Universität Leipzig
Softwaretechnik-Praktikum
Sommersemester 2014

Projektvertrag

Projekt	Graphical SPARQL Builder
Gruppe	s14.swp.gsb
Verantwortlich	Siegfried Zötzsche
Erstellt am	12. April 2014

Inhaltsverzeichnis

1. Einleitung **2**

2. Zielbestimmung **2**

3. Voraussetzungen **2**

4. Designübersicht und Funktionalität **2**

5. Arbeitspakete und Meilensteine **3**

6. Qualitätssicherung **4**

A. Glossar **5**

1. Einleitung

Eine der wichtigsten Leistungen des World Wide Web ist sicherlich das Angebot immenser Mengen frei zugänglichen Wissens. Ein Teil dieses Wissens liegt dabei in einer Vielzahl von Datenbanken, welche dem Semantic Web zugeordnet sind.

Dieses Konzept steht für die Bereitstellung von Daten auf eine maschinell verarbeitbare Weise, wodurch Suchanfragen, Austausch und Verknüpfungen von Daten, sowie strukturierte Erzeugung und Verwertung von Metadaten möglich werden.

Für die Suche in solchen RDF-basierten Datenbanken stellt SPARQL als Anfragesprache ein mächtiges Werkzeug dar – für den eingelernten Nutzer. Dieses Projekt verfolgt als Ziel die Entwicklung eines Graphical-SPARQL-Builders (GSB) und spricht im Vergleich dazu vor allem diejenigen an, denen es an Erfahrungswerten in Bereichen der SPARQL-Syntax bzw. RDF-Graphen allgemein mangelt.

2. Zielbestimmung

Der GSB als Webanwendung soll eine eingängige Benutzeroberfläche bereitstellen, welche die Erstellung von Anfragen an SPARQL-Endpunkte ermöglicht. Diese Anfragen sollen mithilfe graphischer Elemente und intuitiver Werkzeuge schrittweise aufgebaut werden können – weitestgehend abstrahiert von der darunterliegenden SPARQL-Syntax. Dies soll die Formulierung komplexer Anfragemuster ohne aufwendige Einarbeitung ermöglichen.

3. Voraussetzungen

Ursprünglich ging dem GSB kein Projekt voraus, somit muss es komplett neu entwickelt werden. Seit dem Projektangebot haben wir im Vorprojekt einen Prototypen des GSB geschaffen, welcher nun die Basis für das endgültige Produkt stellt.

Als technische Voraussetzungen für den GSB ist die Verfügbarkeit eines realen SPARQL-Endpoints von Nöten, um die Anwendung ausgiebig testen zu können. Hier bei hoffen wir, dass wir auf die Daten und Ontologien des ERM Projects [1] beziehungsweise Auszügen derselbigen zugreifen können, da diese eine hohe Qualität aufweisen. Falls wir diesen Zugang nicht rechtzeitig erhalten, werden wir auf die dbpedia (siehe [3]) zurückgreifen.

Bei der Entwicklung werden wir weiter HTML und JavaScript zusammen mit dem AngularJS-Framework benutzen.

4. Designübersicht und Funktionalität

Design

Das Design basiert abgesehen von den im Folgenden beschriebenen Abweichungen auf dem Design des Vorprojekts. Dieses kann in der Onlineversion des Vorprojekts betrachtet werden (`http://pcai042.informatik.uni-leipzig.de/~swp14-gsb/`). Zusätzlich werden die Subject-Boxen durch Drag & Drop verschiebbar sein.

Verbindungen zwischen Subjects können wie im Vorprojekt durch Properties angegeben werden und werden im Hauptprojekt durch Linien zwischen den jeweiligen Subjects gekennzeichnet.

Durch Buttons in einer ein- und ausblendbaren Leiste in den Subject-Boxen werden die Properties um arithmetische Operatoren (`+`, `-`, `*`, `/`, `COUNT`, `MAX`, `MIN`, `SUM`) erweitert.

Funktionalität

Die Hauptfunktionalität des GSB liegt in der Generierung semantisch und syntaktisch korrekter SPARQL-Anfragen aus einer grafischen Repräsentation der Anfrage. Der User muss den grafischen Input leisten und erhält vom GSB dafür verschiedene Werkzeuge

- Das Tool ist in der Lage aus RDF-basierten Datenbanken Klassen und ihre Eigenschaften herauszufiltern und sie dem User zur Verfügung zu stellen. Hierbei kann der User eine Volltextsuche auf den Daten nutzen.
- Der GSB stellt zudem eine Vielfalt an klickbaren Lösungen für Sprachelemente aus SPARQL und für Arbeitshilfen bereit. Exemplarisch dafür stehen folgende Funktionalitäten
  - Ergebnisspalten können zum Beispiel mit einem Augensymbol ein- und ausgeblendet werden.
  - Zu jeder Klasse und jeder Eigenschaft kann außerdem ein Beschreibungstext über ein Informationssymbol angezeigt werden.
  - Grafische Elemente können über ein intuitives Mülleimersymbol wieder gelöscht werden.
- Um ein langfristig produktives Arbeiten mit dem GSB zu gewährleisten, ist eine Speicherfunktion implementiert, mit welcher User ihre grafisch zusammengestellten Anfragen im JSON-Format abspeichern und später wieder im GSB laden können.

Der GSB verfügt über Möglichkeiten der Individualisierung über eine Konfigurationsdatei, mit der Einstellungen zur Sprachwahl, Einschränkungen der bereitgestellten Datenbank, Expertenansicht und zahlreiche GUI-Anpassungen vorgenommen werden können. Für die Konfiguration sollte ein Administrator verantwortlich sein. Der GSB ist so konstruiert, dass er als Single-Page-Anwendung einfach in bestehende Websites eingegliedert werden kann.

5. Arbeitspakete und Meilensteine

Vorprojekt (30% des Gesamtprojekt)

Muss-Ziele

/M1/001 Entwicklung eines graphischen Modells für SPARQL-Anfragen

/M1/002 Beispielhafte Umsetzung ausgewählter SPARQL-Anfragen mit AngularJS

Graphische Umsetzung (40%)

Muss-Ziele

/M2/001 Umsetzung notwendiger SPARQL-Anfragen

Kann-Ziele

/K2/001 Umsetzung nicht geforderter SPARQL-Anfragen

/K2/002 Modulare Entwicklung um Erweiterbarkeit sicherzustellen

/K3/003 Speichern von Anfragen / Beispielanfragen

/K3/004 i18n

Endpoint-Anbindung (20%)

Muss-Ziele

/M3/001 Anbindung über Konfigurationsdateien

/M3/002 Beispielhafte Anbindung des GSB an dbpedia Endpoint

Kann-Ziele

/K3/001 Anbindung an mehrere Endpoints

/K3/002 Caching von Daten um Reaktionszeiten zu verkürzen

Benutzerhandbuch (10%)

Muss-Ziele

/M4/001 Schriftliches Benutzerhandbuch

Kann-Ziele

/K4/001 Video-Anleitung für Benutzer

6. Qualitätssicherung

Für eine ausführliche Erläuterung der Einzelpunkte sei an dieser Stelle auf das Projektangebot, Abschnitt Qualitätssicherung, verwiesen.

Produktqualität	sehr gut	gut	normal	nicht relevant
Funktionalität	×			
Zuverlässigkeit			×	
Benutzbarkeit	×			
Effizienz		×		
Anpassbarkeit			×	
Übertragbarkeit			×	

A. Glossar

API Application Programming Interface – Programmierschnittstelle. Eine API beschreibt, wie Software-Komponenten bzw. Programme miteinander interagieren können/-sollten. Anders ausgedrückt: eine API ist der Teil eines Softwaresystems, der anderen Programmen zur Verfügung gestellt wird um mit dem Softwaresystem zu interagieren.

DBpedia DBpedia ist eine Datensammlung im RDF Format, deren Datensätze aus der Wikipedia extrahiert wurden. Ziel ist es strukturierte Daten für Webanwendungen zur Verfügung zu stellen. [2, 3, 4]

Endpoint Ein Endpoint ist eine Schnittstelle zwischen der Datensammlung und der Abfragesprache. Nachdem eine Anfrage an den Endpoint gesendet wurde (query) sendet selbiger die Ergebnisse zurück. Ein Beispiel für einen SPARQL-Endpoint ist der "Virtuoso SPARQL Query Editor". [5]

GSB Graphical SPARQL Builder, der Name des Projekts. [6]

i18n internationalization and localization – Anpassung der Software an andere Sprachen und Kulturen ohne Quelltext zu ändern. Sprach- und Kulturspezifika werden über Konfigurationsdateien angepasst.

Ontologie Ontologien sind formalisierte Vokabulare von Begriffen. Diese Vokabulare beziehen sich meist auf eine bestimmte Domäne (Gegenstandsbereich) oder Nutzergruppe. Sie liegen in einer sprachlichen Form vor und umfassen die Begriffe einer Domäne sowie Beziehungen zwischen den Begriffen. [7, 8, 9]

OWL Die Web Ontology Language (OWL, aktuelle Version OWL2) ist eine Beschreibungssprache um Ontologien für das semantische Web zu erstellen und zu publizieren. OWL2-Ontologien können für Informationen verwendet werden, die in RDF geschrieben sind und werden hauptsächlich in Form von RDF-Dokumenten ausgetauscht. [7]

RDF "Resource Description Framework" kurz RDF ist eine Strukturierung von Daten nach dem Muster Subjekt-Prädikat-Objekt. Alle RDF-Daten werden in diesem Tripel-Format gespeichert. RDF gilt als eines der Basis-Elemente des semantischen Webs. Repräsentationen, also syntaktische Standards, des RDF-Prinzips sind N3 (Notation3), Turtle (Terse RDF Triple Language) sowie RDF/XML. Turtle und N3 gelten im Vergleich zu RDF/XML als benutzerfreundlicher. [10, 11, 12]

SPARQL "SPARQL Protocol And RDF Query Language" kurz SPARQL ist eine Abfragesprache für das Datenformat RDF. SPARQL ist graphbasiert und gilt nach dem W3C als Standard für RDF-Abfragen. [13, 14]

Triplestore Ein Triplestore ist ein System zur Speicherung, Verwaltung und Bearbeitung einer Sammlung von RDF-Tripeln. Ein Triplestore bietet gewöhnlich APIs, Reasoning-Verfahren sowie Abfragemöglichkeiten. [9]

Literatur

[1] http://aksw.org/Projects/ERM.html

[2] http://de.wikipedia.org/wiki/DBpedia

[3] http://de.dbpedia.org/

[4] http://wiki.dbpedia.org/Datasets

[5] http://dbpedia.org/sparql

[6] http://pcai042.informatik.uni-leipzig.de/~swp14-gsb/

[7] http://www.w3.org/TR/2012/REC-owl2-overview-20121211/

[8] http://de.wikipedia.org/wiki/Ontologie

[9] http://www.iais.fraunhofer.de/fileadmin/user_upload/Abteilungen/OK/PDFs/Alieva_Magisterarbeit.pdf

[10] http://www.w3.org/TR/rdf-primer/

[11] http://de.wikipedia.org/wiki/Resource_Description_Framework

[12] https://en.wikipedia.org/wiki/RDF/XML

[13] http://www.w3.org/TR/rdf-sparql-query/

[14] http://en.wikipedia.org/wiki/SPARQL