LinkedSpending

SWP14-LS

- Entwurfsbeschreibung -

Inhaltsverzeichnis

Matti Berthold, Thomas Günther, Rachid Ijjaali, Kevin Kubsch,
Tobias Pötschke, Florian Reiner, Tom Vorwerk

1. Allgemeines

OpenSpending ist ein Projekt mit dem Ziel Finanzdaten über Regierungs- und Unternehmensausgaben zu sammeln und ansprechend zu visualisieren. LinkedSpending verknüpft diese Datenmenge mit dem SemanticWeb, indem es diese in das RDF-Format überführt, wodurch die Zusammenhänge besser verarbeitet und somit komplexere Visualisierungen automatisch erstellt werden können.

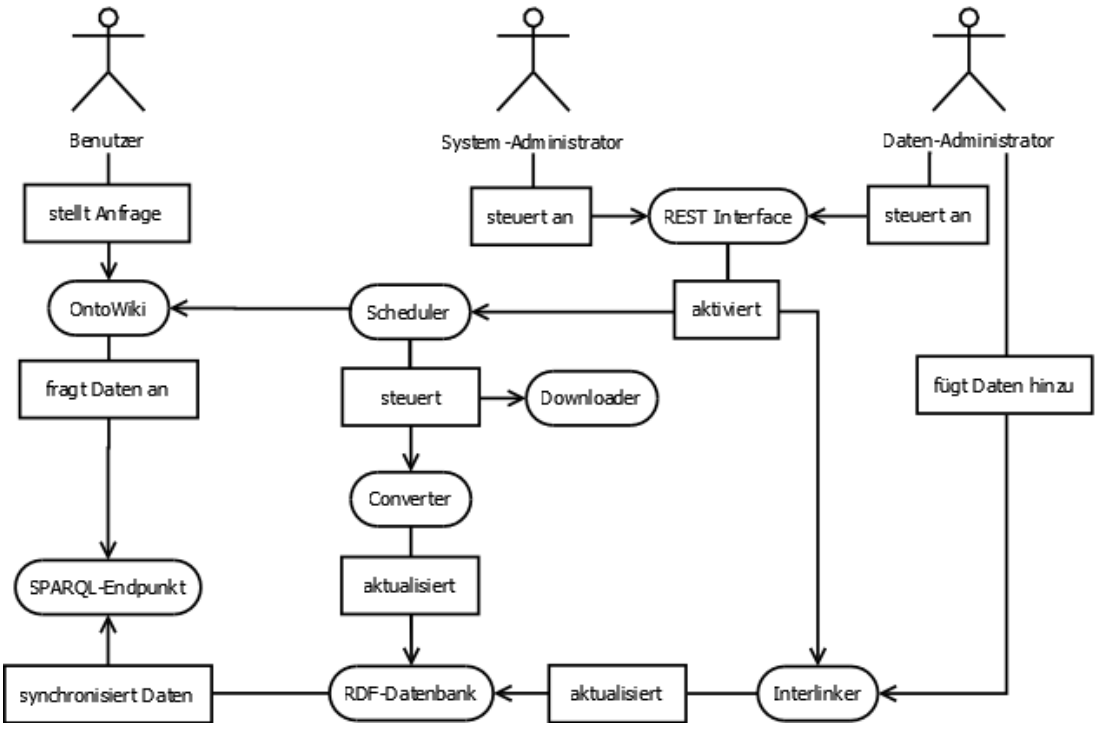
2. Produktübersicht

Daten werden von der OpenSpending-Plattform heruntergeladen, verarbeitet und in das RDF-Format umgewandelt. Die zeitliche Abfolge und Fehlerverarbeitung wird von einem Scheduler verwaltet. Dieser ist außerdem dafür zuständig entsprechende Statusmeldungen an das OntoWiki zu senden.

Die konvertierten Daten werden weiterhin permanent mit externen Datensätzen aus dem SemanticWeb verknüpft. Diese Arbeit des so genannten Interlinkers ist prinzipiell unabhängig vom vorhin beschriebenen Prozess der Datensammlung.

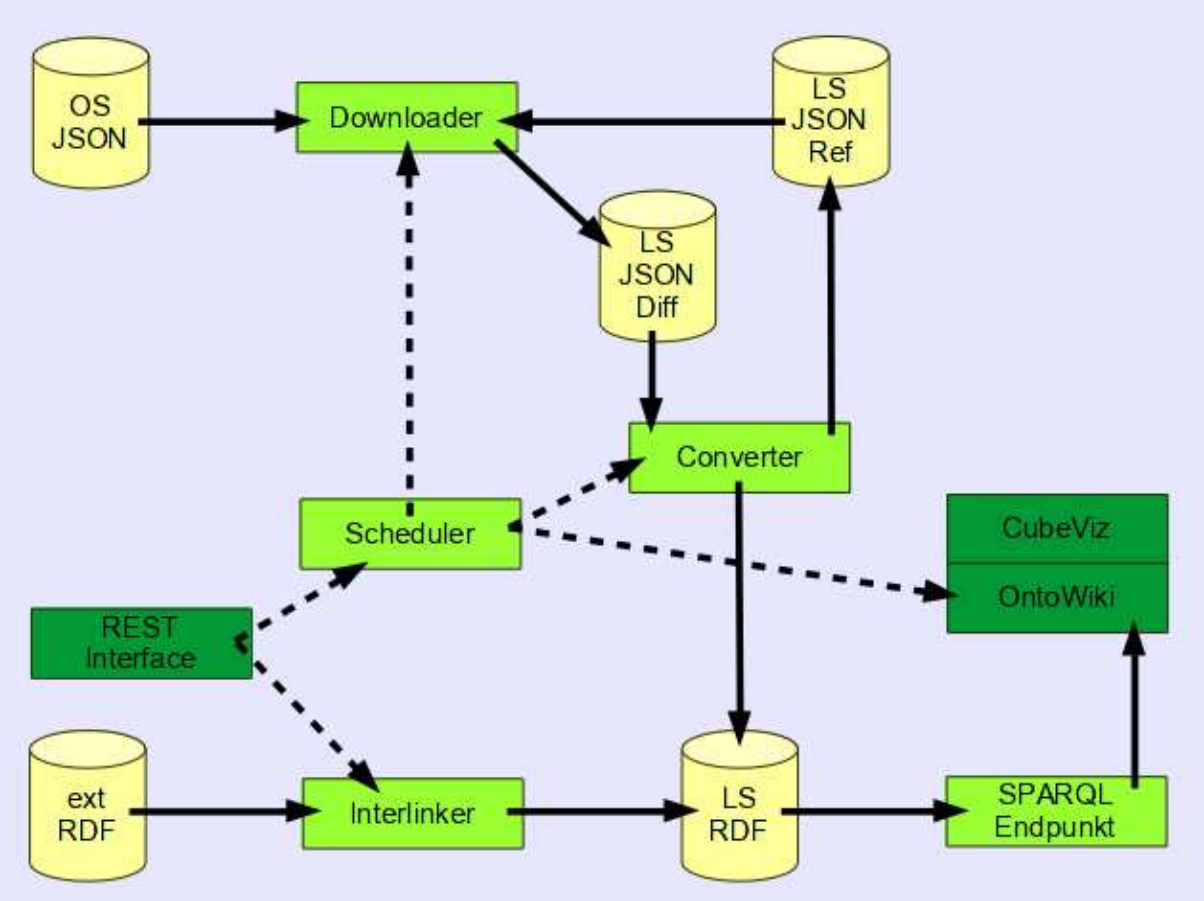
Beide Vorgänge sorgen für die Aktualisierung eines SPARQL-Endpunktes welcher beispielsweise von OntoWiki verwendet wird.

Zur einfachen Kommunikation mit dem Programm steht eine REST-Schnittstelle zur Verfügung. Mit dieser kann Scheduler und Interlinker zur gesteuert werden.



3. Struktur- und Entwurfsprinzipien

3.1 grundsätzliche Struktur



3.2 Prinzipien einzelner Komponenten

Downloader:

- sammelt Daten von OpenSpending (OS JSON)
- vergleicht die Daten mit Referenz (LS JSON Ref)
- erzeugt Sammlung der Differenzen (LS JSON Diff)
- meldet Probleme und Erfolge an Scheduler

Converter:

- arbeitet Differenz-Daten (LS JSON Diff) in RDF-Datenbank ein (LS RDF)
- pflegt Referenz-Daten (LS JSON Ref)
- meldet Probleme und Erfolge an Scheduler

Scheduler:

- bestimmt welche Daten der Downloader aktualisieren soll
- steuert zeitliche Abfolge von Downloader und Converter
- verarbeitet mögliche Probleme
- sendet Statusberichte an OntoWiki

Interlinker:

- verlinkt externe RDF-Daten (ext RDF) mit vorhandenen Daten (LS RDF)

RDF-Datenbank (LS RDF):

- synchronisiert SPARQL-Endpunkt mit sich

REST-Schnittstelle:

- ermöglicht Start/Stop-Anfragen an Scheduler und Interlinker
- erlaubt Auswahl bestimmter Daten für priorisierte Verarbeitung

4. Testkonzept

4.1 Komponententests

Hierbei werden einzelne Klassen und Methoden auf ihre grundlegende Funktionalität überprüft. Nach jeder Programmmodifikation werden diese durch Maven automatisch durchgeführt. Erst nach erfolgreichem Komponententest sind weiterführende Schritte erlaubt, bei Fehlern wird der Build sofort abgebrochen. Diese Unit-Tests werden schon während der Entwicklung der jeweiligen Klasse vom gleichen Entwickler geschrieben. Es ist darauf zu achten dass diese nicht von anderen Klassen abhängig sind und außerdem schnell ausgeführt werden können.

Für die Tests der einzelnen Klassen kommt JUnit zum Einsatz. Zu den zu testenden Klassen werden Testklassen mit mehreren Testmethoden erzeugt. Die entsprechenden Testklassen befinden sich in einer identischen Ordnerhierarchie wie die Hauptklassen. Die Testklasse von src/main/$package/X.java ist hierbei unter src/test/$package/XTest.java zu finden.

4.2 Integrationstests

Nach erfolgreichem Abschluss des Komponententests erfolgt der Integrationstest. Dieser prüft, ob die einzelnen Komponenten korrekt kommunizieren und zusammenarbeiten. Bei Fehlern werden diese entsprechend vermerkt, der Build wird aber dennoch fortgesetzt. Integrations-Tests sollten von einem komponentenfremden Entwickler geschrieben werden, entweder parallel zur Komponentenentwicklung oder kurz darauf.

4.3 Systemtests

Dieser Test wird im Gegensatz zum Komponenten- und Integrationstest aus der Sicht des Anwenders durchgeführt und nicht aus Sicht des Projektteams. Das Ziel des Systemtestes ist die Überprüfung des Gesamtsystems auf Erfüllung der erwarteten Anforderungen. Es werden die funktionalen als auch die nicht-funktionalen Anforderungen geprüft. Als Basis dieses Testes dient das Pflichtenheft. Ein Teil der Systemtests kann automatisiert werden, zusätzliche manuelle Tests sind aber dennoch Pflicht.

4.4 Abnahmetest

Der Abnahmetest schließt das Testkonzept ab und testet das finale Softwareprodukt. Im Abnahmetest wird die Software vom Auftraggeber auf vertragliche Akzeptanz und Benutzerakzeptanz geprüft.

5. Glossar

OntoWiki:
Das Ontowiki ist ein semantisches Programm für Wissensmanagement im Semantic Web Kontext.

RDF (Resource Description Framework):
Das Resource Description Framework bezeichnet eine technische Herangehensweise im Internet zur Formulierung logischer Aussagen über. Im RDF-Modell besteht jede Aussage aus den drei Einheiten Subjekt, Prädikat und Objekt.

REST (Representational State Transfer):
REST bezeichnet ein Programmierparadigma für Webanwendungen welche besagt dass eine URL genau einen Seiteninhalt als Ergebnis einer serverseitigen darstellt.

OpenSpending:
OpenSpending ist eine offene Plattform zum Datenaustausch welche versucht alle Transaktionen zwischen Regierungen und Unternehmen aufzuzeichen und visuell zu präentieren.

SemanticWeb:
Das Semantic Web ist eine Instanz von semantischen Netzen und außerdem eine Erweiterung des WWW. Ziel ist es die Bedeutung von Informationen für Computer verwertbar zu machen und damit automatisch für die interessierten Nutzer im Zuge einer Abfrage zu ordnen.

SPARQL (SPARQL Protocol and RDF Query Language):
SPARQL ist eine graph-basierte Abfragesprache für RDF. Mit SPARQL ist es möglich komplexere Anfragen zu stellen als es reine Textsuchen ermöglichen würden.