ODCleanStore
Infrastructure for Storing, Cleaning, Linking and Providing Aggregated Data
(SW Project Proposal)

Supervisor: Tomáš Knap (tomas.knap@mff.cuni.cz)
Team members: 5 students
Language: Java
OS: Windows 7 / Windows Server 2008 / Linux

Motivation

The advent of Linked Data [1,2] in the recent years accelerates the evolution of the Web into a giant information space where the unprecedented volume of resources will offer to the information consumer a level of information integration and aggregation that has up to now not been possible. Consumers can now 'mashup' and readily integrate information for use in a myriad of alternative end uses. Indiscriminate addition of information can, however, come with inherent problems such as (1) the provision of poor quality, (2) inaccurate, (3) irrelevant, or (4) fraudulent information. All will come with an associate cost which will ultimately affect decision making, system usage and uptake.

The ability to assess the quality of information on the Web, thus, presents one of the most important aspects of the information integration on the Web and will play a fundamental role in the continued adoption of Linked Data principles [2].

Goal of the Project

The goal of the project is to build a Java application which will store, clean, link, and score incoming RDF data and provide aggregated and integrated views on the data to Linked Data consumers. The application will have a graphical user interface for application administration. The main parts of the application are:

Data storage

The application will store the incoming data, together with its metadata, to OpenLink Virtuoso [3] data storage, the most popular RDF data storage with a solid support. This task requires configuring OpenLink Virtuoso and setting up/developing mechanisms (e.g. web services, JAR libraries) to communicate with the storage (to store/retrieve the data). We will use two important data spaces - to store the incoming data (dirty database) and to store clean data (clean database).

Cleaning and scoring the data (implemented by Error Localization component)

The application will check whether the incoming data conform with:

- the ontology used to describe these data
- custom policies defined for the data described by that ontology
- custom policies defined for the data coming from a particular data source

Based on that, we correct syntactical errors in the incoming data, score the incoming data and either send the data to the clean database, or drop the data. As part of the project realization, we will provide several sample sets of policies applicable to real world data sources.

Linking the data (implemented by Object Identification & Record Linkage component)
Since the data describing the same concepts (e.g. persons, cars, emotions) can be identified by various identifiers (URIs), the application will support specification of rules, which will apply to the incoming data and try to reveal whether the new incoming data represents a new concept (not already involved in the clean database) or a concept already involved in the clean database; in the latter case, the application will create a link ("same as" link) specifying that the given two sets of data are representing the same concept. The component will also support creation of arbitrary types of links between data (not just "same as" links). We will use Silk engine [4] and its specification language [5] to specify the policies for this component.

Providing data (implemented by Query Execution & Conflict Resolution component)
The main purpose of the project is to provide data aggregated from various sources and according to consumers' needs.

Data consumers can retrieve data about various concepts via their URI identifiers (see Linked Data principles [2]) or by specifying keywords. The response contains all the data known about the relevant concepts, together with the provenance metadata (who created/published it, when etc.) and with a score; the score is based on the results of the application of policies in the Error Localization component.

When the data is retrieved and prepared for the consumer, we will solve conflicts among the data (various data sources may provide conflicting values for the same RDF properties) by the preferred *conflict resolution policy*, which can be specified by the consumer; otherwise, we will use default conflict resolution policies.

Consumers can also query the application using SPARQL query language [10]. In this case, however, the information about metadata and data quality scores may be limited and the conflict resolution will not be supported.

Ontology maintenance & mapping
The project will maintain ontologies describing the consumed data and enable creation of mappings between these ontologies (e.g. to express that one property is "same as" as another property) which is a crucial aspect when aggregating data. In particular, mappings between ontologies will be taken into account during query answering in the Query Execution component – the answer of the query containing a property from one ontology in its definition may also involve resulting data described by another ontology mapped to the ontology in the definition of the query via a proper ontology mapping.

Roles
The application will support several users' roles:

- administrator - rights to assign other roles, particularize settings of the application, components

- ontology creator - rights to adjust ontologies, ontology mappings, adjust policies for the Error Localization component specific for the given ontology (the second type of policies, see Error Localization component)
- policy creator - rights to adjust policies for the Error Localization component specific for the given data source (the third type of policies), rights to write policies for Object Identification & Record Linkage component
- scraper - rights to insert data, list registered ontologies
- user - rights to query the application (via URI, keywords, or SPARQL query)

Graphical user interface
The application will involve graphical user interface enabling:

- management of all kinds of policies (policies for Error Localization and Object Identification & Record Linkage components, conflict resolution policies, and ontology mapping policies), debugging of policies for Error Localization component by showing which data match the given policies
- management of roles and rights of the application
- management of settings of the components and of the whole application

Expected Utilization of the Team:

(Including analysis, documentation, and testing of the introduced parts):

- Error Localization component (1 person)
- Query Execution & Conflict Resolution component (1 person)
- Object Identification & Record Linkage component (includes ontology mappings) (0.3 persons, reusing Silk engine)
- Application's core - creating and launching components; communication with data consumers/providers; storage configuration; roles management (1.7 persons)
- Graphical User Interface (1 person)

Expected Work Plan:

We will implement prototype of the project with restricted functionality as soon as possible (in Month 4). The work plan is as follows, assuming deadline and the final version of the project in Month 9:

- general analysis, specification, architecture; specification of the components (Months 1 - 2)
- implementing prototype with restricted functionality (Months 3 - 4)
- testing prototype (Month 4)
- implementing full specification (Months 5 - 8)
- testing full specification (Months 8 - 9)
- documentation (user, programmer, configuration guide) (Months 4 - 9)

Other Requirements on the Project

- The final documentations of the project will be in English.
- The application will be freely available under Apache Software License [6].
- It should be easy to incorporate other future components, such as a component computing popularity of the data sources.
- It should be easy to adjust query execution/conflict resolution component to take into account different types of custom policies submitted by the data consumer, such as context (provenance) policies (e.g. "Distrust data coming from a data source http://example.com", "Prefer data with the license ...") and content policies ("Distrust data older than 1 year").

Other Notes

During the work on the project, we will stay in touch with Digital Enterprise Research Institute (DERI) in Ireland [8], one of the biggest Semantic Web Research institutes in the world with a specialized Linked Data Research Centre [9]. We will also discuss the proposed techniques with the Carlo Batini's team at University of Milano-Bicocca [7].

References

[1] Bizer, Ch., Heath, T. and Berners-Lee, T. Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems 5, 1-22 (2009).

[2] Berners-Lee, T. Linked Data - Design Issues. http://www.w3.org/DesignIssues/LinkedData.html

[3] http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/

[4] http://www4.wiwiss.fu-berlin.de/bizer/silk/

[5] http://www.assembla.com/spaces/silk/wiki/Link_Specification_Language

[6] http://www.apache.org/licenses/LICENSE-2.0

[7] http://www.unimib.it/go/page/Italiano/Elenco-Docenti/BATINI-CARLO

[8] Digital Enterprise Research Institute, National University of Ireland, Galway. http://www.deri.ie/

[9] Linked Data Research Centre, DERI. http://linkeddata.deri.ie/

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