Learning der Gesellschaft für Informatik e.V. (GI), 16. – 18. September 2003 in Garching, Köllen Druck+Verlang GmbH, Bonn, 2003, ISBN 3-88579-366-0

TRAHASCH, S. (2002): Ariadne - Digitale Bibliothek für die (virtuelle) Hochschule, Albert-Ludwigs-Universität Freiburg Institut für Informatik, Workshop: Standardisierung im eLearning, Johann Wolfgang Goethe-Universität, Frankfurt/Main, 10./11. April 2002, http://www.httc.de/nmb/images/Trahasch-v1.pdf , Abruf am 2003-06-26

TUM-MOODLE (2014): eLearning@TUM, moodle, medientzuntrum der Technischen Universität München, https://www.moodle.tum.de/, Abruf am 2014-01-16

UNESCO (2011): UNESCO (United Nations Educational, Scientific and Cultural Organization) Institut für Lebenslanges Lernen, Hamburg, http://www.unesco.de/uil.html, Abruf am 2011-08-15

UNICODE (2013): The Unicode Standard, About Versions of the Unicode Standard, http://www.unicode.org/standard/versions/, Abruf am 2013-07-05

VLIST, E. v. d. (2001): Using W3C XML Schema, xml.com, The Guide to W3C XML Schema, A Collection of Articles from XML.com, 2001, O'REILLY

VOIGT, D., TAVANGARIAN, D. (2003): <ML>³ Multidimensional LearningObjects and Modular Lectures Markup Language, Technischer Bericht, Lehrstuhl Rechnerarchitekur, Institut für Technische Informatik, Fachbereich Informatik, Universität Rostock, 10.09.2003, http://www.ml-3.org/ML3/1.2/docu/language/ml3_documentation_german.pdf, Abruf am 2013-08-16

W3C (2011): World Wide Web Consortium (W3C), http://www.w3.org/, Abruf am 2011-10-31

W3C-HTML-CSS (2013): World Wide Web Consortium (W3C): HTML & CSS, http://www.w3.org/standards/webdesign/htmlcss, Abruf am 2013-10-13

W3C-OWL (2011): World Wide Web Consortium (W3C): OWL (Web Ontology Language) Working Group, World Wide Web Consortium, http://www.w3.org/2007/OWL/wiki/OWL_Working_Group, Abruf am 2011-10-31

W3C-SCHEMA (2013): World Wide Web Consortium (W3C): XML Schema Current Status, http://www.w3.org/standards/techs/xmlschema, Abruf am 2013-02-17

W3C-SEMANTIC-WEB (2011): World Wide Web Consortium (W3C): W3C Semantic Web Activity, World Wide Web Consortium, http://www.w3.org/2001/sw/, Abruf am 2011-10-31

W3C-WIKI-WEB-STANDARDS-MODEL (2014): World Wide Web Consortium (W3C) Wiki: The web standards model - HTML CSS and JavaScript, http://www.w3.org/wiki/The_web_standards_model_-_HTML_CSS_and_JavaScript, Abruf am 2014-11-30

W3C-XML-Technology (2013): World Wide Web Consortium (W3C): XML Technology, http://www.w3.org/standards/xml/, Abruf am 2013-02-02

W3SCHOOLS-E4X (2013): W3Schools, XML - E4X, http://web.archive.org/web/20140107115603/http://w3schools.com/xml/xml_e4x.asp, Abruf am 2013-11-17

WALMSLEY, P. (2002): Definitive XML Schema, Prentice Hall PTR, 2002, ISBN: 0-13-065567-8

WANG, D. (2005): AJAX: Die (alte) neue Technologie für Webapplikationen der nächsten Generation, Nicht nur holländische Fußballkunst, in: XMLmagazin & web services, XML-Praxis, Entwickler Magazin 6.2005, S. 154 - 158, www.entwickler-magazin.de, 2005

WEBTREKK WEBSTATISKTIK (2014): Webtrekk Deutsche Webstatistik 4. Quartal 2013, http://www.webtrekk.com/fileadmin/pdf/pm/2014/2014-07-01_Webtrekk_Studie_Q4_2013.pdf, Abruf am 2014-01-13

WEITL, F., HÖCK, S. M. (2003): Modulerstellung mit LMML-d, Konzepte, Sprachstrukturen und praktische Anleitung, IFIS, Universität Passau, Stand: 10.10.2003

WEITL, F., SÜSS, C., KAMMERL, R. (2002): Didaktische Strukturierung von Online-Inhalten, Ein einfaches Referenzmodell und seine Umsetzung in eine XML-Sprache am Beispiel des Learning Material Markup Language Frameworks, Technischer Bericht, Draftversion! (Stand 26.03.2002), IFIS - Institut für Informationssysteme und Softwaretechnik. Universität Passau, 2002

WENZ, C. (2008): JavaScript und Ajax, Das umfassende Handbuch, 8., aktualisierte und erweiterte Auflage, Galileo Press, Bonn 2008, ISBN: 978-3-8362-1128-4

WENZ, C. (2010): Ajax, schnell + kompakt, entwickler.press, 2010, ISBN: 978-3-86802-045-8

WIEPCKE, C. (2006): Computergestützte Lernkonzepte und deren Evaluation in der Weiterbildung. Blended Learning zur Förderung von Gender Mainstreaming, Hamburg: Kovac 2006, ISBN: 3-8300-2426-6
Abbildung „Blended Learning Mix“ in Wikipedia unter der Creative Commons-Lizenz Namensnennung-Weitergabe unter gleichen Bedingungen 3.0 Unported lizenziert, 2006, http://de.wikipedia.org/w/index.php?title=Datei:Blended_Learning.jpg&filetimestamp=20080425193017, Abruf am 2012-08-31

WIKIPEDIA-AJAX-DE (2013): Ajax (Programmierung). In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 18. Dezember 2013, 10:25 UTC. URL: http://de.wikipedia.org/w/index.php?title=Ajax_(Programmierung)&oldid=125554705 (Abgerufen: 3. Januar 2014, 16:56 UTC), Abruf am 2014-01-03

WIKIPEDIA-AJAX-EN (2013): Ajax (programming). (2013, December 27). In Wikipedia, The Free Encyclopedia. Retrieved 20:36, December 27, 2013, from http://en.wikipedia.org/w/index.php?title=Ajax_(programming)&oldid=587868378, Abruf am 2013-12-27

WIKIPEDIA-AJAX-FRAMEWORKS-EN (2013): List of Ajax frameworks. (2013, December 8). In Wikipedia, The Free Encyclopedia. Retrieved 18:25, December 30, 2013, from http://en.wikipedia.org/w/index.php?title=List_of_Ajax_frameworks&oldid=585179546, Abruf am 2013-12-30

WIKIPEDIA-ASCII (2013): American Standard Code for Information Interchange, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 24. Juni 2013, 08:47 UTC. URL: http://de.wikipedia.org/w/index.php?title=American_Standard_Code_for_Information_Interchange&oldid=119858592, Abruf am 2013-07-05

WIKIPEDIA-BLOG (2014): Blog. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 23. Juli 2014, 14:25 UTC. URL: http://de.wikipedia.org/w/index.php?title=Blog&oldid=132420270 (Abgerufen: 12. August 2014, 11:30 UTC), Abruf am 2014-08-12

WIKIPEDIA-BOOKMARKLETS (2007): Wikipedia: Technik/Browser/Unterstützung, http://de.wikipedia.org/wiki/Wikipedia:Browser-Unterst%C3%BCtzung, Abruf am 2007-06-17

WIKTIONARY-SUCHFELD (2014): Wiktionary-Suchfeld, Letzte Änderung dieser Seite: 31. Mai 2014 um 09:52, http://de.wiktionary.org/wiki/Wiktionary:Werbung#Wiktionary-Suchfeld, Abruf am 2014-06-10

WIKIPEDIA-BROWSER-STATISTIK (2014): Usage share of web browsers. (2014, January 5). In Wikipedia, The Free Encyclopedia. Retrieved 16:24, January 13, 2014, from http://en.wikipedia.org/w/index.php?title=Usage_share_of_web_browsers&oldid=589299587, Abruf am 2014-01-13

WIKIPEDIA-DIDAKTIK (2012): Begriffsbeschreibung Didaktik, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 7. Juli 2012, 10:26 UTC. URL: http://de.wikipedia.org/w/index.php?title=Didaktik&oldid=105280907, Abruf am 2012-08-19

WIKIPEDIA-E4X-DE (2013): ECMAScript for XML. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 13. Juni 2013, 20:12 UTC. URL: http://de.wikipedia.org/w/index.php?title=ECMAScript_for_XML&oldid=119529570, Abruf am 2013-11-17

WIKIPEDIA-E4X-EN (2013): ECMAScript for XML. (2013, September 17). In Wikipedia, The Free Encyclopedia. Retrieved 11:43, November 17, 2013, from http://en.wikipedia.org/w/index.php?title=ECMAScript_for_XML&oldid=573288457, Abruf am 2013-11-17

WIKIPEDIA-E-LEARNING-DE (2012): Begriffsbeschreibung E-Learning, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 28. Juli 2012, 09:48 UTC. URL: http://de.wikipedia.org/w/index.php?title=E-Learning&oldid=106093435, Abruf am 2012-08-13

WIKIPEDIA-E-LEARNING-EN (2014): E-learning. (2014, August 10). In Wikipedia, The Free Encyclopedia. Retrieved 10:28, August 11, 2014, from http://en.wikipedia.org/w/index.php?title=E-learning&oldid=620684077, Abruf am 2014-08-11

WIKIPEDIA-EREIGNIS (2013): Ereignis (Programmierung). In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 3. April 2013, 19:33 UTC. URL: http://de.wikipedia.org/w/index.php?title=Ereignis_(Programmierung)&oldid=116865717 (Abgerufen: 2. Januar 2014, 10:16 UTC), Abruf am 2014-01-02

WIKIPEDIA-FC (2014): „Umgedrehter Unterricht“. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 14. Mai 2014, 22:53 UTC. URL: http://de.wikipedia.org/w/index.php?title=Umgedrehter_Unterricht&oldid=130414364 (Abgerufen: 13. August 2014, 08:40 UTC), Abruf am 2014-08-13

WIKIPEDIA-Framework (2013): Framework, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 10. September 2013, 10:33 UTC. URL: http://de.wikipedia.org/w/index.php?title=Framework&oldid=122403181 (Abgerufen: 22. März 2014, 13:05 UTC), Abruf am 2014-03-22

WIKIPEDIA-IL (2014): Integriertes Lernen. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 27. April 2014, 16:38 UTC. URL: http://de.wikipedia.org/w/index.php?title=Integriertes_Lernen&oldid=129884690 (Abgerufen: 13. August 2014, 10:07 UTC), Abruf am 2014-08-13

WIKIPEDIA-IRON (2014): SRWare Iron, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 6. Januar 2014, 15:42 UTC. URL: http://de.wikipedia.org/w/index.php?title=SRWare_Iron&oldid=126184467 (Abgerufen: 15. Januar 2014, 12:54 UTC), Abruf am 2014-01-15

WIKIPEDIA-ISO-8859 (2013): ISO 8859, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 1. Juni 2013, 19:15 UTC. URL: http://de.wikipedia.org/w/index.php?title=ISO_8859&oldid=119115166, Abruf am 2013-07-05

WIKIPEDIA-ISO-639 (2014): ISO 639. (2014, March 9). In Wikipedia, The Free Encyclopedia. Retrieved 17:17, March 9, 2014, from http://en.wikipedia.org/w/index.php?title=ISO_639&oldid=598849897, Abruf am 2014-03-09

WIKIPEDIA-JAVASCRIPT (2013): JavaScript, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 15. Oktober 2013, 18:13 UTC. URL: http://de.wikipedia.org/w/index.php?title=JavaScript&oldid=123483740, Abruf am 2013-10-19

WIKIPEDIA-LMS (2013): Lernplattform, In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 12. Dezember 2013, 12:11 UTC. URL: http://de.wikipedia.org/w/index.php?title=Lernplattform&oldid=125370413 (Abgerufen: 13. Januar 2014, 15:55 UTC), Abruf am 2014-01-13

WIKIPEDIA-LMS-LISTE (2013): Liste von Lernplattformen. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 12. Dezember 2013, 13:00 UTC. URL: http://de.wikipedia.org/w/index.php?title=Liste_von_Lernplattformen&oldid=125371878 (Abgerufen: 10. Januar 2014, 18:46 UTC), Abruf am 2014-01-10

WIKIPEDIA-LOM-EN (2013): Learning object metadata. (2013, December 26). In Wikipedia, The Free Encyclopedia. Retrieved 15:44, January 5, 2014, from http://en.wikipedia.org/w/index.php?title=Learning_object_metadata&oldid=587736525, Abruf am 2014-01-05

WIKIPEDIA-METHODIK-PÄDAGOGIK (2012): Begriffsbeschreibung Methodik (Pädagogik), In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 14. Juni 2012, 21:53 UTC. URL: http://de.wikipedia.org/w/index.php?title=Methodik_(P%C3%A4dagogik)&oldid=104401719, Abruf am 2012-09-06

WIKIPEDIA-MIKROLERNEN (2011): Begriffsbeschreibung Mikrolernen. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 11. Oktober 2011, 10:03 UTC. URL: http://de.wikipedia.org/w/index.php?title=Mikrolernen&oldid=94638121, Abruf am 2012-08-21

WIKIPEDIA-MOOC (2014):Massive open online course. (2014, August 11). In Wikipedia, The Free Encyclopedia. Retrieved 18:11, August 12, 2014, from http://en.wikipedia.org/w/index.php?title=Massive_open_online_course&oldid=620821757 Abruf am 2014-08-12

WIKIPEDIA-PLE (2013): Personal learning environment. (2013, July 10). In Wikipedia, The Free Encyclopedia. Retrieved 22:00, January 11, 2014, from http://en.wikipedia.org/w/index.php?title=Personal_learning_environment&oldid=563707571, Abruf am 2014-01-11

WIKIPEDIA-UC-BROWSER (2014): UC Browser. (2014, January 2). In Wikipedia, The Free Encyclopedia. Retrieved 12:00, January 14, 2014, from http://en.wikipedia.org/w/index.php?title=UC_Browser&oldid=588741319, Abruf am 2014-01-14

WIKIPEDIA-UML (2012): Begriffsbeschreibung UML. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 3. August 2012, 22:05 UTC. URL: http://de.wikipedia.org/w/index.php?title=Unified_Modeling_Language&oldid=106352746, Abruf am 2012-09-14

WIKIPEDIA-VOLLTEXTRECHERCHE (2013): Begriffsbeschreibung Volltextrecherche. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 3. April 2013, 22:35 UTC. URL: http://de.wikipedia.org/w/index.php?title=Volltextrecherche&oldid=116893103, Abruf am 2014-05-03

WIKIPEDIA-WEBCAST (2012): Begriffsbeschreibung Webcast. Seite „Webcast“. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 25. Juni 2012, 15:05 UTC. URL: http://de.wikipedia.org/w/index.php?title=Webcast&oldid=104797558, Abruf am 2012-09-06

WIKIPEDIA-BROWSER-EN (2014): List of web browsers. (2014, June 18). In Wikipedia, The Free Encyclopedia. Retrieved 12:39, June 21, 2014, from http://en.wikipedia.org/w/index.php?title=List_of_web_browsers&oldid=613487627, Abruf am 2014-06-21

WIKIPEDIA-XML-EDITOREN (2014): Comparison of XML editors. (2014, January 15). In Wikipedia, The Free Encyclopedia. Retrieved 13:22, January 16, 2014, from http://en.wikipedia.org/w/index.php?title=Comparison_of_XML_editors&oldid=590833520, Abruf am 2014-01-16

WILBERS, K. (2011): E-Learning didaktisch gestalten, Handbuch E-Learning (Kapitel 4.0), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 35. Erg.-Lfg. Januar 2011, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2011, ISBN: 978-3-87156-298-3

WILBERS, K (2007): Design und Evaluation von Bildungsportalen, in: Bildungsportale, Potenziale und Perspektiven netzbasierter Bildungsressourcen, Herausgeber: Dr. Birgit Gaiser, Prod. Dr. Dr. Friedrich W. Hesse, Dr. Monika Lütke-Entrup, 2007, Oldenbourg Wissenschaftsverlag GmbH, 2007, ISBN: 978-3-486-58426-4

WILEY, D. A. (2000): Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. In D. A. Wiley (Ed.), The Instructional Use of Learning Objects: Online Version, 2000, http://www.reusability.org/read/chapters/wiley.doc, Abruf am 2012-09-16

WINKELMANN, Y. (2006): Aufgaben in virtuellen Lernumgebungen und Werkzeuge zu deren Erstellung, Diplomarbeit an der Technischen Universität Dresden, Februar 2006, AG Didaktik der Informatik/Lehrerbildung, Institut für Software- und Multimediatechnik, Fakultät Informatik, 2006, http://dil.inf.tu-dresden.de/fileadmin/dil-web/forschung/e_learning/studentische_arbeiten/diplom_wink_web.pdf, Abruf am 2013-08-24

WTO (1994): World Trade Organization Agreement on Technical Barriers to Trade, 1994, http://www.wto.org/english/docs_e/legal_e/17-tbt.pdf, Abruf am 2003-06-20

XAMPP (2014): XAMPP Apache + MySQL + PHP + Perl, Apache Friends, Open-Source-Paket, https://www.apachefriends.org/de/index.html, Abruf am 2014-06-15

YOUNG, J. (2000): XML Schritt für Schritt. Unterschleißheim: Microsoft Press Deutschland, 2000, ISBN: 3-86063-765-7

YOUSEF, A. M. F., CHATTI, M. A., SCHROEDER, U., WOSNITZA, M., JAKOBS, H. (2014): MOOCs - A Review of the State-of-the-Art. In Proceedings of the CSEDU 2014 conference, Vol. 3, pp. 9-20. INSTICC, 2014, http://learntech.rwth-aachen.de/dl1193, Abruf am 2014-08-12

ZAKAS, N. C. (2005): Professional JavaScript for Web Developers, Programmer to Programmer, Wrox, Wiley Publishing, Inc., 2005, ISBN-13: 978-0-7645-7908-0

ZAKAS, N. C., MCPEAK, J., FAWCETT, J. (2006): Ajax Professionell, Programmer to Programmer, Redline GmbH, Heidelberg, 2006, ISBN: 978-8266-1669-3

ZAWACKI-RICHTER, O., BÄCKER, E. M., BARTMANN, S. (2010): Lernen in beweglichen Horizonten: Internationalisierung und interkulturelle Aspekte des E-Learning, Handbuch E-Learning (Kapitel 4.38), Das aktuelle Nachschlagewerk aus Wissenschaft und Praxis, 32. Erg.-Lfg. Januar 2010, Andreas Hohenstein/Karl Wilbers (Hrsg), Köln: Deutscher Wirtschaftsdienst (eine Marke von Wolters Kluwer Deutschland), 2010, ISBN: 978-3-87156-298-3

ZHAO, XINYOU; WAN XIN; OKAMOTO TOSHIO (2010): Adaptive Content Delivery in Ubiquitous Learning Environment, Graduate School of Information Systems, The University of Electro-Communications, Japan, E-mail : totoyou@ieee.org, The 6th IEEE International Conference on Wireless, Mobile and Ubiquitous Technologies in Education (IEEE WMUTE 2010), 2010, http://www.myresearch.biz/sharefile/file/WMUTE2010.pdf, Abruf am 2014-08-11

ZIMMERMANN, S. (2012): Ein interdisziplinärer Ansatz zum Conformance Testing mit beispielhafter Umsetzung in den Bereichen E-Procurement und E-Learning, Diplomarbeit zur Erlangung des Grades eines Diplom-Informatikers im Studiengang Informatik, Universität Koblenz Landau, 2012, http://kola.opus.hbz-nrw.de/volltexte/2012/836/pdf/diplomarbeit.pdf, Abruf am 2014-01-12

ZUMBACH, J., MARESCH G. (Hrsg.) (2010): Aktuelle Entwicklungen in der Didaktik der Naturwissenschaften – Ansätze aus der Biologie und der Informatik, Studienverlag Ges. m. b. h. H., 2010, ISBN 978-3-7065-4882-3