Informationsmanagement mit XML

Praktischer Ansatz zur Entwicklung einer Anwendung mit Semantik Web Techniken

02. Juli 2010

Erstellt von:

Karl Glatz, 20570
Hochschule Ravensburg-Weingarten

Sebastian Wiedenroth, 20006
Hochschule Ravensburg-Weingarten

Benjamin Merkle, 20569
Hochschule Ravensburg-Weingarten

Andreas Kimpfler, 21027
Hochschule Ravensburg-Weingarten

Erklärung

Diese Arbeit wurde an der Hochschule Ravensburg-Weingarten als Hausarbeit im Rahmen der Vorlesung "Informationsmanagement mit XML" angefertigt.

Hiermit versichern wir, die vorliegende Ausarbeitung und Implementierung selbständig angefertigt und dabei nur die angegebenen Quellen und Hilfsmittel verwendet zu haben.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Weingarten, 02. Juli 2010

Karl Glatz

Sebastian Wiedenroth

Benjamin Merkle

Andreas Kimpfler

Inhaltsverzeichnis

Abstract

Es gibt viele Ansätze und Forschungsprojekte, die das Semantic Web vorantreiben wollen. Dennoch gehören Technologien des Semantic Web längst nicht zum Alltagswerkzeug eines Webentwicklers. Diese Arbeit gibt einen Überblick über einige Techniken und verwendet diese im Rahmen einer Beispiel-Anwendung zur Evaluierung von Praktikumsstellen für Studenten des Studiengangs Angewandte Informatik. In der Anwendung kommen verschiedene Techniken aus dem Semantic Web Umfeld zum Einsatz. Die für die Anwendung benötigten Daten werden dabei aus internen Datenbanken, dem Umfragesystem, sowie dem Verwaltungsportal für die Vorlesungen der Hochschule entnommen und aufbereitet. Die gesammelten Daten werden semantisch Aufbereitet im Web zur (freien) Verfügung gestellt. Die von uns entwickelte Ontologie versucht möglichst große Teile standardisierter Ontologien zu verwenden. Die Beispiel-Anwendung verwendet Methoden aus der künstlichen Intelligenz um die Daten zu Klassifizieren.

There are many approaches and research projects that intend to push the Semantic Web. However semantic web technologies are not yet part of the average web developers toolset. This paper gives an overview of some techniques and uses them as a part of an example application for the evaluation of internships for students studying applied computer science. Different semantic web techniques are used in the application. The data needed for the application comes from internal databases, the survey system, and in addition from the management portal for the lectures of the university and get prepared for further use. We publish the resulting semantic data to the world wide web for free use. The developed ontology uses well known ontology standards where possible. The application uses methods from artificial intelligence to classify the data.

1 Einleitung

1.1 Motivation und Zielsetzung

Über das Semantic Web gibt es bereits zahllose theoretische Arbeiten und Bücher. Auch an der Hochschule Ravensburg-Weingarten wurde in den letzten Jahren mehr über das Semantic Web geschrieben, als dass eine Projektgruppe versucht hat diese Technologien in einem Projekt praktisch umzusetzen. Daher ist die praktische Betrachtung der Technologien für uns besonders von Bedeutung. Dies manifestiert sich neben dem Titel dieser Ausarbeitung auch in der Aufteilung des Dokuments.

Neben dem kurzen Kapitel Grundlagen (2) stellen wir direkt die Projektidee vor (3). Im Verlauf dieses Kapitels folgt eine Analyse der Idee im Bezug auf benötigte und verfügbare Daten sowie deren Formate (3.3). Im Abschnitt 3.3.2 wird erläutert wie es um die Verfügbarkeit von semantischen Daten an der Hochschule bestellt ist. Dies ist deswegen so wichtig, weil es sich direkt auf das Ziel unseres Projektes auswirkt. Der Abschnitt Analyse skizziert die Ontologie (3.4) für die Hochschuldaten und anschließend rückt der Fokus auf die Umsetzung dieser Idee. In der Umsetzung (4) werden Toolkits und Frameworks zum einfachen Arbeiten mit Sematic Web Technologien im Einsatz beschrieben (4.2 und 4.3). Abschließend fassen wir unsere Ergebnisse zusammen (5.1) und geben einen kurzen Ausblick (5.2).

1.2 Arbeitsverteilung

Name	**Kapitel/Abschnitt**	**Weitere Aufgaben**
Benjamin Merkle	2; 3.5; Glossar	Diagramme
Karl Glatz	1; 2.1; 3	Konzeption, Konfiguration, JavaScript
Sebastian Wiedenroth	2.5; 3.4; 4	Konzeption, Python Code, JavaScript
Andreas Kimpfler	Abstract; 3.3.4; 5	Dokument-Style, Server, Dummy Daten

Tabelle 1: *Arbeitsverteilung der Projektteilnehmer*

2 Semantik Web Grundlagen

2.1 Einführung: Semantic Web und Linked Data

Oft gelesen, selten verstanden. Das **Semantic Web** ist kein neues Web, es kann wie das Web 2.0 nur als eine Erweiterung aufgefasst werden. Dabei ist diese nur zu einem geringen Teil technischer Natur. Die grundlegenden, technischen Konzepte und Ideen bestehen inzwischen einige Zeit. Auch deren Umsetzung ist in der Informatik-Zeitrechnung fast schon Geschichte. Trotzdem gibt es kaum Anwendungen, die wirklich Gebrauch vom Semantic Web machen. Dies begründet sich vor allem darin, dass es bisher wenige Informationen als so genannte **Linked Data** (verknüpfte Daten) gibt, was sich jedoch von Tag zu Tag ändert. Weitere Gründe für die verhaltene Entwicklung in der sonst so von Trend und Boom bestimmten IT-Branche ist auch die technische Akzeptanz, z.B. ist RDF nicht direkt intuitiv zu begreifen.

Im Folgenden werden wir von simplen Begriffserklärungen bis hin zu Projekten, die sich den unterschiedlichsten Aufgabenstellungen des Semantic Web widmen, beleuchten.

Semantic Web Erweiterung des World Wide Webs. Im Web veröffentlichte Daten sollen durch Standardformate für Maschinen les- und verarbeitbar werden. Ursprung dafür ist der Artikel [Bern01] des Web-Begründers Tim Berners-Lee. Wikipedia schreibt ([WPSEMWEB]):

> "Die Informationen im Web sollen von Maschinen interpretiert und automatisch maschinell weiterverarbeitet werden können. Informationen über Orte, Personen und Dinge sollen mit Hilfe des Semantischen Webs von Computern miteinander in Beziehung gesetzt werden können. [...] Bei der Verknüpfung der Informationen in einem Semantischen Web können neue Zusammenhänge entdeckt werden, die zuvor nicht erkennbar waren."

Linked Data Fachbegriff eines bewährten Verfahrens (best practice) zur Beschreibung von Quellen. Das Verfahren definiert die Darstellung, den Zugriff und die Verbindungen der einzelnen Teilstücke von Daten, Informationen und Wissen zu einem großen Ganzen, dem Semantic Web. Das Zusammenfügen erfolgt dabei mittels Uniform Resource Identifier (URI) und dem Resource Description Framework (RDF).

Die folgende Abbildung 1 zeigt an einem Beispiel, anhand verschiedenster Datenquellen, wie der Zugriff und die Verbindungen erfolgen und wie man dies in einem Graphen darstellen kann.

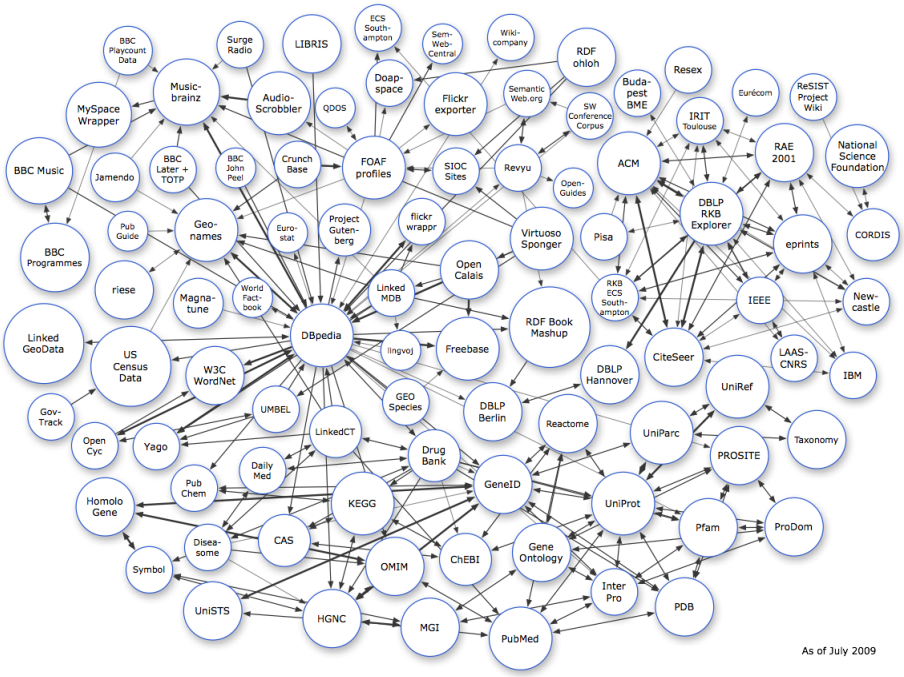

Abbildung 1: *Linking Open Data: Offene Datenquellen und deren Verbindungen; Quelle: [LINKEDATA]*

2.2 Triples

Ein Triple besteht im Bezug auf Semantic Web und insbesondere im Bezug auf RDF und OWL aus den drei Elementen Subjekt, Prädikat und Objekt. Die Idee dahinter ist, dass eine Ressource über Subjekte oder Klassen verfügen kann, die bestimmte Eigenschaften haben. Das Subjekt wird mittels einer Uniform Resource Identifier (URI) identifiziert. Es kann sich dabei um ein Dokument, einen Gegenstand oder eine Information handeln. Die Eigenschaft, welche eine Ressource hat ist das Prädikat und der Wert dieser Eigenschaft ist das Objekt. Prädikate und Objekte können Zeichenketten sein, aber auch andere Ressourcen. So kann eine rekursive Struktur aufgebaut werden. Ein Triple wird öfters auch Statement genannt. Eine Möglichkeit Triples darzustellen, zu übertragen oder zu speichern, bietet das N-Triples Format. Dieses zeilenbasierte Format serialisiert das RDF-Format als Klartext. Dieses Format wurde 2004 als offizielle Recommendation vom World Wide Web Consortium (W3C) verabschiedet. Turtle ist noch eine Erweiterung von N-Triples, da diese zu wenig Funktionalität bieten. Beispielsweise wurde eine Kurzschreibweise hinzugefügt. Beide Formate sind im wesentlichen Teil des von Tim Berners-Lee vorgeschlagenen Formats N3 (Notation 3), aber beide beschränken sich darauf nur zulässige RDF-Graphen darzustellen.

2.3 RDF

Resource Description Framework (RDF) ist ein Framework um Informationen/Metadaten über bestimmte Informationen zu speichern. RDF speichert also sozusagen „Daten über Daten“ (über Daten). Die beschriebenen Informationen sind meist Webressourcen. Dabei spielt es keine Rolle ob die Daten nur einfache Informationen sind oder ein komplexes Objekt. Zum Beispiel können Informationen über den Autor oder das Erstellungsdatum eines Dokuments dargestellt und Informationen über den Inhalt des Dokumentes selber beschrieben werden. Das Hauptziel ist mittels RDF Metadaten automatisch auszuwerten und zu bearbeiten. Weitere Ziele sind das Auffinden von Informationen im Web (Katalogisierung) und Bewertungen von Inhalten (Rating). Außerdem ist RDF ein Baustein des Web of Trust (WOT), dabei werden über eine Community Webseiten als sicher vermerkt. RDF bietet zusätzlich eine gute Interoperabilität zwischen verschiedenen Metadaten-Systemen. RDF wird vom W3C bereits seit 1999 entwickelt und ist offen für weitere Erweiterungen. Das RDF-Framework verwendet die in 2.2 Triples um die Informationen zu beschreiben.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```

Listing 1: *Beispiel für ein einfaches RDF-Dokument*

Im Codelisting 1 (entnommen aus [RDF-Prim]) findet sich ein kurzes Beispiel für die Beschreibung der Person «Eric Miller« in RDF-XML. Darin enthalten sind URIs, welche Ressourcen im Web identifizieren können, Eigenschaften (properties) wie z.B. “mailbox” oder “fullname” und die Werte der Eigenschaften (values) z.B. “Eric Miller”. Die XML Elemente werden auch Terms genannt. Der zugehörige Graph ist in 2 abgebildet.

Durch die Verknüpfung von Daten in Anwendungen werden die Zusammenhänge maschinenlesbar dargestellt. Diese Zusammenhänge können dazu verwendet werden, um dem User weitere Hilfen oder Informationen zu bieten.
Die wichtigsten Eigenschaften von RDF sind:

- Unabhängigkeit Jede Domäne kann eigene Prädikate für ihre Subjekte definieren.
- Austauschbarkeit Durch die Verwendung von XML wird der Austausch und die Kommunikation, zwischen Anwendungen oder Domänen, erheblich vereinfacht.

Abbildung 2: *Darstellung des Beispielgraphen*

- Skalierbarkeit Große Mengen von Daten können durch die Einteilung in Triples schnell und effizient verarbeitet werden.
- Prädikate sind Ressourcen Durch den rekursiven Aufbau können Prädikate eigene Prädikate haben, welche automatisch weiterverarbeitet werden.
- Objekte können Ressourcen sein Wie bei den Prädikaten gilt hier auch der rekursive Aufbau, damit auch diese wieder eigene Ressourcen sein und weiterverarbeitet werden können.

Das Ganze hat den Vorteil, dass sich die Daten aus verschieden Anwendungen exportieren lassen, ohne das die Daten der Daten ihre Bedeutungen verlieren. RDF selbst definiert kein Vokabular für die Beschreibung der Meta Daten, dafür eignet sich beispielsweise Dublin Core (DC). Noch zu erwähnen ist, dass RDF nicht durch eine XML DTD definiert ist sondern durch die Erweiterte Backus-Naur-Form (EBNF). Dadurch ist RDF unabhängig von XML.

RDF Syntax RDF Dateien können auf zwei verschiedene Arten geschrieben werden, als »Abbreviated» oder »Serialization Syntax». Bei der Serialization werden die Werte als XML-Elemente definiert. In der Abbreviated Syntax werden die Properties als Attribute definiert, dabei ensteht das Problem, dass mache Attribute dann mehrfach in einem XML Element auftauchen. Nachfolgend werden kurz die wichtigsten Elemente oder Terms beschrieben. Dabei wird nur auf die Abbreviated Syntax eingegangen.

Die Beispiele wurden [W3SRDF] entnommen. Es gibt fünf wesentliche Kategorien für die XML Notation[XML03]:

- Das RDF Element *RDF*, welches den Rahmen der Dokuments festlegt. Es ist das Wurzelelement und beinhaltet die verschiedenen Namespaces.
- *Description* definiert das Subjekt des Triple, dieses wird im *about* Attribut angegeben. Auch sind weitere Informationen, die die Ressource beschreiben enthalten.
- textitContainer, welche Eigenschaften zusammenfassen. Die drei vorhandenen Containertypen sind *rdf:Bag*, *rdf:Seq* und *rdf:Alt*. *rdf:Bag* (Tasche, Sack) wird bei ungeordneten Mengen von Aufzählungen verwendet. Bei *rdf:Seq*(für Sequenz) handelt es sich um geordnete Listen, z.B. alphabetisch. *rdf:Alt*(Alternative) wird verwendet, wenn die Aufzählung nur einen vordefinierten Wert annehmen kann. Container können wieder beschrieben werden. Beispielsweise kann der Autor eines *Container* angefügt werden.
- Der Elementtyp *type* kann auf die Beschreibung eines RDF-Schema verweisen.
- Anwendungsspezifische Elementtypen zur Beschreibung von Eigenschaften.

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

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Empire Burlesque"
cd:artist="Bob Dylan" cd:country="USA"
cd:company="Columbia" cd:price="10.90"
cd:year="1985" />

</rdf:RDF>
```

Listing 2: *Beispiel für die RDF Syntax*

In dem oben aufgeführten Codebeispiel wird eine CD von Bob Dylan definiert. Die Elemente *artist*, *country*, *company*, *price* und *year* werden im Namespace *http://www.recshop.fake/cd#* definiert. Dieser liegt außerhalb von RDF.

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

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns\#"
xmlns:cd="http://www.recshop.fake/cd\#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Empire Burlesque">
  <cd:artist rdf:resource="http://www.recshop.fake/cd/dylan" />
  ...
  ...
</rdf:Description>

</rdf:RDF>
```

Listing 3: *Beispiel für geschachtelte Ressourcen in RDF*

Im Beispiel 3 wird gezeigt wie ein geschachtelter Term aufgebaut ist. Dabei hat die *artist* Eigenschaft keinen Wert, sondern eine Referenz auf eine andere Ressource welche weitere Informationen über den Künstler enthält.

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

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns\#"
xmlns:cd="http://www.recshop.fake/cd\#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Beatles">
  <cd:artist>
    <rdf:Seq>
      <rdf:li>George</rdf:li>
      <rdf:li>John</rdf:li>
      <rdf:li>Paul</rdf:li>
      <rdf:li>Ringo</rdf:li>
    </rdf:Seq>
  </cd:artist>
</rdf:Description>

</rdf:RDF>
```

Listing 4: *Beispiel für eine Sequenz in RDF*

In 4 wird eine Sequenz von Werten verwendet. In RDF werden diese Werte von Listen *members* genannt.

Für weitere Informationen über die Syntax und Beispiele können unter [RDF-Prim] oder [W3SRDF] nachgelesen werden.

RDF Schema RDF benötigt Erweiterungen um konkret Klassen und Eigenschaften für eine Anwendung nutzen zu können. Für diesen Zweck wurde RDF-Schema (RDFS) entwickelt. Mit RDFS kann eine Grammatik definiert werden, um die Inhalte von RDF-Elementen für eine bestimmte Domäne zu modellieren. Es kann also für "kontrolliertes Vokabular" applikationsspezifisch erzeugt und eingesetzt werden. Einfache n können auf diese Weise formalisiert werden. Der gleiche Gegenstand wird innerhalb einer Applikation oder Ressource immer gleich benannt. Mehr zu Ontologien im Abschnitt über OWL 2.4. Mit RDFS kann darüber hinaus noch überprüft werden, ob eine Ressource von einem Typ ist, welcher die verwendeten Statements erlaubt oder nicht. Neben RDFS gibt es noch weitere Ontologien wie Web Ontology Language (OWL) oder Frame Logic (F-Logic).

2.4 OWL

Die Web Ontology Language (OWL) ist eine weitere Spezifikation des W3C um Ontologien mit Hilfe formaler Syntax zu erstellen. Bevor nun näher auf OWL eingegangen wird, zuerst noch ein paar Worte zu Ontologien im Allgemeinen. Eine Ontologie ist die "Konzeptionalisierung und Kodierung von Expertenwissen mit Hilfe eines strukturierten Glossars und einem vereinbarten Vokabular"[Kred07]. Mit Ontologien werden Informationen und Wissen zusammengefasst und strukturiert, um die Daten austauschbar und maschinenlesbar zu machen. Anschließend können diese Wissensbasen durchsucht, zusammengefügt und editiert werden. Auch können neue Instanzen und Wissensbasen erzeugt werden. Technisch gesehen basiert OWL auf der RDF-Syntax. Es ist ausdrucksstärker als RDFS und basiert historisch auf DARPA Agent Markup Language (DAML) und Ontology Inference Layer (OIL). OWL stellt drei Untersprachen zur Verfügung, OWL DL, OWL Full und OWL Lite. Die Mächtigkeit der OWL DL (Description Language) ist die Gleiche wie bei der Aussagenlogik (Logik 1. Stufe). Dies bedeutet, dass die Sprache berechen- und entscheidbar ist. OWL Full ist mächtiger als (Logik 2. Stufe) deshalb ist sie nur semi-entscheidbar. Die Lite Version ist ähnlich wie OWL DL, nur dass die Kardinalitäten nur 0 oder 1 sind. Mehr zum Thema OWL [**?**]

2.5 SPARQL

SPARQL Protocol and RDF Query Language (SPARQL)[1] (gesprochen "sparkle") ist eine Abfragesprache für Graphen. Der Name ist ein rekursives Akronym für "SPARQL Protocol and RDF Query Language" und benennt somit die zwei Bestandteile. Zum einen beschreibt SPARQL ein Protokoll[2], zum anderem aber auch eine Abfragesprache[3].

[1] http://www.w3.org/2009/sparql/wiki/Main_Page
[2] http://www.w3.org/TR/rdf-sparql-protocol/
[3] http://www.w3.org/TR/rdf-sparql-query/

Die Hauptidee hinter der Abfrage Sprache ist einfaches Pattern-Matching. Ein Untergraph wird mit Platzhaltern beschrieben und wo immer ein Untergraph übereinstimmt wird ein Resultat zurückgegeben.

Anfragen werden an sogenannte SPARQL-Endpoints gestellt, diese verarbeiten die Anfrage und liefern die Resultate zurück. Die meisten Datenquellen bieten öffentliche SPARQL-Endpoints an über die der Datenbestand abgefragt werden kann. Der Endpoint von DBPedia (siehe 2.6.2) ist z.B. unter http://dbpedia.org/sparql erreichbar.

SPARQL-Endpoints sind [4] Webservices, eine Abfrage ist somit nur ein einfacher HTTP-GET Request. Im Protokoll-Teil der Spezifikation wird dies genauer beschrieben.

Die Resultate können, abhängig von der eingesetzten Endpointsoftware, in verschiedenen Formaten zurückgegeben werden. Die gängigsten Formate sind XML, JSON und Plaintext für "ASK" und "SELECT" Queries, sowie RDF/XML, N-Triples, N3 und Turtle für "DESCRIBE" und "CONSTRUCT" Abfragen.

Um das SPARQL-Protokoll nicht selbst implementieren zu müssen gibt es Bibliotheken für viele Programmiersprachen. Für JavaScript gibt es sparql.js[5], und für Python gibt es SPARQLWrapper[6], welcher einfach anzuwenden ist.

2.6 Projekte

Verschiedene Projekte befassen sich bereits mit den Aufgaben und Möglichkeiten des Semantic Web. Diese haben teilweise verschiedene Herangehensweisen für die verschiedenen Aufgaben. Teilweise sind diese auch nur Grundlagen für andere Projekte, einige davon werden nun nachfolgend kurz vorgestellt.

2.6.1 Suchen

Die Suchmaschinen, welche die Informationen zusammentragen sind ein wichtiger Bestandteil. In dieser Arbeit werden kurz zwei Beispielsuchmaschinen vorgestellt, welche Ontologien zusammentragen und für Abfragen zugänglich machen.

Watson Watson ist eine von Knowledge Media Institute entwickelte Suchmaschine zur Auffindung von Ontologien und semantischen Inhalten. Es ist nach eigenen Aussagen das Gateway für das Semantic Web, welches von den Anforderungen des Semantic Web und den Erfahrungen aus früheren Systemen geleitet wird und bildet den Ausgangspunkt für die Auffindung von Ontologien. Aufgaben sind dabei das Sammeln von Daten, deren Analyse und Auswertung, sowie die Beantwortung von Anfragen. Nähere Informationen sind unter [WATSON] zu finden.

[4] http://en.wikipedia.org/wiki/Representational_State_Transfer
[5] http://www.thefigtrees.net/lee/sw/sparql.js
[6] http://sparql-wrapper.sourceforge.net/

Sindice Sindince ist eine Suchmaschine, welche die über das ganze Web verstreuten Definitionen des Semantic Web sammelt. Dabei spielt es keine Rolle von welcher Art von Seite die Informationen zusammengetragen werden. Verarbeitet werden Daten in den Formaten RDF, RDFa und Microformats. Diese zusammengetragenen Informationen können wie bei Watson auch abgefragt werden.

2.6.2 Datahubs

Datahubs sind Projekte, welche Informationen zusammentragen und in semantischer Form abfragbar machen. Diese Informationen können von Maschinen gelesen und verbunden werden um dem Benutzer Informationen über viele Bereiche hinweg bereitstellen zu können.

DBpedia Die DBPedia ist ein Community Projekt um Wissensinhalte aus Wikipedia zu extrahieren und die Informationen maschinenlesbar im Web zu veröffentlichen. Interessant ist dabei auch, dass die Daten als Linked-Data auch auf andere Datenquellen verweisen. Die von Wikipedia eingesetzte Software "Mediawiki" kann selbst allerdings noch nicht mit Semantic-Web Daten umgehen. Um dies zu erreichen gibt es das Semantic-Mediawiki[7].

Freebase Freebase[8] ist eine sehr große eigenständige Wissensdatenbank. Daten werden aus vielen verschiedenen Quellen, wie z.B. MusicBrainz[9], Wikipedia und NNDB[10] als Linked-Data importiert - können jedoch auch von Benutzern wie ein Wiki direkt editiert werden.

[7] http://semantic-mediawiki.org
[8] http://www.freebase.com
[9] http://musicbrainz.org
[10] http://www.nndb.com

3 Projektidee und Analyse

3.1 Hintergrund: Praxisphase eines Bachelor Studenten Angewandte Informatik

Der Bachelor Studiengang Angewandte Informatik (AI) an der Hochschule Ravensburg-Weingarten ist auf 6 Semester ausgelegt. Dies wird dadurch möglich, dass die Studenten in ihrem 6. Semester neben 3 Monaten Praxisphase die übrigen 3 Monate zum Schreiben ihrer Bachelorarbeit nutzen müssen. Viele Studenten gehen daher nach der von der Hochschule empfohlenen Methode vor und absolvieren beides bei einer Firma. Das verringert die Einarbeitungszeit und ermöglicht es, schnell ein Thema für die Bachelor Arbeit zu finden. Dadurch gewinnt die Auswahl einer passenden Firma an Bedeutung.

Die Hochschule versucht schon frühzeitig Firmen und Studenten durch Veranstaltungen wie den Karrieretagen zusammenzubringen. Trotzdem stehen die Studenten am Anfang ihres 5. Semesters vor der Aufgabe jetzt möglichst schnell eine Firma zu finden.

Schwerpunkte Dabei ist es nicht möglich bei jeglichen Firmen zu arbeiten. Die Aufgaben in der Firma müssen weitestgehend mit dem gelernten Wissen in den Schwerpunkten übereinstimmen. Die Schwerpunkte werden von den Studenten bereits im 3. Semester gewählt. Dabei müssen zwei aus drei zur Zeit möglichen Schwerpunkten ausgewählt werden. Daher muss der Student nicht nur eine Firma finden, die seine eigenen Interessen möglichst gut abbildet, sondern es sollten auch die gewählten Schwerpunkte berücksichtigt sein.

Firmenbewertungen Nach Abschluss der Praxisphase werden die Firmen nach einigen, vom Leiter des Praktikantenamts vorgegebenen, Kriterien bewertet. Diese Bewertungen finden seit einiger Zeit online mit dem Umfragetool INKIDU[11] statt.

Professoren Sowohl für die Praxisphase als auch für eine Bachelor-Arbeit benötigen die Studenten einen betreuenden Professor, welche als Dozenten bei bei bestimmten Veranstaltungen auftreten.

3.2 Idee: Automatische Empfehlung einer Firma

Grundidee Der Student soll, basierend auf den **Veranstaltungen** (z. B. Vorlesungen) die er im Rahmen seines gewählten **Schwerpunkts** besucht hat, eine Liste der Firmen erhalten, die am Besten zu diesen Veranstaltungen passen. Optional soll der Student seine Interessen angeben können, welche ebenfalls bei der Empfehlung berücksichtigt werden.

Welche Firma gut zu den gewählten Schwerpunkten und den Präferenzen des Studenten passt, wird anhand der Bewertungen aus vorherigen Praxisphasen ermittelt. Welcher Professor gut zu einer gewählten Firma passt, soll anhand der Veranstaltungen bei denen ein Professor Dozent ist entschieden werden.

[11] http://www.inkidu.hs-weingarten.de

Zu beiden Empfehlungen, Firma und Professor, soll es jeweils auch Alternativen geben, also das Ergebnis dieser Klassifizierung (siehe 3.5) soll eine sortierte Liste sein. Bei der die Firma, die am besten "passt", als erstes genannt ist. Zu jeder Firma soll eine sortierte Liste mit Professoren angegeben werden.

3.3 Analyse: Datenformate und Vorgehensweise

Nachdem die Grundidee skizziert ist, stellt sich die Frage woher die Daten kommen, um eine solche Anwendung zu bauen. Aber nicht nur die Quelle der Daten ist wichtig, auch deren Format für die Weiterverarbeitung. Genau zu dieser Fragestellung wollen wir hier eine Antwort finden.

3.3.1 Warum Ontologien/Linked Data/Semantic Web nutzen?

Der klassische Ansatz für ein Problem, wie das im vorherigen Abschnitt beschriebene, ist eine Anwendung mit einer relationalen Datenbank. Hier können die ganzen Daten aus den Quellen gesammelt werden und mit SQL und etwas "Geschäfts-Logik" kann die Anwendung einfach realisiert werden.

Warum sollte man sich also die Mühe machen und sperrige XML basierte Dokumente erstellen wie RDF? Wieso sollte man neue Abfragesprechen wie SPARQL lernen? Der Nutzen im Verhältnis zum Aufwand, wird bei der Betrachtung *einer Anwendung* nicht direkt deutlich.

Macht man sich aber klar, dass die Daten auch für andere Anwendungen von Bedeutung sein könnten dann wird schnell klar, dass es doch Sinn macht diese Daten in den vom W3C empfohlenen Sprachen zu beschreiben. Problematisch ist dieser Ansatz weil Unternehmen und Behörden so nicht denken. Erst wenn es einen Bedarf an Daten von externer Seite gibt, wird eine, meist spezifische, Schnittstelle geschaffen.

3.3.2 Open Linked-Data an der HS Weingarten

Im Idealfall wären alle Daten schon öffentlich in einem Standardformat zugänglich. Es fehlt also zunächst eine API für Daten der einzelnen Dienste an der Hochschule. Ideal wäre es, wenn die Dienste die Daten als Linked-Data veröffentlichen würden. Dies würde es ermöglichen, die Daten aus den verschiedenen Quellen (Hochschulintern sowie mit externen Quellen; siehe dazu 2.6.2) direkt zu kombinieren.

Das Semantic Web kann nur dann eine Bedeutung erlangen, wenn Webangebote, wie das unserer Hochschule, anfangen die Daten auch semantisch zur Verfügung zu stellen. Bisher gibt es von offizieller Stelle dazu keine Planungen. Daher ist eines unserer Ziele in diesem Projekt:

> **Öffentliche Daten der Hochschule semantisch aufzubereiten und zur Verfügung zu stellen**.

3.3.3 Datenquellen

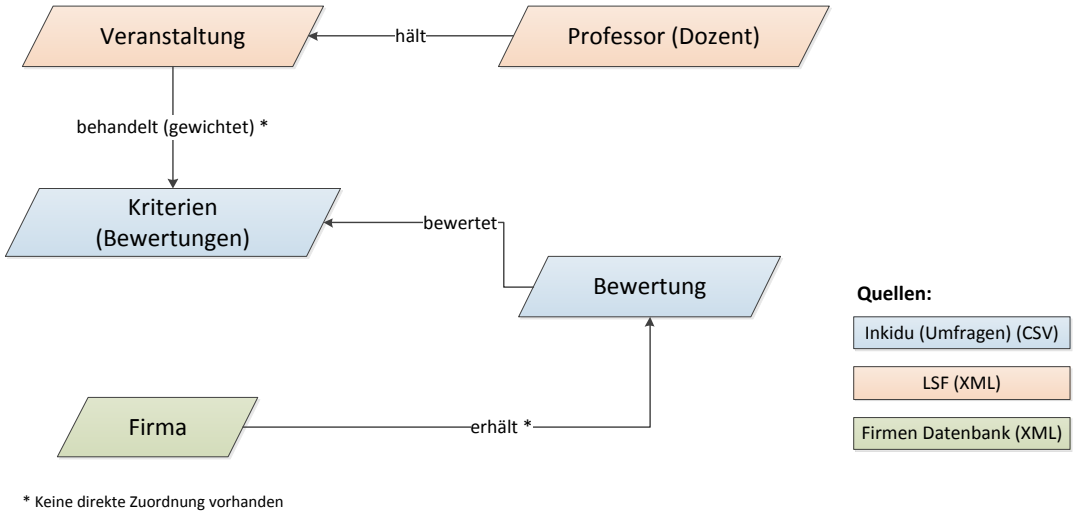

Abbildung 3: *Benötigte Daten und deren Beziehungen (vereinfacht)*

Abbildung 3 zeigt die Daten und deren Beziehungen. In dieser stark vereinfachten Darstellung kann man die Zusammenhänge zwischen den Daten leicht erkennen. Auch deren Quellen, sowie das Format wird in der Grafik kurz angedeutet.

Die Daten an der Hochschule über Veranstaltungen oder Professoren werden im sogenannten Lehre, Studium, Forschung (LSF) der Firma HIS verwaltet. Dabei werden einige Informationen auch öffentlich als HTML ausgeliefert. Leider werden diese Informationen nicht als Triples (z.B. in RDF) veröffentlicht. Es gibt lediglich auf Anfrage beim Rechenzentrum eine XML Schnittstelle bei der die benötigten Daten aus der Datenbank verfügbar gemacht werden können.

Wie in Abbildung 3 bereits unter Quellen zu sehen, gibt es 3 primäre Datenquellen für unsere Anwendung an der Hochschule. Die mit * gekennzeichneten Verbindungen sind nicht vorhanden bzw. lassen sich nicht direkt aus den Daten ableiten.

LSF Das LSF[12] bietet einen XML Export. Leider ist diese Schnittstelle noch nicht so weit ausgebaut als das wir sie für unser Projekt nutzen könnten. Daher wurden die LSF Daten exemplarisch von Hand ermittelt. Verbindungen zwischen Veranstaltung und Dozenten (Professoren) sind jedoch eindeutig.

Fimen Datenbank Die Firmendatenbank (PDB) wird von Hand von der LeiterIn des Praktikantenamts gepflegt. Manuell werden die Daten als XML auf einem Webserver abgelegt.

[12] http://www.lsf.hs-weingarten.de

INKIDU Das Online Umfragesystem INKIDU[13] ermöglicht einen Character Separated Values (CSV) Export der Umfrageergebnisse. Dieses System erlaubt dem Nutzer nur Freitextfelder und bietet daher keine Verbindung zwischen den Umfrageergebnissen und den Daten die zu einer Firma in der Firmendatenbank gespeichert sind. Ideen für die Zuordnung finden sich im Abschnitt 3.3.4.

3.3.4 Algorithmen

Um aus den Daten "Linked Data" zu machen, können verschiedene Techniken angewandt werden. Primär geht es darum, eine Zuordnung zwischen nicht direkt verbundenen Datensätzen zu finden. Ein weiterer Aspekt bei der Veröffentlichung von Daten ist der Datenschutz. Da es sich bei den Bewertungen um Meinungen von Studenten über Firmen handelt, dürfen diese natürlich nicht direkt veröffentlicht werden.

Firmenzuordnung Wie bereits mehrfach erwähnt, gibt es keine Zuordnung zwischen den Firmen in der Firmen Datenbank und den Bewertungen der Studenten. Dadurch, dass die Daten in verschiedensten Formaten vorliegen und auch nicht in einem einheitlichen Konzept abgelegt wurden, bzw. keine eindeutige Identität haben, ist es nur möglich diese anhand eines Pattern-Matching-Verfahrens zu vergleichen und die größtmögliche Ähnlichkeit als Referenz zu verwenden.
Dabei wird ein String, der einen Firmennamen repräsentiert, anhand der im String vorkommenden einzelnen Worte unterteilt und dabei ausgehend von rechts nach links mit dem vorliegenden String verglichen. Wichtig ist dabei, dass unbedingt von rechts angefangen werden muss den String zu vergleichen, da in der deutschen Sprache Firmennamen immer mit der Gesellschaftsform am Ende definiert werden. Nehmen wir einmal das Beispiel

```
Mayer Informatik GmbH
```

so würde dies maximal drei Durchläufe zum Vergleich ergeben:

```
Mayer Informatik GmbH
Mayer Informatik
Mayer
```

Dabei kann es natürlich passieren, dass zum Ergebnis `Mayer` mehrere Einträge in der Firmendatenbank gefunden werden, aber dies ist anhand der vorliegenden Daten leider nicht zu vermeiden. Ist nach dem dritten Durchlauf immer noch kein Matching möglich, so wird der Versuch erfolglos abgebrochen. In beiden Fällen wird die Firma im Ergebnis nicht angezeigt.

[13] http://inkidu.de

Bayesian average Der CSV Export der INKIDU Umfragen enthält jede Firmenbewertung einzeln. Aus Datenschutzgründen müssen diese Bewertungen vor der Veröffentlichung auf einen Wert pro Firma aggregiert werden. Um eine faire Gewichtung zwischen Firmen mit vielen Bewertungen und welchen mit wenigen zu erreichen wird die "Bayesian average" (1) berechnet.

$$\bar{x} = \frac{Cm + \sum_{i=1}^{n} x_i}{C + n} \tag{1}$$

Hierbei ist C die durchschnittliche Anzahl der Bewertungen, m der A-priori Durchschnitt und n die Anzahl der Bewertungen für dieses Element. Eine praktische Erklärung ist in [Weichs06] zu finden.

3.4 Datenmodellierung im Projekt

3.4.1 Ontologien

Ziel beim Aufbau der Ontologie für unseren Anwendungsfall ist es, so viele standardisierte "Terms" zu verwenden wie möglich. Sich also stets bei anderen, bekannten und etablierten Ontologien zu bedienen. Es soll nachher keine völlig neue Ontologie entstehen, sondern nur die Teile für die es keine "Terms" gibt erweitert werden. Im Sinne des Semantic Web Gedanken, sollten Daten wie Bezeichnungen von Objekten möglichst mit den bekannten Terms bezeichnet werden.

In dem Vortrag von Tim Berners Lee vom Mai 2010 [W3CQALD] greift er die Problematik der Zurückhaltung vieler Entwickler in Sachen Semantic Web auf. Viele Entwickler haben das Semantic Web missverstanden, prangert der Web-Erfinder an und versucht es anhand einer Tüte Kartoffelchips für jeden verständlich zu erklären. Es wird keine große universalgültige Ontologie geben, doch das darf uns nicht davon abhalten die Techniken nicht einzusetzen. *Open Your Data NOW!*

Im Folgenden werden Ontologien, die wir im Projekt benutzt haben, vorgestellt:

DC Dublin Core[14] beschreibt grundlegende Metadaten-Elemente. Die Elemente wie Titel, Ersteller, Thema, Beschreibung, Datum und Sprache sind allgemeingültig genug um viele Dinge wie Bücher, Filme, Bilder oder Web Seiten zu beschreiben. Dublin Core ist auch als ISO Standard 15836 festgelegt.

FOAF Friend of a Friend[15] ermöglicht es Menschen und ihre Beziehungen untereinander maschinenlesbar darzustellen. Dazu gehört zum einen Menschen zu beschreiben, also Eigenschaften wie Namen, Alter, Email Adresse und Website zu definieren. Zum anderem werden Relationen zwischen Personen modelliert. Dies kann entweder über direkte "knows" Beziehung zwischen Menschen geschehen oder über Gruppenzugehörigkeiten.

[14] http://dublincore.org
[15] http://www.foaf-project.org

Die Spezifikation[16] wird stets weiterentwickelt und wurde zuletzt im Januar 2010 aktualisiert. Viele soziale Netzwerke wie LiveJournal und identi.ca[17] bieten FOAF Informationen über ihre Nutzer an. Daraus resultiert, dass FOAF einen Großteil der im Web verfügbaren semantischen Daten repräsentiert. Auch in Zukunft wird FOAF eine wichtige Rolle spielen, wenn soziale Netzwerke via FOAF+SSL[18] oder mit Diaspora[19] komplett dezentral werden.

AIISO Die Academic Institution Internal Structure Ontology (AIISO)[20] ist eine Ontologie mit welcher die Struktur von akademischen Instituten beschrieben wird. Es können Lehreinheiten wie Kurse, Fächer und Module beschrieben werden. Diese können, ganz im Sinne von Linked-Data, wiederum andere Lehreinheiten enthalten und auf diese verweisen. Des weiteren können Fakultäten, Departments, Institute und ihre Zusammenhänge beschrieben werden.

SKOS Simple Knowledge Organization System (SKOS)[21] ist ein einfaches Knowledge Organization Systems (KOS) für das Semantic Web. Die vom W3C selbst entwickelte Spezifikation[22] erlaubt es ein eigenes Vokabular zu veröffentlichten. Somit ist es möglich Konzepte und Begriffe für ein Klassifizierungsschema oder eine Taxonomie zu definieren.

Eigene Definition Ein Großteil der verwendeten Daten kann also mit bereits bestehenden Ontologien modelliert werden. Einzig für die Relevanz-Information zwischen einem Topic (skos:Concept) und einem anderem Subjekt wurde ein eigenes Format definiert. Die Zusammenhänge der Daten werden in Abbildung 4 verdeutlicht.

Es ist empfehlenswert bekannte und bereits bestehende Ontologien einzusetzen, da somit das Verbinden mehrerer Datenquellen vereinfacht wird. Auch wenn dies keine harte Anforderung ist, da man mit “owl:sameAs” relativ einfach mappings zwischen unterschiedlichen Ontologien herstellen kann, ist es dennoch sinnvoll auf bereits bestehenden Arbeiten aufzubauen. Aus diesem Grund werden weitere Empfehlungen für Ontologien in [ONTOPAT] aktiv mit der Community als Wiki diskutiert.

[16] http://xmlns.com/foaf/spec/
[17] http://identi.ca
[18] http://esw.w3.org/Foaf%2Bssl
[19] http://www.joindiaspora.com
[20] http://vocab.org/aiiso/schema
[21] http://www.w3.org/2004/02/skos/
[22] http://www.w3.org/TR/skos-reference/

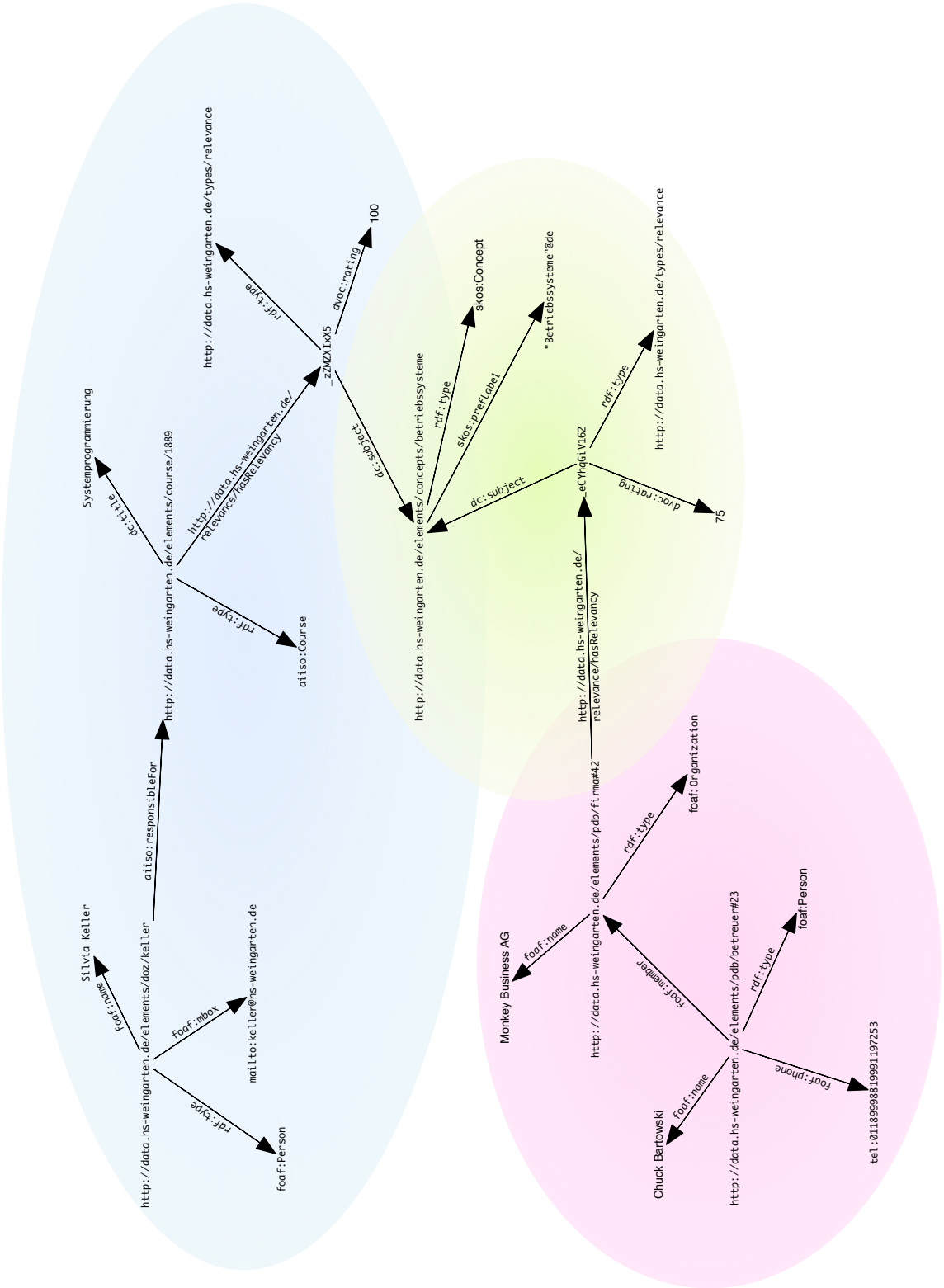

Abbildung 4: *Aufbau des Datenmodells an einem konkreten Beispiel*

3.4.2 Daten veröffentlichen

Die Daten werden als RDF Dateien auf einem Webserver abgelegt. Zusätzlich soll ein SPARQL Endpoint dafür sorgen, das jeder direkte Abfragen über Daten machen kann.

Die Subdomain “data” hat sich für die Veröffentlichung von (semantischen) Daten bewährt. Prominente Beispiele:

- data.gov - US Regierung
- data.gov.uk - UK Regierung
- data.worldbank.com - Weltbank
- data.nytimes.com - New York Times

Daher wäre die Adresse **data.hs-weingarten.de** wünschenswert.

3.5 Klassifizierung der Firmen und Professoren

Sind alle Daten als »Linked Data« verfügbar, muss unsere Anwendung die Firmen und Professoren den gewählten Schwerpunkten und Präferenzen des Studenten zuordnen. Diese Zuordnung wird auch als **Klassifikationsproblem** bezeichnet. In [Ertel08] finden sich im Kapitel 8 detaillierte Erklärungen zu den einzelnen Verfahren. Die erste Schwierigkeit ist ein geeignetes Verfahren für die Klassifizierung zu finden. Das Verfahren muss die größtmögliche Ähnlichkeit der Daten in der Datenbank, mit den eingegebenen Daten finden. Es orientiert sich an den bisherigen Daten (Bewertungen). Diese werden daher auch als Trainingsdaten bezeichnet.
Ein einfaches Verfahren diese Ähnlichkeit zwischen den gegebenen und den gesuchten Daten zu finden, ist die Nearest-Neighbour-Klassifikation. Dabei werden beispielsweise mit Hilfe der euklidischen Distanz die Abstände zwischen dem gesuchten Punkt im n-dimensionalen Raum und den gespeicherten Punkten gesucht. Der neue Punkt wird dann gleich klassifiziert wie der Punkt der ihm am nächsten ist. In Abbildung 5 ist in einem Voronoi-Diagramm dargestellt. Jeder Punkt wird von einem konvexen Polygon umgeben welches das Gebiet definiert, in dem ein neuer Punkt gleich diesem Punkt klassifiziert wird.

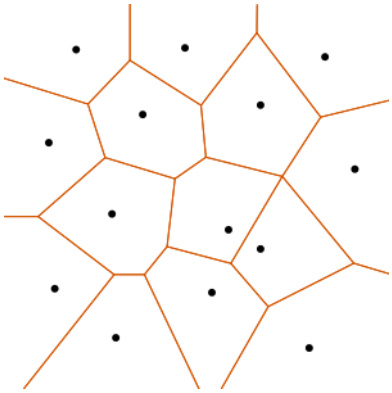

Abbildung 5: *Beispiel eines Voronoi-Diagramm*

Das Problem dabei ist das durch Ausreißer, Rauschen oder Fehleingaben sich eine Überanpassung einstellen kann. In unserem Fall könnte eine sehr gute oder schlechte Bewertung die neue Bewertung stark beeinflussen und eine Firma ins Abseits schieben. Der k-Nearest-Neighbour-Algorithmus ist eine Verbesserung, die nicht nur den nächsten Nachbarn sondern die k nächsten Nachbarn betrachtet. Für k=3 also die 3 nächsten Nachbarn.

Der beschriebene Algorithmus (inkl. Verbesserung) zählt zu den *lazy learning* (faules Lernen) Verfahren, weil bei der Berechnung immer alle Trainingsdaten benutzt werden. Dies ist bei kleinen Datenmengen jedoch unkritisch.

4 Umsetzung

Dieser Abschnitt beschreibt die technischen Hilfsmittel und Vorgehensweisen bei der Umsetzung der Projektidee. Die Architektur kann in drei Bereiche geteilt werden. Zuerst werden die Daten aus verschiedenen Quellen zusammengetragen, aufbereitet und in dem zentralen Triplestore gespeichert. (Siehe 6) Der zweite Bereich beschäftigt sich damit, wie die semantischen Daten der Welt zugänglich gemacht werden. Zuletzt wird die Beispielanwendung "Semantic Internship Advisor" vorgestellt, welcher auf die Daten zugreift.

4.1 Architektur Übersicht

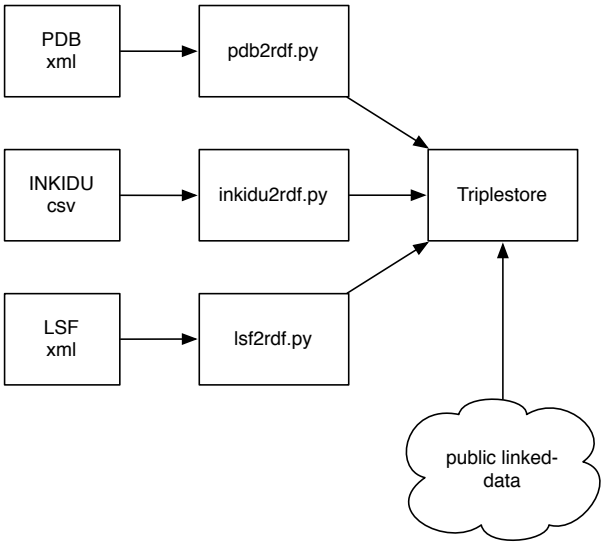

Abbildung 6: *Architektur: Daten aufbereiten*

4.2 Daten aufbereiten: Python Skripte

Die Konvertierung der Datenquellen nach RDF ist mit Hilfe von drei kleinen Python Skripten umgesetzt. Für jede Datenquelle wurde ein Skript verfasst.

4.2.1 RDFLib

Dank RDFLib[23] ist das Erstellen eines Graphen und die Ausgabe als RDF/XML sehr einfach zu realisieren. RDFLib ist eine Python Bibliothek, welche Parser und Serialisierer für viele Varianten von RDF enthält. Des weiteren ist ein "in-memory" Graphstore enthalten. Es besteht aber ebenso die Möglichkeit den Graphen persistent zu speichern.

[23] http://www.rdflib.net/

4.2.2 Arbeitsweise

Die Skripte arbeiten in zwei schritten. Im ersten Schritt werden die Originaldaten geladen und in eine interne Datenstruktur gebracht. Für XML Quellen wird das Python Modul »xml.dom.minidom« als Parser eingesetzt. Für das CSV Format bietet Python in seiner Standardbibliothek einen Parser.

Im zweiten Schritt wird die Datenstruktur abgearbeitet und in Triples strukturiert. Diese Triples werden dann in einen temporären Graphen eingefügt.

```
1 for betreuer in firma['betreuer']:
2   betreuer_node = PDB_BETREUER[betreuer['id']]
3   g.add((betreuer_node, RDF.type, FOAF['Person']))
4   g.add((betreuer_node, FOAF['member'], firma_node))
5   g.add((betreuer_node, FOAF['name'], Literal(betreuer['name'])))
```

Listing 5: *Alle Betreuer einer Firma werden zum Graphen "g" hinzugefügt*

Im Codelisting 5 wird für jeden Betreuer einer Firma eine "betreuer_node" im Namespace PDB_BETREUER erstellt. Über diese Node werden dann drei Aussagen (Triples) getroffen. Zuerst wird der Typ als foaf:Person festgelegt. In Zeile 4 wird festgelegt, dass der Betreuer Mitglied einer Firma ist. Zuletzt wird der Name als Literal gesetzt. Diese drei Aussagen werden in den Graphen "g" mit der "add" Methode hinzugefügt.

4.2.3 Verknüpfen der Datenquellen

In dem Umfragetool INKIDU werden die Firmen einzig über ein Freitextfeld identifiziert, welches vom Umfrageteilnehmer ausgefüllt wird. Herauszufinden für welche Firma aus der PDB eine Bewertung gilt ist somit maschinell nicht trivial.

Das PDB Skript generiert für jede Firma eine eindeutige URI. Während dem Konvertieren der INKIDU Daten wird versucht die URI der Firma zu finden. Dazu wird dem Freitextfeld nur das erste Wort entnommen. Darauf hin wird eine SPARQL Abfrage auf die konvertierten PDB Daten gestartet. Da SPARQL Abfragen reguläre Ausdrücke enthalten können ist es möglich sehr mächtiges Stringmatching durchzuführen.

```
1  def get_firma(haystack):
2    sparql = SPARQLWrapper(SPARQL_ENDPOINT)
3    sparql.setQuery("""
4      PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
5      PREFIX rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
6      SELECT ?firma
7      WHERE {
8        ?firma foaf:name ?firma_name .
9        ?firma rdf:type foaf:Organization .
10       FILTER regex(?firma_name, "^%s", "i" )
11     }
```

```
12    """ % (haystack.split()[0], ))
13    sparql.setReturnFormat(JSON)
14    results = sparql.query().convert()['results']['bindings']
15    if len(results) == 1:
16      return results[0]['firma']['value']
17    return None
```

Listing 6: *Funktion um mit Hilfe von SPARQL Firma zu finden*

In Listing 6 ist eine Funktion gezeigt, welche die eben beschriebene Vorgehensweise implementiert. Zuerst wird eine SPAQRL-Verbindung zur Endpoint-URL aufgebaut. Die Anfrage definiert zwei Namespaces. Der Friend-of-a-Friend (siehe 3.4.1) Namespace wird als "foaf" abgekürzt. Ebenso wird der RDF Namespace festgelegt. Gesucht wird eine Firma deren Name mit dem ersten Wort des Freitextfeldes beginnt und die vom Typ "foaf:Organisation" ist. Kann diese Anfrage eindeutig beantwortet werden wird der Identifier zurückgegeben.

4.3 Daten verfügbar machen: Jena und Joseki

Bei **Jena**[24] handelt es sich um ein Framework für das Semantic Web in der Programmiersprache Java. Diese wurde ursprünglich in den »HP Labs Semantic Web Research«[25] entwickelt und ist seit Oktober 2009 ein unabhängiges Open-Source Projekt. Es ermöglicht, ähnlich wie die RDFLib (4.2.1), das bequeme lesen und schreiben von RDF Dateien. Darüber hinaus bietet Jena auch einen SPARQL Query-Endpoint (Server) namens **Joseki**. Alle Module in Jena sind so implementiert, dass die Teile auch einzeln zur Entwicklung einer eigenen Anwendung genutzt werden können.

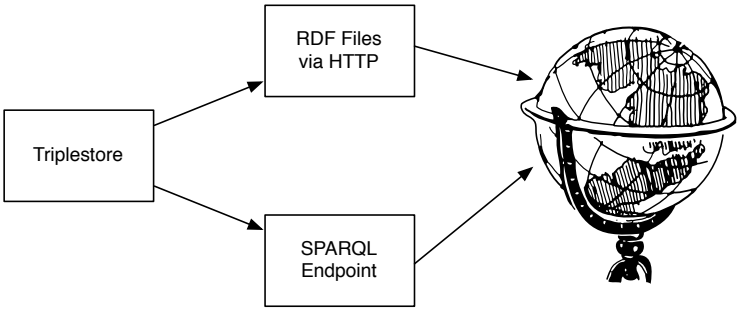

Abbildung 7: *Architektur: Daten veröffentlichen*

[24] http://openjena.org/
[25] http://www.hpl.hp.com/semweb/

4.3.1 Joseki Server

Die Joseki Web-Anwendung stellt das zentrale Element unseres Servers dar. Es erfüllt die Aufgabe, die gesammelten Daten dem gesamten Web bereit zu stellen. Unsere Anwendung (siehe 3 und 4.4) ist nur eine von vielen möglichen Anwendungen die diese Daten dann nutzen kann.

Da wir die Daten sowohl als statische RDF-Dateien zum Download, also auch über das SPARQL-Protokoll anbieten, werden die HTTP Verbindungen von einem Apache Webserver angenommen. So werden die statischen Dateien, sowie die Beispiel-Webanwendung vom Apache ausgeliefert. Stellt der Webserver anhand der aufgerufenen URL fest, dass es sich um eine SPARQL Abfrage handelt, wird die Verbindung an den Joseki Server weitergegeben.

4.4 Die Anwendung: Semantic Internship Advisor

Die Anwendung wurde komplett in JavaScript realisiert. Dazu werden die beiden Bibliotheken »jQuery«[26] und »sparql.js« verwendet. Die Anwendung macht direkte Abfragen auf den Joseki SPARQL Endpoint (siehe 4.3). Beim Aufruf der Anwendung werden die Themen und Schwerpunkte über eine SPARQL Abfrage geladen und daraus dynamisch eine HTML Form aufgebaut. Hat der Benutzer seine Schwerpunkte und Präferenzen festgelegt werden die Firmen, die am ehesten dazu passen geladen und inklusive möglichen Betreuern angezeigt.

Abfragen In Listing 7 ist zu sehen wie eine SPARQL Abfrage unter JavaScript aussehen kann. In diesem Beispiel wird zuerst ein SPARQL Objekt mit dem Endpoint als Parameter angelegt. Daraufhin werden wie üblich die verwendeten Namespaces deklariert. In Zeile 7 wird das eigentliche Query ausgeführt, wobei wie in JavaScript üblich die Callback-Handler inline als Parameter angegeben werden. Bei Erfolg (Zeile 13) kann die Antwort verarbeitet werden. Hier werden die einzelnen Schwerpunkte als Option in ein Auswahlfeld hinzugefügt.

```
1  var s = new SPARQL.Service("http://fh-data.frubar.net/query/hswgt");
2  s.setPrefix("rdf", "http://www.w3.org/1999/02/22-rdf-syntax-ns#")
3  s.setPrefix("aiiso", "http://purl.org/vocab/aiiso/schema#")
4  s.setPrefix("dc", "http://purl.org/dc/elements/1.1/")
5
6  var q = s.createQuery();
7  q.query("SELECT ?m ?title FROM <http://fh-data.frubar.net/rdf/lsf.rdf> \
8               WHERE { ?m rdf:type aiiso:Programme . ?m dc:title ?title }",
9    {
10     failure: function() {
11       alert('Laden der Schwerpunkte fehlgeschlagen');
12     },
```

[26] http://jquery.com

```
13      success: function(json) {
14        for (var x in json.results.bindings) {
15          $('#schwerpunkt').append($("<option></option>").
16            attr("value",json.results.bindings[x].m.value).
17            text(json.results.bindings[x].title.value));
18        }
19      }
20    });
```

Listing 7: *SPARQL Abfrage mit JavaScript*

Folgende (geplante) Eigenschaften sind nicht implementiert:

- Vorschlag eines Betreuers (Professor) pro Firma
- Daten vom LSF werden nicht verarbeitet
- Kein automatisches Laden von INKIDU Umfrageergebnissen
- Kein durchstöbern (browsen) der Daten möglich

Die Daten vom LSF konnten uns im Zeithorizont des Projektes leider Aufgrund von fehlenden Kapazitäten im Rechenzentrum nicht bereitgestellt werden.

Zuordnung Die Zuordnung von Firmen zu den gewählten Schwerpunkten ist aus mehreren Gründen nicht wie in 3 beschreiben realisiert worden:

- Keine »echten« Daten aus dem LSF.
- Fehlende Bewertung der Kriterien pro Veranstaltung

Aus diesen Gründen wurde für die Demo-Anwendung die Schwerpunkte direkt mit Kriterien (in der Anwendung Präferenzen genannt) bewertet. Diese Bewertung wurde von uns »nach Gefühl« vorgenommen.

Die Bewertungen werden aufgrund der in Abschnitt 3.3.4 kurz angesprochenen Datenschutzbedenken nicht direkt veröffentlicht, sondern als ein Mittelwert. Daher wird auch nicht direkt der in 3.5 vorgestellte Nearest Neighbour Algorithmus verwendet. Das Prinzip ist zwar das selbe, jedoch gibt es für jede Klasse (entspricht Firma) nur einen Trainingsdatensatz. Somit werden diese nur aufsteigend nach den Abständen sortiert.

4.4.1 Semantische Benutzeroberflächen

Da Semantic Web Anwendungen meist datenorientierter als herkömmliche Webanwendungen sind, sollten die Userinterface Möglichkeiten ebenfalls neu überdacht werden. Für den Semantic Internship Advisor wird deshalb ein neues HTML5 Slider-Element verwendet um die Präferenzen einzustellen.

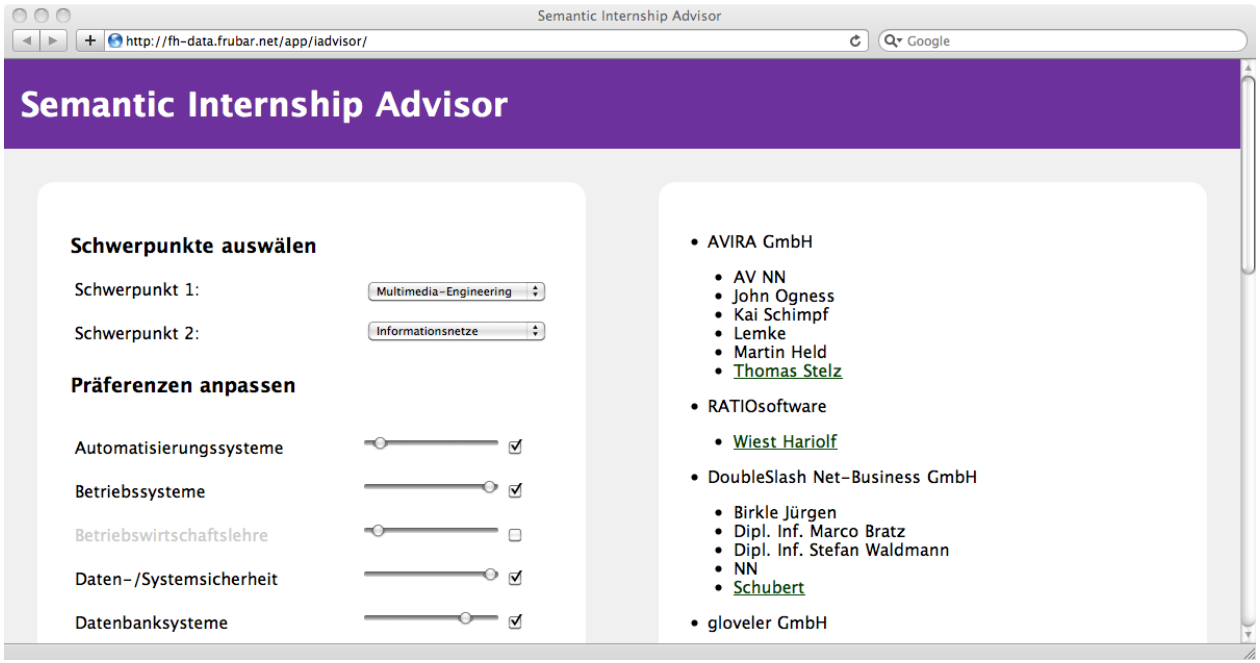

Abbildung 8: *Screenshot der Anwendung*

```
1 <input type="range" />
```

Listing 8: *HTML 5 Slider*

Eine ausführlichere Betrachtung des Themas bietet [Aufr08]. Mit SMILE[27] gibt es bereits eine eigene Widget-Bibliothek für das Semantic Web.

Eine sehr frühe Demo-Version unserer Anwendung ist unter `http://fh-data.frubar.net/app/iadvisor/` zu finden.

4.4.2 Mögliche Erweiterungen

Maps Es könnte z.B. aus den vorhandenen Adressdaten direkt ein Map-Widget eingebunden werden. APIs für diesen Zweck gibt es sowohl vom freien OpenStreetMaps (siehe Abbildung 9) als auch von Google.

Weitere Firmeninformationen Es wäre denkbar über Freebase oder DBPedia (2.6.2) Informationen wie »Anzahl der Mitarbeiter« abzufragen. Dabei gibt es nur ein Zuordnungsproblem. Gemäß dem Linked Data Ansatz müssten unsere RDFs eigentlich ein *owl:sameAs* mit einer Referenz auf den Eintrag in z.B. DBPedia haben. Da wir diese Informationen jedoch nicht manuell nachtragen möchten, würde sich hier eine Suche nach Firmenname anbieten. SPARQL hilft dabei jedoch nicht, denn eine Freitextsuche ist hier (zumindest im

[27] `http://www.simile-widgets.org/`

Allgemeinen) nicht möglich. Es gibt jedoch Dienste wie http://lookup.dbpedia.org die so etwas ermöglichen.

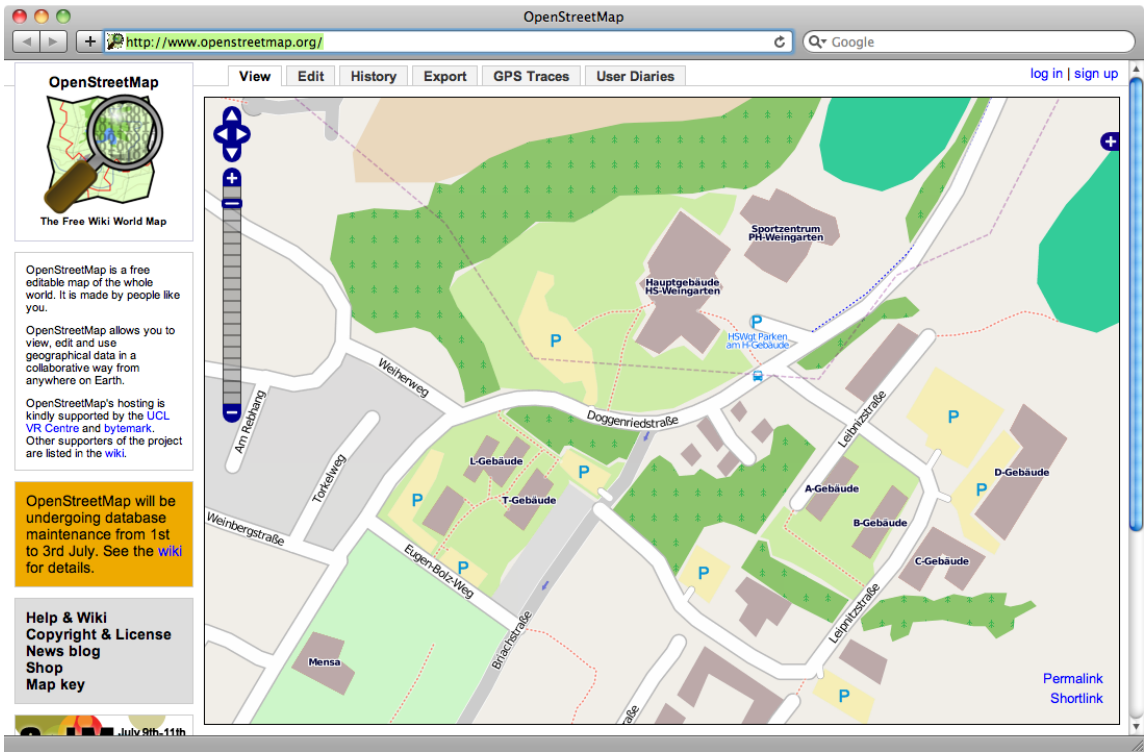

Abbildung 9: *OpenStreetMap Beispiel*

5 Zusammenfassung

5.1 Fazit

Die Erfahrungen mit der vom Team erstellten Anwendung zeigt deutlich, dass es zwar ungewohnt, jedoch nicht allzu kompliziert ist, Daten semantisch aufzubereiten und zu verarbeiten. Anwendungsentwicklung mit Semantic Web Techniken ist einfach und kann viel Spaß bereiten. Alle Projektteilnehmer haben viel gelernt.

Positiv überrascht wurden wir durch die Flexibilität und Kooperationsbereitschaft verschiedener Instanzen. Hierbei möchten wir uns noch einmal beim Rechenzentrum der HS-Weingarten und Marius Ebel vom INKIDU-Team für die Zusammenarbeit bedanken. Das Rechenzentrum hat uns den, für die Entwicklung den nötigen (virtuellen) Server, bereitgestellt. Hierfür möchten wir uns ebenfalls bedanken.

Aufgrund des kurzen zeitlichem Rahmens dieser Arbeit konnten leider nicht alle Ziele erreicht werden. Auf die Daten des LSF System mussten wir leider verzichten, da der Export vom Rechenzentrum so kurzfristig nicht umgesetzt werden konnte.

Ebenso konnten wir die in dieser Arbeit erstellten Daten nur über Umwege veröffentlichen. Die sehr restriktiven Firewall-Regeln des Rechenzentrums erschwerten das Bereitstellen der Anwendung unter der Internet-Adresse der Hochschule und war nur über einen externen Proxy möglich.

5.2 Ausblick

Ein Ziel für zukünftige Semantic Web Projekte an der Hochschule Ravensburg-Weingarten muss sein, das Angebot an Daten zu erhöhen und diese bereitzustellen. Vor allem sollten Daten direkt aus dem LSF exportiert werden, da es sich bei dieser Anwendung um die zentrale Dokumentationsquelle aller relevanten Daten der Organisationsstruktur der Hochschule handelt.
Würde die Hochschule einen zentrales System für semantische Datenverarbeitung bereitstellen, welcher z.B. unter data.hs-weingarten.de auch aus dem Internet erreichbar ist, so könnte man alle gewünschten Daten semantisch aufbereitet der Öffentlichkeit zur Verfügung stellen. Zudem wäre die Hochschule hier ein Vorreiter in der Nutzung von semantischen Web-Techniken und dies würde langfristig dazu führen, dass zum einen weitere Forschungsprojekte, gegebenenfalls auch aus der Industrie, entstehen, welche auch Geld für selbige mitbringen. Zudem würde es den Ruf der Hochschule, gerade im Bereich Informatik nochmals erhöhen und langfristig auch dafür sorgen, dass andere Hochschulen und Universitäten in Daten semantisch veröffentlichen. Aus diesen Daten, welche dann deutschlandweit von allen Hochschulen existieren ließen sich dann Vergleiche anstellen, Rankings generieren uvm. Den Möglichkeiten sind hier sogut wie keine Grenzen gesetzt.
Ein weiterer Aspekt dieser Veröffentlichung wäre, dass dann jeder auf Basis dieser Daten eigene Anwendungen erstellen kann, die z.B. den Studenten das Studium vereinfachen.

Das Projekt ist als OpenSource Projekt im Web[28] verfügbar. Interessierte können an

[28] https://frucman.frubar.net/info-xml/

dem Projekt auch nach Ablauf der Projektphase teilhaben.

Glossar

Academic Institution Internal Structure Ontology (AIISO)
Ontologie für die Beschreibung der Struktur akademischer Einrichtungen

Character Separated Values (CSV)
Format bei dem die einzelnen Elemente durch spezielle Zeichen, z.B. Strichpunkte, getrennt werden

DARPA Agent Markup Language (DAML)
Auszeichnungssprache für das Semantic Web

Dublin Core (DC)
Sammlung von Metadatenelementen zu Beschreibung von Dokumenten/Ressourcen

Erweiterte Backus-Naur-Form (EBNF)
Formale Metasyntax zur Darstellung kontextfreier Grammatiken

Frame Logic (F-Logic)
Formale Sprache zur Formulierung von Ontologien

INKIDU
Inkidu ist ein an der HS Ravensburg-Weingarten entwickeltes Internet Umfragesystem

Knowledge Organization Systems (KOS)
Ein KOS oder Begriffssystem fasst Begriffe oder Klassen aus abgetrennten System zusammen.

Lehre, Studium, Forschung (LSF)
Verwaltungsportal für Hochschulen mit Stundenplänen, Prüfungsplänen uvm.

Microformats
Microformats ist ein Markup-Format für die semantische Beschreibung von (X)HTML Webseiten

Ontologie
Menge von Begrifflichkeiten und deren Beziehungen für eine bestimmte Domäne

Ontology Inference Layer (OIL)
Webbasierte Auszeichnungssprache für das Semantic Web

Pattern-Matching
Sucht nach einem bestimmten vorgegebenen Muster in einer diskreten Struktur

RDF-Schema (RDFS)
Erweitert RDF und formalisiert einfache Ontologien für eine Domäne

Resource Description Framework (RDF)
Framework für die Beschreibung von Objekten (Ressourcen) und deren Eigenschaften in Triples

RESTful
Erfüllt die Anforderungen an einen REST Webservice first

Simple Knowledge Organization System (SKOS)
Auf RDF basierende Sprache zur Formalisierung von Klassifikationen und Thesauri

SPARQL Protocol and RDF Query Language (SPARQL)
Graph-basierte Abfragesprache für RDF

Uniform Resource Identifier (URI)
Identifikator um jegliche Ressource im WWW eindeutig zu identifizieren

Web of Trust (WOT)
Mit Hilfe einer Community wird ein Vertrauensnetzwerk aufgebaut, um Webseiten und Inhalte als nicht gefährlich einzustufen und um Personen zu authentifizieren.

Web Ontology Language (OWL)
Beschreibt Zusammenhänge zwischen Termen in RDF

World Wide Web Consortium (W3C)
Standardisierungsgremium für das World Wide Web

XML
eXtensible Markup Language, Auszeichnungssprache für hierarchische Daten

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Literatur

[Bern01] Tim Berners-Lee, James Hendler, Ora Lassila «*The Semantic Web: a new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. In: Scientific American, 284 (5), S. 34–43, May 2001*»http://www.scientificamerican.com/article.cfm?id=the-semantic-web

[Sega09] Toby Segaran, Colin Evans, and Jamie Taylor «*Programming the Semantic Web*»

[Ertel08] Wolfgang Ertel «*Grundkurs Künstliche Intelligenz* »http://www.hs-weingarten.de/~ertel/kibuch

[XML03] H. Wittenbrink, W. Köhler, O. Bergmann et al. «*XML. SPC TEIA Lehrbuch Verlag, Berlin, 2003. ISBN 3-935539-69-X*»

[Aufr08] Michael Aufreiter «*Informationsvisualisierung und Navigation im Semantic Web: Eine Analyse bestehender Visualisierungstechniken im Hinblick auf Eignung für das Semantic Web* »http://quasipartikel.at/wp-content/uploads/2009/05/informationsvisualisierung_im_semantic-web1.pdf

Weblinks

[RDF-Prim] Frank Manola, Eric Miller «*RDF Primer W3C Recommendation 10 February 2004, besucht am 29.06.2010* »http://www.w3.org/TR/rdf-primer/

[WATSON] «*WATSON Ontologie Suchmaschine für das Semantic Web, besucht am 29.06.2010* »http://watson.kmi.open.ac.uk/

[W3SRDF] «*W3Schools Tutorial zu RDF, besucht am 29.06.2010* »http://www.w3schools.com/rdf/rdf_main.asp/

[ONTOPAT] «*Ontology Design Patterns* »http://ontologydesignpatterns.org/

[W3CQALD] Tim Berners-Lee «*Linked data: It's is not like that; it's like a bag of potato chips.* »http://www.w3.org/QA/2010/05/linked_data_its_is_not_like_th.html

[WPSEMWEB] «*Wikipeida: Semantisches Web* »http://de.wikipedia.org/wiki/Semantisches_Web

[LINKEDATA] «*A home for, or pointers to, resources from across the Linked Data community* »http://linkeddata.org/

[Weichs06] Markus Weichselbaum «*Bayesian Rating - how to implement a weighted rating system* »http://www.thebroth.com/blog/118/bayesian-rating

[Kred07] Heinz Kredel «*Metadaten: DC, RDF und OWL 2007, besucht am 29.06.2010* »http://krum.rz.uni-mannheim.de/inet-2007/sess-308.html

[OWL09] Michael K. Smith,Chris Welty, Deborah L. McGuinness «*OWL Web Ontology Language Guide, Deutsche Übersetzung, besucht am 29.06.2010* »http://www.semaweb.org/dokumente/w3/TR/2004/REC-owl-guide-20040210-DE.html