Begriffe

PIM

Die Abkürzung steht für Personal Information Manager und ist eine Software, welche persönliche Daten verwaltet. Darunter zählen Kontakte, Termine und Aufgaben, aber auch Briefe, Faxe, etc. pp. Eine PIM-Software muss dabei sowohl benutzerfreundlich gestaltet sein, als auch große Freiheiten im Bezug auf die Nutzung gewähren. Jeder möchte andere Datenfelder nutzen oder auch seine Daten synchronisieren.

Authenifizierung

Die Authentifizierung ist ein Vorgang bei die Identität von jemandem, beispielsweise eine Person, überprüft wird. Eine einfache Form einer Authentifizierung ist das Einloggen bei einem *Social Network*.

Ontologie

Mit einer Ontologie wird eine Menge von Informationen bezeichnet, welche in einer strukturierten Form vorliegen. Im Gegensatz dazu bildet eine Taxonomie nur eine hierarchische Untergliederung. Neben den Informationen an sich gibt es noch Regeln, welche die Beziehungen beschreiben.

Netzwerkschicht

Sie wird auch als Bitübertragungschicht bezeichnet und hat die Aufgabe mechanische und elektronische Hilfsmittel für die Steuerung von physikalischen Verbindungen zu Verfügung zu stellen. Das für uns wichtige ist noch, dass hier das Übertragungsmedium rein fällt: Internet (per WLAN oder Ethernet), per Funk oder Bluetooth.

Resource

Dieser Begriff wird je nach Gebiet mit anderen Bedeutungen belegt. Im Zusammenhang mit URI's und RDF ist eine Resource eine Webseite, Datei, ein Aufruf von Webservices, aber auch z.B. ein E-Mail-Empfänger. Sie kann über eine URI eindeutig identifiziert werden.

URL / URI

Ein URI ist ein eindeutiger Bezeichner für eine Resource. Dabei kann man eine bestimmte Internetseite meinen oder eine E-Mail-Adresse. Vielen ist in dem Zusammenhang nur der Begriff URL geläufig. Dieser bezeichnet eine bestimmte Unterart von URI's, nämlich solche die spezielle Orte oder Pfade dahin beschreiben.

RDF

Der Begriff *Resource* ist essentiell bei RDF. Dies ist wie oben im Absatz *URI* schon beschrieben, alles, was man mittels einer URI benennen kann. Die oben angesprochenen Tripel bilden die strukturelle Grundlage eines RDF-Ausdruckes.
Damit legt *RDF* die syntaktische Grundlage fest, auf der dann die vorliegenden Daten interpretiert werden. Dies geschieht dann wiederum über RDF-S (siehe Konzepte).

Domäne

Unter einer Domäne versteht man ein Wissensgebiet innerhalb eines Fachbereichs. Das Wissen ist in Themen und Begriffe aufgeteilt und wohl strukturiert. Ein Beispiel wäre die Informatik.

OntoWiki

OntoWiki ist eine Plattform zur agilen Verwaltungen von Semantic Web Wissensbasen. Es ermöglicht die visuelle Präsentation einer Wissensgrundlage als Informationsdiagramm, mit verschiedenen Blicken auf Fallinformationen. Es ermöglicht intuitive Autorisierung von semantischem Inhalt mit einem Inline-Editiermodus für die Bearbeitung von RDF Inhalt, ähnlich wie WYSIWIG für Textdokumente. Es fördert die sozialen Mitarbeitsaspekte durch zählen der Änderungen, die Kommentierfähigkeiten und das diskutieren jedes einzelnen Punktes einer Wissensgrundlage, es ermöglicht die Popularität des Inhaltes zu bewerten und abzuschätzen und die Aktivitäten der Nutzer zu akzeptieren.

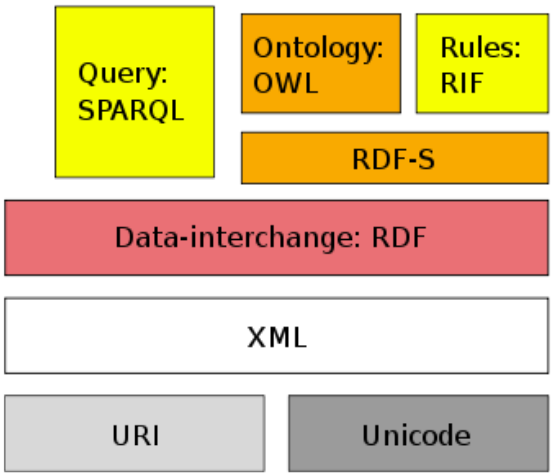
Unicode

Unicode wurde mit dem Ziel entwickelt die unterschiedlichen und inkompatiblen Kodierungen in verschiedenen Ländern oder Kulturkreisen beseitigen zu können. In der deutschen gibt es z.B. das ä,ö oder ß, in der englischen Kodierung wiederum nicht. Würde man nun beispielsweise ein ä in einer englischen Kodierung darstellen wollen, würden nur Hieroglyphen angezeigt werden. Unicode bietet nun die Möglichkeit diesen Missstand aufzuheben.

Konzepte

Semantic-Web

Der Begriff ist an sich etwas irreführend. Zerlegen wir ihn erstmal und betrachten seine Einzelteile. Fangen wir mit dem *Web* im Namen an. Dies bezieht sich auf das WorldWideWeb, kurzum dem Internet. Jeder kennt es und nutzt es, sei es in Form von sozialen Netzwerken (StudiVZ) oder zum Suchen von Informationen. Es dient zudem noch als Medium um Informationen von A nach B zu senden.



(siehe Bild 1 – Verweis)

Kommen wir nun zu dem *Semantic*. Die Semantik, so haben wir früher gelernt, beschreibt den Inhalt einer Zeichenkette. (Die Syntax war das mit der Struktur) Je nachdem in welchem „Zusammenhang“ man sich gerade befindet kann vorliegende Information anders interpretiert werden. Der Satz „Möchtest du mit mir gehen?“ illustriert das gut. Er kann einerseits bedeuten, dass man jemanden, den man mag, zu einer Beziehung anregen möchte, aber auch, dass man jemanden einfach als Begleitung erfragt.
Die Motivation hinter dem *SemanticWeb* besteht nun darin, die Elemente (Texte, Bilder, etc. pp.) auf Internetseiten mit zusätzlichen Informationen an zureichern und ihnen somit eine Bedeutung zu geben. Dies geschieht mit einer Auszeichnungssprache und ermöglicht aus dem vorliegenden Haufen von Worten und Bildern inhaltliche Schlüsse zu ziehen und Beziehungen untereinander zu erkennen. Ein weiterer Grund ist die automatisierte Erfassung solcher Seiten durch Maschinen. Sie können mit den meisten Internetseiten nicht viel anfangen, da sie die vorliegenden Elemente (Texte, Bilder) keinem bestimmten Kontext zuordnen können. Aber mithilfe der oben erwähnten zusätzlichen Informationen ist es für sie möglich bestehendes Wissen in Kontexte einzuordnen und mit anderem Wissen zu verknüpfen.

URI – Uniform Resource Identifier

Eine URI ist ein eindeutiger Bezeichner zu einer bestimmten Resource. Eine Resource kann zum Beispiel eine Webseite sein, wie *http://www.beispiel.de.* Das *http* steht für eine bestimmte Schemata und www.bespiel.de beschreibt den Pfad. Es gibt noch weitere URI-Schemata, unter anderem für den E-Mail-Versand (mailto) oder einen Dateiupload (ftp). Sie alle sind anders strukturiert und werden je nach Art anders von einem Programm interpretiert.
Vielen ist in dem Zusammenhang nur der Begriff URL geläufig. Dieser bezeichnet eine bestimmte Unterart von URI's, nämlich solche die spezielle Orte oder Pfade dahin beschreiben.

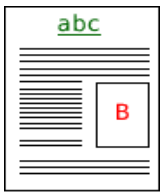
XML und RDF

Allgemeines

2001 schrieb Berners-Lee et al. in einem Artikel des Scientific American: *das Semantische Web ist ein Erweiterung des herkömmlichen Webs, in der Informationen mit eindeutigen Bedeutungen versehen werden, um die Arbeit zwischen Mensch und Maschine zu erleichtern.*

Um Informationen einer Webseite mit Bedeutungen versehen zu können benötigt man neben HTML, welches die Struktur der Seite beschreibt, noch die Auszeichnungssprache RDF, welche wie HTML auf XML basiert. RDF ist ein Akronym für Resource Description Framework und basiert auf sogenannten *triples*, *Konstrukte* der Form Subjekt-Prädikat-Objekt.

Nehmen wir als Beispiel mal eine Internetseite: **abc** (*abc ist Text*) wäre hier nicht nur eine Überschrift, sondern auch der Titel der Internetseite (*abc ist Titel von Internetseite*). Oder das Element **B** ist nicht nur ein Bild, sondern eines welches den Autor der Internetseite darstellt.



Ein Objekt kann aber wiederum selbst wieder ein Subjekt in einem anderen *triple* sein. Damit lassen sich beliebig komplexe Ontologien aufbauen.

Ein Beispiel

Mit folgenden Tripeln kann man ausdrücken, dass <foo bar > einen bestimmten Titel und Autor hat (genauer in N3; das *has* dient dem Verständnis):

<foo bar > has <http://purl.org/dc/elements/1.1/title> "Zwei Wörter für Beispiele" .
<foo bar> has <http://purl.org/dc/elements/1.1/publisher> "Konrad".

In konkretem Code sieht dass dann so aus:

```
<?xml version="1.0" encoding="UTF-8" ?>

<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 rdf:about="foo bar">
                <dc:title>Zwei Wörter für Beispiele</dc:title>
                <dc:autor>Konrad</dc:autor>
        </rdf:Description>

</rdf:RDF>
```

Das erinnert stark an HTML-Code, genauer gesagt ist das XML-Code, welcher mit RDF ausgestattet wurde. Auf diese Weise kann man alle Elemente eines XML-Dokumentes ausstatten und eigene Ontologien erstellen. Um solche Strukturen abfragen zu können kann man *SPARQL* nutzen, welches als nächstes vorgestellt wird.

RDF-S

RDF wird für die Erstellung einer Syntax genutzt, um eine gemeinsame Datenbasis zu haben. Um die dann vorliegenden Daten zu interpretieren können benötigt man ein Schema (auch Vokabular genannt). Dies ist aber nicht vordefiniert, sondern kann je nach Anwendungsfall anders aufgebaut und strukturiert werden. Dabei wird die Scheme-Definition Language genutzt. In ihr wird u.a. auch für jede Eigenschaft definiert, welche Werte sie annehmen kann, welche erlaubt sind, was diese jeweils für eine Bedeutung haben, welche Beziehungen zu anderen Eigenschaften bestehen und welche Arten von Ressourcen eine bestimmte Eigenschaft verwenden darf.

Möchte man beispielsweise Elemente die zu einer gemeinsamen Gruppe gehören, wie etwa Bücher, mit gleichen Eigenschaften zu versehen, benötigt man die RDF-S (RDF-Schema). Sie bietet erst die Möglichkeit Elemente untereinander semantisch in Verbindung zu setzen.

In RDF-S liegt das mengen theoretische Klassenmodell zugrunde, wobei die Klassen separat von den Eigenschaften modelliert werden.

Ein Beispiel

```
<rdf:Property rdf:about="http://www.w3.org/2000/01/rdf-schema#label">

        <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#"/>

        <rdfs:label>label</rdfs:label>

        <rdfs:comment>A human-readable name for the subject.</rdfs:comment>

        <rdfs:domain rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>

        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>

</rdf:Property>
```

Dieser Code-Block definiert eine Eigenschaft (property) und beschreibt u.a. ihre Bezeichnung (label) und ein Kommentar dazu (comment).

Aufbauend auf RDF-S gibt es *OWL*, welche die Sprache RDF-S um weitere Konstrukte erweitern soll, um die Ausdrucksstärke zu erhöhen.

SPARQL

SPARQL ist eine Abfragesprache für RDF. Sie ähnelt von der Syntax her der Datenbankabfragesprache SQL. Nachdem sie abgesetzt wurde erhält man als Ergebnis eine tabellenartige Struktur.

Beispielabfrage (aus wikipedia):

```
PREFIX abc: <http://example.com/exampleOntology#>

SELECT ?capital ?country

WHERE
{
        ?x abc:cityname ?capital.
        ?y abc:countryname ?country.
        ?x abc:isCapitalOf ?y.
}
```

Gehen wir der Reihe nach durch. Die erste Zeile (mit dem PREFIX) bedeutet, dass man eine Variable anlegt und mit einem Wert belegt: abc wird der Wert *http://example.com/exampleOntology#* zugewiesen. In der Zeile mit dem *SELECT* schreibt man alle das rein was man aus den Datenbeständen auswählen möchte. In dem Beispiel wäre das der Stadtname und das Land.

Der *WHERE*-Block beinhaltet alle Bedingungen, die ein Element erfüllen muss, um in die Ergebnismenge aufgenommen zu werden. Das Beispiel enthält ?x und ?y, 2 Variablen für das Konstruieren der Bedingung. Das ?x repräsentiert dabei die Stadt und ?y ein Land.

Die Bedingung fordert folgendes:

Selektiere mir alle Städte (?capital) und alle Länder (?country), wo gilt:

- gerade ausgewählte Stadt ist gleich der Stadt ?x,
- gerade ausgewähltes Land ist gleich ?y,
- und die Stadt ?x ist die Hauptstadt von ?y.

Als Ergebnis wird beispielsweise geliefert:

capital	**country**
Moskau	Russland
Berlin	Deutschland
...	...

OWL

Die Abkürzung OWL steht für Web Ontology Language, obwohl sie eigentlich WOL lauten müsste. Die Ursache für diesen Verdreher ist unter *Verweis 1* in den Quellen zu finden. OWL basiert auf der *RDF-S*-Syntax und erweitert diese, indem beispielsweise Sprachkonstrukte eingeführt werden, die Ausdrücke ähnlich der Prädikatenlogik ermöglichen. Weiterhin wird das Klassenkonzept mit der Vererbung und Spezialisierung unterstützt und erweitert dieses um damit komplexere Beziehungen zu modellieren.

Ein Beispiel der Wikipedia etwas aufbereitet (Beispiel 1 - Verweis)

Das Beispiel beschreibt die Konzepte <Person>, <Gender> und <Woman>. Eine Frau ist definiert als eine <Person>. Sie besitzt die Eigenschaft <gender>, welche den Wert <female> hat. Die Eigenschaft muss außerdem der Klasse <Gender> angehören. Die Instanz <Supertoll> ist somit als <Person> beschrieben eine Frau (<Woman>).

OWL und RDF-S-Code:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" [ ... ] >
        <owl:Class rdf:ID="Gender"/>    * 1 *
        <owl:Class rdf:ID="Person"/>

        <owl:Class rdf:ID="Woman">
                <rdfs:subClassOf rdf:resource="#Person"/>          * 2 *
                <owl:equivalentClass>
                        <owl:Restriction>
                                <owl:onProperty rdf:resource="#Gender"/>                    * 3 *
                                <owl:hasValue rdf:resource="#female" rdf:type="#Gender"/>
                        </owl:Restriction>
                </owl:equivalentClass>
         </owl:Class>
        <owl:ObjectProperty rdf:ID="gender" rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty"> [ ... ]
        </owl:ObjectProperty>
        <owl:DatatypeProperty rdf:ID="name" rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty"> [ ... ]
        </owl:DatatypeProperty>

        <owl:DatatypeProperty  rdf:ID="firstname"
                               rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
                <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>          * 4 *
                <rdfs:domain rdf:resource="#Person"/>
        </owl:DatatypeProperty>
        <Person rdf:ID="Supertoll" firstname="Gudrun" name="Superfrau">
                <Gender rdf:resource="#female"/>                                    * 5 *
        </Person>
</rdf:RDF>
```

Erläuterungen:

*** 1 *** Es werden 2 Klassen ohne Inhalt erstellt: *Gender* und *Person*.

*** 2 *** Dann wird eine Klasse *Woman* erstellt. Diese ist abgeleitet von der Klasse Person, was mit dem Code <***rdfs:subClassOf rdf:resource="#Person"***/> ausgedrückt wird.

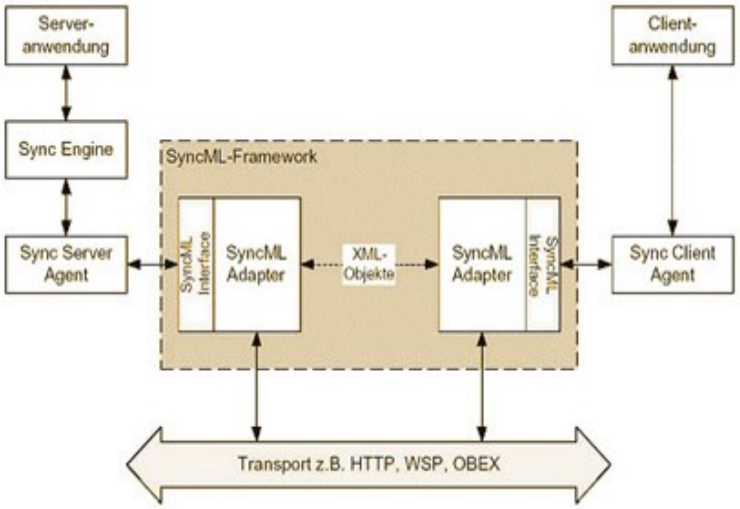
*** 3 *** Die Klasse *Woman* bekommt nun die Eigenschaft Gender zugewiesen und deren Wert wird auf *female* gesetzt. Dies ist eine „Einschränkung“, was mit dem <***owl:Restriction***> ausgedrückt wird.

*** 4 *** Hier wird eine neue Eigenschaft namens *firstname* erstellt. Hier wird die *range* und *domain* definiert. Letzteres ordnet die Eigenschaft der Klasse *Person* zu, womit jede Instanz von Person oder abgeleiteter Klassen ebenfalls diese Eigenschaft besitzen.

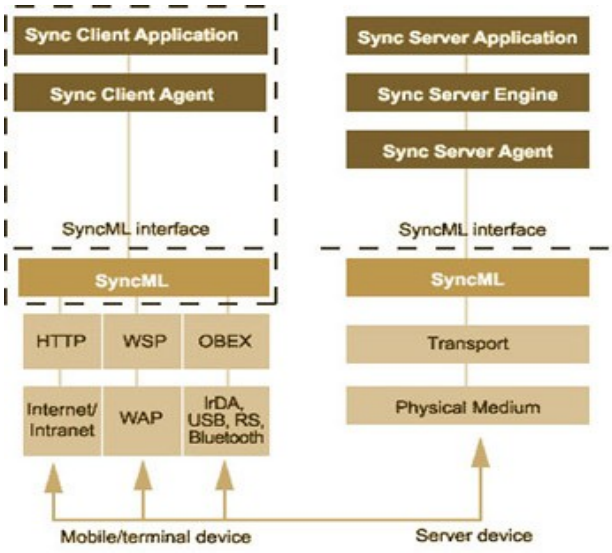
*** 5 *** Nun wird eine konkrete Person angelegt. Sie besitzt einen Vor- und Nachnamen (firstname, name) sowie ein Geschlecht (gender). Wie man nun sieht, wird in der Person direkt der Name und der Vorname gesetzt, wo hingegen das Geschlecht extra definiert wird.

SyncML

Die Abkürzung SyncML steht für Synchronization Markup Language und basiert auf XML. Sie ein Plattform-unabhängiger Standard zur Datensynchronisation zwischen Computern. Bei den Daten kann es sich um beliebige Informationen handeln, Kontakte, E-Mails, etc. pp.
Es wird keine Plattform vorgeschrieben und die Netzwerkschicht (*TCP/IP*, *WSP oder Obex*) ist auch egal, womit man Daten sowohl per PC, als auch mittels einen mobilen Endgerätes (Handy, PDA) synchronisieren kann.



(Bild 2 – Verweis)



(Bild 3 – Verweis)

Wie bei einer Synchronisierung üblich werden die zu synchronisierenden Daten abgeglichen und es wird ermittelt, welche letztendlich synchronisiert werden müssen. Mithilfe von Nachrichten tauschen sich Server und Client aus. Der Client initiiert normalerweise den Start einer Synchronisation.

Die angesprochenen SyncML-Nachrichten ähneln in ihrer Struktur der E-Mail-Nachricht. Es gibt einen Header, welcher Empfänger und Senderinformationen enthält, sowie für den Server eindeutige Synchronisations-IDs. Dem Kopf folgen die Synchronisationsbefehle zum Hinzufügen, Löschen und Ersetzen von Daten.

Beschreibung der zu studierenden Applikation

So wie es im ersten Kundengespräch ersichtlich wurde, wird ein Modul benötigt, welches eine Synchronisation von PIM-Daten zwischen einem Client und Server ermöglicht. Der Client (ob PC, Handy, PDA etc. pp.) soll frei wählbar sein, der Server wird durch ein OntoWiki-System gestellt.

Es wird gefordert, dass sich der Client mit vorher auf dem Server (OntoWiki) hinterlegten Anmeldedaten am Server authentifiziert und eine Synchronisation erbittet. Nachdem eine erfolgreiche Authentifizierung erfolgte, wird die Synchronisation gestartet. Dabei muss eine Synchronisierung in beide Richtungen möglich sein, dass heißt, dass man Daten sowohl hoch laden, als auch herunterladen kann. Es wird dabei mittels SyncML kommuniziert.

Das zu schreibende Modul setzt beim Erbitten der Synchronisation an. Wir stellen die Authentifizierung und kümmern uns dann um den Abgleich der Daten. Die Anfragen, die in SyncML gestellt werden, müssen dann in SPARQL umgesetzt und an die API des OntoWiki weitergereicht werden. Neue Daten müssen wieder in SyncML umgesetzt und dann an den Client gesendet werden.

Weiterhin arbeiten wir nicht direkt auf den Datenbeständen sondern nur mittels SPARQL auf der API. Anfragen und Befehle werden über SPARQL abgegeben. Dies ist nötig, da der Datenbestand wechseln kann, also beispielsweise von einer MySQL-Datenbank zu einem Virtuoso.

Uns wurde mitgeteilt, dass es bereits einen geeigneten SyncML-Server gibt, den wir aber noch inhaltlich prüfen müssen, um zu sehen, was wir davon nutzen können.

Quellen, Bilder und Literatur

- *http://de.wikipedia.org/wiki/Semantic_Web*
- *http://de.wikipedia.org/wiki/Uniform_Resource_Identifier*
- *http://de.wikipedia.org/wiki/Resource_Description_Framework*
- *http://www.w3.org/TR/rdf-sparql-query/#basicpatterns*
- *http://de.wikipedia.org/wiki/SPARQL*
- *http://books.google.de/books?id=lwa7TuJ_RR8C&printsec=frontcover&source=gbs_summary_r&cad=0 (S. 203)*
- *Verweis 1: http://lists.w3.org/Archives/Public/www-webont-wg/2001Dec/0169.html*
- *Bild 1 – Verweis: http://de.wikipedia.org/w/index.php?title=Datei:SW_layercake_2006.svg&filetimestamp=20070623222219*
- *Beispiel 1 – Verweis: http://de.wikipedia.org/wiki/Web_Ontology_Language#Beispiel*
- *Verweis 2: http://books.google.de/books?id=cGyHOzDyCnMC&pg=PA187&lpg=PA186&dq=syncml*
- *Bild 2 – Verweis: http://de.wikipedia.org/w/index.php?title=Datei:Komm.jpg&filetimestamp=20060614193058*
- *Bild 3 – Verweis: http://de.wikipedia.org/w/index.php?title=Datei:Schema.jpg&filetimestamp=20071107185113*