

Semantic Web Content Management

Norman Beck

Mellow Message Medienproduktion GmbH
Riemannstr. 29b, 04107 Leipzig, Germany
E-mail: beck@mellowmessage.de

Sören Auer

Universität Leipzig
Fakultät für Mathematik und Informatik
Anwendungsspezifische Informationssysteme
Augustusplatz 10-11, 04109 Leipzig, Germany
E-mail: auer@informatik.uni-leipzig.de

Kurzfassung

Im Rahmen der Semantic Web Initiative des W3C sind in jüngster Vergangenheit die RDF-basierten Standards RDFS und OWL verabschiedet worden, die als Wissensrepräsentationssprachen gemäß dem Ontologie-Paradigma im Web fungieren. Aufbauend auf diesen Technologien wird in dieser Arbeit gezeigt, wie im Zusammenspiel mit einem Ontologie-Editor die Grundlage für ein wissensbasiertes Web Content Management geschaffen werden kann. Dabei finden die üblichen Paradigmen des Web Content Managements Berücksichtigung und Vorteile der Modellierung von Wissensbasen gegenüber einem relationalen oder auch XML basierten Inhaltsmodell werden genutzt. Dieser Ansatz leistet damit einen Beitrag, Web Content Management vom reinen Informationsmanagement zum Wissensmanagement weiterzuentwickeln.

1 Einleitung

Mit dem Aufkommen zahlreicher Content Management Systeme, die sich insbesondere im Bereich des Web Content Managements zunehmender Beliebtheit erfreuen, sind in den vergangenen Jahren einige Konzepte zur Ablage der zu verwaltenden Daten entstanden. Die verschiedenen Systeme setzen hierbei vorzugsweise auf die Speicherung der Rohdaten in einem relationalen Datenbanksystem, um damit eine strikte Trennung zwischen Inhalt, Layout und Struktur gewährleisten zu können. Die Vorteile dieser Vorgehensweise liegen einerseits in der Einfachheit der Implementierung und andererseits in der hohen Akzeptanz im Bereich datenbankgestützter Software.

Zur Verbesserung der Interoperabilität sind auch im Bereich des Content Managements Bestrebungen entstanden, einheitliche Schnittstellen für Import- und Exportfunktionalitäten zur Verfügung zu stellen. Als Austauschformat wird dabei auf XML zurückgegriffen [22]. Für die Serialisierung der Inhalte nach XML kommen hierbei entweder Mapping-Tools zum Einsatz, die es erlauben, die Rohdaten aus der Datenbank in ein valides XML-Dokument zu transformieren oder die Daten werden unabhängig von einem Datenbanksystem als XML-Dateien im Dateisystem abgelegt. Neuere Systeme unterstützen sogar die Ablage der Daten in einer nativen XML-Datenbank. All diese Ansätze haben den gemeinsamen Vorteil, dass letztendlich mit einem standardisierten Format Daten zwischen den verschiedenen Systemen ausgetauscht werden können. In der Realität werden hier aber sehr oft Grenzen erreicht, insbesondere dann, wenn das Verständnis der übermittelten Daten nicht auf beiden Seiten identisch ist. Im Sektor des E-Business sind für solche Anwendungsfälle bereits erste Lösungen (Business Frameworks, z.B. ebXML [21]) verfügbar, welche ein gemeinsames Vokabular und somit auch ein gemeinsames semantisches Verständnis über die Geschäftsdaten und Geschäftsprozesse anbieten.

Eine weitere Möglichkeit, die Interoperabilität zwischen verschiedenen Content Management Systemen zu gewährleisten, besteht in der Verwendung der Standards des *Semantic Web*, einer Initiative des *World Wide Web Consortiums* (W3C). Die Standards und Techniken, die unter dem Oberbegriff *Semantic Web* subsumiert sind, erlauben Maschinen die semantische Referenzierung und Beschreibung von Inhalten, die weltweit eindeutig identifizierbar sind. Die Standards basieren auf der eXtensible Markup Language XML.

In der vorliegenden Arbeit wird erläutert, inwiefern die Semantic Web Standards RDF/S und OWL im Web Content Management eingesetzt werden können, um eine möglichst einfache Interoperabilität zwischen verschiedenen Systemen zu erhalten. Dabei sollen insbesondere die Aspekte der Wiederverwendbarkeit von Web Content und das Integrieren heterogener Datenquellen Beachtung finden. Nach einem kurzen Überblick über Wissensmodellierung mit RDF/S und OWL sowie deren gegenwärtige Einsatzszenarien beschäftigt sich Kapitel 3 mit der Repräsentation von Web Content durch Ontologien. Hier soll erläutert werden, wie der Inhalt der zu verwaltenden Webseiten und deren gesamte Struktur unter Einhaltung der üblichen Paradigmen (sinnvolle Trennung von Inhalt, Layout und Struktur sowie die Abbildung des Content Life-cycles) mit dem Semantic Web Ansatz zur ontologiebasierten Wissensrepräsentation verwaltet werden können. Dabei soll auch ein bestehender webbasierter Ontologie-Editor vorgestellt werden. Anschließend wird der wissensbasierte Ansatz mit seinen Vor- und Nachteilen diskutiert.

2 Wissensmodellierung

Ontologien

Der Begriff *Ontologie* leitet sich vom griechischen *onta* (das Seiende) und *logos* (Lehre) her. Er bedeutet demnach *Die Lehre vom Seienden*. Der Begriff wurde ursprünglich in der Philosophie verwendet und steht dort für die Lehre von den Möglichkeiten und Bedingungen existierender Dinge. In der Informatik wird versucht, mit Ontologien das Wissen über einen bestimmten Ausschnitt der Realität darzustellen. Eine wichtige Definition für die Ontologie in der Informatik gab T. Gruber in [1]:

*An ontology is a specification
of a conceptualization.*

Gruber stellt die Ontologie damit als eine explizite formale Spezifikation einer gemeinsamen Konzeptualisierung dar. Zur Beschreibung einer

solchen Ontologie bedarf es entsprechend [17] einer Ontologiesprache, die Folgendes leistet:

1. Die Sprache kann Konzepte (Klassen bzw. Entitäten) und deren Eigenschaften beschreiben.
2. Sie kann darüber hinaus Beziehungen zwischen diesen Konzepten beschreiben.
3. Die Ontologiesprache kann Constraints (Bedingungen) über die Eigenschaften und Beziehungen beschreiben.

Das Einsatzspektrum der Ontologien in der Informatik ist breit gefächert. Nach Uschold und Gruninger [2] fokussieren Ontologien vor Allem auf die Bereiche Kommunikation, automatisches Schließen sowie Repräsentation und Wiederverwendung von Wissen. Analog zum Web Content Management wäre die Verwendung von Ontologien also einerseits in der Repräsentation und Wiederverwendung von Web Content zu sehen und andererseits in der Kommunikation also dem Import und Export von bspw. Geschäftsdaten, die semantisch interpretiert werden können. Das automatische Schließen spielt in diesem Zusammenhang eine eher geringe Rolle.

Mit Einführung der Semantic Web [7] Initiative durch das World Wide Web Consortium (W3C) ist es möglich geworden, Ontologien in Web-Umgebungen zu nutzen. Zu den Standards dieser Initiative zählen insbesondere die RDF-basierten Ontologiesprachen [3] wie RDFS und OWL, die in den nächsten Abschnitten näher erläutert werden sollen.

Resource Description Framework – RDF

Das Resource Description Framework RDF [4] ist seit Februar 1999 eine W3C-Empfehlung, die auf Vorschläge von Netscape und Microsoft zurückgeht. RDF legt ein abstraktes Verfahren fest, mit dem Zusammenhänge zwischen Entitäten und deren Eigenschaften beschrieben werden können. Die Grundlage von RDF bildet sein Datenmodell, das einerseits an keine konkrete Syntax gebunden ist, andererseits verlangt der Austausch von Metadaten aber eine solche. In diesem Zusammenhang wurde vom W3C eine RDF-Syntaxspezifikation auf Basis der eXtensible Markup Language XML entwickelt. Diese Spezifikation erlaubt die Serialisierung von RDF-Metadaten nach XML und demnach einen Austausch auf standardisierter Ebene.

RDF ist als Metasprache für Metadaten zu verstehen. Bei der Entwicklung von RDF standen dabei folgende Punkte im Vordergrund:

1. Informationen sollen besser katalogisiert werden können und Abhängigkeiten zwischen Dokumenten sollen transparenter werden.
2. Automatisierte Anwendungen sollen dazu beitragen, dass dem Anwender nur Daten im WWW angeboten werden, die er als interessant klassifiziert. Dementsprechend soll RDF die Metadaten so beschreiben, dass sie von Maschinen verstanden werden können.

Das Datenmodell des RDF ist an die grundlegende Form von Aussagen angelehnt:

S hat die Eigenschaft P mit Wert O.

Dabei wird *S* analog zu den natürlichsprachigen Grammatiken als *Subjekt*, *P* als *Prädikat* und *O* als *Objekt* bezeichnet. Wegen dieser Dreiteilung werden RDF-Aussagen auch immer wieder als *Triples* oder *SPO-Statements* bezeichnet. Die folgenden Grundtypen können als Bestandteile (Subjekt, Prädikat, Objekt) einer RDF-Aussage auftreten:

1. **Ressourcen:** Ähnlich dem objektorientierten Paradigma in Java *Alles ist ein Objekt* gilt als Motto von RDF: *Alles ist eine Ressource*. Eine Ressource kann eine beliebige Entität sowohl aus der realen Welt als auch aus dem Internet symbolisieren. Jede Ressource muss dabei durch einen eindeutigen Unified Resource Identifier (URI) global identifizierbar sein.
2. **Literale:** Literale stellen konkrete Werte (z.B. Text oder Ziffern) dar.

Ressourcen können an allen Stellen eines Triples auftauchen, Literale hingegen nur als Objekt. Eine ausführliche Beschreibung zu RDF und den zugehörigen Terminologien findet sich in [4].

RDF-Schema - RDFS

Das Datenmodell von RDF eignet sich über die Beschreibung von Metadaten hinaus auch als Basis zur Repräsentation von komplexen Zusammenhängen. Ein Nachteil an RDF allein ist jedoch, dass selbst einfachste Aussagen ohne zu-

sätzliche Informationen über die Bedeutung der verwendeten Vokabeln nicht möglich sind. Mit RDFS [5] wurde eine Terminologie definiert, die es erlaubt, eine gewisse Grundstruktur in RDF-Dokumente zu bringen. Diese Struktur besteht aus zwei Komponenten:

1. **Typisierung:** Mittels RDF allein können Ressourcen, die Eigenschaften darstellen, nicht von anderen Ressourcen unterschieden werden. Hier kommt RDFS zum Einsatz. RDFS erlaubt die Definition von *Klassen*, *Eigenschaften* und *Instanzen*. Somit kann zuerst ein Modell, d.h. eine Struktur des zu repräsentierenden Wissens geschaffen werden, das dann instanziiert wird. Eine Instanz ist dabei ähnlich der objektorientierten Programmierung ein einer Klasse zugeordnetes Objekt mit bestimmten Wertzuweisungen für ihre Eigenschaften.

1. **Einschränkungen von Eigenschaften:** Durch die Typisierung können mittels RDFS spezielle Ressourcen als Eigenschaften deklariert werden. Diese Eigenschaften können mit der Angabe von Definitions- und Wertebereichen weiter spezifiziert werden. Als Definitions- und Wertebereich von Eigenschaften dienen dabei Klassen. Eine Klasse wird also nicht mit ihren Eigenschaften definiert, sondern es werden die Argumentrahmen der Eigenschaften beschrieben, also welche Klassen als Subjekt und Objekt für eine Eigenschaft fungieren können. Das Klassenmodell von RDFS wird somit hingegen der Objektorientierung als Eigenschaft-basiert (*property-centric*) bezeichnet.

Durch seinen beschreibenden Charakter zählt RDFS zu den *schema specification languages*. In RDF-Schema werden aber nicht nur rein syntaktische Bedingungen über die Struktur eines Dokumentes festgelegt, wie es etwa bei XML-Schema der Fall ist. RDFS erlaubt darüber hinaus die Interpretation einer Aussage bspw. durch einen *Reasoner*.

Eine ausführliche Beschreibung zu RDFS und den zugehörigen Terminologien findet sich in [5].

Web Ontology Language

Die Web Ontology Language (OWL) [6] wurde als Erweiterung zu RDF/RDFS (kurz: RDF/S)

entwickelt. In OWL sind weitere Sprachkonstrukte eingeführt worden, die es erlauben, Ausdrücke in einem entscheidbaren Fragment der Prädikatenlogik 1. Stufe (Description Logic) zu formulieren. Mengenoperationen wie z.B. Vereinigung oder Durchschnitt von Klassen können hierbei auf logische Axiome zurückgeführt werden. Durch die Integration von so genannten Restriktionen ist es zusätzlich möglich geworden, detaillierte Beschreibungen von Eigenschaften in einem Modell vorzunehmen. So kann bspw. ein und dieselbe Eigenschaft verschiedene Wertebereiche für verschiedene Klassen haben.

OWL selbst ist als Schema in RDF/S definiert. Jede OWL-Ontologie ist somit auch ein gültiges RDF-Dokument, das serialisiert und ausgetauscht werden kann.

Es werden drei Unterarten von OWL nach ihrem Funktionsumfang unterschieden, die je nach erforderlicher Effizienz und Anwendungsfall zum Einsatz kommen:

1. **OWL Lite:** Die "Light-Version" wurde mit dem Ziel geschaffen, eine minimale Untermenge von OWL zu schaffen, die für Entwickler einen idealen Kompromiss zwischen Nützlichkeit und Effizienz darstellt.
2. **OWL DL:** Die OWL Description Logic (OWL DL) beinhaltet den vollständigen OWL-Wortschatz, der unter einer Anzahl einfacher Beschränkungen interpretiert wird.
3. **OWL FULL:** OWL Full beinhaltet ebenfalls den vollständigen OWL-Wortschatz. Allerdings wird dieser hier etwas umfangreicher interpretiert. Sämtliche Beschränkungen aus OWL DL entfallen. So kann bspw. eine Klasse wiederum als Instanz verwendet werden.

Eine ausführliche Beschreibung zu OWL und den zugehörigen Terminologien findet sich in [6].

Gegenwärtige Einsatzszenarien

Die Standards RDFS und OWL sind erst in jüngster Vergangenheit vom W3C verabschiedet worden und ihre möglichen Einsatzbereiche sind äußerst vielfältig. In jeglichem Bereich der Wissensrepräsentation und auch der Kommunikation können RDF-basierte Ontologien als Alternativen zu den herkömmlichen Standards gesehen wer-

den. Eine weite Verbreitung von RDFS und OWL darf daher in naher Zukunft zu erwarten sein.

Die Metasprache RDF hingegen existiert schon seit 1999 und erfreut sich seitdem zunehmender Beliebtheit. Ein bekanntes Beispiel für die Verwendung von RDF ist mit der *Dublin Core* [8] Initiative vom Online Computer Library Center (OCLC) gegeben. Dublin Core ist ein Metadatenformat zur Beschreibung von Dokumenten und anderen Objekten im Internet. Das *Dublin Core Metadata Set* besteht momentan aus 15 Elementen und kann mit verschiedenen Syntaxformaten dargestellt werden, von denen RDF eine übliche Variante ist.

Ein weiteres Einsatzbeispiel von RDF ist das RDF Site Summary (RSS) [9]. Das RSS stellt eine einfache Art der Content Syndication dar. Es wird verwendet, um Inhalte von Internetseiten in maschinenlesbarer Form bereitzustellen. Auch in Web Content Management Systemen konnte sich RSS zum Publizieren von Inhalten etablieren [18].

3 Repräsentation von Web Content mit Ontologien

Einleitung

In den letzten Jahren haben sich für die Erstellung von Webseiten verschiedene Konzepte entwickelt, die alle gleichsam auf dem Prinzip des Content Lifecycles [10] beruhen. Als Content einer Webseite werden hierbei sinnhaft interpretierte Rohdaten, z.B. Textmaterial, bezeichnet, die entweder redaktionell, also durch einen Redakteur eingepflegt werden oder aber auch aus anderen Quellen, z.B. ERP-Systemen, importiert werden.

Mittels Web Content Management Systemen sollen Inhalte systematisch verwaltet und publiziert werden. Dazu durchläuft jeder Content mehrere Abschnitte von der Erstellung bis hin zu seiner Archivierung. Dieser Zyklus wird als Content Lifecycle bezeichnet und soll im Folgenden erläutert werden (Abb. 1).

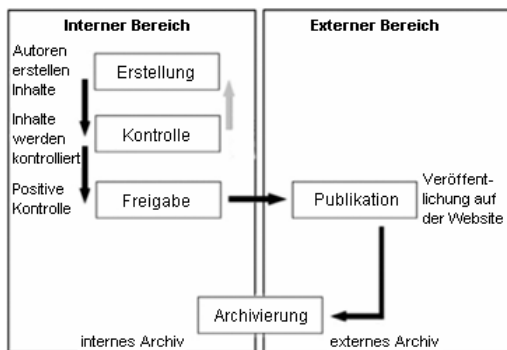


Abb. 1 – Content Lifecycle

Der in Abbildung 1 dargestellte Content Lifecycle ist in zwei Bereiche gegliedert: einen internen und einen externen Teil. Der interne Teil stellt dabei den Content-Anbieter dar, der externe den Content-Nutzer.

1. **Content-Erstellung/Bearbeitung:** Der erste Schritt im Content Lifecycle ist die Erstellung bzw. der Import des Contents. Ein Autor kann hierbei Inhalte mit Hilfe verschiedener Werkzeuge erstellen oder auch den Import von Content aus einer fremden Quelle initiieren.
2. **Kontrolle:** Dieser zweite Schritt im Content Lifecycle ist eine wichtige Stufe für die Qualität und Richtigkeit der Inhalte auf einer Webseite. Durch das so genannte Vier-Augen-Prinzip kann das Publizieren fehlerhafter Inhalte vermieden werden.
3. **Freigabe und Wiedervorlage:** Je nach Ergebnis im Kontrollvorgang kann der Inhalt dem Autor zur Bearbeitung wieder vorgelegt werden oder die Inhalte werden auf der Webseite publiziert.
4. **Publikation:** In dieser Phase steht der freigegebene Inhalt dem Content-Nutzer auf der Webseite für einen festgelegten Zeitraum zur Verfügung.
5. **Archivierung:** Nach Ablauf des festgelegten Zeitraumes soll der erstellte Inhalt von der Webseite genommen werden. Die einfachste Variante wäre es, die Beschreibungen einfach zu löschen. Im Hinblick auf spätere statistische Auswertungen oder das Wiederherstellen eines Beitrages ist diese Methode aber wenig sinnvoll. Eine konsequente Archivierung alter Inhalte kann eini-

ge Vorteile erzielen: Es ist später möglich, ältere Versionen der Webseite einzusehen und es können neue Beiträge mit älteren als Vorlage erstellt werden. Dabei werden zwei Arten von Archivierung unterschieden. Zum einen gibt es eine interne Archivierung für statistische Auswertungen oder auch zur Wiederverwendung und zum anderen existiert ein öffentliches Archiv, das dem Content-Nutzer die Möglichkeit geben soll, in älteren Artikeln zu stöbern.

Der eben erläuterte Content Lifecycle ist ein theoretisches Modell, das den optimalen Ablauf für die Distribution von Inhalten auf einer Webseite beschreibt. In den existierenden Web Content Management Systemen haben sich hierzu verschiedene Methoden entwickelt, dieses Modell abzubilden und weitgehend zu automatisieren. In den folgenden Abschnitten soll nun erläutert werden, wie unter Verwendung der in Abschnitt 2 vorgestellten Standards Webinformationsangebote und deren Inhalte abgebildet werden können.

Modellierungsanforderungen

Um die Modellierung einer Webseite mittels Wissensbasen zu erläutern, sollen in diesem Abschnitt zuerst die Anforderungen an die verwendeten Ontologien gestellt werden, um als Grundlage für ein Web Content Management System zu fungieren. Die ontologische Wissensbasis muss im Zusammenspiel mit einem Ontologie-Editor Folgendes leisten:

1. **Struktur:** Es muss die komplette Struktur der Webseite widerspiegelt werden, d.h. wie einzelne Inhaltselemente (schließlich dargestellt als Webseite) in Beziehung zueinander stehen.
2. **Inhalt:** Neben der Repräsentation von verschiedenen Arten von Inhalten in einer Content-Ontologie muss es möglich sein, Inhalte redaktionell zu pflegen und Inhalte aus anderen Wissensbasen zu importieren bzw. zu referenzieren.
3. **Import/Export:** Die Content-Ontologie muss den Import und Export von Inhalten in standardisierten Formaten erlauben.
4. **Content Lifecycle:** Es muss die Voraussetzung für eine möglichst getreue Abbildung des Content Lifecycles geschaffen werden.

5. **Benutzerverwaltung:** Es muss ein Benutzer- und Gruppenmanagement verfügbar sein. Hierzu wird eine System-Ontologie verwendet, die im Abschnitt *Web Content Management mit pOWL* näher erläutert werden soll.
6. **Layout:** Es muss gewährleistet sein, dass Änderungen am Layout der Website einfach zu realisieren sind. Hierzu muss eine Layout-Ontologie existieren, in der Templates bearbeitet werden können.

Seitenstruktur und Layout

Aufbauend auf das RDFS-Klassenprinzip lässt sich die Struktur einer Website durch Instanzen von einer oder mehreren Klassen innerhalb einer *Struktur-Ontologie* definieren. Die Websitestruktur lässt sich dabei als Baum betrachten, dessen Wurzelknoten durch die Startseite gegeben ist. Dieser Knoten kann mit weiteren Knoten, die thematisch untergeordnete Seiten repräsentieren, verknüpft sein, die ihrerseits auch wieder mit untergeordneten Seiten verknüpft sein können. Aufgabe der Ontologie ist es damit, eine hierarchische Baumstruktur zu modellieren, in der für jede Seite ihre Ebene und ihre Position innerhalb dieser Ebene angegeben werden können. Für die Abbildung der Seitenstruktur schlagen die Autoren die Definition einer Klasse in einer Struktur-Ontologie mit folgenden Eigenschaften vor:

1. **Eltern-Element:** Jeder Knoten in der Baumstruktur hat entweder ein Eltern-Element oder befindet sich auf oberster Ebene der Website.
2. **Position:** Jeder Knoten erhält eine eindeutige Position auf seiner Ebene. Somit kann eine Hierarchie innerhalb einer Ebene erzielt werden.

Durch dieses Prinzip ist es später möglich, Änderungen an der Websitestruktur durch atomare Operationen durchzuführen. Soll eine Seite an eine andere Position auf der Website verschoben werden, so müssen lediglich diese beiden Eigenschaften der zu verschiebenden Seite geändert werden. Gleiches gilt für das Löschen oder das Einfügen von neuen Seiten.

Als weitere Eigenschaften einer Seite müssen zusätzlich noch die folgenden aufgenommen werden:

1. **Inhalt:** Jede Seite kann auf verschiedene Arten von Inhalten verweisen. Hierzu wird entweder eine Instanz oder eine Klasse aus der Content-Ontologie bzw. aus einer fremden Ontologie referenziert.
2. **Status:** Diese Eigenschaft dient als Workflow-Marker für die Seite und schafft demnach die Voraussetzung für die Abbildung des Content Lifecycles auf Seitenebene. Je nach Bearbeitungszustand kann der Status die Werte *In Bearbeitung*, *Freigegeben*, *Archiv* oder *Offline* annehmen.

Die obigen Eigenschaften sind für jede Seite in der Baumstruktur obligat. Durch die Definition von Subklassen mit zusätzlichen Eigenschaften ist es möglich, weitere Angaben für eine Seite zu machen. Somit können Seiten bspw. mit verschiedenen Metadaten versehen werden und die Deklaration verschiedener Seiten-Typen wie z.B. Übersichtsseiten oder Detailseiten wird ermöglicht.

Die Layoutinformationen für die einzelnen Seiten werden durch Instanzen von Template-Klassen definiert. Hierzu wird eine *Layout-Ontologie* eingeführt, die die notwendigen Klassen zur Verfügung stellt. Da die einzelnen Inhalte einer Seite durch Instanzen aus der Content-Ontologie gegeben sind, ist es sinnvoll, für jede einzelne Content-Klasse ein Template anzugeben, das festlegt, wie die Instanz auf der Website dargestellt werden soll. Darüber hinaus können zusätzlich Templates für sämtliche Eigenschaften einer Content-Klasse angegeben werden. Eine nähere Erläuterung der Templatezuweisung findet sich im Abschnitt *Web Content Management mit pOWL*.

Verwaltung der Inhalte

Nachdem die Abbildung der Struktur einer Website erläutert wurde, stellt der folgende Abschnitt dar, wie Web Content mittels Ontologien verwaltet und mit den entsprechenden Seiten verknüpft wird. Ein grundlegender Ansatzpunkt im Web Content Management ist dabei die Trennung von Inhalt, Layout und Struktur. Die Autoren schlagen dazu vor, neben der oben definierten Struktur-Ontologie eine Content-Ontologie zu verwenden, welche die jeweiligen Content-Klassen zur Verfügung stellt. Um auf diese Content-Klassen auch den Content Lifecycle abzubilden, ist es

erforderlich, eine Klasse *Content* zu definieren, die als Eigenschaft den *WorkflowStatus* beinhaltet. Von dieser Klasse werden dann alle anderen Content-Klassen abgeleitet sein.

Das Anlegen von Inhalten für eine Website kann nun auf zwei verschiedene Arten erfolgen. Einerseits können Inhalte redaktionell durch einen Autor gepflegt werden, andererseits sollte die Möglichkeit gegeben werden, Inhalte aus fremden Ontologien zu importieren. Das Importieren aus fremden Ontologien erfolgt im einfachsten Fall durch die Referenzierung ihrer Inhalte, d.h., ein Knoten in der Seitenstruktur verweist auf eine Instanz aus der fremden Ontologie. Diese Referenzierung kann mit dem OWL-Sprachelement *Same-As* erfolgen. Im Fall der redaktionellen Pflege hingegen muss entweder für jeden neuen Inhaltstyp eine neue Klasse mit ihren entsprechenden Eigenschaften definiert werden oder es wird eine bereits bestehende Klasse aus einer anderen Ontologie als Vorlage verwendet. Über die Angabe der *OWL-Equivalent* Beziehung kann dabei für eine neue Klasse festgelegt werden, dass sie äquivalent zu einer anderen ist.

Nach dem Definieren der Content-Klassen können neue Inhalte erstellt werden, indem Instanzen zu diesen Klassen mit den entsprechenden Wertzuweisungen für die Eigenschaften angelegt werden. Als ein mögliches Beispiel sollen im Folgenden die Daten der Mitarbeiter des Lehrstuhls für AIS an der Universität Leipzig administriert werden. Dazu wird zuerst die Content-Klasse *Staff* erstellt und mit den Eigenschaften *StaffName*, *StaffFirstname*, *StaffTitle* und *StaffAddress* belegt. Mit Hilfe eines Ontologie-Editors wird nun für jeden Mitarbeiter eine Instanz von der Klasse *Staff* erzeugt. Jeder Instanz wird dabei ein Name zugewiesen und die genannten Eigenschaften können in Abhängigkeit von ihrem Datentyp mit entsprechenden Werten belegt werden. Um die neu angelegten Mitarbeiterdaten auch auf der Website darzustellen, müssen diese in den Knoten der Baumstruktur referenziert werden. Dazu kann für jeden einzelnen Mitarbeiter ein Knoten in den Baum eingefügt werden, dessen Inhalt auf die entsprechende Instanz in der Content-Ontologie verweist. Auf der Website werden dann einfach die Werte der Eigenschaften für die Mitarbeiter-Instanz angezeigt.

Eine weitere Möglichkeit, um bspw. eine Übersicht über alle Instanzen einer Content-Klasse zu

bekommen, besteht darin, in der Struktur-Ontologie eine neue Klasse zu definieren, die das Referenzieren von ganzen Content-Klassen erlaubt. Somit könnte in obigem Beispiel ein Knoten in der Baumstruktur eine Übersicht über alle gespeicherten Mitarbeiter des Lehrstuhls geben, die dann jeweils zu ihrer Detailansicht verlinkt sind.

Abbildung 2 verdeutlicht im Folgenden das Zusammenspiel zwischen Struktur, Content, Layout und einem webbasierten Ontologie-Editor. Hierbei stellt der Ontologie-Editor den Kern für das ontologiebasierte Web Content Management dar. Er ermöglicht in Ebene 2 die Definition von Klassen auf Struktur-, Layout- und Contentebene sowie die Verwaltung des Systems. Von diesen Klassen werden in Ebene 3 Instanzen erzeugt, die in Ebene 4 zu einer Website verknüpft werden. Dem Ontologie-Editor wird zur Administration eine System-Ontologie zur Verfügung gestellt, in der das Benutzer-Management, die Zugriffsrechte und Widget-Konfigurationen festgelegt werden können. Eine nähere Erläuterung zu System-Ontologie und Widgets findet sich im nächsten Abschnitt.

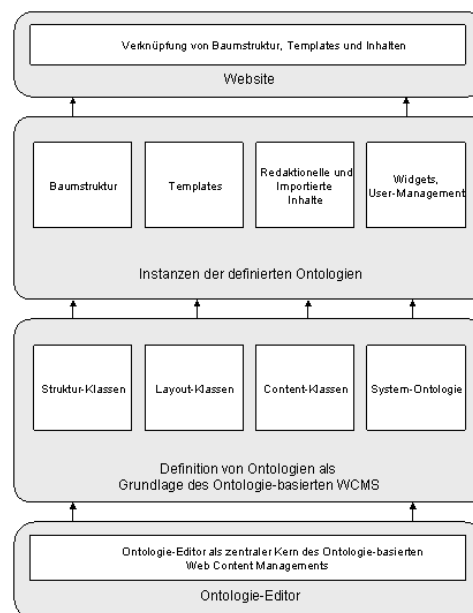


Abb. 2 – Zusammenspiel verschiedener Ontologien beim Web Content Management.

Web Content Management mit pOWL

Der webbasierte RDFS/OWL-Editor pOWL [11] wird von den Autoren um eine Web Content Management Komponente gemäß den vorgestellten Ansätzen erweitert. Ein Prototyp des Open Source Editors ist unter [12] zu finden.

pOWL wird mit der am weitesten verbreiteten Technologie zur Realisierung von dynamischen

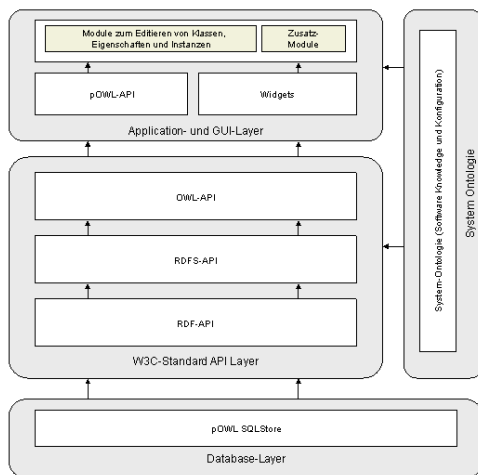


Abb. 3 – Aufbau des Semantic Web Frameworks pOWL

Webanwendungen – PHP [25] – entwickelt und ist demnach auf den gängigen Webserverssystemen verfügbar. Um auf breiter Basis angewendet zu werden, ist nach unserer Ansicht eine Unterstützung dieser Systeme notwendig. Damit zeichnet sich für pOWL ein großer Vorteil als kollaboratives Werkzeug für RDF-basierte Wissensbasen ab.

Das pOWL Semantic Web Application Framework besteht aus den in Abbildung 3 dargestellten Komponenten:

1. **pOWL SQL-Store:** Der SQL-Store bildet im pOWL-Framework die Datenschicht. Er verwendet hierbei den abstrakten Datenbanklayer AdoDB [24] und sichert damit die Zusammenarbeit des Frameworks mit beliebigen relationalen Datenbanksystemen, die durch AdoDB unterstützt werden.
2. **W3C-Standard API Layer:** In dieser Schicht finden sich die Schnittstellen zum

Zugriff auf die W3C-Standards RDF, RDFS und OWL. Für den Umgang mit RDF kommt in pOWL die *RAP* (RDF API für PHP [13]) zum Einsatz. Diese ist ein unabhängig entwickeltes Softwareprojekt, das die Speicherung der einzelnen SPO-Statements sowie das Parsen und die Serialisierung der RDF-Daten für pOWL übernimmt.

Die RDFS- und OWL-API sind die wesentlichen Bestandteile der Applikation, die die notwendigen Funktionalitäten zur API-gestützten Arbeit mit Ontologien abdecken.

3. **pOWL Application und GUI-Layer:** Diese Schicht beinhaltet die pOWL-API und sinnvolle Widget-Plugins, auf deren Basis Module zum Editieren von Modellen, Triples, Klassen, Eigenschaften und Instanzen realisiert wurden. Widgets sind dabei kleine, konfigurierbare Formularelemente. Die pOWL Application und GUI-Schicht stellt ein flexibles, leicht erweiterbares Userinterface dar.

4. **System Ontologie:** Zur Konfiguration des Editors selbst wird eine System-Ontologie verwendet. In dieser Ontologie sind Klassen und Eigenschaften definiert, mit denen bspw. das Benutzer- und Rechtemanagement sowie die Konfiguration und Zuweisung von Widgets getätigt werden können. Die Zuweisung der Widgets geschieht hierbei in Abhängigkeit der Nutzergruppe und des Datentyps. Eine weitere Klasse *Label* in der System-Ontologie gewährleistet die Mehrsprachigkeit der Applikation. Die System-Ontologie geht jedoch weit über die eigentliche Konfiguration hinaus. Durch die Verwendung einer eigenen Wissensbasis zur Konfiguration entspricht die System-Ontologie einer neuen Möglichkeit zum Software-Design. Sie stellt einen homogenen Ansatz zur Repräsentation von Wissen über und für eine Software-Applikation dar.

Zusätzlich zu den existierenden Modulen für Modelle, Triples, Klassen, Eigenschaften und Instanzen wird derzeit das Web Content Management Modul *CMS* entwickelt. Die Anbindung des Moduls ist dabei durch den einfachen Erweiterbarkeitscharakter des pOWL Application und GUI-Layer gewährleistet. Darüber hinaus stellt

pOWL auch einen großen Umfang von Funktionen bereit, die vom WCMS-Modul zur Benutzerverwaltung, Konfiguration und Versionisierung verwendet werden. Abbildung 4 zeigt den Aufbau des WCMS-Moduls. Das Modul reiht sich hierbei in die pOWL Application und GUI-Schicht als zusätzliches Anwendungsmodul ein (vgl. Abbildung 3).

oder Publikationsdaten in standardisierten Formaten. Das Importieren und Exportieren fremder Ontologiedaten wird über entsprechende Plugins realisiert.

WCMS Layout-Ontologie: In der Layout-Ontologie sind Klassen für Templates definiert. In der momentanen Version des WCMS-Moduls können Templates für Content-Klassen, Eigen-

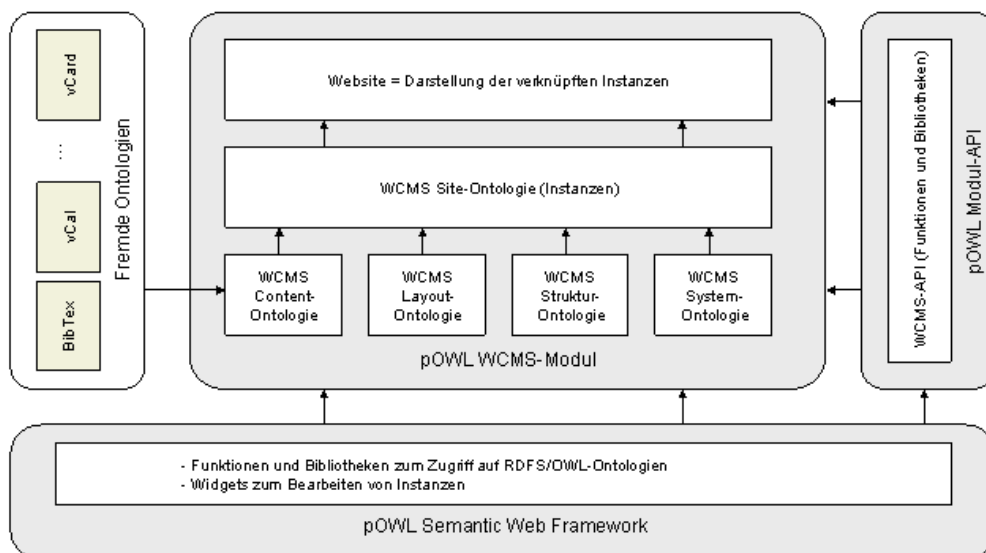


Abb. 4 – WCMS Modul für pOWL

Das pOWL WCMS Modul besteht aus einer Modul-API, vier Modul-spezifischen Ontologien und einer erweiterten System-Ontologie. Die Modul-API stellt die notwendigen Funktionen zum Abbilden der Website-Struktur, zur Darstellung der Inhalte und zum Parsen der Templates zur Verfügung. Die zum Modul gehörigen Ontologien decken den bereits erläuterten Ontologie-basierten Ansatz zum Web Content Management ab:

WCMS Content-Ontologie: In dieser Ontologie können Klassen für die Inhalte der Website definiert werden. Jede Klasse erbt hierbei von einer Klasse *Content*, in der Eigenschaften für das Einhalten des Content Lifecycles deklariert sind. Darüber hinaus können fremde Ontologien wie Bibtex [26] vCal [27] oder vCard [28] importiert werden, deren Instanzen dann auf der Website dargestellt werden können. Dies ermöglicht unter anderem den Import und Export von Mitarbeiter-

schaften und Übersichtsseiten angegeben werden. Für die weiter oben definierte Klasse *Staff* könnte ein Template folgender Maßen gestaltet sein:

```
<div class="staff">
  Name: {StaffName}<br />
  Vorname: {StaffFirstname}<br />
  Anschrift: {{StaffAddress}}<br />
</div>
```

Beim Parsen/Ausgeben der Mitarbeiter-Instanz auf der Website würden nun die *Marker* (Bereiche innerhalb der geschweiften) Klammern durch die Werte der Eigenschaften für die Instanz ersetzt werden. Die doppelt geschweiften Klammern für *StaffAddress* bedeuten hierbei, dass ein gesondertes Eigenschaftstemplate für diese Eigenschaft existiert. Hier wird also zuerst das Template für die Eigenschaft geparkt und dann in das Klassentemplate für *Staff* integriert.

WCMS Struktur-Ontologie: In dieser Ontologie sind die Klassen *Page* und *PageInstanceOverview* definiert, mit denen die Struktur der

Website abgebildet werden kann. *PageInstance-Overview* ist hierbei eine Unterklasse von *Page*, die es erlaubt, die Instanzen einer Klasse aufzulisten. Alle Klassen innerhalb der Struktur-Ontologie erfüllen die Bedingungen, die im Ontologie-basierten Ansatz definiert wurden.

WCMS System-Ontologie: Für jedes Zusatzmodul im pOWL Semantic Web Application Framework kann eine eigene System-Ontologie existieren, die die Konfiguration des Moduls beinhaltet. Mit dieser Ontologie können bspw. das Benutzermanagement und die Konfiguration von speziellen Widgets vorgenommen werden.

den. Das Layout der Website ist durch Instanzen der Layout-Klassen gegeben. Die Vorgehensweise, dass zuerst alle Ontologien in eine andere importiert werden, hat den Vorteil, dass einfach und schnell neue Websites umgesetzt werden, indem lediglich weitere Content-Ontologien und Site-Ontologien angelegt werden, die wiederum die vier definierten Ontologien importieren.

Abschließend soll in diesem Kapitel noch kurz das in Abbildung 5 dargestellte Frontend des WCMS-Moduls erläutert werden. Das Content Management Modul bietet dem Benutzer im linken Bereich des Editors die Navigation durch die Struktur der Website. Hierfür wurde das von

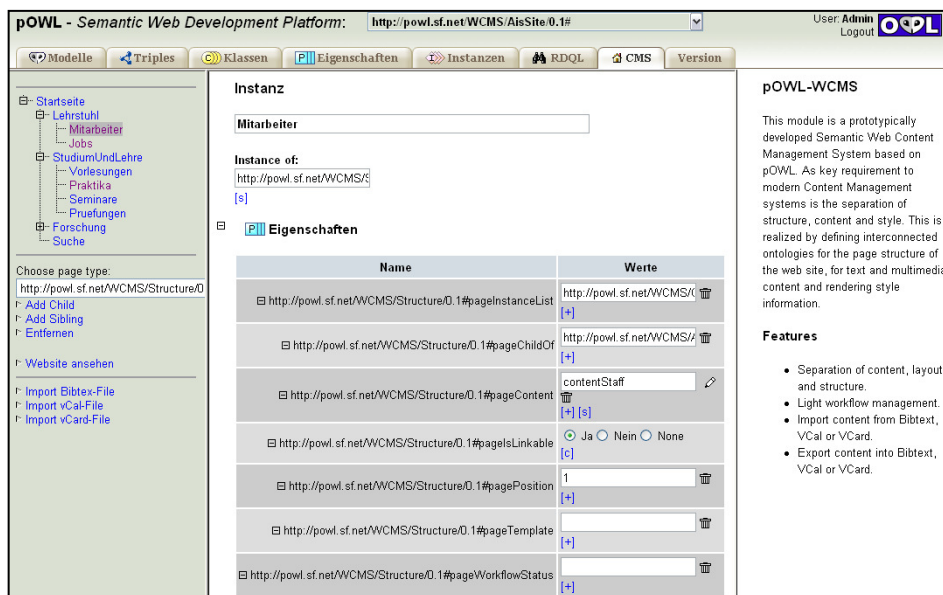


Abb. 5 – pOWL WCMS Frontend

WCMS Site-Ontologie: Die ersten vier erläuterten Ontologien stellen lediglich die definierten Klassen mit ihren Eigenschaften zur Verfügung. Um eine Website zu erstellen, werden diese vier Ontologien in der WCMS Site-Ontologie importiert. Hier können nun Instanzen der importierten Klassen angelegt werden, die dann miteinander verknüpft die Website darstellen. Hierzu wird die Website zuerst als Baumstruktur durch Instanzen der Struktur-Klassen modelliert. Den einzelnen Knoten der Baumstruktur können anschließend Instanzen der Content-Klassen zugewiesen wer-

pOWL zur Verfügung gestellte TreeView-Widget zur Visualisierung von Baumstrukturen verwendet. In diesem Menü kann nun eine Seite ausgewählt werden, die dann im Hauptbereich der Anwendung den Instanzen-Editor der pOWL-Applikation lädt. Hier können nun die Inhalte der betreffenden Seite referenziert und der aktuelle Status im Content Lifecycle angegeben werden.

4 Diskussion des ontologiebasierten Ansatzes

Das vorangegangene Kapitel hat gezeigt, dass die ontologiebasierte Contentrepräsentation eine funktionale Grundlage für ein Web Content Management System darstellen kann. Die dafür notwendigen Anforderungen können erfüllt und umgesetzt werden. Insbesondere wird hierbei unter Nutzung der Semantic Web Paradigmen eine strikte Trennung zwischen Struktur und Inhalt einer Website eingehalten, die eine leichte Wiederverwendung von bereits angelegten Inhalten erlaubt.

Zu den Zielen semantischer Technologien zählen insbesondere die Bestrebungen, die Bedeutung von Inhalten in den Vordergrund zu stellen und dem Benutzer einen bedarfsgerechten Zugriff auf diese Inhalte zu bieten. Während gewöhnliche Web Content Management Systeme hauptsächlich auf die Beschaffung, Erzeugung, Aufbereitung, Verwaltung und Präsentation von Dokumenten abzielen, erlaubt der ontologiebasierte Ansatz darüber hinaus die teilweise Interpretation von Web Content durch Maschinen, womit z.B. Inkonsistenzen bei einer Begriffsbildung semi-automatisch erkannt werden können. Damit wächst das Content Management vom reinen Informationsmanagement zum Wissensmanagement. Durch die Integration von heterogenen Datenquellen können hierbei vernetzte Wissensquellen aufgebaut werden, wobei die Daten nicht nur physisch zusammengeführt werden sondern auch inhaltlich aufeinander abgestimmt werden können. Mögliche Namens- oder Strukturkonflikte, die beim Integrieren heterogener Daten auftreten, können dabei in den Ontologien aufgelöst werden.

Ein weiterer Vorteil in dem vorgestellten Ansatz ist durch den intelligenten Zugriff auf Content für Maschinen gegeben. Ein großer Teil von Informationen in einem Unternehmen befindet sich in verschiedenen Dokumenten. Für den Anwender des Systems erschließt sich die Bedeutung dieser Inhalte lediglich über den Kontext. Durch die Nutzung von Ontologien ist es hier nun möglich, diesen Dokumenten eine für Maschinen verwertbare Struktur (z.B. über Metadaten) hinzuzufügen. Damit können WCMS für verschiedene Nutzergruppen Inhalte intelligent kombinieren

und bedarfsgerecht aufbereiten. Im konkreten Anwendungsfall lässt sich dadurch bspw. das Ergebnis einer Suchanfrage anreichern, indem Beziehungen und Regeln zwischen Dokumenten sowie die Struktur der Ontologie (Synonyme, Unterbegriffe, etc.) ausgenutzt werden.

Mit der Semantic Web Initiative sind zur ontologiebasierten Wissensrepräsentation RDF-basierte Standards entstanden, die eine weitere Anreicherung für das Web Content Management darstellen. Die Nutzung der standardisierten Ontologiesprachen wie RDFS und OWL ermöglicht eine einfache Interoperabilität zwischen verschiedenen Systemen. Im Bereich des E-Business wäre somit bspw. das Definieren von gemeinsamen Vokabularen zum identischen Verständnis von auszutauschenden Geschäftsdaten auf beiden Kommunikationsseiten durch die Definition der Wissensbasis gegeben. Ebenso sind Import- und Exportfunktionalitäten in bestehenden Systemen einfach zu realisieren, da keine aufwändigen Schnittstellen oder Mapping-Tools mehr implementiert werden müssen.

Wie bei allen neuen Standards sind aber auch im Bereich der RDF-basierten Ontologiesprachen noch zu wenige Werkzeuge verfügbar, die einen sinnvollen Umgang mit den neuen Technologien erlauben. RDFS und OWL sind erst in jüngster Zeit vom W3C verabschiedet worden. Es herrscht noch Mangel an Sekundärliteratur und Use Cases, zu dessen Beseitigung die Autoren mit dieser Ausarbeitung einen Beitrag leisten wollen. Die gegenwärtigen Ontologie-Editoren und APIs wie z.B. Protegé [14], das JENA-Framework [15] oder KAON [23] erfüllen zwar den Zweck des Verwaltens von Wissensbasen, unterstützen jedoch kaum kollaboratives Arbeiten. Aber insbesondere im Zusammenhang mit Web Content Management spielt die kollaborative Arbeit an Wissensbasen eine zentrale Rolle.

Ein weiteres Problem, das sich auch bei ersten Tests mit bestehenden Ontologie-Editoren abzeichnete, ist die zunehmende Größe von Ontologien. Die Modellierung beliebiger Entitäten aus der realen Welt verlangt nach umfangreichen Ontologien, wie es etwa bei der Ontologie der Universal Standard Products and Services Classification (UNSPSC) mit mehr als 80.000 Statements der Fall ist. Im Praxiseinsatz zeichneten sich hier aber Geschwindigkeitsengpässe bei den bestehenden Ontologie-Editoren ab, die in erster

Linie auf das Speicherungsmodell der Triples zurückzuführen sind. Dieser Umstand zeigt, dass auch im Bereich der Speicherung von großen RDFS/OWL-Ontologien noch Weiterentwicklungen notwendig sind. Erste Lösungsvorschläge sind hierzu im Redland Application Framework [19] oder SESAME-System [20] zu finden.

Zusammenfassung

Die vorliegende Ausarbeitung hat einen Überblick über die neuen Standards der Semantic Web Initiative zur Wissensrepräsentation gegeben und deren möglichen Einsatz im Bereich des Web Content Managements dargestellt.

Hierzu wurden in den einleitenden Kapiteln die Begriffe Ontologie, RDF, RDFS, OWL kurz erläutert und deren gegenwärtige Einsatzszenarien aufgezeigt. Anschließend wurde in Kapitel 3 untersucht, welche Anforderungen an Wissensbasen im Zusammenspiel mit einem Ontologie-Editor gestellt werden, um als Grundlage im Web Content Management zu fungieren. An Hand dieser Anforderungen konnten danach Strategien für eine mögliche Vorgehensweise zum ontologiegestützten Web Content Management entwickelt werden. Hierbei wurden insbesondere eine Struktur- und eine Content-Ontologie beschrieben, die eine flexible Möglichkeit darstellen, Web Content Management auf einem wissensbasierten Ansatz aufzubauen. Die Analyse dieser Möglichkeit in Kapitel 4 hat gezeigt, dass die Technologien des Semantic Webs das reine Informationsmanagement in üblichen WCMS zu einem Wissensmanagement vervollständigen können. Die Vorteile des erarbeiteten Ansatzes waren dabei insbesondere in der Wiederverwendbarkeit von angelegten Inhalten, der bedarfsgerechten Aufbereitung von Content, dem Integrieren von heterogenen Datenquellen und der einfachen Interoperabilität zwischen kommunizierenden Applikationen zu finden. Die Realisierbarkeit wurde an Hand der prototypischen Umsetzung als CMS-Modul für den webbasierten Ontologie-Editor pOWL gezeigt.

Zusammenfassend kann gesagt werden, dass die Nutzung von ontologiebasierten Wissensrepräsentationen erhebliche Potentiale für das Web Content Management birgt und Synergien zwischen diesen beiden Domänen erschließt.

Literatur

- [1] Tom Gruber, „A Translation Approach to Portable Ontology Specifications“, Academic Press, 1993.
- [2] Mike Uschold, Michael Grüninger, „Ontologies: Principles, Methods and Applications“, AIAI-TR-191 at The University of Edinburgh, 1996.
- [3] Dan Connolly, Jim Handler, Guus Schreiber, „Web-Ontology Working Group“ - <http://www.w3.org/2001/sw/WebOnt/>, 2004
- [4] Eric Miller, Ralph Swick, Dan Brickley, „Resource Description Framework“ - <http://www.w3.org/RDF/>, 2004
- [5] Dan Brickley, R.V. Guha, „RDF Schema“ - <http://www.w3.org/TR/rdf-schema/>, 2004
- [6] Deborah L. McGuinness, Frank van Harmelen, „Web Ontology Language OWL“ - <http://www.w3.org/TR/owl-features/>, 2004
- [7] W3C, „Semantic Web“ - <http://www.w3.org/2001/sw/>, 2004
- [8] Dublin Core, „Dublin Core Metadata Initiative“ - <http://www.dublincore.org/>, 2004
- [9] „RDF Site Summary RSS“ - <http://web.resource.org/rss/1.0/spec>, 2004
- [10] Oliver Zschau, Dennis Traub, Rik Zahradka, „Web Content Management - Websites professionell planen und betreiben“, Galileo Press GmbH, 2. Auflage 2002
- [11] „pOWL – Semantic Web Development Plattform“, <http://powl.sourceforge.net/>, 2004
- [12] „pOWL – Prototype“, <http://powl.sourceforge.net/powl>, 2004
- [13] „RAP – RDF API for PHP“, <http://www.wiwiss.fu-berlin.de/suhl/bizer/rdfapi/>, 2004
- [14] „Protégé – An Ontology and Knowledgebase Editor“, <http://protege.stanford.edu/>, 2004
- [15] „Jena – A Semantic Web Framework for Java“, <http://jena.sourceforge.net/>, 2004
- [16] Computerwoche, „COMPUTERWOCHE ONLINE: Semantik als Schlüssel für Content-Management“,

<http://www.computerwoche.de/index.cfm?type=detail&artid=45151&category=20&Pageid=255>, 2004

- [17] Anne Jannasch, „Web Knowledge Management“, TU Kaiserslautern, 2004
- [18] FEiG & PARTNER, „Content-Syndication durch RSS-Feeds“, http://www.contentmanager.de/magazin/artikel_347_content_syndication_durch_rss_feeds.html, 2004
- [19] Dave Beckett, „Redland Application Framework“, <http://www.redland.opensource.ac.uk/>, 2004
- [20] Arjohn Kampmann, Jeen Broekstra, „Sesame“, <http://sourceforge.net/projects/sesame/>, 2004
- [21] Manjunath Kamath, Nikunj P. Dalal, Ramasamy R. Chinnanchetty, „The Application of XML-Based Markup Languages in Enterprise Process Modeling“, Oklahoma State University, 2002
- [22] Jayavel Shanmugasundaram, Eugene Shekita Rimon, Michael Carey, Bruce Lindsay, Hamid Pirahesh, Berthold Reinwald, „Efficiently Publishing Relational Data as XML Documents“, IBM Almaden Research Center, 2000
- [23] Raphael Volz, Daniel Oberle, Steffen Staab, Boris Motik, „KAON SERVER - A Semantic Web Management System“, Universität Karlsruhe, 2003
- [24] J. Lim, „ADODB Library for PHP“, <http://php.weblogs.com/ADODB>, 2004
- [25] „PHP“, <http://www.php.net/>, 2004
- [26] „Bibtex“, <http://purl.org/net/nknouf/ns/bibtex#>, 2004
- [27] „iCal“, <http://www.w3.org/2002/12/cal/ical#>, 2004
- [28] „vCard“, <http://www.w3.org/2001/vcard-rdf/3.0#>, 2004