Entwurfsbeschreibung

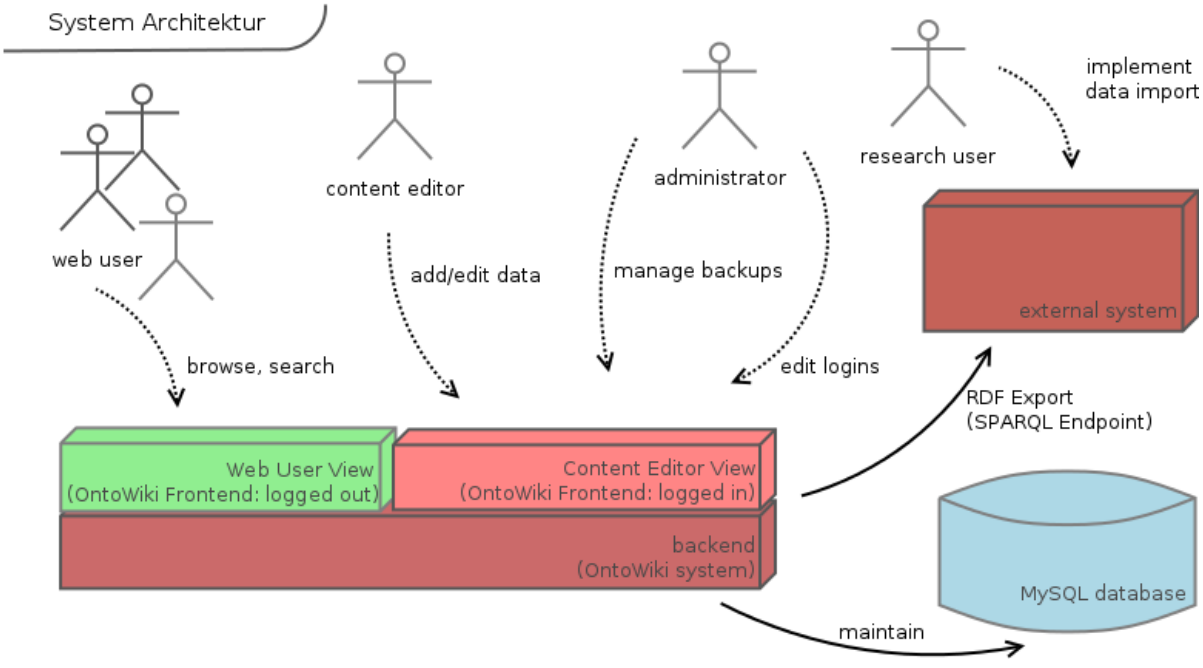
Gliederung:

1. Allgemeines

Die technische Umsetzung der Personendatenbank der *Projektgruppe Wissenschaftsbeziehungen im 19. Jahrhundert zwischen Deutschland und Russland auf den Gebieten Chemie, Pharmazie und Medizi*n der Stiftung *Sächsische Akademie der Wissenschaften zu Leipzig* soll technisch umgesetzt werden. Es existiert bereits eine Datenbasis in HTML Dokumenten.
Die Aufgabe ist nun eine technische Umsetzung der im Projektvertrag definierten Anforderungen zu entwerfen.

2. Produktübersicht

Unser Produkt gliedert sich in zwei Teile. Zum einen ein Konvertierungstool um die Altdaten leicht in die neu zu schaffende Plattform zu integrieren und zum anderen diese selbst.
Die Plattform wird mit dem OntoWiki Framework realisiert. Dies hat die Vorteile, dass wesentliche Anforderungen, wie Account Management, Schnittstelle zu einer Datenbank und wesentliche Designelemente für die Visualisierung leicht genutzt werden können. Viel wichtiger ist aber, das OntoWiki einen einfachen Umgang mit semantischen Daten ermöglicht. Und dies ist das Kernelement der Aufwertung der bisherigen Datenbank.



3. Grundsätzliche Struktur- und Entwurfsprinzipien

3.1 Visualisierung der Inhalte

Die Visualisierung der Daten wird über einzelne vorgefertigte Views, die in Ontowiki implementiert sind realisiert.
Es finden Projektspezifische Anpassungen an des allgemeinen Design sowie tiefgreifende Änderungen in der konkreten Darstellung der Datensätze statt. Dies soll eine individuelle Gliederung der nicht hierarchisierten semantischen Daten zu einzelnen Personen ermöglichen.
Grundprinzip dieser Umsetzung ist die Nachbildung der bisherigen Darstellung auf der bisherigen Website (http://drw.saw-leipzig.de/personendatenbank.html).
Es sind zwei unterschiedliche Views zu unterscheiden. Zum einen eine einfache strukturierte Darstellung für nicht angemeldete Nutzer.
Zum anderen eine erweiterte Sicht mit weiteren Editierungsmodulen für angemeldete Nutzer.

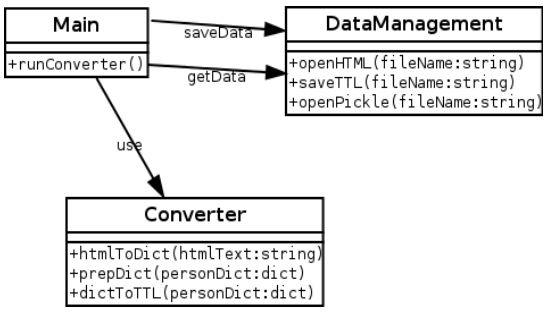
3.2 Erweiterbarkeit

Durch Schnittstellen für Plug-Ins ist das OntoWiki einfach erweiterbar und bietet dem Nutzer eine Möglichkeit, auch auf interne bzw. externe Daten oder Applikationen zuzugreifen.
Alle Anpassungen an das OntoWiki Framework werden durch solche Plugins realisiert. Dies bietet insbesondere den Vorteil, einzelne Komponenten für andere Projekte zu übernehmen und zum anderen nach Projektabschluss die Webapplikation in Folgeprojekten durch weitere Module zu ergänzen.
Die Möglichkeit in OntoWiki mit verschiedenen Wissensbasen zu arbeiten soll für unser Projekt genutzt werden um zum einen die Datenbasis vom definierten Vokabular zu trennen, und zum anderen bietet es die Möglichkeit eine Unterscheidung zwischen der gesammten Datenbasis und einer reduzierten Datenbasis, der dem nicht angemeldeten Nutzer zur Verfügung steht einzuführen.
Dies ermöglicht auch nicht oder erst zu einem späteren Zeitpunkt zu veröffentlichende Daten strukturell von denen zu Trennen, die allen zugänglich sind.
Insbesondere bietet dies auch flexibele zukünftige Nutzungsmöglichkeiten mit Vokabularen und Wissensbasen zu realisieren.

4. Struktur der einzelnen Pakete

4.1 Konvertierungstool

Das Konvertierungstool ist ein externes Tool, welches die zur Verfügung gestellten .pickle Dateien (100 Personen) und die existierenden HTML Datein in das Datenformat .ttl überführt, so dass sie in die Datenbank importiert werden können.
Da bereits bestehende Daten mit Python generiert wurden wird das Tool in dieser Programmiersprache geschrieben.



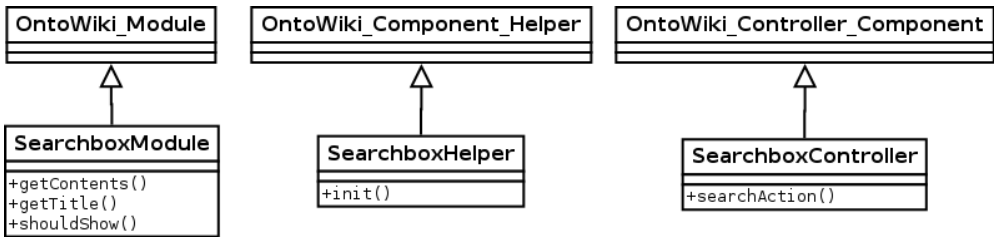
/LF130/ Produktfunktion: Überführung der HTML-Daten in eine Datenbank und Erfassung des Vokabulars

4.2 Extensions für OntoWiki

Informationen über die Funktionsweise von Extension finden Sie in der Entwurfsbeschreibung des Vorprojekts.

searchBox

Die Extension searchbox wird eine an die Fachdomäne angepasste Suche implementieren. Sie wird als Module entworfen und die Methode searchAction() einen View als Tab im großen Fenster der



Seite aufrufen. Dort wird dann das Ergebnis in Listenform angezeigt.

/LF110/ Geschäftsprozess: Erweiterte Suche

publicate

Diese Extension besteht aus einem Module und einem Plugin. Das Module stellt für den Content Editor die Funktion zur Verfügung ein Model in ein neues zu konvertieren.
Das Plugin stellt die eigentliche Funktion des Konvertierens zur Verfügung. Dabei werden aus einem bestehenden Model Relationen gefiltert, die nicht übernommen werden sollen und evtl. Rückrelationen, die noch nicht eingefügt sind erstellt. Danach wird das Ergebnis als neues Model angelegt oder ein bestehendes überschrieben.

/LF90/ Geschäftsprozess: Veröffentlichungsmodus festlegen

/LL10/ Zugriffskontrolle der erstellten Daten

instantRessourceCreation

Dieses Plugin soll beim Hinzufügen einer Ressource die Möglichkeit bieten schnell eine Noch nicht existierende Ressource zu erstellen und diese sofort einzubinden. Diese Funktion wird unter anderem benötigt, wenn über den zu erstellenden Eintrag eine weitere Aussage formuliert werden soll. OntoWiki bietet diese Möglichkeit nicht direkt. Der beste Weg ist dort erst eine Ressource anzulegen und diese dann einzubinden. Das beeinträchtigt den Workflow des Content Editors sehr.
/N 20/
Beschreibung: Die Benutzeroberfläche für der Content Editor muss einfach zu bedienen und den Nutzer bei der Eingabe neuer Daten unterstützen.

transliterate

Dieses Plugin greift am gleichen Punkt, wie *instantRessourceCreation.* Hier kann bei der Bearbeitung von Ressourcen ein Zusätzliches Label generiert werden, was den Inhalt des Labels transkribiert und mit der Annotation @kyr bzw. @trans versieht.

/LF140/ Produktfunktion: Transliterationen von Namen (eng-de, kyrillisch)

semantizeHelper

Bietet die Möglichkeit in bereits eingefügten Ressourcen im Label nach Labeln zu suchen, die in anderen Ressourcen verwendet werden. Diese werden mit einem auszuwählenden Property eingefügt.

/LL20/ Bessere semantische Aufbereitung von Daten

4.3 Genutzte Ontowiki Funktionen

Viele der Anforderungen werden bereits von OntoWiki zur Verfügung gestellt.
Realisiert durch Nutzung des Frameworks:

/LF10/ Geschäftsprozess: Anlegen von Content Editoren

/LF30/ Geschäftsprozess: Anmeldung/Abmeldung am System

/LF40/ Geschäftsprozess: Personen und Daten zu Personen Anlegen

/LF50/ Geschäftsprozess: Personen und Personendaten Bearbeiten

/LF60/ Geschäftsprozess: Hinzufügen der Namen in unterschiedlichen Schreibweisen und Sprachen

/LF70/ Geschäftsprozess: Namen von Orten und Fachgebieten in verschieden Varianten

/LF80/ Geschäftsprozess: Hinzufügen neuer Kategorien

/LF100/ Geschäftsprozess: Volltextsuche

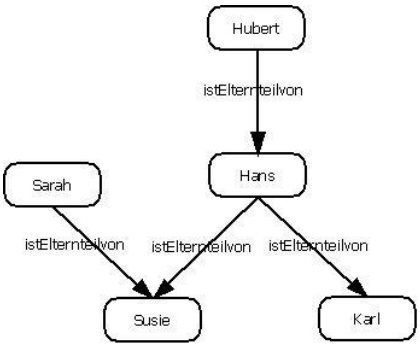
/LF120/ Produktfunktion: Übersichtliche Darstellung

5. Datenmodell

Die Datenhaltung von OntoWiki erfolgt vollständig über den sogenannten Triplestore. In dieser Datenbank, die in Form eines MySQL oder eines Virtuoso Servers existieren kann, sind sowohl die von OntoWiki selbst verwendeten Texte und Daten gespeichert, als auch die Datenbestände, die von OntoWiki verwaltet werden. Aufgrund von Performanzbedenken entschieden wir uns für die MySQL Variante.
Die Speicherung der Daten in dieser Datenbank wird in der folgenden Grafik schematisch dargestellt.

Der Index ist hierbei eine fortlaufende Nummer, die als Primärschlüssel dient. Die eigentlichen Daten sind als Tripel der Form Subjekt-Prädikat-Objekt gespeichert. Das Subjekt bezeichnet hierbei stets eine Instanz einer Klasse, das Objekt eine andere Klasse oder ein Literal (String, Bool, Int, ...) und das Prädikat die Beziehung zwischen

den beiden. Mittels dieser Strukturen können sehr komplexe Beziehungen beschrieben und mithilfe von geeigneten Algorithmen weitere Informationen gewonnen werden. Es folgt der aus den Daten in Grafik 1 konstruierbare Baum.

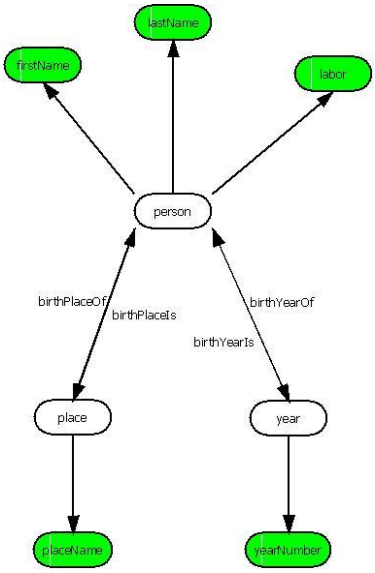


Das Auslesen dieser Beziehungen wird, wie an anderer Stelle erwähnt, vom Erfurt Framework übernommen.
Aus der elementaren Beziehung "ist Elternteil von" kann man nunmehr weitere Informationen entnehmen, z.B. indem man abfragt welche Person Elternteil von einem Elternteil von Susie ist. Diese spezialisierte Form der Abfrage ist mittels der Abfragesprache SPARQL möglich.

Beispiel für SPARQL: (Abfrage aller Personen, die Elternteil von einem Elternteil von Susie sind)

SELECT $Großelternteil WHERE { $Großelternteil abstammung:istElternteilvon Susie}

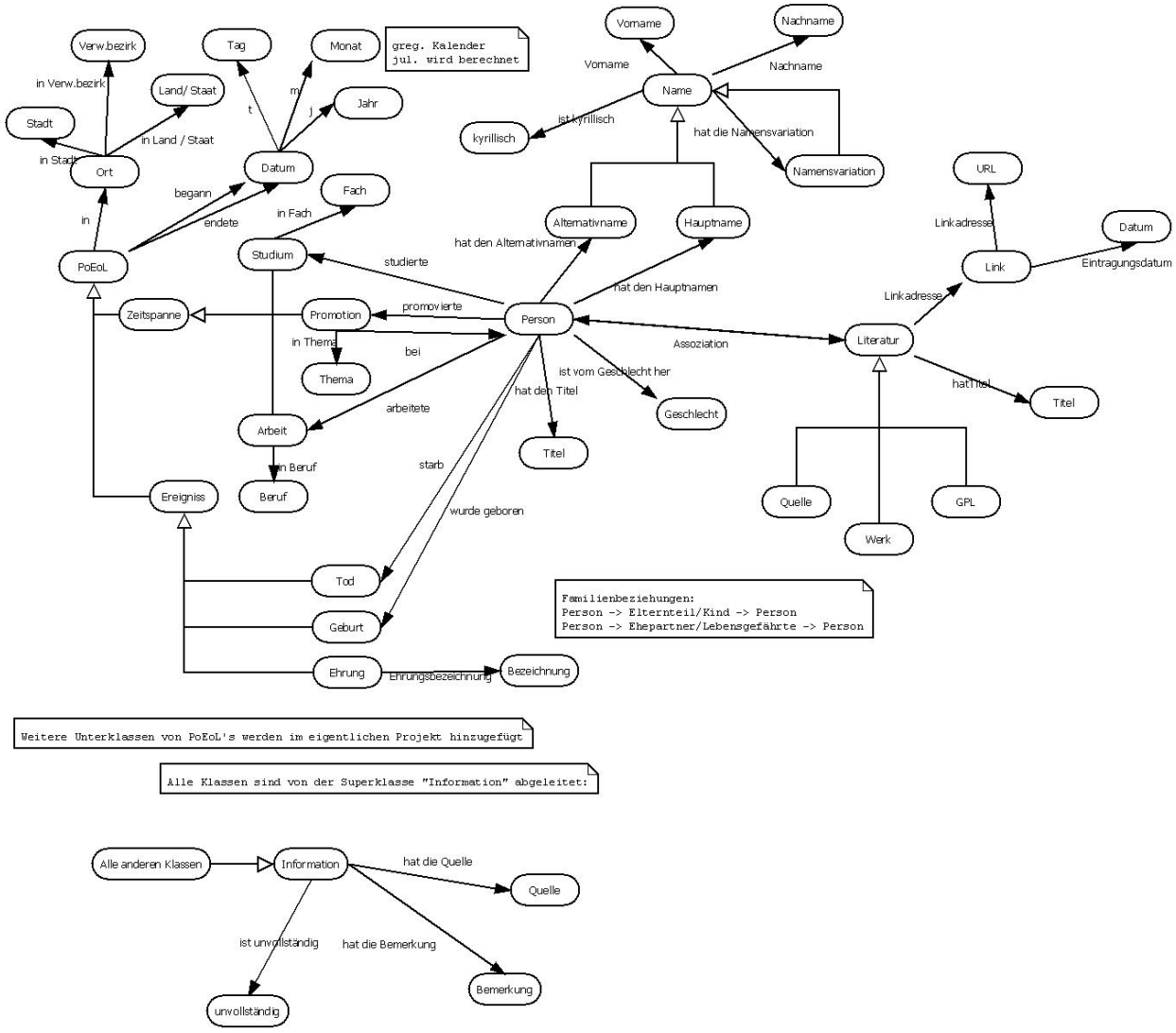
Es ist durch Erweiterungen möglich, Datenbestände sowie Klassenmodelle aus XML Dateien wie RDF, OWL oder Turtle zu importieren oder als solche zu exportieren. Ein Klassenmodell wird dabei als Vokabular bezeichnet, und gibt einen Überblick über alle Objekte sowie über alle Beziehungen zwischen diesen (Vererbung, funktionale Relationen, ...). Für die grafische Darstellung verwenden wir UML-konforme Zeichen. Es folgt eine Grafik unseres für das Vorprojekt entwickelten Vokabulars. Es ist aus Zeit- und Aufwandsgründen natürlich erheblich einfacher als das im Hauptprojekt entwickelte, jedoch eignet es sich bereits gut, um die Funktionalität der Daten zu verdeutlichen.



Die farblosen Flächen sind Klassen von Objekten, denen über die Pfeile, die Object Properties darstellen, andere Objekte und Daten zugeordnet sind. Sie können innerhalb der semantischen Interpretation als Subjekte oder Objekte auftreten.
Die Object Properties treten an die Stelle von Prädikaten, die Instanzen von Klassen miteinander verbinden.

Die grün hinterlegten Flächen stellen Data Properties, also literale Daten, dar. Sie können nur als Objekte im Schema dienen, da sie nicht eindeutig identifizierbar sind. Dennoch sind sie die einzige Möglichkeit wirkliche Daten abzuspeichern.

Durch die inversen Object Properties birthPlaceOf und birthPlaceIs sowie birthYearOf und birthYearIs ist es möglich, ohne erheblichen Mehraufwand Orten die Personen zuzuordnen, die dort geboren wurden oder starben. Hierzu ist es nötig für Orte und Jahre eigene Klassen zu erstellen, wobei die spezifischen Informationen in placeName und yearNumber gespeichert werden.



6. Testkonzept

6.1 Der Testprozess

Komponententests

Komponenten wie etwa Klassen und Methoden werden bei diesen Tests auf ihre Korrektheit und Funktionalität getestet. Dabei werden Testreihen und Beispiele für die einzelnen Klassen und Methoden entwickelt. Dies sollte in der Regel vom Programmierer selbst aus geschehen. Bei jedem Durchlauf eines Tests wird das Ergebnis für evtl. Fehler und Funktionsdefizite sorgfältig dokumentiert. Die Fehlerquelle sollte dann gefunden und wenn möglich behoben werden. Nach erfolgreichen Komponententests ist der Testprozess weiterzuführen.

Integrationstests

In diesem Testabschnitt werden die einzelnen Komponenten auf Zusammenarbeit getestet. Getestet wird nach jeder Woche welche fertiggestellten Module bereits mit anderen zusammenarbeiten können und ob dabei Fehler auftreten bzw. diese kompatibel sind. Werden bei Integrationstests Fehler lokalisiert werden die einzelnen Module angepasst.

Systemtests

Sind Komponenten- und Integrationstests erfolgreich abgeschlossen, so wird eine Vorabversion aus den bestehenden Modulen zusammengestellt. Mit dieser Vorabversion wird aus Nutzersicht geprüft ob die Anforderungen aus dem Lastenheft erfüllt werden. Werden dabei fehlende Funktionalitäten festgestellt sind diese hinzuzufügen. Werden Module dabei verändert, müssen Komponenten- und Integrationstests erneut durchgeführt werden.

6.2 Was ist zu testen?

Plugins

Unter Plugins versteht man, im falle von OntoWiki, kleine aus PHP Dateien aufgebaute Programmteile welche auf „Eventhandling“ ausgelegt sind. Plugins werden wir gebrauchen um nötige Schnittstellen zu erzeugen.
Plugins bestehen wie schon erwähnt aus PHP Dateien die an entsprechender stelle in der OntoWiki MVC Dateistruktur eingefügt werden müssen. Plugins werden wir damit Großteils mit PHPUnit testen. Plugin Tests im eigentlichem Sinn Komponententests werden vom Programmierer selbst erstellt und Dokumentiert. Erst nach erfolgreichem Komponententests werden sie unserem OntoWiki-Repository hinzugefügt und auf Integrationsfähigkeit getestet.

Extensions

Unter Extensions versteht man eine übergeordnete Klasse von Plugins welche auf Funktionsaufrufe und Steuerbarkeit ausgelegt sind. Extensions werden wir benötigen um OntoWiki nach den Kunden wünschen zu Konfigurieren (Einfache GUI, Abfragen und Eintragemöglichkeit in die Datenbank erleichtern).
Ähnlich der Plugins sind Extensions zum Großteil aus PHP Dateien aufgebaut und werden damit auch mit PHPUnit getestet. Extensions werden ebenfalls erst nach erfolgreichem Komponententest in das OntoWiki-Repository eingefügt und auf Integrationsfähigkeit getestet.

XML Validator

Da wir in unserem Projekt mit extrem vielen Daten innerhalb einer .html Datei konfrontiert sind werden wir die Daten dieser .html Datei automatisch generieren lassen. D.h. Wir werden in einer bekannten Programmiersprache einen Skript erstellen um die Daten der .html Datei in XML Form zu bringen. Da die .html Datei teils per Hand geschrieben wurde erwarten wir keine konsistenten Daten. Um diese Konsistenz Fehler schneller zu bearbeiten werden wir XML Validator verwenden um evtl. Syntaxfehler in der XML zu beheben.

Glossar

Altdaten
Das sind alle Daten, die bisher erfasst wurden und die in Form der bestehenden HTML-Daten zur Verfügung gestellt werden.

Backend
Das Backend ist der Kern des Systems, dass die Datenverwaltung und Datenhaltung realisiert. Dem Administrator und den Content Editoren werden hier alle sie betreffenden Funktionalitäten zur Verfügung gestellt.

Content Editor
Ein Content Editor hat die Möglichkeit den Datenbestand einer Applikation zu ändern, verfügt also über den Zugang zu dem Backend .

Frontend
Das Frontend ist die Schnittstelle des Systems zu dem Web User. Hier werden die Daten für die Öffentlichkeit visuell dargestellt.

OntoWiki
Das OntoWiki ist ein Wikisystem, in dem alle Daten als Tripel gespeichert werden, was ihre Verknüpfungen ermöglicht. Es ist auch ein Werkzeug, das der kollaborativen Arbeit mehrerer Nutzer dient. Web User können verschiedene Inhalte anlegen und editieren und auch dazu Annotationen bzw. Kommentare schreiben.

SPARQL Endpoint
Durch diese Schnittstelle wird es externen Projekten ermöglicht direkt auf die Daten zuzugreifen.

Vokabular
Ein Vokabular bildet eine Menge von gewählten Bezeichnungen, welche die Datenstruktur in einer Ontologie beschreiben.

Web User
Ein Web User ist ein Nutzer einer Webapplikation, er kann den Datenbestand nicht ändern, sondern die Funktionen einer Applikation nuten z.B.: recherchieren und Suchanfragen stellen.

Extension
Eine Extension dient zum Aufruf von Funktionen in einer Softwareanwendung.

Plugin
Ein Plugin ist ein Softwaremodul, das von einer Softwareanwendung während seiner Laufzeit entdeckt und eingebunden werden kann, um dessen Funktionalität zu erweitern.