Universität Zürich
Institut für Informatik

Dynamic and Distributed
Information Systems

Diplomarbeit 1. September 2007

Developing a Web Portal for Case Studies

Roman Zweifel
von Jona SG, Schweiz

Student-ID: 00-921-924
R_Zweifel@access.unizh.ch

BetreuerIn: **Katharina Reinecke**

Prof. Abraham Bernstein, PhD
Institut für Informatik
Univesität Zürich
http://www.ifi.unizh.ch/ddis

Danksagung

Ich möchte allen herzlich danken, welche mich während der Zeit an der Universität Zürich und insbesondere während meiner Diplomarbeitszeit unterstützt haben.

Besonderen Dank geht an meine Betreuerin Katharina Reinecke für die Ermöglichung und Hilfe während meiner Diplomarbeit, als auch an Herrn Professor Bernstein. Beiden möchte ich für die Betreuung herzlich danken.

Ebenfalls besonderen Dank möchte ich meiner Familie und meiner Freundin aussprechen. Beide haben mich während der ganzen Arbeit tatkräftig motiviert.

Herzlichen Dank auch an alle weiteren Personen, die mich während meiner Arbeit unterstützt haben.

Zusammenfassung

Das e-Learning Angebot hat an den höheren Schulen in den letzten Jahren stark zugenommen. Universitäten offerieren ihren Studenten ganze Kurse und Lernmaterialien über das Internet. Es ist sozusagen eine neue Lernform entstanden.

Je mehr solcher Angebote entstehen, desto schwieriger wird es, die richtigen Materialien und Kurse zu finden. Heute wird dies oft über eine Suche oder Kategorisierungen gelöst. Neue Entwicklungen wie beispielsweise die Fassettensuche oder semantische Annotationen werden selten verwendet.

Diese Arbeit beschreibt das CasIS Portal. Es dient zur Bereitstellung von diversen Informatik Fallstudien für den Masterstudiengang der Informatik an der Universität Zürich. Zur verbesserten Verwendbarkeit wurden insbesondere einer einfachen Auffindbarkeit der Ressourcen Rechnung getragen. Mit Hilfe des Semantic Web und einer umfangreichen Fassettennavigation wird es dem Studenten ermöglicht, seinem Wissensstand entsprechende Lernressourcen einfacher zu finden.

Abstract

In the last few years the e-learning offers from the higher education schools have increased. They supply whole courses and learning materials in the internet for their students. So the internet gives the possibility to have another learning method.

The more learning ressources exist in a portal, it is much more difficult for the students to find the right materials and courses. Today the main approach is to offer a full text search or a categorisation of course materials. New developments like faceted browsing or semantic annotation is rarely used.

This thesis describes the CasIS portal. It is developed for the master studies in Computer Science at the University of Zurich. It supplies different case studies in the dissimilar areas of Information Systems. For a better usability the searching of the right ressources is very important. With the aid of the semantic web and a facetted browsing tool, the student gets the ability to find the appropriate ressources easily.

Inhaltsverzeichnis

1

Einleitung

1.1 Das Projekt Swiss Virtual Campus

Das Projekt Swiss Virtual Campus (SVC) [Campus, 2007] wurde im Jahre 1999 gegründet und hat zum Ziel, e-Learning an Schweizer Universitäten zu fördern. Insbesondere werden den Studierenden qualitativ hochstehende Unterrichtsmaterialien und Unterrichtsmethoden zur Verfügung gestellt. Diese Lernmaterialien dürfen zwischen den Hochschulen ausgetauscht und weiterverwendet werden. Dies steigert die Effizienz und trägt zur Qualitätssicherung bei. Die virtuellen Kurse haben für die Studierenden zusätzliche Vorteile, unter anderem zeit- und ortsunabhängiges Lernen, die Kontrolle über das eigene Lerntempo sowie zusätzliche Interaktivität. Derzeit (Stand April 2007) befinden sich 82 Kurse in mehreren Sprachen online und behandeln Themen aus den unterschiedlichsten Fachgebieten.

1.1.1 Das Projekt „Cases in Information Systems“

Das Projekt „Cases in Information Systems“ (CasIS) [CasIS, 2007] ist ein Projekt des SVC und hat zum Ziel, Fallstudien in hoher Qualität bereitzustellen. Das Projekt unterstützt die Fallstudienbearbeitung in Lehrveranstaltungen der Wirtschaftsinformatik auf Masterebene. Es werden unterschiedliche Module angeboten: vier Module für einen bestimmten realen Fall sowie ein digitaler Methoden- und Analysewerkzeugkasten. Die realen Fallstudien wurden in Zusammenarbeit mit Partnern aus der Praxis entwickelt und in der eigens entworfenen XML Sprache CaseML definiert. Durch die Erstellung eines Onlineportals sollen die Fallstudien realistischer und vor allem auch interaktiver gestaltet werden können. Das Portal beinhaltet vier verschiedene Module:

- Eintrittstest - klärt das Vorwissen des Studenten ab
- Vorbereitungsmodule - helfen Lücken zu schliessen
- Fallstudien - der eigentliche Inhalt zur Bearbeitung und zur Übung
- Toolbox - Methoden und Werkzeuge zur Analyse und Problemlösung

Aufgrund des Eintrittstests erhält der Student einen entsprechenden Vorschlag zur Bearbeitung der Vorbereitungsmodule.

1.2 Motivation

Im Bereich des e-Learning wurden in den letzten Jahren sehr viele Anstrengungen unternommen. Insbesondere an Hochschulen und Universitäten wird das e-Learning Angebot stark erweitert. Diese Arbeit stellt ebenfalls ein solches e-Learning Angebot bereit. Die besondere Motivation liegt darin, dass aktuelle e-Learning Portale nur wenig Gebrauch von neuen Forschungsentwicklungen machen. Insbesondere das Semantic Web wird derzeit noch stark vernachlässigt.

Ein weiteres Manko aktueller Portale ist das Suchen geeigneter Lernressourcen. Dazu werden oft eine Volltextsuche sowie eine hierarchische Kategorisierung nach Themengebieten verwendet. Die Volltextsuche birgt den Nachteil, dass falsche Suchbegriffe zu schlechten Resultaten führen und allenfalls keine Lernressource gefunden wird, obwohl eine geeignete vorhanden ist. Die Kategorisierung drängt den Benutzer wiederum in einen Teilbereich der zur Verfügung stehenden Ressourcen. Wünschenswert wäre eine Möglichkeit, in der der Benutzer über alle Daten hinweg eine Auswahl treffen kann. Dies ist mit einer Kategorisierung nicht möglich. Deshalb soll im CasIS Portal eine verbesserte Suchmöglichkeit implementiert werden, welche es dem Benutzer einfach ermöglicht, passende Lernressourcen zu finden. Eine interessante Entwicklung ist hierbei die Fassettennavigation, welche in Kapitel 6.6 vorgestellt wird.

1.3 Struktur der Arbeit

In Kapitel 2 werden die Grundlagen von Web 2.0 und Ajax vorgestellt, um einen ersten Eindruck über die heutigen Modelle und Techniken zu erhalten. Dieser Abschnitt dient als Ausgangspunkt für Kapitel 3. Es wird aufgezeigt, welche Schwachpunkte das derzeitige Web aufweist und wie diese mit Hilfe von Semantic Web behoben werden können.

Kapitel 4 beschreibt traditionelle Lernmethoden und vergleicht diese mit aktuellen e-Learning Möglichkeiten. Im nächsten Schritt werden diese e-Learning Methoden dem Semantic Web gegenübergestellt und Vorteile und Nachteile des Semantic Web im Bereich des e-Learning erläutert.

Nach dieser grundlegenden Einführung in die Themenbereiche, werden in Kapitel 5 die Anforderungen an unser Portal besprochen und der Funktionsumfang aufgezeigt. Diese Funktionen werden als Grundlage für das Kapitel 6 verwendet.

Kapitel 6 beschreibt alle verwandten Arbeiten in Bezug auf unser Projekt. Zuerst werden semantische (e-Learning) Anwendungen vorgestellt und deren Verwendbarkeit im CasIS Projekt besprochen. Im nächsten Schritt werden einzelne geforderte Komponenten analysiert, welche noch nicht mit den vorgestellten Anwendungen realisiert werden können. In diesem Abschnitt erfolgt somit eine Evaluation der benötigten Software innerhalb des CasIS Portals.

Nach entsprechender Auswahl der Softwarekomponenten wird in Kapitel 7 die Implementation des CasIS Portals erläutert. Dabei wird ein Überblick über die Benutzeroberfläche geboten als auch eine technische Architekturansicht des Systems.

In Abschnitt 8 folgen die Erkenntnisse aus der Entwicklung und in Kapitel 9 wird die weitere Verwendung des Portals erläutert.

2

Grundlagen von Web 2.0 und Ajax

> „The Web 2.0 lesson: leverage customer-self service and algorithmic data management to reach out to the entire web, to the edges and not just the center, to the long tail and not just the head.“ [O'Reilly, 2005]

2.1 Web 2.0

Der Begriff Web 2.0 ist nur sehr wage zu umschreiben. Er entstand bei einer Konferenz zwischen O'Reilly und MediaLive International. Der Begriff Web 2.0 wurde von Tim O'Reilly in [O'Reilly, 2005] beschrieben. Dabei geht es um eine Plattform, welche bestimmten Prinzipen und Praktiken folgt. Diese sind unter anderem die folgenden:

- Innerhalb des Web 2.0 sollen Dienste angeboten werden und keine Softwarepakete. Der Vorteil liegt darin, dass der Kunde mit dem Unternehmen in Kontakt steht und aktuelle Informationen erhalten kann. Ein neuer Dienst wird somit sehr schnell erkannt und vielleicht auch genutzt. Bei Softwarepaketen hat der Nutzer keinen Kontakt mit dem Unternehmen und muss zuerst selbst eine aktuelle Software beschaffen, wenn er neue Funktionen benötigt.
- Durch die Partizipation und Mitentwicklung der Nutzer, welche Daten zur Verfügung stellen und abrufen, entsteht eine enorme Benutzergemeinschaft. Jeder, der einen Internetanschluss hat, kann und soll teilnehmen.
- Der Dienst sollte bei grösserer Nachfrage der Nutzer ausbaufähig und möglichst günstig skalierbar sein.
- Die Intelligenz und das Wissen sowie die individuellen Daten der Nutzer sollen zusammengeführt und allgemein zugänglich gemacht werden.

Genauere Beschreibungen und Ausführungen zum Thema Web 2.0 finden sich in [McCormack, 2002] oder [Alby, 2006].
Für die Entstehung und Entwicklung sowie das Fortbestehen des Web 2.0 sind drei Bereiche von besonderer Bedeutung. Diese werden in den folgenden Abschnitten genauer beschrieben.

2.1.1 Das Geschäftsmodell

Es existieren unterschiedliche Geschäftsmodelle für das Internet. Eine Erläuterung sämtlicher Geschäftsmodelle würde den Rahmen dieser Arbeit sprengen, weshalb auf die ausführlichen Beschreibungen in [Hammer and Wieder, 2003] verwiesen wird. Die richtige Auswahl des Geschäftsmodelles hat massgeblichen Einfluss auf den Erfolg des Unternehmens. Im Internet ist insbesondere das werbefinanzierte Modell sehr weit verbreitet. Ziel dabei ist es, möglichst viele Kunden auf eine Webseite zu locken, um dann gezielt Werbung einzublenden. Beispiele dafür sind flickr[1] oder t-online[2].

2.1.2 Die Benutzer

Ein wichtiger Aspekt im Bereich des Web 2.0 ist die Einbindung der Nutzer durch Partizipation. Es geht dabei nicht mehr darum, Informationen als Dienstleister selbst anzubieten, sondern den Benutzern des Dienstes die Möglichkeit zu geben, ihre eigenen Daten in den Service einzubringen. Anwendungen wie ein Benutzerforum, YouTube[3], MySpace[4], del.icio.us[5], flickr, eBay[6], Wikipedia[7] oder BitTorrent[8] folgen genau diesem Prinzip. Sie stellen lediglich den Dienst bereit, die wichtigen Daten werden von den jeweiligen Benutzern dem Dienst zur Verfügung gestellt. Je mehr Nutzer bereit sind, ihr Material und ihr Wissen beim Dienst zu hinterlegen, desto grösser wird die Benutzergemeinschaft und desto wertvoller wird der entsprechende Dienst. In diesem Zusammenhang spricht man von einem positiven Netzwerkeffekt.

2.1.3 Die Technik

Im Zusammenhang mit Web 2.0 wird meistens die Technik „Asynchronous JavaScript and XML“ (AJAX)[9] erwähnt. Diese Technik wird in Abschnitt 2.2 erläutert. Ein besonderer Vorteil von Ajax ist die Möglichkeit, Daten nachträglich von einem Server nachzuladen, ohne dass der Benutzer etwas davon merkt. Weitere wichtige Techniken sind Sprachen wie Hypertext Preprocessor (PHP)[10], Java Server Pages (JSP)[11] oder Active Server Pages (ASP.NET)[12] sowie JavaScript[13], welche dem Entwickler einfache Möglichkeiten bieten, Daten dynamisch, flexibel und sehr schnell darzustellen.

2.2 Ajax

AJAX beschreibt eine Vorgehensweise im World Wide Web, die es ermöglicht, einzelne Teile einer Webseite mit neuen Inhalten vom Server zu füllen. Dies hat zwei wichtige Vorteile:

[1] http://www.flickr.com
[2] http://t-onilne.de
[3] http://www.youtube.com
[4] http://www.myspace.com
[5] http://del.icio.us
[6] http://www.ebay.com
[7] http://de.wikipedia.org
[8] http://www.bittorrent.com
[9] http://developer.mozilla.org/en/docs/AJAX
[10] http://www.php.net
[11] http://java.sun.com/products/jsp
[12] http://msdn2.microsoft.com/en-us/asp.net/default.aspx
[13] http://developer.mozilla.org/en/docs/JavaScript

- Zwischen dem Browser und dem Server müssen weniger Daten transferiert werden. Dies führt dazu, dass die Applikation wesentlich schneller wird und die Benutzer weniger warten müssen.
- Neue Daten können im Hintergrund nachgeladen werden, ohne dass der Benutzer dies merkt.

Hinter dem Begriff AJAX verbirgt sich keine neue Technologie. Es handelt sich dabei um sehr bekannte Techniken des World Wide Web:

- HTML / XHTML und CSS zur Darstellung der Daten
- Document Object Model (DOM) zur dynamischen Darstellung der angefragten Daten und zur Manipulation der einzelnen Elemente innerhalb von HTML / XHTML und CSS
- XML und XSLT als Datenübertragungsformat und XSLT für Transformationen
- XMLHttpRequest zur asynchronen Datenanfrage an den Server
- JavaScript als Bindeglied zwischen den einzelnen Bereichen

Im folgenden soll der genaue Ablauf einer AJAX Anwendung dargestellt werden. Einen Überblick gibt die Abbildung 2.1.

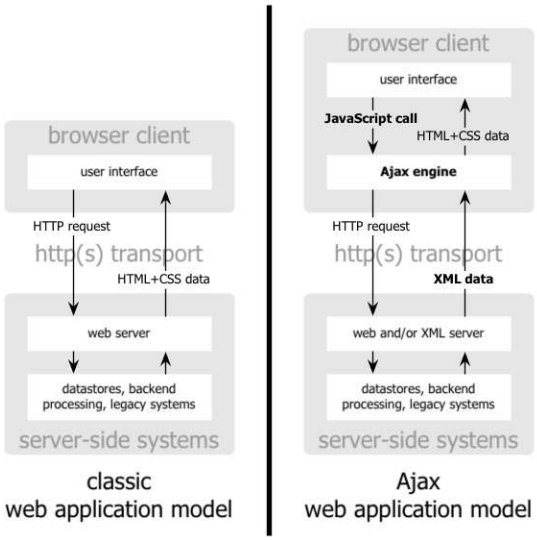

Abbildung 2.1: Ablauf einer Ajax Anwendung [Garrett, 2005]

Der linke Teil zeigt eine klassische Webanwendung ohne AJAX. Es wird jeweils eine HTTP Anfrage direkt an den Server geschickt und eine komplette Seite mit HTML und CSS zurückgesendet. Auf der rechten Seite wird die typische AJAX Anwendung dargestellt. Eine Anfrage erfolgt über das JavaScript-Objekt `XMLHttpRequest` und die AJAX Engine. Dabei wird lediglich ein einzelner Teil der Webseite angefordert und nicht die komplette Seite. Die angeforderten Daten werden im XML Format zurückgeliefert und können wieder mit Hilfe von JavaScript und dem Document Object Model (DOM)[14] in die Webseite eingebunden werden.

[14] http://www.w3.org/DOM/

In einer klassischen Webseite mit Kopfbereich, Navigation, Inhalt und Fussbereich würde dies bedeuten, dass beim Anklicken eines Links lediglich der Inhaltsbereich aktualisiert wird, jedoch nicht die anderen Bereiche. Diese müssen demnach nur einmal vom Server geladen werden.

3

Grundlagen des Semantic Web

„The Semantic Web is not a separate Web but an extension of the current one, in which information is given well defined meaning, better enabling computers and people to work in cooperation“ [Berners-Lee et al., 2001]

3.1 Probleme des derzeitigen World Wide Web

Die Verwendung des World Wide Web (WWW) hat sich seit der Entstehung stark geändert. Insbesondere in den letzten Jahren sind neue Anwendungsgebiete und Entwicklungen hervorgetreten. Grosse Unternehmen im Bereich des World Wide Web wie Google[1], Yahoo[2], Amazon[3] oder Ebay[4] offerieren komplette Dienstleistungen und Services. Insbesondere Suchdienste wie Google oder Yahoo werden heute häufig frequentiert. Das Auffinden von relevanten Informationen und deren Verwendung ist für den Menschen von heute zu einem fast unüberwindbaren Hindernis geworden. Die Information ist vorhanden, jedoch ist es sehr schwierig diese Informationen zu finden. In [Grigoris A., 2004] werden die aktuellen Probleme des derzeitigen WWW behandelt. Als erster negativer Punkt wird aufgeführt, dass das heutige Web lediglich für Menschen lesbar ist und mit sehr wenig Struktur verbunden ist. Die besten Softwaretools für das derzeitige Web sind Suchmaschinen wie Google, Altavista[5] oder Yahoo. Ihr Ansatz, nach den entsprechenden Schlüsselwörtern im Webinhalt zu suchen, birgt allerdings auch Risiken. Oft werden sehr viele nicht relevante Treffer zurückgeliefert, welche manuell durch Menschen verarbeitet und selektioniert werden müssen. Es wäre viel besser, wenn der Computer diese aufwändige Aufgabe bereits durch eine „genauere“ Selektionierung vornehmen könnte. Ein weiteres Problem liegt darin, dass die Bedeutung des Inhalts einzelner Webseiten nicht maschinenlesbar ist und die Interpretation der Texte und deren Bedeutung für den Computer kaum möglich ist. Auch die verschiedenen multimedialen Inhalte des Internets, wie Bilder und Videos, sind für den Computer kaum zu interpretieren oder zu katalogisieren. Gemäss [Daconta Michael C., 2003] besteht ein weiterer Mangel darin, dass die derzeitigen Webseiten und deren Inhalte kaum wiederverwendet werden können. Beispielsweise können Datenbankinhalte oft nur von einem einzigen Programm verwendet werden und sind für andere Dienste nicht oder nur schwer zugänglich.

[1] http://www.google.com
[2] http://www.yahoo.com
[3] http://www.amazon.com
[4] http://www.ebay.com
[5] http://www.altavista.com

3.2 Semantic Web

3.2.1 Der Begriff „Semantik"

Der Begriff „Semantik" stammt aus dem Griechischen und befasst sich mit der Bedeutung eines Wortes oder Satzes. Beispielsweise kann der Begriff „Bank" mehrdeutig ausgelegt werden. Es kann sich um eine Sitzbank handeln, oder um eine Bank im finanziellen Sinne. Die Bedeutung des Wortes wird für den Menschen erst aus dem Kontext heraus ersichtlich. Computer können diesen Kontext oft nicht interpretieren und benötigen deshalb zusätzliche Informationen.

3.2.2 Vision des Semantic Web

Die Idee des Semantic Web liegt darin, die Bedeutungen von Daten genauer zu definieren und dem Computer somit die Möglichkeit zu bieten, die Informationen besser zu „verstehen". Die unstrukturierten und verteilten Webinhalte sollen möglichst von Maschinen verarbeitet werden können. Die Webseite des World Wide Web Consortium (W3C) zu Semantic Web formuliert die Ziele des Semantic Web folgendermassen: [W3CSemWeb, 2001]

> „The Semantic Web is about two things. It is about common formats for integration and combination of data from diverse sources, where on the original Web mainly concentrated on the interchange of documents. It is also about language for recording how the data relates to real world objects."

Das Semantic Web verfolgt demnach zwei Ziele. Erstens sollen unterschiedliche Daten aus verschiedenen Quellen integriert werden. Dies gibt die Möglichkeit, die Daten anderen zur Verfügung zu stellen. Zweitens soll eine Verknüpfung zwischen den Daten und realen Objekten erstellt werden.

3.3 Semantic Web Layer

Die einzelnen Schichten des Semantic Web sind in unterschiedliche Layer eingeteilt. Diese verschiedenen Schichten sind in Abbildung 3.1 abgebildet. Der Aufbau in einzelnen Layern hat den Vorteil, dass komplexe Aufgaben aufgeteilt werden können. Dieser Ansatz wurde beispielsweise beim OSI - Referenzmodell sowie bei der Java Virtual Machine verwendet. Jede unterliegende Schicht offeriert der überliegenden Schicht ihren Dienst an. Dies bewirkt eine Entkopplung der einzelnen Aufgaben und ermöglicht das Austauschen einzelner Schichten. [Grigoris A., 2004] definiert zwei Prinzipien bei der Entwicklung neuer Schichten:

- Abwärtskompatibilität: Agenten innerhalb einer Schicht sollten auch die Daten der darunterliegenden Schicht interpretieren und benutzen können
- Aufwärtsverständnis: Agenten innerhalb einer Schicht sollten auch Teile der darüberliegenden Schicht verstehen und benutzen können.

In den folgenden Abschnitten erfolgt eine Erläuterung der einzelnen Schichten.

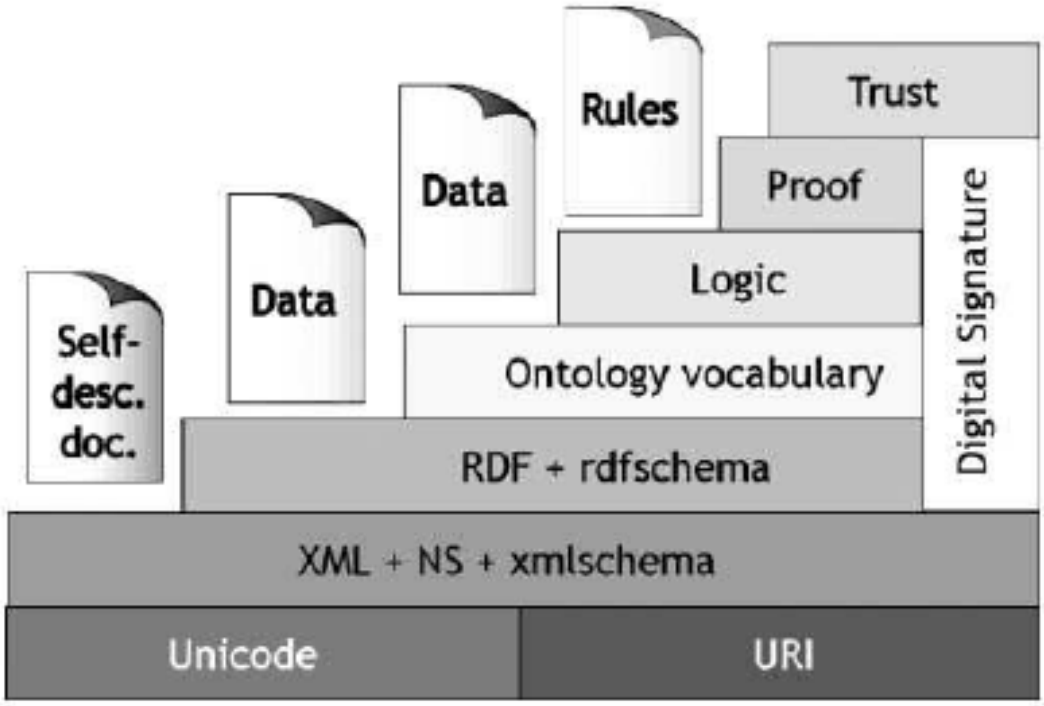

Abbildung 3.1: Semantic Web Layer [Grigoris A., 2004]

3.3.1 Unicode / Uri

Der Unicode Standard [UnicodeOrg, 2007] wurde entwickelt, um weltweit Daten aus den verschiedensten Sprachen auszutauschen oder weiter zu verarbeiten. Er vereint somit die häufigsten Sprachen beziehungsweise deren einzelne Zeichen in einem Standardformat. Dabei wird jedem Zeichen eine eindeutige Zahl (und eine eindeutige Bit-Repräsentation) zugeordnet. Diese Zuordnung kann beispielsweise auf 8, 16 oder 32 Bit pro Zeichen erfolgen. Im Bereich des WWW ist beispielsweise die UTF-8 Codierung weit verbreitet, da diese kompatibel ist mit dem 7-Bit-ASCII Standard.
Besondere Vorteile des Unicode Zeichensatzes sind Plattform-, Programm- als auch Sprachunabhängigkeit. Somit ist Unicode für den weltweiten Datenaustausch unersetzlich geworden und ist anerkannter Industriestandard.
Das Semantic Web speichert seine Daten im Klartext und benutzt dazu den Unicode Standard. Somit ist eine effiziente und weltweite Nutzbarkeit gewährleistet.

Das Semantic Web benutzt diesen Standard für die „Darstellung" seiner Daten. Diese sollen weltweit effizient weiterverwendet werden können.

Unter dem Uniform Resource Identifier (URI) [Network Working Group, 2007] wird eine Sequenz von Zeichen verstanden, welche eine abstrakte oder physikalische Ressource identifiziert. Beispiele für solche URI's sind:

- http://www.google.ch
- http://www.ifi.unizh.ch

- mailto:john@doe.com

Innerhalb des Semantic Web ist es notwendig, die beschriebenen Ressourcen einfach referenzieren zu können. URI's bieten dazu die nötigen Mittel.

3.3.2 XML und XML-Schema und Namensräume

Bei der eXtensible Markup Language (XML) handelt es sich um eine einfache, sehr flexible Beschreibungssprache für Dokumente und Daten. Der Vorteil von XML liegt darin, dass die Sprache unabhängig von Betriebssystemen ist und in unterschiedlichen Anwendungen eingesetzt werden kann [Born, 2005]. Wie bereits in Darstellung 3.1 auf Seite 9 ersichtlich, baut XML auf Unicode und den URI's auf und nutzt somit deren Vorteile. Der weitere Nutzen von XML liegt in folgenden Bereichen:

- frei definierbare Elemente
- Trennung von Inhalt und Darstellung
- Transformation in unterschiedliche Formate, z.B. XHTML mit Hilfe von XSLT
- Austausch von Daten zwischen Systemen, z.B. SOAP

XML stellt somit einen geeigneten Rahmen zur Verfügung, Daten sehr flexibel darzustellen und auszutauschen. Bei der Bearbeitung von XML-Dokumenten mit Hilfe von Parsern[6] wird lediglich die Wohlgeformtheit geprüft. Unter Wohlgeformtheit ist die korrekte Syntax des Dokumentes zu verstehen, z.B. dass für jedes öffnende auch wieder ein schliessendes Element vorhanden ist oder dass die Elemente genau gleich geschrieben sind. Wie lässt sich aber festlegen, welche Elemente mit welchem Wert verwendet werden dürfen? An dieser Stelle setzt das sogenannte XML-Schema an und bietet eine Gültigkeitsüberprüfung. Nur die im Schema definierten Elemente in der vorgegebenen Struktur sind erlaubt und dürfen auch nur die definierten Werte enthalten. Es lässt sich sowohl der Datentyp, als auch dessen Ausprägung definieren. Zum Beispiel kann definiert werden, dass ein Integerwert innerhalb eines bestimmten Wertebereichs sein muss. Weitere Informationen zum Thema XML und XML-Schema finden sich im Buch [Born, 2005].

Namensräume (Name Spaces, NS) werden benutzt um gleichnamige XML Tags voneinander unterscheiden zu können. Dabei wird ein einfacher Präfix vor das eigentliche XML Tag gestellt. Die Definition sieht folgendermassen aus: [Born, 2005]

- `xmlns:<pefix>=<URI>`

Ein konkretes Beispiel würde folgendermassen aussehen

- `<rdf:RDF xmlns:rdf="http://ww.w3.org/1999/02/22-rdf-syntax-ns#">`

Das zweite Beispiel zeigt die Einleitung eines Resource Description Framework Dokuments (RDF-Dokuments) und die Definition des entsprechenden Namensraumes „rdf" zusammen mit der entsprechenden URI im WWW.

In Bezug auf das Semantic Web hat XML allerdings einen sehr grossen Nachteil. Es bietet keinerlei Semantik [Grigoris A., 2004]. Auch die Schachtelung der einzelnen Elemente lässt nicht auf eine Semantik schliessen, da diese Schachtelung sehr unterschiedlich vorgenommen werden kann. Deshalb benötigt es zusätzliche Mechanismen, um diese Semantik beschreiben zu können.

[6] Software zur Verarbeitung von strukturierten Daten

3.3.3 RDF und RDF-Schema

Das Resource Description Framework (RDF) ist der eigentliche Kern des Semantic Web und baut auf XML auf. Ein wesentlicher Unterschied zu XML ist, dass bei RDF die Reihenfolge der Elemente keine Rolle spielt [Grigoris A., 2004].
Gemäss [W3CRDF, 2004] handelt es sich bei RDF um eine Sprache, welche Informationen über Ressourcen im World Wide Web bereitstellt. RDF verfolgt das Ziel, diese Daten möglichst maschinenlesbar zur Verfügung zu stellen und einen einfachen Datenaustausch zu ermöglichen.
RDF basiert auf sogenannten Tripeln. Diese können in die Bereiche Subjekt, Prädikat und Objekt unterteilt werden:

- Das **Subjekt** beinhaltet die Ressource, welche über eine URI identifiziert werden kann. Über sie wird eine Aussage getroffen.
- Das **Prädikat** stellt eine bestimmte Beziehung zwischen Subjekt und Objekt her.
- Das **Objekt** wird vom Prädikat referenziert. Beim Objekt kann es sich dabei um eine URI oder um ein Literal handeln.

Die Kombination dieser drei Elemente wird als Statement bezeichnet. Ein kleines Beispiel ist in Listing 3.1 dargestellt.

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description about="http://www.ifi.unizh.ch">
    <dc:creator>Institut fuer Informatik, Universitaet Zuerich</dc:creator>
  </rdf:Description>
</rdf:RDF>
```

Listing 3.1: RDF Beispiel

In diesem Beispiel wird der Ersteller einer Webseite beschrieben, wobei die einzelnen Elemente folgendermassen zugeordnet werden:

- **Subjekt:** http://www.ifi.unizh.ch
- **Prädikat:** dc:creator
- **Objekt:** Institut für Informatik, Universität Zürich

Mit Hilfe des RDF Schema erhält der Entwickler zusätzlich die Möglichkeit, die Applikationsdomäne sowie die Semantik dieser Domäne festzulegen [Grigoris A., 2004]. Das Schema definiert die benutzbaren Vokabulare sowie die einzelnen Eigenschaften [W3CRDF, 2004]. Es bietet folgende Möglichkeiten:

- **Subklassenhierarchien:** `rdfs:subClassOf`) Die Klasse ist eine genauere Beschreibung einer anderen Klasse.
- **Eigenschaftenhierarchien:** `rdfs:subPropertyOf`) Die Eigenschaft ist eine genauere Beschreibung einer anderen Eigenschaft.
- **Domaineinschränkungen:** (`rdfs:domain`) Gibt an, welche Eigenschaften auf welches Subjekt angewendet werden dürfen.

- **Wertbeschränkungen:** (`rdfs:range`) Gibt an, welche Eigenschaften auf welches Objekt angewendet werden dürfen.

Eine komplette Übersicht über die Klassen und Eigenschaften in RDF-Schema stellt die offizielle Webseite des W3C [W3CSchema, 2004] bereit. Ein kleines Beispiel zum Verständnis der Funktionsweise von RDF-Schema zeigt Listing 3.2.

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

  <rdfs:Class rdf:ID="Drink" />

  <rdfs:Class rdf:ID="Softdrink" />
    <rdfs:subClassOf rdf:resource="#Drink" />
  </rdfs:Class>
</rdf:RDF>
```

Listing 3.2: RDF-Schema Beispiel

Das Beispiel zeigt eine Getränkehierarchie mit Hilfe von `rdfs:subClassOf`. „Drink" bildet dabei die obere und „Softdrink" die untere Klasse in der Hierarchie.

Das entspechende Prinzip wurde somit von XML und XML-Schema übernommen. Trotzdem gewährt RDF mit seinen oft binären Verknüpfungen und das RDF-Schema mit seinen eingeschränkten Möglichkeiten zu wenig Spielraum. Ein RDF-Schema ist auf eine Subklassenhierarchie sowie eine Eigenschaftenhierarchie limitiert. Diese Einschränkung wird mit der nächsthöheren Schicht, der Ontologieschicht, behoben.

3.3.4 Ontology vocabulary

Unter Ontologie versteht [Daconta Michael C., 2003] S.190 folgende Definition:

> „An ontology models the vocabulary and meaning of domains of interest: the objects (things) in domains; the relationships among those things; the properties, functions, and processes involving those things; and constraints on and rules about those things."

Eine Ontologie versucht demnach, Dinge aus der Realität möglichst genau zu beschreiben. Zusätzlich sollen die Beziehungen und Bedeutungen (Semantik) unter den einzelnen Dingen beschrieben werden. Allerdings werden mit einer Ontologie nicht die eigentlichen Dinge (Instanzen) beschrieben sondern lediglich die möglichen Strukturen dieser Instanzen.

Ontologien können in unterschiedliche Semantikklassen eingeteilt werden. [Daconta Michael C., 2003] definiert dazu ein sogenanntes Ontologiespektrum von Ontologien mit schwacher bis starker Semantik:

- **Taxonomie:** Unter Taxonomie versteht [Daconta Michael C., 2003] die Klassifizierung aufgrund von Beziehungen. Diese sind allerdings nicht wirklich genau definiert und können beispielsweise `subclass of` oder `part of` lauten, auch wenn diese Eigenschaft vielleicht nicht genau zutrifft oder eine noch genauere Beziehungsbeschreibung wünschenswert wäre.

- **Thesaurus:** Als nächsthöhere Stufe wird der Thesaurus aufgeführt. Darunter wird ein Wörterverzeichnis verstanden, das die Klassifizierung von Datenmaterial ermöglicht [Bertelsmann, 1988]. Die Tabelle 3.1 liefert einen Überblick über die einzelnen Klassifizierungen.

- **Conceptual Model:** Diese Modelle beinhalten bereits viele Informationen, wie beispielsweise Subklassenbeziehungen. RDF und RDF-Schema bewegen sich in diesem Bereich.

- **Local Domain Theory:** Mit Hilfe von Logik soll es möglich werden, sehr viel und sehr komplexes und konsistentes Wissen abzubilden. Dem Computer soll die Möglichkeit geboten werden, selbst Schlüsse und Herleitungen herstellen zu können. Aufgebaut wird auf Axiomen sowie Inferenzregeln, welche schlussendlich dazu genutzt werden, Theoreme zu bilden. Mit OWL ist ein mächtiges Werkzeug für diesen Bereich entstanden. Mit Hilfe von Reasonern[7] ist es weiterhin möglich, aus OWL entsprechende Schlussfolgerungen abzuleiten. Ein Beispiel für diesen Bereich wäre die Web Ontology Language[8] (OWL) mit seinen drei Varianten OWL light, OWL DL sowie OWL Full.

Semantische Relation	**Definition**	**Beispiel**
Synonym	Term X hat beinahe die gleiche Bedeutung wie Term Y.	„Report“ ist ein Synonym für ein „Dokument“.
Homonym	Term X wird gleich buchstabiert wie Term Y, welcher jedoch eine andere Bedeutung hat.	Der „Tank“ ist ein Militärfahrzeug sowie ein Homonym für den Begriff „Tank“ als Behälter für Flüssigkeiten.
umfassender (hierarchisch: `parent of`	Term X ist umfassender als ein Term Y.	„Organisation“ ist umfassender als „Finanzinstitut“.
enger gefasst (hierarchisch: `child of`	Term X ist enger gefasst als ein Term Y.	„Finanzinstitut“ ist enger gefasst als „Organisation“.
Assoziation	Term X ist in Assoziation mit Term Y. Es besteht irgendeine unspezifizierte Beziehung zwischen diesen Termen.	Ein „Nagel“ ist assoziiert mit „Hammer“.

Tabelle 3.1: Semantische Beziehungen im Thesaurus, frei übersetzt aus [Daconta Michael C., 2003] S.160

Für Menschen ist die Semantik eines Wortes oft aus dem Kontext ableitbar. Wie bereits in 3.2.1 erwähnt, ist der Begriff „Bank“ mehrdeutig. Wird er allerdings im Zusammenhang mit Geld aufgeführt, muss es sich um ein Finanzinstitut handeln und nicht um eine Sitzbank. Diese Schlussfolgerung wäre für den Computer nicht möglich. Er benötigt zusätzliche Informationen, um diese Verbindungen zwischen Wörtern und Bezeichnungen herstellen zu können. Mit OWL wurde eine Sprache entwickelt, die diese Verbindungen herstellen kann und für den Computer die Zusammenhänge verständlich macht. Somit wird es für den Computer möglich, genau diese Schlussfolgerungen durchführen zu können.

[7] Software, welche das Schlussfolgern innerhalb von Ontologien ermöglicht
[8] OWL wird ausführlich in [W3COWL, 2004] beschrieben

3.3.5 Logic, Proof

Der Begriff der Logik stammt aus dem Griechischen und bedeutet die „Lehre von der Folgerichtigkeit“ [Bertelsmann, 1988]. Bei der Logik geht es demnach darum, bestimmte Schlussfolgerungen herleiten zu können. Ein kleines Beispiel zu einer logischen Schlussfolgerung zeigt die Tabelle 3.2.

Beschreibung	**Sachverhalt**
Wenn die Ampel grün ist, dann darf durchgefahren werden.	Bedingung
Es ist grün.	Zustand
Es darf durchgefahren werden.	Schlussfolgerung (Modus ponens)

Tabelle 3.2: Beispiel einer logischen Schlussfolgerung

Solche Schlussfolgerungen sollen auch im Semantic Web möglich sein. Insbesondere soll es möglich sein, diese Schlussfolgerungen automatisch von Maschinen vornehmen zu können. Ontologien bieten bereits ein mächtiges Mittel zur genauen Beschreibung von Dingen. Mit Hilfe von Reasonern[9] ist es möglich, diese Schlussfolgerungen herzuleiten. Für das Semantic Web ist es zusätzlich interessant, diese logischen Beschreibungen auszutauschen, damit Softwareagenten diese weiterverwenden können. Deshalb sind entsprechende Beschreibungssprachen nötig. Ein entsprechender Ansatz ist RuleML[10] aus dem Jahre 2000. Dabei werden in der Notation von XML entsprechende Regeln definiert, welche anhand von DTD's oder XML- Schema validiert werden können. Da RuleML auf XML basiert, ist ein einfacher Austausch der einzelnen Regeln gewährleistet. Ein weiterer Ansatz ist Common Logic[11] (CL). Dieser Ansatz basiert auf dem Knowledge Interchange Format (KIF). Ziel von KIF war es, ein einziges Austauschformat für Wissen zu schaffen.
Mit Hilfe dieser Regeln und den Definitionen in den Ontologien ist es nun möglich, entsprechende Validierungen durchzuführen. Diese sind allerdings sehr komplex.

3.3.6 Trust, Digital Signature

Die grössere Vernetzung von Daten im Semantic Web benötigt zusätzliche Sicherheitsmechanismen. Die zahlreichen Agenten und Benutzer müssen überprüft werden können, bevor sie Zugang zu sensiblen Daten erhalten. Diese Überprüfung kann mit Hilfe von digitalen Signaturen erfolgen. Es ist wichtig, dass die Benutzer den Anwendungen vertrauen können und dass sensible Daten nicht an unbefugte Personen gelangen.

3.4 Anwendungen des Semantic Web

Die Anwendungen des Semantic Web sind sehr vielfältig. [Grigoris A., 2004] nennt folgende Beispiele:

[9] Ein Reasoner ist eine Softwarekomponente, welche genau diese Schlussfolgerung aufgrund bereits bestehender (semantischer) Daten durchführen kann
[10] http://www.ruleml.org
[11] http://cl.tamu.edu

- **Verbesserung von Suchergebnissen:** Mit Hilfe der zusätzlichen Metadaten wird es für den Computer einfacher, die richtigen Daten zu finden. Er kann Synonyme und Homonyme erkennen, Assoziationen herstellen und Schlussfolgerungen ziehen. Dies ermöglicht eine verbesserte Trefferquote.
- **vereinfachte Datenintegration:** Zusammenführen von unterschiedlichen Datenquellen (Datenbanken, Dateien) mit Hilfe einer einheitlichen zentralen Ontologie, welche die Daten und deren Semantik definiert.
- **e-Learning:** Das Semantic Web bietet die Möglichkeit, das Lernen individueller und flexibler zu gestalten. Eine ausführliche Beschreibung erfolgt in Kapitel 4.
- **automatisierte Web Services:** Im WWW gibt es derzeit unzählige Web Services. Das Problem dieser Services ist, dass sie oft isoliert benutzt werden müssen. Die Zusammenführung und Nutzung mit Hilfe von entsprechenden Softwareagenten wäre wünschenswert.
- **Annotation von Multimediadaten:** Multimediadaten wie Bilder, Musik und Video können sehr schlecht katalogisiert werden. Sie benötigen zusätzliche Metadaten um den Inhalt, den Autor, das Jahr oder andere benötigten Daten zu beschreiben.

Weitere interessante Beispiele stammen aus den Bereichen Wissensmanagement oder Semantische Portale. Ein gut bekanntes Beispiel ist sicherlich auch das Rich Side Summary (RSS), das auf RDF basiert und heute häufig für Nachrichten eingesetzt wird. Eine ausführliche Beschreibung verschiedener Beispiele und Projekte können unter [SWAD-E, 2001] nachgelesen werden. Die Flexibilität und die Erweiterbarkeit des Semantic Web lassen viel Spielraum für weitere Anwendungen.

3.5 Probleme des Semantic Web

Das Semantic Web ist noch nicht fertig entwickelt. Die oberen Schichten sind nur teilweise implementiert und es existieren nur wenige Werkzeuge um das Semantic Web effizient zu nutzen. Zusätzlich sind viele Bereiche noch Gegenstand der Forschung.

4

e-Learning und Semantic Web

> „eLearning is just-in-time education integrated with high velocity value chains. It is the delivery of individualized, comprehensive, dynamic learning content in real time, aiding the development of communities of knowledge, linking learners and practitioners with experts“ [Drucker, 2000]

4.1 Traditionelles Lernen

Traditionelles Lernen basiert in Anlehnung an [Grigoris A., 2004] auf folgenden Eigenschaften:

- Der Instruktor definiert den Inhalt, Zeitplan sowie die Geschwindigkeit des Lernens.
- Der Zugang erfolgt linear. Dies bedeutet, dass das Wissen in einer vorgegebenen Reihenfolge konsumiert wird.
- Das Lernen erfolgt zeit- und ortsabhängig.

Genau diese Eigenschaften treffen auch heute noch auf den klassischen Schulunterricht zu. Dies führt dazu, dass alle Klassenmitglieder unabhängig ihrer individuellen Eigenschaften genau gleich behandelt werden und das Lernen eher als Massenveranstaltung angesehen werden kann. Der Lernprozess ist sehr wenig personalisiert und somit auch nicht auf die einzelnen Lerngeschwindigkeiten der Klassenmitglieder angepasst.

4.2 e-Learning

e-Learning[1] hat seinen Ursprung im rechnergestützten Unterricht[2], welches einen Versuch darstellte, die Ausbildung zu automatisieren, den bezahlten Instruktor zu ersetzen sowie das Lerntempo auf die eigenen Bedürfnisse der Lernenden anzupassen [L.Stojanovic et al., 2001]. E-Learning bietet darüber hinaus weitere Möglichkeiten, welche die benannten Nachteile aus Abschnitt 4.1 aufheben. Die Tabelle 4.1 zeigt einen guten Vergleich zwischen Training und e-Learning und hebt

[1] elektronisch unterstütztes Lernen
[2] Computer Based Training (CBT)

die zusätzlichen Stärken von e-Learning hervor. Insbesondere die Zeit- und Ortsunabhängigkeit bietet den Lernenden einen grossen Vorteil. Somit ist es auch nicht verwunderlich, dass e-Learning insbesondere bei höheren Ausbildungen an Universitäten und Fachhochschulen eingesetzt wird. Dort treffen unterschiedliche Personen mit unterschiedlichen Ausbildungen und Vorkenntnissen aus unterschiedlichen Ortschaften aufeinander. Ein personalisiertes und unabhängiges Lernen ist dort besonders hilfreich.

Dimension	Training	eLearning
Auslieferung	Push: Instruktor definiert vorgehen	Pull: Student definiert vorgehen
Ansprechbarkeit	Antizipatorisch: Problem sollte bekannt sein	Reaktionär: gibt direkt Antwort zu einem Problem
Zugang	Linear: fixer Verlauf	Nicht-Linear: direkter Zugang zu Wissen
Symmetrie	Asymmetrisch: Training erscheint als separate Aktivität	Symmetrisch: Lernen als integrierte Aktivität
Modalität	Diskret: innerhalb von Modulen, mit fixem Start und End	Anhaltend: Lernen parallel zum Geschäftsprozess, ohne Ende
Autorität	Zentral: Inhalt wird vom Lehrer entwickelt und freigegeben	Verteilt: Inhalt aus der Interaktion von Benutzern und Lehrern
Personalisierung	Massenproduktion: Inhalt für viele bestimmt	Personalisiert: Inhalt wird individuel vom Benutzer definiert
Anpassungsfähigkeit	Statisch: Inhalt und Organisation bleibt in ihrer ursprünglichen Form	Dynamisch: Inhalt variert aufgrund von Benutzereingaben, Erfahrungen, neuen Praktiken, Geschäftsprozessen sowie Heuristiken

Tabelle 4.1: Unterschiede zwischen Training und e-Learning, in Anlehnung an [Drucker, 2000]

Derzeit gibt es unterschiedliche Standards im Bereich des e-Learning. Darunter ist IEEE Learning Object Metadata (LOM)[3], ARIADNE[4], IMS[5] sowie der umfassende Standard Sharable Content Object Reference Model (SCORM)[6]. Insbesondere der letzte Standard hat sich weit verbreitet und wird von vielen Lernplattformen unterstützt. Diese Metadatenstandards unterstützen die formale Semantik nicht ausreichend. Sie unterstützen lediglich die Interoperabilität innerhalb der Domäne, jedoch nicht zwischen unterschiedlichen und verteilten Domänen [L.Stojanovic et al., 2001].

4.3 e-Learning und Semantic Web

Das oben beschriebene Problem könnte mit Hilfe des Semantic Web behoben werden. Es bietet eine sehr starke Semantik sowie die Möglichkeit der maschinenbasierten Verarbeitung der verfügbaren Metadaten. Zusätzlich sind semantische Abfragen sowie konzeptionelle Navigation des Lernmaterials möglich [Grigoris A., 2004]. In Tabelle 4.2 findet sich eine gute Übersicht über

[3]http://ltsc.ieee.org/wg12/
[4]http://www.ariadne-eu.org
[5]http://www.imsproject.org
[6]http://www.adlnet.gov

die Unterschiede zwischen e-Learning und e-Learning mit Hilfe von Semantic Web. Damit eine starke Semantik gewährleistet werden kann, werden Ontologien benötigt. [Grigoris A., 2004] beschreibt dazu drei nötige Ontologien:

- Die **Inhaltsontologie** beschreibt die grundsätzlichen Konzepte der Domäne, in der das Lernen stattfindet.
- Die **Pädagogische Ontologie** definiert beispielsweise die Art des Lernmaterials wie Lektion, Tutorial, Beispiel oder Übung.
- Die **Strukturontologie** beschreibt die logische Struktur des Lernmaterials, wie vorhergehend oder nachfolgend.

[L.Stojanovic et al., 2001] nennt ebenfalls drei Ontologien, welche sich teilweise mit den obigen decken:

- eine Ontologie über das Lernmaterial selber (Inhalt)
- wie das entsprechende Thema präsentiert werden soll (Kontext)
- die Struktur des Lernmaterials

Mit Hilfe dieser verschiedenen Dimensionen lässt sich die Lernumgebung ausreichend beschreiben.

Anforderungen	e-Learning	Semantic Web
Auslieferung	Pull: Student definiert vorgehen	Lernmaterial ist verteilt im WWW und unterliegt Standardontologien, welche mit Hilfe von semantischen Anfragen abgefragt werden können.
Ansprechbarkeit	Reaktionär: gibt direkt Antwort zu einem Problem	Softwareagenten benutzen allgemeine Servicesprachen, um die Auslieferung von Lernmaterial aufgrund des aktullen Problems zu gewährleisten.
Zugang	Nicht-Linear: direkter Zugang zu Wissen	Benutzer beschreibt Problem (Lernziel, Vorwissen etc.) und nutzt semantische Abfragen, um das passende Lernmaterial zu erhalten.
Symmetrie	Symmetrisch: Lernen als integrierte Aktivität	Das Semantic Web bietet die Möglichkeit, als Plattform alle Geschäftsprozesse (auch das Lernen) einer Organisation zu integrieren.
Modalität	Anhaltend: Lernen parallel zum Geschäftsprozess, ohne Ende	Aufgrund von personalisierten Agenten wird die Lernumgebung dynamisch angepasst und in den Geschäftsprozess integriert.
Autorität	Verteilt: Inhalt aus der Interaktion von Benutzern und Lehrern	möglichst dezentralisiert
Personalisierung	Personalisiert: Inhalt wird individuell vom Benutzer definiert	Der Benutzer sucht das Lernmaterial aufgrund seiner individuellen Bedürfnisse. Die Ontologie dient dabei als Bindeglied zwischen Benutzeranfragen und dem Lernmaterial.
Anpassungsfähigkeit	Dynamisch: Inhalt variert aufgrund von Benutzereingaben, Erfahrungen, neue Praktiken, Geschäftsprozessen sowie Heuristiken	Verteiltes Wissen in den unterschiedlichsten Formen, ständige Verbesserungen des Lernmaterials.

Tabelle 4.2: Unterschiede zwischen e-Learning und Semantic Web, in Anlehnung an [L.Stojanovic et al., 2001]

5

Anforderungen an das CasIS Portal

5.1 Anforderungen

In diesem Abschnitt werden die grundlegenden Anforderungen an das CasIS System behandelt. Dabei werden funktionale Anforderungen, Leistungsanforderungen als auch Qualitätsanforderungen erläutert und dargestellt.

5.1.1 Funktionale Anforderungen

Benutzerrollen

Das CasIS System basiert auf zwei unterschiedlichen Benutzerrollen.
Administratoren erhalten volle Rechte für das CasIS System. Sie haben somit die Möglichkeit, Einstellungen am Portal vorzunehmen, wie beispielsweise das Sperren von Ressourcen oder das Auswählen von Fallstudien, als auch die Möglichkeit, sich als normale Benutzer innerhalb der Fallstudien zu bewegen. Zusätzlich erhält der Administrator Zugriff auf Zusatzinformationen innerhalb der Fallstudien, die von den Benutzern nicht eingesehen werden dürfen.
Die zweite Benutzergruppe besteht aus den eigentlichen Benutzern. Sie erhalten lediglich eine eingeschränkte Sicht. Ihnen stehen die Funktionen zur Fassettennavigation sowie zur Volltextsuche jederzeit zur Verfügung.
Es soll kein Gastbenutzer eingerichtet werden. Dies bedeutet, dass das System nur von angemeldeten Benutzern verwendet werden kann.

CasIS Prozess

Die Bearbeitung erfolgt in einem vorgegebenen Prozess, welcher aus vier unterschiedlichen Modulen besteht:

- **Eingangstest:** Dieser evaluiert, wie weit sich der Student mit einer bestimmten Materie bereits auskennt. Aufgrund dieses Tests wird schlussendlich ein entsprechender Pfad durch die unterschiedlichen Module und Lernressourcen[1] vorgeschlagen.

[1] als Lernressource wird allgemein der Inhalt eines der vier CasIS Module verstanden

- **Vorbereitungsmodule:** Die Vorbereitungsmodule helfen dem Studenten, sich auf die Aufgaben in der Fallstudie vorzubereitung und zusätzliches Wissen anzueignen.
- **Fallstudien:** In der Fallstudie geht es darum, das bereits Gelernte an einem konkreten Fall anzuwenden und diesen Fall möglichst effizient zu lösen. Dabei kann der Student die bereitgestellte Toolbox als Unterstützung verwenden.
- **Toolbox:** Die Toolbox bietet zahlreiche Werkzeuge, die den Studenten dabei unterstützen, die Fallstudien erfolgreich zu lösen.

Diese vier Module bilden den Kern der ganzen Applikation. Die einzelnen Lernressourcen in den Modulen werden mit XML und XML-Schema beschrieben. Die Transformation in HTML und Latex erfolgt derzeit über die Extensible Stylesheet Language[2] (XSL) sowie über eine XSL Transformation[3] (XSLT). Jedes einzelne Modul wird mit einem separaten XML und XML-Schema beschrieben. Somit entstehen insgesamt vier nötige Transformationen.
Bezüglich der Navigation innerhalb der Module ist der Benutzer absolut frei, bis auf eine Ausnahme. Hat der Benutzer den Eingangstest absolviert, kann er nicht mehr zurück. Innerhalb der anderen Bereiche bestehen freie Navigationsmöglichkeiten. Diese Freiheit ermöglicht dem Benutzer, jede Ressource im Portal entsprechend zu nutzen. Abbildung 5.1 zeigt den CasIS Prozess.

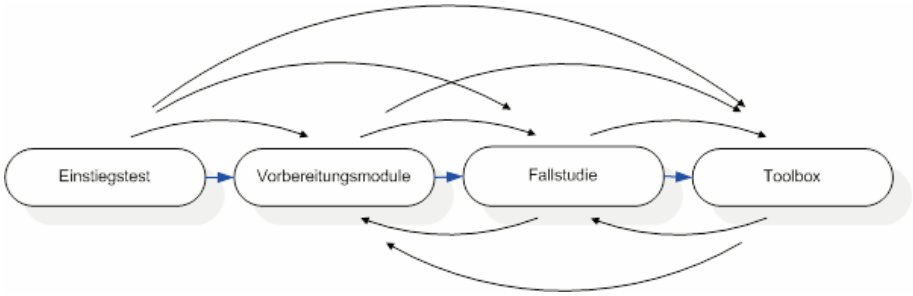

Abbildung 5.1: Der CasIS Prozess

Auswahl von Prozessmodulen

Der Administrator soll die Möglichkeit haben, einzelne Module aus dem Portal zu entfernen. Beispielsweise kann er den Einstiegstest entfernen und nur die anderen drei Module benutzen. Es muss immer mindestens ein Modul für die Benutzer zur Verfügung stehen. Der Administrator entscheidet frei, welche Module zur Verfügung stehen.

Adaptivität

Die einzelnen Bereiche der Anwendungen sind innerhalb von Benutzergruppen anpassbar. Dies bedeutet beispielsweise, dass für die eine Benutzergruppe die Navigation links angeordnet wird, während bei der anderen Gruppe die Navigation rechts steht. Die einzelnen Komponenten können somit links oder rechts angeordnet werden.

[2] http://www.w3.org/TR/xsl
[3] http://www.w3.org/TR/xslt

Schnelle Reaktionszeit

Eine schnelle Reaktionszeit bietet dem Benutzer ein angenehmes Navigieren innerhalb des CasIS Portals. Lange Wartezeiten verärgern die Benutzer und senkt den Lernwillen. Eine angenehme Oberfläche und schnelle Reaktionszeiten sind erforderlich.

Suche

Innerhalb des Portals werden zahlreiche Fallstudien in HTML zur Verfügung gestellt. Diese werden dem Benutzer möglichst leicht zugänglich gemacht. Aus diesem Grunde werden zwei verschiedene Suchmöglichkeiten angeboten.

- Das Angebot einer Fassettennavigation (siehe Abschnitt 6.6) bietet dem Nutzer die Möglichkeit, auf einfache Weise unterschiedliche HTML Module aufgrund verschiedener Kriterien auszuwählen. Die einzelnen Fassetten dienen dabei als Wegweiser und als Unterstützung. Die Daten werden von den jeweiligen Administratoren erfasst.
- Zusätzlich zur Fassettennavigation wird eine Volltextsuche angeboten. Diese ermöglicht es dem Benutzer auch Text, welcher in der Fassettennavigation nicht enthalten ist, ohne weiteres zu durchsuchen. Dies ist nötig, da im CasIS Portal viele Daten in statischen HTML Dokumenten vorhanden sind und dieses Wissen dem Benutzer ebenfalls zur Verfügung stehen soll. Ein leichter Zugang kann mit einer Volltextsuche gewährleistet werden. Das Suchinterface ist leicht verständlich und intuitiv benutzbar.

Farben

Die Farben innerhalb des CasIS Portals werden von dem System „Foundamentals of Information Systems“ (FOIS)[4] übernommen. Weitere Farben im entsprechenden Farbton dürfen verwendet werden. Die Farben werden in einer separaten CSS Datei abgelegt.

Hochladen weiterer Lernressourcen

Für den leichten Umgang mit den Lernressourcen ist es erforderlich, auf einfache Art und Weise neue Lernressourcen anbieten zu können. Im Administrationsbereich ist deshalb eine Funktion zu implementieren, die die Lernressource entgegennimmt und direkt in die Fassettennavigation und die Suchfunktion einbindet, damit der neue Inhalt den Studenten zur Verfügung steht.

Browserunterstützung

Das CasIS Portal ist browserübergreifend programmiert und bietet jedem Webbrowser eine ansprechende Benutzeroberfläche.

5.1.2 Leistungsanforderungen

Mengen / Raten

Die Anzahl der Benutzer wird auf rund 300 pro Tag geschätzt. Die Anzahl der Anfragen dürfte somit technisch ohne weiteres zu bewältigen sein.

[4] http://www.ifi.unizh.ch/fois

5.1.3 Qualitätsanforderungen

Funktionalität

Die unterschiedlichen Funktionen werden dem Benutzer einfach zugänglich gemacht. Um eine bessere Übersichtlichkeit zu gewährleisten, werden verwandte Funktionen gruppiert.

Zuverlässigkeit

Für den Betrieb in Verbindung mit Olat[5] ist die Zuverlässigkeit von Olat abhängig, da für die Authentifizierung dessen Variablen verwendet werden. Im eigenständigen Betrieb wird ein 24 Stunden pro Tag und sieben Tage die Woche Betrieb angestrebt, damit die Plattform den Studierenden jederzeit zur Verfügung steht.

Benutzbarkeit

Den Benutzern wird eine einfache zentrale Navigation geboten. Diese ist intuitiv benutzbar und bildet den in Abschnitt 5.1.1 erläuterten CasIS Prozess ab.

Änderbarkeit

Eine gute Modularisierung und Strukturierung wird angestrebt. Dies ermöglicht eine einfache Änderung der Komponenten. Farben werden beispielsweise in einer separaten CSS Datei abgelegt.

5.1.4 Rahmenbedingungen

Technisch

- Das Portal bietet zwei unterschiedliche Betriebsmöglichkeiten. Einerseits in Verbindung mit Olat und andererseits auch ohne Olat als eigenständige Applikation.
- Zum Einsatz kommen Semantic Web als auch Web 2.0 Technologien.

Kulturell

Das CasIS Portal wird aufgrund unterschiedlich sprachlicher Benutzer in der englischen Sprache implementiert. Eine Mehrsprachigkeit ist derzeit noch nicht vorgesehen, eine leichte Erweiterbarkeit mit anderen Sprachen ist erwünscht.

[5] Online Learning and Training (Olat) ist die Lernplattform der Universität Zürich

6

Verwandte Arbeiten

In diesem Abschnitt sollen bereits bestehende Arbeiten im Bereich von semantischen Portalen sowie im Bereich von semantischen e-Learningsystemen vorgestellt werden. Zusätzlich wird ein Überblick über die Annotierungsmöglichkeiten geboten. Annotierungsmethoden werden in CasIS benötigt, um die einzelnen Lerninhalte, welche beispielsweise in HTML oder XML vorliegen, entsprechend annotieren zu können.

6.1 Grundlagen der Webprogrammierung

In der heutigen Webprogrammierung werden sehr viele grössere Aufgaben mit Webframeworks erledigt. Ein Webframework unterstützt den Entwickler bei der Programmierung von üblichen Szenarios, wie sie bei der Webprogrammierung häufig vorkommen. Beispielsweise kann ein Webframework die Speicherung der Daten in einer Datenbank erleichtern oder das Erstellen der Webseiten für den Browser vereinfachen und beschleunigen. Zusätzlich bindet das Framework den Benutzer an eine bestimmte Vorgehensweise bei der Programmierung, was eine bessere Wartbarkeit und Lesbarkeit bewirkt, da sich alle Entwickler an diese Vorgehensweise halten müssen. Ein weiterer Vorteil liegt in der Wiederverwendbarkeit von Programmteilen und in einer besseren Skalierbarkeit.
Derzeit werden Webapplikationen hauptsächlich nach dem Model View Controller Konzept (MVC) entwickelt. Dieses teilt die Webapplikation in drei verschiedene Bereiche auf, welche hier kurz beschrieben werden.

- Die Datenschicht (Model) beinhaltet die Daten und deren Manipulationen. Wie diese Daten angeboten werden wird nicht genauer spezifiziert. Oft werden Datenbanken verwendet. Das Model ist somit für die Datenlieferung zuständig und ist unabhängig von den anderen beiden Bereichen.
- Die Steuerungsschicht (Controller) übernimmt die Steuerung und delegiert die einzelnen Aufgaben. Er wird als zusätzliche Komponente zwischen die Datenschicht und die Präsentationsschicht eingebunden.
- Die Präsentationsschicht (View) ist verantwortlich, die erhaltenen Daten für den Benutzer darzustellen. In einer Webanwendung werden darunter die unterschiedlichen Outputformate wie beispielsweise (X)HTML, XML oder PDF zusammengefasst.

Die Separierung in unterschiedliche Bereiche hat unter anderem folgende wichtigen Vorteile:

- Separierung von unterschiedlichen Bereichen ermöglicht eine bessere Austauschbarkeit der einzelnen Bereiche. Eine Veränderung der Model-Schicht wird somit möglich, ohne die View-Schicht komplett ändern zu müssen.

- Bessere Wiederverwendbarkeit, da der Programmcode zur Datenmanipulation unabhängig von der View-Schicht weiter verwendet werden kann.

- Bessere Arbeitsteilung, da die unterschiedlichen Aufgaben an die entsprechenden Personen delegiert werden können. Ein Datenbankprogrammierer muss sich demnach nicht um die Präsentation kümmern, sondern stellt lediglich die Daten zur Verfügung. Der Programmierer, welcher zuständig ist für die Gestaltung, kann sich dieser Daten bedienen und die Präsentationsschicht entsprechend zusammenstellen.

Das Model View Controller Prinzip ist sehr weit verbreitet und es gibt sehr viele Implementierungen in den unterschiedlichsten Programmiersprachen. Beispiele für Model View Controller Frameworks in der Webentwicklung in der Programmiersprache Java sind Struts1/2[1], Stripes[2], WebWork[3], Spring MVC[4] oder Grails[5].
Insbesondere in der Präsentation (View) hat sich neben dem reinen Model View Controller Prinzip eine weitere Entwicklung gezeigt. Dabei wird die Präsentationsschicht nicht einfach als HTML ausgegeben, sondern es werden Komponenten gebildet. Diese haben eine ähnliche Eigenschaft, wie sie bereits aus üblichen Desktopkomponenten bekannt sind. Ein sehr grosser Vorteil der Komponentenentwicklung in diesem Bereich liegt darin, dass die Komponenten standardisiert sind. Dies ermöglicht eine einfache Wiederverwendbarkeit und Austauschbarkeit. Ein sehr grosser Vertreter ist der Java Server Faces[6] Standard. Eine entsprechende Open Source Referenzimplementierung wird von der Apache Software Foundation[7] bereitgestellt und heisst MyFaces[8].
Auf eine genauere Analyse des Bereiches Java Webentwicklung soll an dieser Stelle verzichtet werden, da dies nicht Bestandteil dieser Arbeit ist. Für die Entwicklung des CasIS Portals wird das Struts2 Framework verwendet. Es erfüllt alle gestellten Anforderungen an das Portal:

- Struts2 implementiert den Model View Controller Standard

- Struts2 offeriert eine einfache Internationalisierungsmöglichkeit

- Struts2 stellt integrierte Ajaxkomponenten zur Verfügung

- stellt eine sehr gute Dokumentation bereit

- unterstützt unterschiedliche Präsentationsmöglichkeiten

Es gilt zu beachten, dass diese Funktionen nicht exklusiv von Struts2 bereitgestellt werden. Das Spring MVC stellt ähnliche Funktionen zur Verfügung und könnte ebenfalls verwendet werden.

[1] http://struts.apache.org
[2] http://mc4j.org/confluence/display/stripes/Home
[3] http://www.opensymphony.com/webwork
[4] http://www.springframework.org
[5] http://grails.codehaus.org
[6] http://java.sun.com/javaee/javaserverfaces/
[7] http://www.apache.org
[8] http://myfaces.apache.org

6.2 Semantische Webanwendungen

In diesem Abschnitt sollen spezielle semantische Webanwendungen vorgestellt werden. Sie bieten einen kurzen Einblick, welche Vorteile und Nachteile semantische Applikationen gegenüber herkömmlichen Applikationen aufweisen.

6.2.1 SEAL

Ein vollständiges semantisches Portal wurde im SEAL (SEmantic PortAL) Projekt [Stojanovic et al., 2001] entwickelt. SEAL beschreibt ein Framework für die Entwicklung von ontologiebasierenden Portalsystemen und basiert auf dem Ontobroker System.

6.2.2 Ontoweaver

OntoWeaver[9] ist ein Framework zur Erstellung von Webseiten. Zum Funktionsumfang gehören Ontologien zur Seitengestaltung und Navigation, eine Präsentationsontologie, ein Framework zur genaueren Anpassung sowie Werkzeuge für die Ontologie- und Seitenentwickler.

6.2.3 SWED

Das Semantic Web Environmental Directory (SWED)[10] listet Informationen zu Unternehmen und deren Projekten auf. Interessant ist dabei insbesondere, dass die Informationen über die Unternehmen nicht in SWED selbst, sondern bei den beteiligten Unternehmen aktuell gehalten werden. SWED dient somit als Aggregator der einzelnen Daten.

6.2.4 ODESeW

Das Semantic Web Portal based on WebODE (ODESeW) [Corcho et al., 2006] basiert auf dem WebODE [Arpirez et al., 2001] Framework. ODESeW basiert auf einem Model View Controller Ansatz und offeriert zusätzlich ein Model zur Definition der Navigation.

6.2.5 Ontoview

[Mäkelä et al.,] beschreibt Ontoview als Framework zur Erstellung eines semantischen Portals. Es basiert auf dem Cocoon Framework[11]. Ontoview hat eine Suchmaschine sowie ein Link - Empfehlungssystem.

6.3 Semantische e-Learning Webanwendungen

6.3.1 OntAWare

OntAWare [Holohan et al., 2005] ist eine Software, welche das Autoring, Management und die Auslieferung von Lerninhalten unterstützt. Dabei werden die einzelnen Inhalte mit Hilfe von

[9] http://kmi.open.ac.uk/projects/akt/ontoweaver
[10] http://www.swed.co.uk/swed/about/sweed_approach.htm
[11] http://cocoon.apache.org

Ontologien beschrieben. Diese Beschreibungen ermöglichen es, die unterschiedlichen Kursinhalte dynamisch zusammenzustellen und zu verwenden. Der Administrator hat somit die Möglichkeit, die einzelnen Kursinhalte dynamisch zusammenzustellen. Dies führt automatisch zu einer hohen Adaptivität.
OntAWare teilt die e-Learning Einheiten in drei Bereiche: Kurse, Lektionen und Tests. Innerhalb eines Kurses können beliebige Lektionen und Tests eingetragen werden. Dies erlaubt den verantwortlichen Lehrern bei der Zusammenstellung der Lerninhalte viel Freiheit.

6.3.2 OntoAIMS

OntoAIMS [Aroyo et al.,] ist ein Framework zur Verarbeitung von elektronischen Lernobjekten und baut auf den Definitionen von AIMS [Aroyo et al.,] auf. Es unterstützt den Autor bei der Bearbeitung und Bereitstellung der Lernobjekte.

6.4 Annotation von Textdokumenten

Für die Annotation von Webseiten kennt das Semantic Web grundsätzlich drei unterschiedliche Annotierungsmöglichkeiten:

- **Manuelle Annotation:** Die manuelle Annotation erfolgt oft über ein Programm, das die zu annotierende Textseite in einen Browser lädt und diese Daten schlussendlich durch Selektieren der entsprechenden Textstelle annotiert werden können. Das Programm offeriert dazu die entsprechenden Ontologieelemente zur Auswahl. Als Beispiel findet sich in Abbildung 6.1 das SMORE [SMORE, 2005] Annotierungswerkzeug.

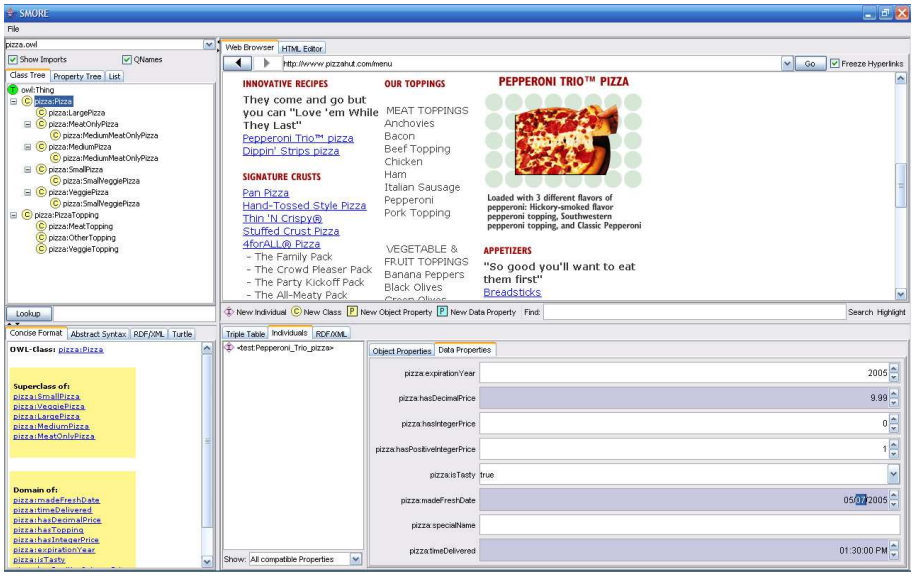

Abbildung 6.1: SMORE Annotierungswerkzeug [SMORE, 2005]

Im linken oberen Bereich befindet sich die Ontologie mit den entsprechenden Klassen zur Auswahl. Rechts oben wird das Textdokument mit dem zu annotierenden Text angezeigt und rechts unten bietet das Programm die Möglichkeit, die entsprechenden Annotationen

vorzunehmen.
Das Positive an der manuellen Annotation ist die einfache Anwendbarkeit. Der Benutzer wählt die entsprechende Textstelle aus und ordnet diese manuell einem Ontologieelement zu. Heute gibt es bereits mehrere Werkzeuge, welche diese Art von Annotation unterstützen [Reeve and Han, 2005]. Dazu gehört neben SMORE beispielsweise: Semantic Word [Tallis, 2003], Annotea [W3CAnnotea, 2001], Annozilla [Annozilla, 2007] (Mozilla Implementierung von Annotea) sowie Melita [Melita, 2005]. Weitere Tools sind beispielsweise in [Ontoweb, 2002] oder [Reeve and Han, 2005] beschrieben. Eine Auflistung von diversen Annotierungswerkzeugen findet sich in [AI, 2007].
Obwohl das manuelle Annotieren am einfachsten durchgeführt werden kann, hat es auch verschiedene Gefahren [Reeve and Han, 2005]:

  - Das manuelle Annotieren ist aufwendig und zeitintensiv.
  - Bei dynamischen Webseiten wird das Annotieren erheblich schwieriger.
  - Die Person, die die Annotation vornimmt, muss mit der Ontologie vertraut sein, um unnötige Fehler zu vermeiden.

- **Semi-automatische Annotation:** Aufgrund der obigen Gefahren wurde die semi-automatische Annotation vorangetrieben. Dabei geht es darum, den Zeitaufwand sowie Fehler zu reduzieren. Zusätzlich sollen wiederkehrende Arbeiten möglichst maschinell verarbeitet werden. Diese semi - automatischen Methoden basieren auf der Verwendung von Wrappern und auf der Analyse von natürlichsprachigem Text [Reif, 2006]. Mit Hilfe der Textanalyse soll es möglich werden, das Annotieren ohne menschliche Hilfe vollziehen zu können. Trotzdem ist es bis heute nicht möglich eine vollständige und vollautomatische Annotation vorzunehmen. Dies ist deshalb der Fall, weil Ontologiedefinitionen sehr komplex werden können und eine vollständige Zuordnung nicht vollautomatisch stattfinden kann. Oft wird noch ein zusätzliches Verbindungselement zwischen dem Dokument und den verwendeten Ontologien benötigt. Beispiele für semi-automatische Annotationswerkzeuge sind S-CREAM / Ont-O-Mat [Handschuh et al., 2002], MnM [MnM, 2004] oder OntoAnnotate [Staab et al., 2001].

- **Vollautomatische Annotation:** Als letzter Schritt wäre die vollautomatische Annotation wünschenswert. Leider wird es kaum möglich sein, den Menschen komplett aus dem Annotierungsprozess auszuschliessen. Dies liegt daran, dass der Zusammenhang zwischen den (Text-) Daten und den Ontologien für den Computer kaum zu interpretieren ist.

Weitere Informationen zur Annotation von unstrukturierten Dokumenten finden sich in [Ontoweb, 2002] sowie [Reeve and Han, 2005].

6.5 Annotation von XML Dokumenten

6.5.1 Grundlagen

In Abschnitt 6.4 wurde das Annotieren von unstrukturierten Dokumenten erläutert. Dies geschieht oft über die manuelle Auswahl einzelner Elemente und die Zuordnung zu einem Ontologieelement. Dies bietet Vor- als auch Nachteile. Insbesondere der Aufwand und die damit verbundenen Kosten bewirken, dass das Annotieren oft vernachlässigt oder ganz weggelassen wird.

Ein anderer Ansatz ist das Annotieren von XML-Dokumenten. Das XML-Format ist unterdessen weit verbreitet und wird oft verwendet, um strukturierte Daten darzustellen und diese auszutauschen. Wie bereits in Abschnitt 3.3.2 dargestellt, folgt ein XML-Dokument einer Wohlgeformtheit. Zusätzlich zur Wohlgeformtheit ist es möglich mit XML-Schema, die zu verwendenden Elemente festzulegen und das XML-Dokument gegen diese Definition zu validieren und auf Korrektheit zu überprüfen. Die vorgegebene Struktur und die vorgegebenen Elemente aus dem XML-Schema haben für das Annotieren einen grossen Vorteil. Die Möglichkeit, diese Elemente direkt mit einer Ontologie zu verbinden.

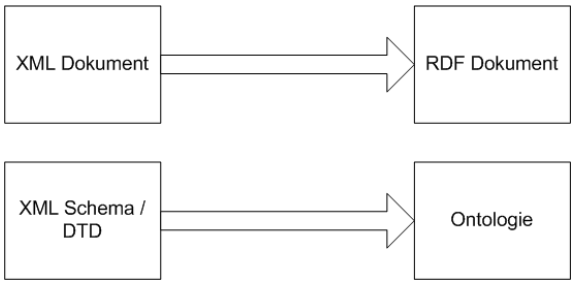

Abbildung 6.2: Verbindung zwischen XML und OWL

Das grundlegende Prinzip zeigt Abbildung 6.2. Das XML-Schema Dokument enthält Informationen, wie das XML-Dokument aussehen muss, damit dieses gültig ist. Mit Hilfe dieser Definition ist es möglich, die einzelnen Elemente aus einem XML Dokument direkt einem Element aus der Ontologie zuzuordnen. Dabei kann es sich beispielsweise um eine Klasse aus der Ontologie handeln.
Der grosse Vorteil dieser Transformationsart liegt darin, dass eine einmal definierte Verbindung immer wieder verwendet werden kann und zwar für jedes beliebige XML Dokument, das gegen dieses XML Schema validiert. Der Inhalt des Dokuments wird direkt in ein entsprechendes RDF Dokument umgewandelt und das XML-Schema zu einer Ontologie transformiert.
Diese Transformation kann auf zwei unterschiedliche Arten erfolgen:

- mit Hilfe von **XSLT**: XSLT bietet die Möglichkeit ein beliebiges XML-Dokument in ein anderes Format wie XHTML, HTML, XML oder RDF bzw. OWL umzuwandeln. Es bestehen bereits mehrere Prozessoren wie beispielsweise Saxon[12] (Java, .NET) oder Xalan[13] (C++, Java), um diese Transformation durchführen zu können. In Abbildung 6.3 wird das genauere Vorgehen erläutert. Der Entwickler hat ein bereits vorhandenes XML-Dokument mit entsprechenden Daten, die transformiert werden sollen. Aufgrund der Definitionen im XSLT werden die einzelnen Elemente aus dem XML-Dokument ausgewählt und in ein neues Dokument mit neuer Formatierung überführt.

- mit einem **eigenen Programm**: dieses liest die XML-Daten aus dem Dokument und schreibt einen neuen Text in einem anderen Format oder wieder in XML.

Beide Varianten haben ihre Vor- und Nachteile. XSLT ist ein etablierter Standard und dürfte sich für fast alle Aufgaben in diesem Bereich eignen. Der grosse Vorteil liegt darin, dass für die Transformation lediglich eine entsprechende Transformationsanweisung vorhanden sein muss. Wird eine andere Transformation benötigt kann einfach diese Anweisung ausgewechselt werden, der

[12] http://saxon.sourceforge.net
[13] http://xalan.apache.org

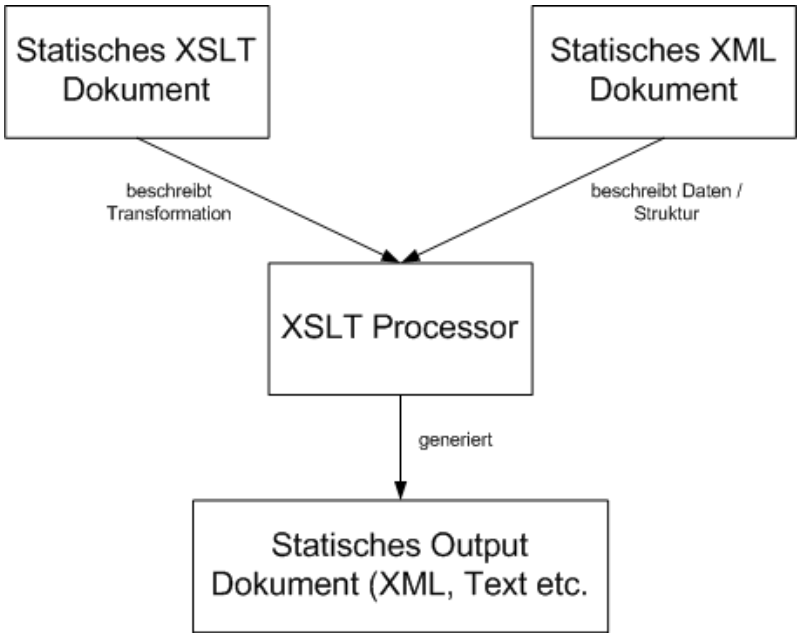

Abbildung 6.3: XSLT Transformation

Prozessor bleibt unberührt. Ein Nachteil von XSLT ist die Beschränkung auf XML. Ausgangspunkt einer Transformation ist immer ein XML-Dokument. Dieses kann mit Hilfe von XSLT in andere Formate wie HTML oder Text transformiert werden. Als weiterer Nachteil von XSLT könnte auch die Komplexität angesehen werden. XSLT ist sehr mächtig und offeriert zahlreiche Funktionen um die Verarbeitung von XML zu ermöglichen. Dies kann bei grösseren und vor allem komplexeren Dokumenten schnell kompliziert werden.
Die eigene Implementierung trägt sämtliche Vor- und Nachteile die eine selbst entwickelte Software mit sich bringt. Zusätzlich ist die Verarbeitung von grösseren XML-Dokumenten nicht ganz einfach zu bewerkstelligen und kann schnell kompliziert werden. Deshalb ist es meistens ratsamer auf XSLT zurückzugreifen.
In den Kapiteln 6.5.2 bis 6.5.4 werden einzelne Werkzeuge vorgestellt, welche die Transformationen von XML nach RDF beziehungsweise OWL durchführen.

6.5.2 WEESA

Das Framework Web Engineering for Semantic Web Applications (WEESA) [Reif, 2005] dient dazu, XML Applikationen direkt mit semantischen Metadaten ergänzen zu können. Abbildung 6.4 zeigt das vorgehen in WEESA. Im ersten Schritt wird eine Verbindung vom XML-Schema zur Ontologie definiert. Im zweiten Schritt werden die Definitionen aus dem ersten Schritt auf die XML-Dokumente angewendet, um die einzelnen RDF-Daten zu generieren. WEESA lässt sich zusätzlich in das XML Framework Cocoon[14] integrieren.

[14]http://cocoon.apache.org

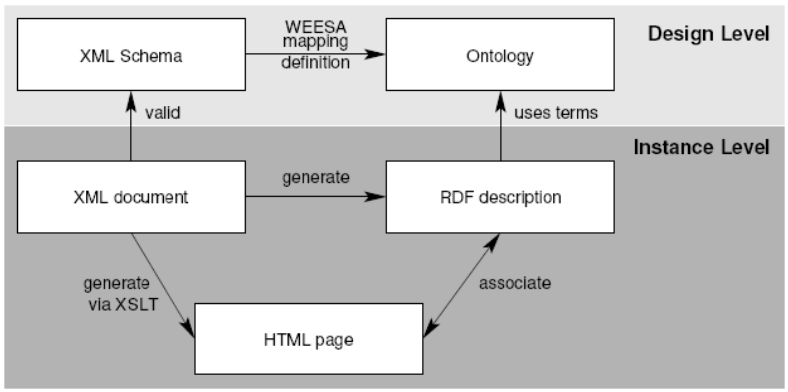

Abbildung 6.4: WEESA [Reif, 2005]

6.5.3 xml2owl

Im Gegensatz zu [Reif, 2005] definiert der Ansatz von XML2OWL [Bohring and Auer, 2005] eine fixe Verbindung zwischen den einzelnen Elementen von XML-Schema nach OWL. Beispielsweise wird das Element `xsd:complexType` direkt dem Element `owl:Class` zugeordnet. Eine Übersicht über die möglichen Verbindungen gibt Tabelle 6.1

XSD	OWL
xsd:elements, containing other elements or having at least one attribute	owl:Class, coupled with owl:ObjectProperties
xsd:elements, with neither subelements nor attributes	owl:DatatypeProperties
named xsd:complexType	owl:Class
named xsd:SimpleType	owl:DatatypeProperties
xsd:minOccurs, xsd:maxOccurs	owl:minCardinality, owl:maxCardinality
xsd:sequence, xsd:all	owl:intersectionOf
xsd:choice	combination of owl:intersectionOf, owl:unionOf and owl:complementOf

Tabelle 6.1: Verbindung von XML nach OWL [Bohring and Auer, 2005]

Diese definierten Verbindungen werden mit Hilfe einer geeigneten XSLT Transformation vorgenommen. Es gilt zu beachten, dass bei diesem Framework weder eine Ontolgie, noch ein XML-Schema vorhanden sein muss. Das Framework kann diese bei Bedarf automatisch generieren. Interessant bei dieser Vorgehensweise ist, dass die Umwandlung vollautomatisch durchgeführt werden kann, da die Verbindungen fix definiert sind. [Rodrigues et al., 2006] sieht darin allerdings zwei grosse Nachteile. Erstens hat der Nutzer keinen Einfluss auf die Transformation und somit auch keine zusätzlichen Gestaltungsmöglichkeiten. Zweitens kann das Framework lediglich die implizite Semantik des XML-Dokumentes bearbeiten, was dazu führt, dass die generierten Ontologien kaum mehr Semantik enthalten als die XML-Dokumente selbst.

6.5.4 jxml2owl

Der Ansatz von jxml2owl [Rodrigues et al., 2006] beschreibt eine Verbindung zwischen einem XML-Dokument und einer bereits bestehenden Ontologie. Die Verbindungen werden dabei vom Benutzer manuell erstellt. Der Benutzer hat dafür vollständige Kontrolle über den Prozess und die Verbindungsmöglichkeiten. Die schlussendliche Transformation erfolgt über XSLT. Zusätzlich bietet jxml2owl einen guten Editor, welcher eine einfache Verbindung zwischen XML-Daten und den Ontologien ermöglicht.

6.5.5 XPath

Eine weitere Möglichkeit besteht darin, nicht das ganze XML-Dokument zu annotieren, sondern lediglich einzelne XML-Elemente für die Metadaten zu verwenden. Diese können schlussendlich für die Suche verwendet werden. Die Definition der einzelnen XPath Ausdrücke müssten von Hand definiert werden.

6.5.6 Manuelle Annotation

Wie bereits in Abschnitt 6.4 dargestellt, ist es auch bei XML Dokumenten möglich, eine manuelle Annotation vorzunehmen.

6.5.7 Bemerkungen

In den Kapiteln 6.5.2 bis 6.5.4 wurden diverse Möglichkeiten zur XML-Annotation vorgestellt. Tabelle 6.2 zeigt eine kurze Zusammenfassung der einzelnen Techniken. Der grösste Unterschied liegt einerseits in der Genauigkeit der Methoden und andererseits in Dokumenten, die benötigt werden, um die Transformation starten zu können. XPath und die manuelle Annotation wurden dabei nicht berücksichtigt, da XPath lediglich einzelne XML-Elemente auswählt und die manuelle Annotation nicht spezifisch für XML-Dokumente ist, sondern für jeden beliebigen Text verwendet werden kann.

	WEESA	xml2owl	jxml2owl
XML nötig	ja	ja	ja
XML-Schema nötig	ja	nein	nein
Ontologie nötig	ja	nein	ja
Transformation mit	eigenem Java Code	XSLT	XSLT
Annotationsart	manuell	automatisch	manuell
Genauigkeit	sehr genau	ungenau	sehr genau

Tabelle 6.2: Übersicht XML nach RDF/OWL

Für das CasIS Portal sind die beiden Projekte WEESA und jxml2owl interessant. Sie bieten die benötigte Genauigkeit, um die Fallstudien adäquat nach Ontologien abbilden zu können. Trotzdem ergibt sich innerhalb des Annotationsprozesses mit Hilfe von WEESA oder jxml2owl ein Problem. Die Annotierungsmethoden sind darauf ausgerichtet, XML-Elemente nach RDF zu transformieren. Die CasIS Fallstudien enthalten jedoch innerhalb der XML-Elemente zusätzlichen Text. Da dieser nicht in einem eigenen XML-Element vorhanden ist, kann dieser ganze Text nicht mehr aufgeteilt und annotiert werden.

Die Annotation wird schlussendlich manuell ausgeführt. Es werden sogenannte Metadaten erfasst, welche gleichzeitig für die Fassettensuche benutzt werden können.

6.6 Fassettennavigation

Der Begriff Fassettennavigation (Faceted Browsing) beschreibt die Möglichkeit, innerhalb von beliebigen Daten aufgrund unterschiedlicher Kriterien navigieren zu können. Die Fassetten bilden somit eine unterschiedliche Blickrichtung auf die zugrundeliegenden Daten. An einem kleinen Beispiel soll die Fassettennavigation erläutert werden:

Eine Lernressource kann auf unterschiedliche Art und Weise beschrieben werden. Einerseits kann ausgesagt werden, zu welchem Lerngebiet sie gehört (Mathematik, Sprachen etc.). Andererseits kann beschrieben werden, wer die Lernressource zur Verfügung stellt oder diese entwickelt hat. Wiederum kann eine Lernressource in Bereiche wie Kurse oder Seminare eingeteilt werden. Diese unterschiedlichen Bereiche können als Fassetten betrachtet werden, welche einen differenzierten Blick auf eine spezifische Lernressource bieten.
Fassetten offerieren im Gegensatz zu einfachen Kategorisierungen zusätzliche Möglichkeiten. In einfachen Kategorisierungen ist nur eine hierarchische Navigation innerhalb der Kategorie möglich. Kombinationen mit anderen Kategorien sind nicht möglich. Genau dieses Manko behebt die Fassettennavigation. Bei der Auswahl einer Fassette werden die anderen jeweils mit dem neuen Filterkriterium aktualisiert und angepasst. Somit kann zwischen den einzelnen Fassetten navigiert werden. Dies ermöglicht eine bessere Navigationsmöglichkeit und verhindert insbesondere leere Resultate, wie diese beispielsweise bei Suchmaschinen möglich sind.
Einige interessante Beispiele sind Longwell[15], /facet[16], SWED[17], Flamenco[18] sowie Exhibit[19]. Diese Werkzeuge werden in nachfolgenden Abschnitten vorgestellt.

6.6.1 Longwell

Zum Funktionsumfang von Longwell gehört eine Fassettennavigation sowie eine Volltextsuche aufgrund von verschiedenen RDF-Daten. Zusätzlich können die Suchergebnisse entsprechend sortiert werden. Longwell ist frei verfügbar und kann von der Webseite heruntergeladen werden. Auf der Webseite finden sich eine Dokumentation sowie eine Demonstration der Software. Ein Eindruck vermittelt Abbildung 6.5.

Eine Grundinstallation ist mit Longwell schnell vollzogen und die unterschiedlichsten Funktionen können ausprobiert werden. Trotz der umfangreichen und sehr flexiblen Funktionen erfüllt Longwell nicht unsere Anforderungen:

- Das Laden der Fassetten ist leider eher langsam.
- In Longwell gibt es keine eingeschränkte Sicht für Benutzer und Administratoren. Eine entsprechende Implementierung wäre zwar möglich, ist allerdings komplizierter als bei anderen Werkzeugen.

[15] http://simile.mit.edu/wiki/Longwell
[16] http://slashfacet.semanticweb.org
[17] http://swed.co.uk
[18] http://flamenco.berkeley.edu/
[19] http://simile.mit.edu/exhibit/

Abbildung 6.5: Longwell: semantische Fassettennavigation

- Die Beschreibung der Benutzerschnittstelle erfolgt in Fresnel[20]. Eine einfache Anpassung der einzelnen Einträge wird dadurch erschwert, da zusätzlich eine weitere Technik nötig wird.

6.6.2 /facet

Mit /facet (gesprochen „Slash Facet“) können wie mit Longwell beliebige semantische Datensätze durchsucht werden. /facet bietet neben einer Volltextsuche die Darstellung von Zeitlinien und Karten. Eine ausführliche Beschreibung findet sich in [Michiel et al., 2006].
/facet fehlt leider eine technische Dokumentation sowie die Möglichkeit, die Benutzerschnittstelle einfach anzupassen. Zusätzlich ergaben sich während der Testphase Probleme mit der Kommunikation zwischen Client und Server, welches öfters mit einer Fehlermeldung quittiert wurde („There was a problem retrieving the XML data“). Wie Longwell hat /facet ebenfalls keine unterschiedlichen Sichten für Benutzer und Administratoren.

6.6.3 SWED

Ein weiteres Projekt ist das Semantic Web Environment Direcory (SWED) [21]. Es stellt eine Fassettennavigation über unterschiedliche Unternehmen und Projekte innerhalb des Vereinigten Königsreiches bereit. Abbildung 6.6 gibt einen ersten Überblick über das Portal.

Das Portal bietet neben der Fassettennavigation eine Volltextsuche sowie die Möglichkeit Daten von anderen Webseiten zu aggregieren. Eine Betrachtung der technischen Dokumentation, die komplizierte Konfiguration und ein erster kleiner Prototyp haben gezeigt, dass die gewünschten Anforderungen mit SWED nicht zu erfüllen sind. Insbesondere die individuelle Manipulation der Fassettennavigation, sowie die unterschiedlichen Sichten für Administratoren und normale Benutzer erwies sich als zu kompliziert.

[20] http://simile.mit.edu/wiki/Fresnel
[21] http://www.swed.co.uk/swed/index.html

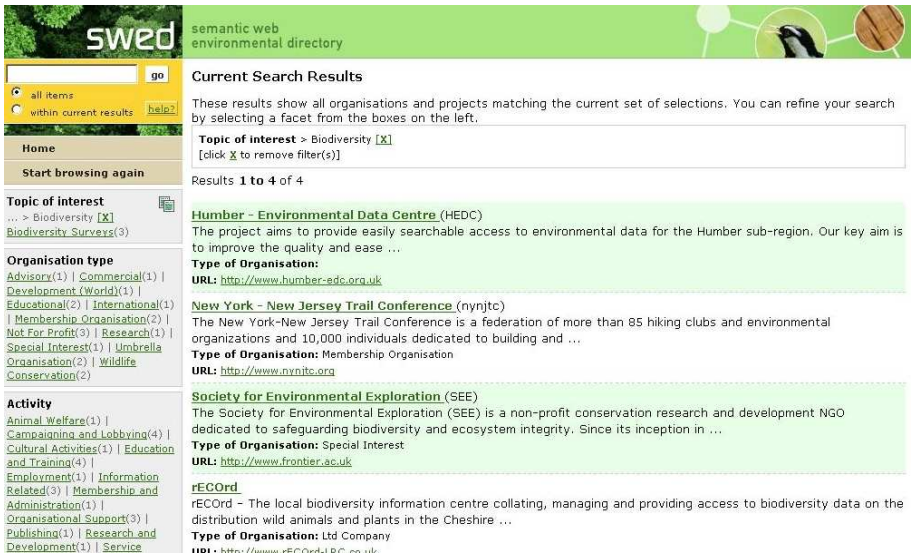

Abbildung 6.6: SWED Portal

6.6.4 Flamenco

Das Ziel von Flamenco[22] ist dem Benutzer eine möglichst einfache Navigation durch grosse Datenbestände anzubieten. Dabei wird ebenfalls das Prinzip der Fassettennavigation umgesetzt. Einzelne Elemente werden Kategorien zugeordnet, welche wiederum zu einer Fassette zusammengefasst werden. Zusätzlich offeriert Flamenco eine Volltextsuche. Flamenco ist in Python programmiert und benutzt mehrere Dateien zur Konfiguration und Verlinkung der Datenbestände. Die Volltextsuche innerhalb von Flamenco wird mit Lucene angeboten. Leider unterstützt Flamenco keine Semantik und kein AJAX. Zudem ist die Konfiguration in mehrere Dateien aufgeteilt. Dies erschwert die Erstellung der unterschiedlichen Fassetten für den Administrator und die normalen Benutzer.

6.6.5 Exhibit

Die vorher beschriebenen Anwendungen basieren alle auf Servertechnologien. Exhibit verfolgt einen anderen Weg. Es ist in Javascript und Ajax geschrieben und wird somit auf der Clientseite ausgeführt. Dies ermöglicht eine sehr schnelle Reaktionszeit und entlastet den Server. Abbildung 6.7 zeigt ein Beispiel einer Exhibit Anwendung.

Ein weiterer Vorteil von Exhibit ist die einfache Konfiguration. Eine einzige HTML Datei ist für die Darstellung und Fassettendefinitionen zuständig. Die Daten selbst werden im JSON Format gespeichert. Zusätzlich wird über eine Exportfunktion die Möglichkeit geboten, die Daten in ein anderes Format zu exportieren, zum Beispiel RDF/XML für das semantische Web.
Ein Nachteil von Exhibit ist die Tatsache, dass keine Volltextsuche innerhalb der Exhibit Daten angeboten wird.

[22] http://flamenco.berkeley.edu

Abbildung 6.7: Exhibit: Fassettennavigation

6.6.6 Bemerkungen

Für das CasIS Projekt sind alle fünf Werkzeuge sehr interessant. Erste Analysen und Vergleiche mit den vorgestellten Anforderungen bieten einen guten Überblick über die einzelnen Werkzeuge. Schlussendlich wird für das Portal Exhibit verwendet. Es hat alle gewünschten Funktionen bezüglich der Fassettennavigation und dazu eine sehr einfache Konfiguration. Ein weiterer Vorteil von Exhibit ist die Tatsache, dass Standardtechnologien wie JavaScript und Ajax verwendet werden. Auf der Webseite findet sich zusätzlich eine ausreichend technische Dokumentation.

6.7 Volltextsuche

Die einzelnen Fallstudien werden in diversen HTML- und XML-Dateien zur Verfügung gestellt. Neben der beschriebenen Facettennavigation ist es deshalb hilfreich eine zusätzliche Volltextsuche über alle vorhandenen Dokumente anzubieten. Die CasIS Fallstudien werden nicht in einer Datenbank gespeichert und somit können einfache Suchmechanismen wie zum Beispiel der SQL Befehl `LIKE` nicht verwendet werden. Aus diesem Grund wurde die Suche auf gängige Volltextsuchprogramme ausgeweitet. Diese werden in den folgenden Abschnitten evaluiert.

6.7.1 Lucene

Lucene[Foundation, 2007] ist eine von der Apache Software Foundation zur Verfügung gestellte Java Bibliothek, welche Texte indizieren kann. Abbildung 6.8 offeriert einen guten Überblick.

Auf der linken Seite findet sich die Indexierung diverser Textdokumente. Lucene selbst hat leider keine Möglichkeit, aus unterschiedlichen Textdokumenten den Inhalt auszulesen. Dies muss mit Hilfe der Applikation eigenhändig erledigt werden. Glücklicherweise gibt es bereits zahlreiche Bibliotheken, welche das Extrahieren von Textinhalten aus den unterschiedlichsten

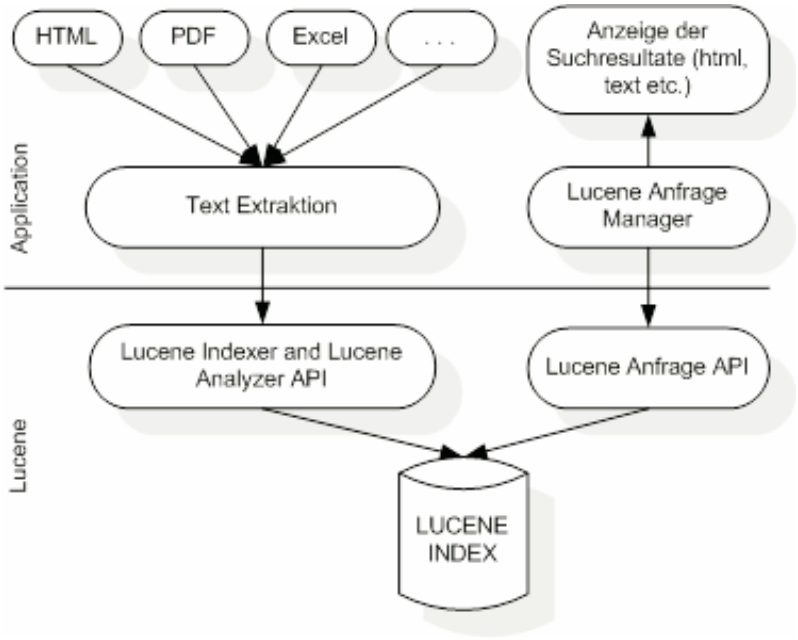

Abbildung 6.8: Lucene Volltextsuche

Dokumenttypen ermöglichen. Ein sehr gutes Beispiel ist Apache POI[23], welches Beispielsweise Textextraktion aus den zahlreichen Microsoft Formaten ermöglicht oder PDFBox[24] zur Textextraktion von PDF Dokumenten. Diese werden mit Hilfe der Lucene Schnittstellen in den Lucene Index eingelesen. Auf der rechten Seite erfolgt die Abfrage über die Abfrageschnittstelle. Dabei stehen diverse Suchalgorithmen zur Verfügung. Lucene bietet dem Nutzer folgende Möglichkeiten[Foundation, 2007]:

- Skalierbarkeit
- performante Indexierung (inkrementelle als auch volle Indexierung)
- effiziente Suchalgorithmen, die beispielsweise Rankierungen von Suchresultaten oder Sortiermöglichkeiten unterstützen

Lucene ist ein Open Source Projekt und somit frei verfügbar. Aufgrund seiner Beliebtheit und Möglichkeiten wurde Lucene auch in die meisten anderen Programmiersprachen portiert. Lucene wird auch in zahlreichen anderen Projekten benutzt, wenn es darum geht, eine Volltextsuche anzubieten. Beispiele sind Nutch[25] oder Solr[26]. Eine ausführliche Behandlung von Lucene findet sich im Buch „Lucene in Action“ [Erik Hatcher, 2004].

[23] http://poi.apache.org
[24] http://www.pdfbox.org
[25] http://lucene.apache.org/nutch
[26] http://lucene.apache.org/solr

6.7.2 Egothor

Egothor[27] ist wie Lucene eine Volltextsuche, welche in Java implementiert ist. Im Gegensatz zu Lucene hat Egothor eine Extraktionsmöglichkeit von unterschiedlichen Dokumentformaten wie PDF, Microsoft Word oder Excel. Eine gute Dokumentation als auch ein Mail-Archiv ergänzen das Angebot.

6.7.3 JSE

Die Java Search Engine[28] (JSE) bietet umfangreiche Funktionen im Bereich der Volltextsuche. Unter anderem Indexierung von Microsoft Word, PDF, HTML oder Textdatein, inkrementelle sowie eine volle Indexierung und enthält einen kompletten Webcrawler[29] zur Textextraktion aus dem Internet. Leider ist die Java Search Engine nur zu Evaluationszwecken frei zugänglich. Für eine andere Verwendung wird eine Gebühr von 300$ erhoben.

6.7.4 Regain

Regain[30] ist eine weitere sehr umfangreiche Volltextsuche. Regain basiert auf Lucene und deckt somit alle Eigenschaften von Lucene ab. Zusätzlich hat es die Möglichkeit, Verzeichnisse nach Dateien zu durchsuchen und diese zu indexieren. Dabei werden unter anderem die Dateiformate Html, Pdf, Microsoft Word, Excel und Powerpoint unterstützt. Regain offeriert eine gute Dokumentation[31], ein eigenes Forum[32] sowie einen Newsletter.

6.7.5 Lius

Lucene Index Update and Search (LIUS)[33] basiert wie Regain auf Lucene und erweitert Lucene um zusätzliche Dokumente, welche indexiert werden können. Darunter fallen Formate wie PDF, Microsoft Word, Excel und Powerpoint sowie Zip Files, MP3 oder JavaBeans. Eine kleine Dokumentation befindet sich auf der Webseite des Entwicklers[34]. Die Entwicklung wird nicht weitergeführt.

6.7.6 Solr

Solr[35] baut ebenfalls auf Lucene auf. Zum Angebot gehört eine Fassettensuche, Hervorhebung von Suchresultaten, Replikation sowie ein Administrationsbereich. Die Indexierung erfolgt über XML via Http, genauso wie bei der Suchanfrage. Solr wird als eigenständiger Server betrieben

[27] http://www.egothor.org
[28] http://www.me.lv/jse/index.html
[29] Ein Webcrawler bietet die Möglichkeit, anhand der Verlinkungen in (X)Html Seiten, weitere Webseiten aufzurufen und diese ebenfalls zu indexieren.
[30] http://regain.sourceforge.net
[31] http://regain.murfman.de/wiki/en/index.php
[32] http://forum.murfman.de/en/viewforum.php?f=3
[33] http://sourceforge.net/projects/lius
[34] http://www.doculibre.com/lius/doc-1.0_en.html
[35] http://lucene.apache.org/solr

6.7.7 Nutch

Ein weiteres Projekt innerhalb von Lucene ist Nutch[36]. Es baut ebenfalls auf Lucene auf. Nutch bietet einen eigenen Webcrawler, Indexierung unterschiedlicher Dokumentformate, eine einfache Benutzerschnittstelle sowie die Hervorhebung von Suchresultaten.

6.7.8 Bemerkungen

In diesem Abschnitt wurden zahlreiche Möglichkeiten aufgeführt, um eine Volltextsuche realisieren zu können. Die einzelnen Fallstudien im CasIS Portal sind hauptsächlich im (X)HTML Format vorhanden. Deshalb ist es wichtig, dass dieser Dokumenttyp von der Volltextsuche unterstützt wird oder einfach implementiert werden kann. Für die Realisation wird Lucene verwendet. Lucene hat folgende Vorteile:

- Die einfache Schnittstelle führt zu einer einfachen Verwendbarkeit und Erweiterbarkeit.
- Eine gute Dokumentation sowie viele Benutzer ermöglichen eine zeitsparende Implementation der Volltextsuche.

6.8 Diskussion

In diesem Kapitel wurden die verwandten Arbeiten bezüglich des CasIS Portals erläutert. Die analysierten Frameworks und Implementierungen trafen nicht ganz die Anforderungen an das CasIS Portal. Insbesondere in folgenden Bereichen bietet kein Framework direkte Unterstützung:

- **Adaptivität:** Kein Framework offeriert eine Möglichkeit zur dynamischen Anpassung des Portals, wie beispielsweise das verschieben einzelner Komponenten. Diese Funktionalität muss entsprechend von Hand implementiert werden.
- **Prozessabbildung:** Kein Framework gibt eine Möglichkeit zur Abbildung des CasIS Prozesses. Dieser einfache Prozess kann auch von Hand implementiert werden.
- **Annotation:** Eine Analyse unserer XML-Dokumente hat ergeben, dass innerhalb der XML-Elemente eher langer Text vorhanden ist. Die in Abschnitt 6.5 vorgestellten Werkzeuge benutzen jedoch das ganze Element und somit den ganzen Text des XML-Elementes. Eine einfache Annotation und Eingliederung in die Fassettennavigation ist somit kaum möglich. Deshalb werden für die Fassettennavigation die benötigten Daten manuell erfasst.

Die anderen Anforderungen an das CasIS Portal können mit den vorgestellten Werkzeugen realisiert werden. Die folgende Liste zeigt einen kurzen Überblick, mit welchen Techniken die Implementierung des CasIS Portals erfolgt:

- Das Struts2 Framework stellt die Grundfunktionalitäten bezüglich der Webentwicklung zur Verfügung und unterstützt den Model View Controller Standard.
- Für die Präsentationsschicht wird Velocity[37] verwendet. Velocity besticht durch seine hohe Geschwindigkeit und Einfachheit in der Programmierung.

[36] http://lucene.apache.org/nutch
[37] http://velocity.apache.org

- Mit Exhibit wird die Fassettennavigation für die Studenten und die Administratoren bereitgestellt. Exhibit hat zahlreiche Funktionen, welche es erlauben, die einzelnen Daten zu gruppieren oder zu sortieren und offeriert eine leistungsfähige Fassettennavigation.
- Lucene stellt die Volltextsuche für die HTML Dateien zur Verfügung. Es offeriert eine einfache Schnittstelle für die Indexierung als auch für die Abfragen.

7

Implementation

In diesem Kapitel wird die Implementation des CasIS-Portals aufgezeigt. Im ersten Teil wird hauptsächlich auf die Benutzerschnittstelle eingegangen. Dies ermöglicht dem Leser einen ersten Eindruck des Portals. Der zweite Teil befasst sich mit der technischen Implementierung und bietet einerseits einen Überblick über die CasIS Software Architektur als auch eine detaillierte Beschreibung zu den einzelnen Komponenten.

7.1 Benutzerschnittstelle

Einen ersten Eindruck bietet Abbildung 7.1.

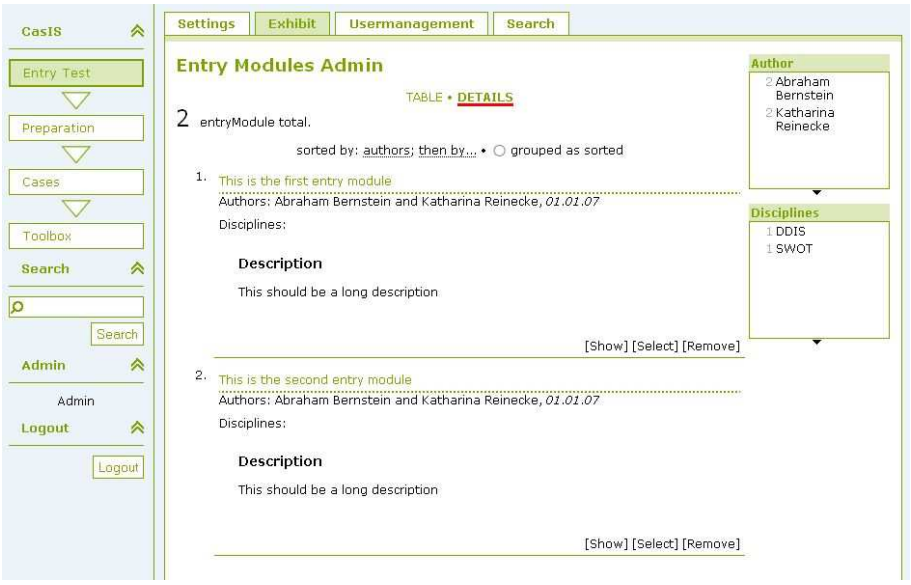

Abbildung 7.1: CasIS Portal

Sie zeigt das Portal im Administratormodus bei der Auswahl der Lernressourcen. Derzeit auf der linken Seite befindet sich der CasIS Prozess mit seinen vier Modulen. Darunter erhält der

Benutzer Zugang zur Volltextsuche. Ist der angemeldete Benutzer ein Administrator, wird ihm zusätzlich ein Link zum Administrationsmenu angeboten. Zum Abmelden vom Portal erhält der Benutzer einen entsprechenden Link am Ende des Navigationsmenus.

7.1.1 Volltextsuche

Jeder Benutzer erhält Zugriff auf die Volltextsuche. Er hat die Möglichkeit einen Suchbegriff einzugeben. Lucene unterstützt dabei zahlreiche Anfragemöglichkeiten. Die häufig benötigten Möglichkeiten werden in Tabelle 7.1 dargestellt. Eine ausführliche Beschreibung der verschiedenen Anfragemöglichkeiten findet sich in [Erik Hatcher, 2004]. Nach einer erfolgreichen Anfrage

Operator	Erklärung	Beispiel
AND	beide verknüpften Begriffe müssen im Dokument vorhanden sein	casis AND facet
OR	einer der beiden verknüpften Begriffe muss im Dokument vorhanden sein	casis OR facet
NOT	der Begriff nach NOT darf nicht im Dokument vorkommen	casis NOT facet
Gruppierung	mit Hilfe von Klammern können Anfragen gruppiert werden	(casis AND facet) NOT exhibit

Tabelle 7.1: Lucene Anfragemöglichkeiten

werden die Resultate am Bildschirm angezeit. Dabei wird das Resultat mit einem kleinen Text versehen und das gesuchte Wort hervorgehoben. Zusätzlich wird gekennzeichnet, wie relevant ein Resultat ist. Bei mehr als zehn Suchresultaten wird automatisch ein Link zur Verfügung gestellt, welcher auf die nächsten Resultate verweist. Ein direkter Link bei einem Suchresultat gestattet dem Anwender, die Lernressource direkt zu öffnen und sich den Inhalt anzeigen zu lassen, sofern er die entsprechenden Rechte besitzt. Abbildung 7.2 zeigt ein einzelnes Suchresultat, sowie den hervorgehobenen Suchbegriff und die Relevanz des Treffers. Ein direkter Link führt zur Lernressource.

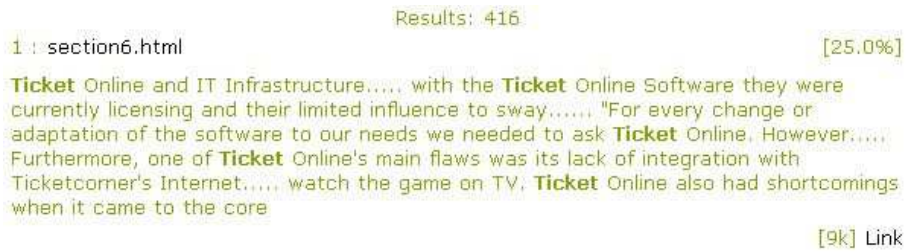

Abbildung 7.2: CasIS Volltextsuche

Das CasIS Portal erlaubt es dem Benutzer, sämtliche HTML-Dateien zu durchsuchen. Die Suche erfolgt aufgrund des Textes im Dokument. Zusätzlich zum eigentlichen Inhalt werden weitere Werte indexiert. Darunter fällt der Dokumentenpfad auf dem Server, der Dateiname, das letzte Änderungsdatum sowie die Grösse der Datei. Es können ohne grosse Probleme weitere Elemente indexiert werden, falls dies erwünscht ist.

Die Suchabfrage erfolgt aufgrund des Inhaltes der Datei. Dies bedeutet, dass bei einer Suchanfrage lediglich der Inhalt der Datei durchsucht wird. Andere indexierte Elemente werden derzeit nicht berücksichtigt.

7.1.2 Fassettennavigation

Neben der Volltextsuche gestattet das CasIS Portal auch eine Suche mit Hilfe von Fassetten. In der am Anfang gezeigten Abbildung (Abbildung 7.1) befinden sich diese auf der rechten Seite. Das Konzept der Fassettennavigation wurde bereits in Abschnitt 6.6 beschrieben. Der Benutzer hat die Möglichkeit, einzelne Fassetten zu selektieren und somit die Sicht auf die darunter liegenden Daten einzuschränken. Es werden nur noch Daten angezeigt, die den Fassetten genügen. Die anderen Daten werden entsprechend gefiltert. Exhibit erlaubt Einfach- und Mehrfachselektionen innerhalb einer Fassette. Die Einfachselektion wird beim Anklicken des Namens des Elements aktiviert. Die Mehrfachselektion erfolgt bei der Selektion der Kontrollkästchen auf der rechten Seite. Abbildung 7.3 zeigt diesen Sachverhalt mit zwei selektierten Elementen innerhalb derselben Fassette.

Abbildung 7.3: CasIS Exhibit Facet

Neben der Fassettennavigation bietet Exhibit dem Nutzer zusätzlich Gruppierungs- und Sortierungsfunktionen an, sowohl für die Tabellenansicht, als auch für die Detailansicht. Dem Anwender werden somit zahlreiche Möglichkeiten zur Verfügung gestellt, Lernressourcen möglichst einfach zu finden.

7.1.3 Administration

Die Administration ist nur für Benutzer der Gruppe Administratoren zugänglich. Sie bietet folgende Möglichkeiten:

- Einstellungen
  - **Applikationseinstellungen:** umstellen zwischen dem Betrieb mit Olat und dem Betrieb ohne Olat. Dies beeinflusst die Möglichkeiten, wie sich die Benutzer am System anmelden können. Im Olat Modus liefert Olat die nötigen Variablen, im Betrieb ohne Olat muss sich der Benutzer mit Benutzernamen und Passwort authentifizieren.
  - **Benutzereinstellungen:** nach der Auswahl der Benutzergruppe (es gibt zwei Benutzergruppen: „admin“ und „user“) können Einstellungen bezüglich des Portals vorgenommen werden. Der Administrator wählt beispielsweise, welche Module oder Lernressourcen für Benutzergruppen zur Verfügung stehen. Er hat auch die Möglichkeit, das Aussehen der Applikation zu verändern und beispielsweise Menus anstatt links rechts

darzustellen. Diese Anpassungen erfolgen jeweils dynamisch und per Benutzergruppe. Zusätzlich wird hier auch die Sprache definiert. Derzeit sind die Texte allerdings nur in englischer Sprache vorhanden.

- Selektion von Lernressourcen in den verschiedenen Modulen über Exhibit sowie Hochladen der neuen Lernressourcen in die einzelnen Module
- Benutzerverwaltung: Erstellen, Editieren oder Löschen von Benutzern sowie Importieren von Benutzern per Datei.
- Die Volltextsuche kann neu gestartet werden. Dies bewirkt, dass alle Dokumente gelöscht und neu eingelesen werden. Eine weitere Option ist das Löschen des Indexes. Folglich ist keine Suche mehr möglich.

Das Administrationsmenu bietet dem Administrator eine einfache Möglichkeit, Anpassungen am Portal selbst vorzunehmen.

7.1.4 Verwendete Farben

Die verwendeten Farben sind vom Projekt „Fundamentals of Information Systems“ übernommen worden. Mit der entsprechenden Erlaubnis wurde das CSS heruntergeladen und im CasIS Portal verwendet. Die nachfolgende Tabelle listet die Standardfarben auf.

Farbe	Hexadezimalwert
	#7DA300
	#ECF4F9
	#DDE7BC
	#E8EED0
	#000000
	#FFFFFF (weiss)

Tabelle 7.2: CasIS Standardfarben

Teilweise wurden die Farben für das Portal übernommen und teilweise wurden neue Farben für eine bessere Gestaltung hinzugefügt.

7.2 Softwarearchitektur

In den vorhergehenden Abschnitten wurde das CasIS Portal hauptsächlich aus Benutzersicht dargestellt. In diesem Abschnitt soll nun der technische Aspekt hinter der Benutzerschnittstelle genauer betrachtet werden. Die Abbildung 7.4 zeigt die einzelnen Komponenten des Systems.

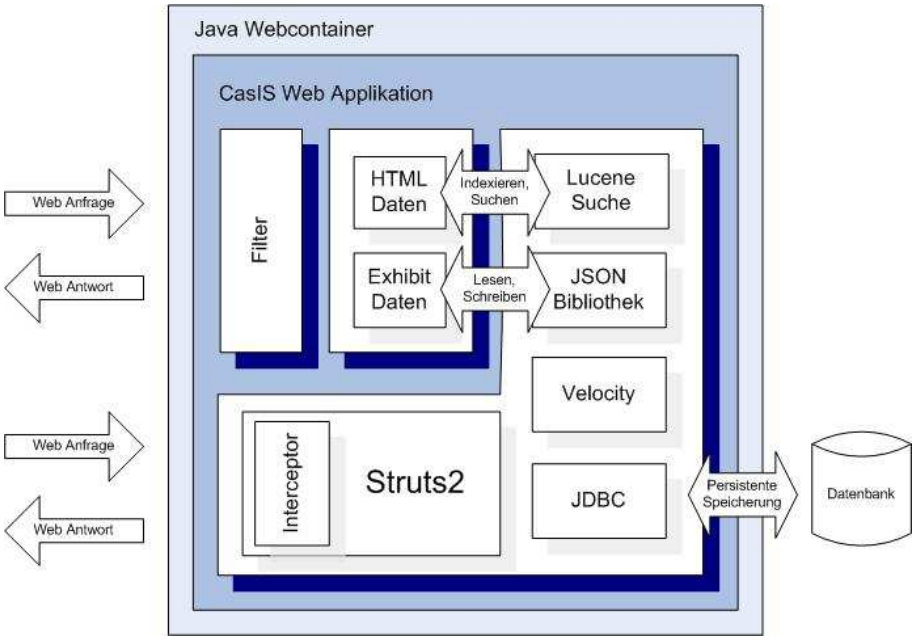

Abbildung 7.4: CasIS System Architektur

7.2.1 HTML Daten

Die HTML Daten beinhalten die einzelnen Lernressourcen. Sie sind im Verzeichnis „data“ abgelegt. Die Verzeichnisstruktur für Lernressourcen folgt dabei folgendem Muster:

- „data“ - Modulname - Lernressourcenummer - Sprache - „index.html“

Unter dem Modulnamen werden die vier Module des CasIS Prozesses verstanden. Die Exhibits zu den einzelnen Modulen befinden sich ebenfalls in diesem Verzeichnis. Die Verzeichnisstruktur ist folgendermassen:

- „data“ - Modulname - „exhibit“ - Benutzergruppe - „exhibit.html“

Die einzelnen HTML Dateien können mit Hilfe der vorangestellten Filterkomponenten geschützt werden. Dies ist notwendig, weil der Ordner von aussen frei verfügbar ist und nicht von dem CasIS Portal geschützt werden kann. Gibt ein Benutzer beispielsweise den Pfad zum Administratorenexhibit an, so hat er ohne die Filter freien Zugriff. Genau dieses Szenario sollen die Filter vermeiden. Die Implementierung erfolgt in dem Packet `casis.security.filters`. Für jedes einzelne Modul wird ein eigener Filter zur Verfügung gestellt, damit eine individuelle Behandlung ermöglicht wird. Die Filter bieten folgende Möglichkeiten:

- sperren des ganzen Moduls
- sperren einzelner Lernressourcen
- sperren des Exhibit Pfades. Das Exhibit gehört immer genau einer Benutzergruppe. Der Zugriff auf ein Exhibit einer anderen Benutzergruppe ist nicht möglich.

Die einzelnen Filter überwachen die angeforderte URL und überprüfen, ob ein nicht erlaubter Wert in dem Verzeichnispfad vorhanden ist. Ist dies der Fall, wird der Zugriff verweigert. Beispielsweise gibt es im Portal eine Datei mit dem Namen teachingNotes.html. Diese Datei darf

nur für Administratoren zugänglich sein. In der Benutzereinstellung findet sich ein Eintrag „Gesperrte Ressourcen“. Dort ist es möglich, kommagetrennte Werte einzugeben. Ein Eintrag „teachingNotes“ würde somit dazu führen, dass diese Ressource für die entsprechende Benutzergruppe nicht mehr aufrufbar ist.
Tomcat[1] bietet leider zuwenig Möglichkeiten in diesem Bereich, weshalb eine eigene Implementierung angestrebt wurde.

7.2.2 Exhibit Daten

Innerhalb des CasIS gibt es zwei unterschiedliche Exhibit's: einerseits das Administratorenexhibit, welches alle verfügbaren Lernressourcen beinhaltet und andererseits das Userexhibit, das eine ausgewählte Selektion des Administratorexhibits darstellt.

Das Datenformat

Als Datengrundlage benutzt Exhibit das JavaScript Object Notation (JSON)[2] Format. Dieses Format beschreibt einen einfachen Standard zur Speicherung von Objekten in JavaScript. Folgende Datentypen werden unterstützt: `string`, `number`, `true`, `false`, `null` sowie die komplexeren Datentypen `array` und `object`. Ein `array` kann alle anderen Datentypen enthalten, während ein `object` eine Auflistung von `string` : Datentyp ist. Ein Beispiel bietet 7.1.

```
{
    items: [
        {
          index: 1,
          label: "This is the first Cases Module",
            type: "casesModule",
            disciplines: [ "Mathematics", "Biologie" ],
            description: "This should be a long description"
    }
    ]
}
```

Listing 7.1: JSON Beispiel

.

„Items“ und „disciplines“ sind vom Datentyp `array`, während die anderen vom Datentyp `object` sind. Wie das Beispiel zeigt, ist auch das Verschachteln der Datentypen erlaubt.

In Exhibit werden alle Elemente in einem `array` gehalten. Die Unterelemente können wiederum beliebig sein. Im CasIS Portal wird allerdings nur eine zweistufige Verschachtelung vorgenommen. Dies bedeutet, dass im Portal lediglich zweidimensionale Arrays benutzt werden, so wie dies im Beispiel 7.1 mit „disciplines“ aufgezeigt wird.

Bei der Selektion der entsprechenden Lernressourcen werden die Daten von Exhibit entsprechend manipuliert. Bei der Selektion wird aufgrund eines eindeutigen Wertes „index“ das Objekt im Administratorexhibit gesucht. Wird dieses gefunden, wird es kopiert und neu dem Userexhibit angehängt. Somit enthält das Administratorexhibit stehts alle möglichen Lernressourcen, während das Userexhibit lediglich eine Kopie der ausgewählten Objekte beinhaltet. Bei der Entfernung einer Lernressource wird wiederum aufgrund des „index“ Feldes das Object im UserEx-

[1] Tomcat bezeichnet einen Java Webapplikationsserver der Apache Software Foundation, http://tomcat.apache.org
[2] http://www.json.org

hibit gesucht. Wird es gefunden, so wird das Objekt einfach gelöscht. Das Administratorexhibit ist davon nicht betroffen.

Da es mit JavaScript allein leider nicht möglich ist, Objekte auch wieder persistent zu speichern, benötigen wir eine Bibliothek, welche das einfache Manipulieren von JSON Daten ermöglicht. Eine solche Bibliothek ist json-lib[3]. Sie erlaubt das einfache Einlesen und Speichern von JSON Daten.

Exhibit Ansicht

Die Ansicht der Exhibit Daten übernimmt eine spezielle HTML Datei. Diese enthält drei Bereiche, welche für die Darstellugng benutzt werden können:

- Das „exhibit-control-panel“ ist verantwortlich für die unterschiedlichen Ansichten innerhalb von Exhibit. In CasIS werden eine Tabellenansicht sowie eine Listenansicht angeboten.
- Das „exhibit-view-panel“ zeigt die Detailansicht eines einzelnen Eintrages innerhalb des Exhibit Datenbestandes.
- Das „exhibit-browse-panel“ zeigt die unterschiedlichen Fassetten, welche innerhalb von CasIS benutzt werden können.

Die Ansicht wird komplett in JavaScript und AJAX zusammengestellt. Die Auswahl der Daten erfolgt mit Hilfe von zwei Operatoren:

- „.“ Operator: traversiert den Exhibit Graphen vorwärts. In unserem Beispiel würde „.label“ den Text „This is the first Cases Module“ ausgeben.
- „!“ Operator: traversiert den Exhibit Graphen rückwärts.

Somit ist es möglich, innerhalb von komplexen Strukturen die gewünschten Daten auszulesen und darzustellen. Eine ausführliche Dokumentation findet sich auf der Webseite des Projektes.

[3] http://json-lib.sourceforge.net

8

Schlussfolgerung

Während dieser Diplomarbeit wurde ein Portal zur Publikation von Fallstudien entwickelt. Dabei ist insbesondere der Navigation und der Suche von entsprechenden Ressourcen Rechnung getragen worden. Aktuelle Portale bieten oft nur ein Volltextsuche sowie eine Kategorisierung der Lernressourcen. Mit Hilfe der Fassettennavigation wurde aufgezeigt, dass es eine weitere effiziente Möglichkeit gibt, dem Benutzer auf einfache Weise Lerninhalte anzubieten. Nach der Entwicklung des Portals können folgende Schlussfolgerungen gezogen werden:

- Die Benutzung entsprechender Frameworks zur Entwicklung von Webanwendungen ist eine enorme Erleichterung. Sie bieten Standards, welche leicht wiederverwendet werden können. So bietet Struts2 bereits integrierte Ajax Komponenten und die Möglichkeit, eine einfache Internationalisierung zu realisieren. Beides Anforderungen an das CasIS Portal.
- Exhibit bietet eine leichte Schnittstelle zur Verwendung im CasIS Portal. Es kann leicht integriert und konfiguriert werden. Die Fassettennavigation ist sehr umfangreich und kann vom Benutzer einfach bedient werden. Exhibit wird auf dem Client ausgeführt und benutzt Ajax zur Kommunikation mit dem Server. Dies senkt die Wartezeiten für die Lernenden. Ein kleiner Nachteil von Exhibit ist das Datenformat. JSON gestattet grundsätzlich eine freie Reihenfolge der Elemente. Dies führt dazu, dass insbesondere bei Schreiboperationen die Reihenfolge der Elemente durcheinander geraten kann. Dies hat allerdings keinen Einfluss auf die Darstellung innerhalb der Fassettensuche.
- Nicht jeder Lerninhalt kann entsprechend annotiert werden. Die Fallstudien enthalten sehr viel Text, der nicht direkt annotiert werden konnte. Die Suche wurde deshalb in zwei Komponenten aufgeteilt. Einerseits die Fassettennavigation, welche auf erfassten Metadaten operiert, und andererseits eine Volltextsuche, die den Inhalt der Dateien durchsucht. Der Benutzer erhält über diese beiden Suchmöglichkeiten leichten Zugang zu allen Lernressourcen.

9

Ausblick

Das CasIS Portal wurde als Portal für unterschiedliche Fallstudien entwickelt. Diese sind derzeit in Bearbeitung und werden schlussendlich in das CasIS Portal integriert. Für jedes einzelne Modul werden entsprechende Fassetten benötigt. Diese können mit Hilfe von Exhibit auf einfache Weise an die Bedürfnisse der Administratoren angepasst werden, falls später eine Änderung erwünscht ist. Exhibit ist dabei sehr flexibel.

In absehbarer Zukunft wird auch ein ausgiebiger Benutzertest nötig. Derzeit wurde das Portal von wenigen Personen, insbesondere der Betreuerin evaluiert. Da die einzelnen Module noch nicht fertiggestellt sind, wurde auf ein richtiger Benutzertest während dieser Arbeit verzichtet. Dieser müsste mit den fertigen Modulen nachgeholt werden.

Die Internationalisierung ist soweit fertiggestellt, allerdings wurde noch keine Mehrsprachigkeit implementiert. Derzeit ist das Portal in Englisch verfügbar. Eine spätere Erweiterung ist einfach zu implementieren. Das verwendete Framework Struts2 bietet dazu sehr einfache Möglichkeiten.

Abbildungsverzeichnis

Tabellenverzeichnis

Codeverzeichnis

Literaturverzeichnis

[AI, 2007] AI, A. I. (2007). Sweet tools. http://www.mkbergman.com/?page_id=325.

[Alby, 2006] Alby, T. (2006). *Web 2.0. Konzepte, Anwendungen, Technologien*. Number 978-3446409316. Hanser Fachbuchverlag, 1 edition.

[Annozilla, 2007] Annozilla (2007). Annozilla. http://annozilla.mozdev.org/index.html.

[Aroyo et al.,] Aroyo, L., Mizoguchi, R., and Tzolov, C. Ontoaims: Ontological approach to courseware authoring.

[Arpirez et al., 2001] Arpirez, J. C., Corcho, O., Fernandez-Lopez, M., and Perez, A. G. (2001). Webode: a scalable workbench for ontological engineering.

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Ora, L. (2001). The semantic web. *Scientific American*.

[Bertelsmann, 1988] Bertelsmann (1988). *Die grosse Bertelsmann Lexikothek*. Bertelsmann, Gtersloh, Deutschland.

[Bohring and Auer, 2005] Bohring, H. and Auer, S. (2005). Mapping xml to owl ontologies. http://citeseer.ist.psu.edu/733967.html.

[Born, 2005] Born, G. (2005). *XML*. Markt+Technik, Martin-Kollar-Strasse, Mnchen, Deutschland.

[Campus, 2007] Campus, S. V. (2007). Swiss virtual campus. http://www.virtualcampus.ch/.

[CasIS, 2007] CasIS, S. V. C. (2007). Casis. http://www.virtualcampus.ch/.

[Corcho et al., 2006] Corcho, O., Cima, A. L., and Perez, A. G. (2006). A platform for the development of semantic web portals.

[Daconta Michael C., 2003] Daconta Michael C., Obrst Leo J., S. K. T. (2003). *The Semantic Web*. Wiley Publishing, Inc., Indianapolis, Indiana, USA.

[Drucker, 2000] Drucker, P. (2000). Need to know: Integrating e-learning with high velocity value chains. A Delphi Group White Paper.

[Erik Hatcher, 2004] Erik Hatcher, O. G. (2004). *Lucene in Action*. Manning Publications Co.

[Foundation, 2007] Foundation, A. S. (2007). Lucene, a fulltext search engine. Technical report, Apache Software Foundation.

[Garrett, 2005] Garrett, J. J. (2005). Ajax: A new approach to web applications.

[Grigoris A., 2004] Grigoris A., H. F. v. (2004). *A Semantic Web Primer*. MIT Press, Cambridge, Massachusetts London, England.

[Hammer and Wieder, 2003] Hammer, C. and Wieder, G. (2003). *Internet Geschftsmodelle mit Rendite*, volume 1. Galileo Press GmbH, 2003 edition.

[Handschuh et al., 2002] Handschuh, S., Staab, S., and Ciravegna, F. (2002). S-cream semi-automatic creation of metadata. http://citeseer.ist.psu.edu/handschuh02scream.html.

[Holohan et al., 2005] Holohan, E., Melia, M., McMullen, D., and Pahl, C. (2005). Adaptive e-learning content generation based on semantic web technology. http://www.computing.dcu.ie/ ontogole/publications/index.shtml.

[L.Stojanovic et al., 2001] L.Stojanovic, S.Staab, and R.Studer (2001). elearning based on the semantic web.

[Mäkelä et al.,] Mäkelä, E., Hyvönen, E., Saarela, S., and Viljan, K. Ontoviews - a tool for creating semantic web portals.

[McCormack, 2002] McCormack, D. (2002). *Web 2.0: The Future of the Internet and Technology Economy and How Entrepreneurs, Investors, Executives & Consumers Can Take Ad*. Number 978-1587622007. Aspatore Books, 1st edition.

[Melita, 2005] Melita (2005). Melita. http://nlp.shef.ac.uk/melita/features.html.

[Michiel et al., 2006] Michiel, H., van Ossenbruggen Jacco, and Lynda, H. (2006). /facet: A browser for heterogeneous semantic web repositories. online.

[MnM, 2004] MnM (2004). Mnm. http://kmi.open.ac.uk/projects/akt/MnM/index.html.

[Network Working Group, 2007] Network Working Group, U. (2007). Uniform resource identifier. http://tools.ietf.org/html/rfc3986.

[Ontoweb, 2002] Ontoweb, O.-B. I. (2002). Deliverable 1.3: A survey on ontology tools. http://citeseer.ist.psu.edu/525623.html.

[O'Reilly, 2005] O'Reilly, T. (2005). What is web2.0 - design patterns and business models for the next generation of software. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html.

[Reeve and Han, 2005] Reeve, L. and Han, H. (2005). Survey of semantic annotation platforms. ACM Press.

[Reif, 2005] Reif, G. (2005). Weesa - web engineering for semantic web applications. http://www.infosys.tuwien.ac.at/Staff/greif/weesa.pdf.

[Reif, 2006] Reif, G. (2006). Semantic annotation. In *Semantic Web - Wege zur vernetzten Wissensgesellschaft*. Springer Verlag.

[Rodrigues et al., 2006] Rodrigues, T., Rosa, P., and Cardoso, J. (2006). Mapping xml to existing owl ontologies.

[SMORE, 2005] SMORE (2005). Smore. http://www.mindswap.org/2005/SMORE/.

[Staab et al., 2001] Staab, S., Maedche, A., and Handschuh, S. (2001). An annotation framework for the semantic web. http://citeseer.ist.psu.edu/article/staab01annotation.html.

[Stojanovic et al., 2001] Stojanovic, N., Maedche, A., and Staab, S. (2001). Seal a framwork for developing semantic portals.

[SWAD-E, 2001] SWAD-E, W. (2001). Swad-europe. online.

[Tallis, 2003] Tallis, M. (2003). Semantic word processing for content authors. Second International Conference on Knowledge Capture (Sanibel, Florida, USA).

[UnicodeOrg, 2007] UnicodeOrg (2007). Unicode. http://www.unicode.org.

[W3CAnnotea, 2001] W3CAnnotea (2001). Annotea. http://www.w3.org/2001/Annotea.

[W3COWL, 2004] W3COWL (2004). Semantic web. http://www.w3.org/TR/owl-features/.

[W3CRDF, 2004] W3CRDF (2004). Rdf primer. http://www.w3.org/TR/rdf-primer/.

[W3CSchema, 2004] W3CSchema (2004). Rdf schema. http://www.w3.org/TR/rdf-schema/.

[W3CSemWeb, 2001] W3CSemWeb (2001). Semantic web. http://www.w3.org/2001/sw/.