Collaborative Project

LOD2 - Creating Knowledge out of Interlinked Data

Project Number: 257943 Start Date of Project: 01/09/2010 Duration: 48 months

Deliverable 3.1.1
Report on Knowledge Extraction from Structured Sources

Dissemination Level	Public
Due Date of Deliverable	Month 6, 28/2/2011
Actual Submission Date	15/03/2011
Work Package	WP3, Knowledge Base Creation, Enrichment and Repair
Task	Task T3.1
Type	Report
Approval Status	Approved
Version	1.0
Number of Pages	95
Filename	deliverable-3.1.1.pdf

Abstract: The deliverable contains a survey of Knowledge Extraction from Structured Sources. It contains a general definition of Knowledge Extraction, an introduction to current progress on conversion of relational databases to RDF and a collection and classification of existing tool support.

Project funded by the European Commission within the Seventh Framework Programme (2007 – 2013)

History

Version	Date	Reason	Revised by
0.1	2010-01-05	Initial version	Jens Lehmann
0.2	2011-01-18	Created Structure	Sebastian Hellmann
0.3	2011-02-01	First draft for R2RML and Virtuoso Chapter	Hugh Williams
0.4	2011-02-04	Created collection of related EU Projects	Jens Lehmann
0.5	2011-02-07	Finished review of existing criteria	Sebastian Hellmann
0.6	2011-02-10	Setup OntoWiki and refined criteria	Jörg Unbehauen
0.7	2011-02-14	Tool collection started	ULEI, OpenLink and SWC
0.8	2011-02-14	Created draft for Wikipedia article	Sebastian Hellmann
0.9	2011-02-20	Finished Chapter 3 and 4	Hugh Williams
1.0	2011-02-28	Tool collection finished	All
1.1	2011-03-02	Improvements on Wikipedia Article	Sebastian Hellmann
1.2	2011-03-04	Fixes on OntoWiki (data.lod2.eu)	Sebastian Tramp
1.3	2011-03-05	Finished Export of data from OntoWiki	Sebastian Hellmann
1.4	2011-03-07	Addition of screenshots	Amrapali Zaveri
1.5	2011-03-10	Introduction and conclusions	Sebastian Hellmann
1.6	2011-03-14	Layout	Sebastian Hellmann

Author list

Sebastian Hellmann (ULEI), Jörg Unbehauen (ULEI), Amrapali Zaveri (ULEI), Jens Lehmann (ULEI), Sören Auer (ULEI), Sebastian Tramp (ULEI), Hugh Williams (OpenLink), Orri Erling (OpenLink), Ted Thibodeau Jr. (OpenLink), Kingsley Idehen (OpenLink), Andreas Blumauer (SWC), Helmut Nagy (SWC)

Executive summary

This report contains a survey of Knowledge Extraction from structured sources such as relational databases, XML and CSV. As the existing literature was either too specialized or incomplete, a general definition of *Knowledge Extraction* was created that covers structured as well as unstructured sources (Chapter 2). A summary of the current progress on conversion of relational databases to RDF is given (Chapter 3), followed by a detailed description of an examplary tool (Chapter 4), which shall enable the reader to gain an in-depth familiarity with the topic. Based on the definition of Knowledge Extraction and existing surveys on knowledge extraction from relational databases, classification criteria were developed and refined in a *Knowledge Extraction Tool Survey Schema* OWL ontology (Chapter 5) Finally, almost 30 existing tools (implementations available) were collected and classified according to this schema (Chapter 6). Based on the work in this deliverable several online resources were created for public dissemination.

The contents of this deliverable should enable the members of the LOD2 consortium and interested third parties to find an appropriate tool for their knowledge extraction use case. As the surveyed tools produce RDF and OWL data, the integration into the LOD2 stack can be performed in an ad-hoc manner by loading the resulting RDF into the LOD2 Knowledge Base.

Contents

Chapter 1

Introduction

In recent years, the availability of data in Semantic Web formats such as RDF and OWL has drastically increased. Nevertheless, the data that is currently available constitutes just a fraction of existing data that could be exposed and distributed as RDF and OWL. As the Web of Data, envisioned by Tim Berners-Lee[1], gains momentum, the demand to "triplify" data is steadily increasing, especially in the areas of commerce, science and government. This "triplification" process, however, is normally not easily implemented and executed. Although tools exist to support the generation of RDF from legacy sources, several obstacles remain and are intensified for automated approaches. The following list gives examples for these obstacles and cost factors:

Identification of private and public data. Legacy sources always contain information which should not be made public on the Web such as passwords, email addresses or technical parameters and configurations. Automatically distinguishing between strictly confidential, important and less relevant information is very hard, if not impossible.

Proper reuse of existing vocabularies. Even the most elaborated approaches to ontology mapping fail in generating certain mappings between the legacy data (e.g. database entities such as table and column names) and existing RDF vocabularies, due to lacking machine-readable descriptions of the domain semantics in the database schema.

Missing schema descriptions. Many legacy sources do neither provide proper documentation nor extensive schema definition (e.g. MySQL does not contain definitions for foreign keys or constraints, XML Data Type definition only provides information about the validity of the data, but not about the Semantics). Syntactic approaches for detecting schema descriptions are likely to fail, since schemas are often grown

[1] `http://www.w3.org/DesignIssues/Semantic.html`

evolutionary and naming conventions are often not enforced. In most cases the structure of the data needs to be manually reverse engineered by a domain expert, who has an understanding of the content and the domain.

URI Generation. The quality of legacy data sources do often not match the requirements for RDF datatypes and URIs. Strings and terms have to be normalized and cleaned to admit a transition to URIs. The choice which entities to use for identifiers (e.g. convert primary keys to URIs) is not always obvious.

To aid users in the "triplification" of their data this deliverable makes the following contributions:

Providing a definition of *Knowledge Extraction* to help understand and define, what "triplification" means. The definition is accompanied with several examples and put into right context respective other technology areas. (Chapter 2)

For further reading, a detailed introduction into two key technologies of the consortium is given. This in detailed example helps to gain deeper insight into the matter. (Chapter 4)

An extensive tool survey aids in finding appropriate tools for Knowledge Extraction tasks.

Additionally information was made available in online resources such as Wikipedia[2] and a tool database (`http://tinyurl.com/KETSurvey`), which was created during the course of this deliverable.

The remainder of the document is structured as follows: Chapter 2 contains a general definition of Knowledge Extraction and several examples.

The current progress of the R2RML language developed by the RDB2RDF W3C Working Group is briefly described in Chapter 3 and there is an introduction into one of the key technologies of the consortium regarding extraction of knowledge from structured sources (Chapter 4).

Furthermore, several surveys were analysed (cf. Section 5.2) and criteria to classify approaches were collected. Based on these criteria a survey schema was developed as an OWL Ontology (cf. Section 5.3)[3] and data about Knowledge Extraction tools were collected online as Linked Data[4] (cf. Chapter 5 and 6).

The final two chapters are concerned with a survey about tool support. The data collected in this deliverable is available in structured form as Linked Data. An OntoWiki[5] was deployed and can be accessed and

[2] `http://en.wikipedia.org/wiki/Knowledge_extraction`
[3] `http://data.lod2.eu/2011/tools/ket/`
[4] `http://tinyurl.com/KETSurvey`
[5] `http://ontowiki.net/Projects/OntoWiki`

edited at `http://data.lod2.eu/2011/tools/` (note the extra / at the end). `http://tinyurl.com/KETSurvey` is the direct link to browse all Knowledge Extraction Tools. Collecting data in a structured form required the development of a schema. In Section 5.1, we will first list the general properties and features we collected about the tools. For this we re-used the *Description of a Project (DOAP)* Vocabulary[6]. As we discovered that some properties were inappropriate or missing we extended and changed certain properties. Although we undertook quite an effort to exhaustively collect all properties for each tool, certain data points could not be found and remain missing (e.g. some tools do not have a mailing list or a bug database). In the next step, we reviewed all surveys about converting relational databases to RDF (see Section 5.2) and extracted the most important features to create the Knowledge Extraction Tool Survey Schema (KET Schema). The properties of this survey ontology can be found in Section 5.3. In Section 5.4, we give a short tutorial on how to access the information online. Additionally, tools from `http://semanticweb.org/wiki/Tools` and `http://www.w3.org/2001/sw/wiki/Tools` are imported into the OntoWiki. In Chapter 6, the collected data can be found in form of a Tool Survey Table (Section 6.1) using the KET Schema and in a list giving all the other information (Section 6.2).

1.1 About

This deliverable is designed to be easily accessible and reusable in a sustainable way. We identified several interest groups who can benefit by a proper public dissemination. These interest groups consist of: 1. the LOD2 Consortium members, 2. the Semantic Web community, 3. the W3C RDB2RDF Working Group, 4. enterprises and web communities, who want to convert their existing data sources, 5. the general public.

Therefore a proper public dissemination needs to fulfill the following requirements:

1. the content of this deliverable should build upon existing resources and if possible strengthen these existing resources.

2. the content should be made available at popular *information crossroads*, not only at the deliverable download site.

3. if data and lists are collected, they should be made accessible in a structured format.

4. the deliverable should use an open license according to the Open Knowledge Definition[7].

[6] `http://trac.usefulinc.com/doap`
[7] `http://www.opendefinition.org/`

5. content should be kept in a location that permits editing and allows for continuous maintenance either by the consortium members or online communities and stakeholders.

Compiling the necessary content in a huge monolithic PDF document is inadequate and thus we inverted the traditional procedure. Instead of collecting content and adding it to a single document, we created and extended several online resources that bear the potential of facilitating discovery, reuse and maintenance of the provided content. This deliverable is – with the exception of Chapter 3, Chapter 4 and Section 5.2 – a snapshot of these online resources at the time of the creation of this PDF.

1.2 License

Chapter 2

Definition of Knowledge Extraction

As we began our research on the topic of knowledge extraction, it became obvious that no surveys exist that either cover all structured sources or provide a clear definition of the "triplification" process and the required prerequisites. Most approaches were driven by very specific use cases that came with a specific data source and required the transformation into RDF. The question that arose immediately was, what the properties of such a transformation were and how they differed from previous efforts. The following aspects were especially under-developed:

Clear boundaries to existing research areas. *Information Extraction* (TextMining), *Extract-Transform-Load* (ETL, Data Warehouse) and *Ontology Learning* were related, but what is the criteria for distinction from the methods analised in this report?

Although, the area of extraction of RDF from relational databases were well developed, they were hardly comparable to extraction methods employed on other sources, thus preventing generalisation.

Especially, the idea that "knowledge" is extracted was hard to grasp. Although, RDF and OWL can serve as knowledge representation formalisms, the mere usage of RDF/OWL as a format can not sufficiently define the notion of "knowledge". The main question is: "What is the result of a "triplification" process? Structured data or represented knowledge and when does structured data become knowledge.

As Wikipedia is becoming increasingly popular as a source for initial research and the collection of background information[1]. We reviewed several articles and found that the coverage of the topic was de facto non-existent.

[1] `http://www.dailyprincetonian.com/2010/03/23/25575/`

Therefore, we decided to bootstrap an article about Knowledge Extraction, which is included in this deliverable on the following pages. The online article can be found at `http://en.wikipedia.org/wiki/Knowledge_extraction`.

Knowledge extraction

Knowledge Extraction is the creation of knowledge from structured (relational databases, XML) and unstructured (text, documents, images) sources. The resulting knowledge needs to be in a machine-readable and machine-interpretable format and must represent knowledge in a manner that facilitates inferencing. Although it is methodical similar to Information Extraction (NLP) and ETL (Data Warehouse), the main criteria is that the extraction result goes beyond the creation of structured information or the transformation into a relational scheme. It requires either the reuse of existing formal knowledge (reusing identifiers or ontologies) or the generation of a schema based on the source data.

The RDB2RDF W3C group [1] is currently standardizing a language for extraction of RDF from relational databases. Another popular example for Knowledge Extraction is the transformation of Wikipedia into structured data and also the mapping to existing knowledge (see DBpedia, Freebase).

Overview

After the standardization of knowledge representation languages such as RDF and OWL, much research has been conducted in the area, especially regarding transforming relational databases into RDF, Entity resolution, Knowledge Discovery and Ontology Learning. The general process uses traditional methods from Information Extraction and ETL, which transform the data from the sources into structured formats.

The following criteria can be used to categorize approaches in this topic (some of them only account for extraction from relational databases):

Source	Which data sources are covered: Text, Relational Databases, XML, CSV
Exposition	How is the extracted knowledge made explicit (Ontology file, Semantic Database)? How can you query it?
Synchronization	Is the knowledge extraction process executed once to produce a dump or is the result synchronized with the source? Static or Dynamic. Are changes to the result written back (Bi-directional)
Reuse of vocabularies	The tool is able to reuse existing vocabularies in the extraction. For example the table column 'firstName' can be mapped to foaf:firstName. Some automatic approaches are not capable of mapping vocab.
Automatisation	The degree to which the extraction is assisted/automated. Manual, GUI, semi-automatic, automatic.
Requires a Domain Ontology	A pre-existing ontology is needed to map to it. So either a mapping is created or a schema is learned from the source (Ontology learning).

Examples

Entity Linking

1. DBpedia Spotlight [2], OpenCalais, the Zemanta API, and Extractiv [3] analyze free text via Named Entity Recognition and then disambiguates candidates via Name Resolution and links the found entities to the DBpedia knowledge repository[4] (DBpedia Spotlight web demo [5]).

   President Obama [6] called Wednesday on Congress [7] to extend a tax break for students included in last year's economic stimulus package, arguing that the policy provides more generous assistance.

   As President Obama is linked to a DBpedia LinkedData resource, further information can be retrieved automatically and a Semantic Reasoner can for example infer that the mentioned entity is of the type Person [8] (using FOAF_(software)) and of type Presidents of the United States [9] (using YAGO). Counter examples: Methods that only recognize entities or link to Wikipedia articles and other targets that do not provide further retrieval of structured data and formal knowledge.

Relational Databases to RDF

1. Triplify, D2R Server and Virtuoso RDF Views are tools that transform relational databases to RDF. During this process they allow to reuse existing vocabularies and ontologies during the conversion process. When transforming a typical relational table named *users*, one column (e.g.*name*) or an aggregation of columns (e.g.*first_name* and *last_name*) has to provide the URI of the created entity. Normally the primary key is used. Every other column can be extracted as a relation with this entity[10] . Then properties with formally defined semantics are used (and reused) to interpret the information. For example a column in a user table called *marriedTo* can be defined as symmetrical relation and a column *homepage* can be converted to a property from the FOAF Vocabulary called foaf:homepage [11], thus qualifying it as an inverse functional property. Then each entry of the *user* table can be made an instance of the class foaf:Person [12] (Ontology Population). Additionally domain knowledge (in form of an ontology) could be created from the *status_id*, either by manually created rules (if *status_id* is 2, the entry belongs to class Teacher) or by (semi)-automated methods (Ontology Learning). Here is an example transformation:

Name	marriedTo	homepage	status_id
Peter	Marry	http://example.org/Peters_page	1
Claus	Eva	http://example.org/Claus_page	2

```
:Peter :marriedTo :Marry .
:marriedTo a owl:SymmetricProperty .
:Peter foaf:homepage  <http://example.org/Peters_page> .
:Peter a foaf:Person .
:Peter a :Student .
:Claus a :Teacher .
```

Extraction from structured sources to RDF

1:1 Mapping from RDB Tables/Views to RDF Entities/Attributes/Values [13]

When building a RDB representation of a problem domain, the starting point is frequently an entity-relationship diagram (ERD). Typically, each entity is represented as a database table, each attribute of the entity becomes a column in that table, and relationships between entities are indicated by foreign keys. Each table typically defines a particular class of entity, each column one of its attributes. Each row in the table describes an entity instance, uniquely identified by a primary key. The table rows collectively describe an entity set. In an equivalent RDF representation of the same entity set:

- Each column in the table is an attribute (i.e., predicate)
- Each column value is an attribute value (i.e., object)
- Each row key represents an entity ID (i.e., subject)
- Each row represents an entity instance
- Each row (entity instance) is represented in RDF by a collection of triples with a common subject (entity ID).

So, to render an equivalent view based on RDF semantics, the basic mapping algorithm would be as follows:

1. create an RDFS class for each table
2. convert all primary keys and foreign keys into IRIs
3. assign a predicate IRI to each column
4. assign an rdf:type predicate for each row, linking it to an RDFS class IRI corresponding to the table
5. for each column that is neither part of a primary or foreign key, construct a triple containing the primary key IRI as the subject, the column IRI as the predicate and the column's value as the object.

Early mentioning of this basic or direct mapping can be found in Tim Berners-Lee's comparison of the ER model to the RDF model. [10]

Complex mappings of relational databases to RDF

The 1:1 mapping mentioned above exposes the legacy data as RDF in a straightforward way, additional refinements can be employed to improve the usefulness of RDF output respective the given Use Cases. Normally, information is lost during the transformation of an entity-relationship diagram (ERD) to relational tables (Details can be found in Object-relational impedance mismatch) and has to be reverse engineered. From a conceptual view, approaches for extraction can come from two directions. The first direction tries to extract or learn an OWL schema from the given database schema. Early approaches used a fixed amount of manually created mapping rules to refine the 1:1 mapping[14] [15] [16] . More elaborate methods are employing heuristics or learning algorithms to induce schematic information (methods overlap with Ontology learning). While some approaches try to extract the information from the structure inherent in the SQL schema[17] (analysing e.g. foreign keys), others analyse the content and the values in the tables to create conceptual hierarchies[18] (e.g. a columns with few values are candidates for becoming categories). The second direction tries to map the schema and its contents to a pre-existing domain ontology (see also: Ontology alignment). Often, however, a suitable domain ontology does not exist and has to be created first.

XML

As XML is structured as a tree, any data can be easily represented in RDF, which is structured as a graph. XML2RDF [19] is one example of an approach that uses RDF blank nodes and transforms XML elements and attributes to RDF properties. The topic however is more complex as in the case of relational databases. In a relational table the primary key is an ideal candidate for becoming the subject of the extracted triples. An XML element, however, can be transformed - depending on the context- as a subject, a predicate or object of a triple. XSLT can be used a standard transformation language to manually convert XML to RDF.

Survey of Methods / Tools

Name	Data Source	Data Exposition	Data Synchronisation	Mapping Language	Vocabulary Reuse	Mapping Automat.	Req. Domain Ontology	Uses GUI
A Direct Mapping of Relational Data to RDF [20]	Relational Data							
CSV2RDF4LOD [21]	CSV	ETL	static	none	true	manual	false	false
Convert2RDF [22]	Delimited text file	ETL	static	RDF/DAML	true	manual	false	true
D2R Server [23]	RDB	SPARQL	bi-directional	D2R Map	true	manual	false	false
DartGrid [24]	RDB	own query language	dynamic	Visual Tool	true	manual	false	true
DataMaster [25]	RDB	ETL	static	proprietary	true	manual	true	true
Google Refine's RDF Extension [26]	CSV, XML	ETL	static	none		semi-automatic	false	true
Krextor [27]	XML	ETL	static	xslt	true	manual	true	false

MAPONTO [28]	RDB	ETL	static	proprietary	true	manual	true	false
METAmorphoses [29]	RDB	ETL	static	proprietary xml based mapping language	true	manual	false	true
MappingMaster [30]	CSV	ETL	static	MappingMaster	true	GUI	false	true
ODEMapster [31]	RDB	ETL	static	proprietary	true	manual	true	true
OntoWiki CSV Importer Plug-in - DataCube & Tabular [32]	CSV	ETL	static	The RDF Data Cube Vocaublary	true	semi-automatic	false	true
Poolparty Extraktor (PPX) [33]	XML, Text	LinkedData	dynamic	RDF (SKOS)	true	semi-automatic	true	false
RDBToOnto [34]	RDB	ETL	static	none	false	automatic, the user furthermore has the chance to fine-tune results	false	true
RDF 123 [35]	CSV	ETL	static	false	false	manual	false	true
RDOTE [36]	RDB	ETL	static	SQL	true	manual	true	true
Relational.OWL [37]	RDB	ETL	static	none	false	automatic	false	false
T2LD [38]	CSV	ETL	static	false	false	automatic	false	false
The RDF Data Cube Vocabulary [39]	Multidimensional statistical data in spreadsheets			Data Cube Vocabulary	true	manual	false	
TopBraid Composer [40]	CSV	ETL	static	SKOS	false	semi-automatic	false	true
Triplify [41]	RDB	LinkedData	dynamic	SQL	true	manual	false	false
Virtuoso RDF Views [42]	RDB	SPARQL	dynamic	Meta Schema Laguage	true	semi-automatic	false	true
Virtuoso Sponger [42]	structured and semi-structured data sources	SPARQL	dynamic	Virtuoso PL & XSLT	true	semi-automatic	false	false
VisAVis [43]	RDB	RDQL	dynamic	SQL	true	manual	true	true
XLWrap: Spreadsheet to RDF [44]	CSV	ETL	static	TriG Syntax	true	manual	false	false
XML to RDF [45]	XML	ETL	static	false	false	manual	false	false

Knowledge discovery

Knowledge discovery describes the process of automatically searching large volumes of data for patterns that can be considered knowledge *about* the data [46] . It is often described as *deriving* knowledge from the input data. Knowledge discovery developed out of the Data mining domain, and is closely related to it both in terms of methodology and terminology [47] .

The most well-known branch of data mining is knowledge discovery, also known as Knowledge Discovery in Databases (KDD). Just as many other forms of knowledge discovery it creates abstractions of the input data. The *knowledge* obtained through the process may become additional *data* that can be used for further usage and discovery.

Another promising application of knowledge discovery is in the area of software modernization which involves understanding existing software artifacts. This process is related to a concept of reverse engineering. Usually the knowledge obtained from existing software is presented in the form of models to which specific queries can be made when necessary. An entity relationship is a frequent format of representing knowledge obtained from existing software. Object Management Group (OMG) developed specification Knowledge Discovery Metamodel (KDM) which defines an ontology for the software assets and their relationships for the purpose of performing knowledge discovery of existing code. Knowledge discovery from existing software systems, also known as software mining is closely related to data mining, since existing software artifacts contain enormous business value, key for the evolution of software systems. Instead of mining individual data sets, software mining focuses on metadata, such as database schemas.

References

[1] RDB2RDF Working Group, Website: http://www.w3.org/2001/sw/rdb2rdf/, charter: http://www.w3.org/2009/08/rdb2rdf-charter, R2RML: RDB to RDF Mapping Language: http://www.w3.org/TR/r2rml/

[2] http://spotlight.dbpedia.org

[3] http://www.extractiv.com/demo.html

[4] "Life in the Linked Data Cloud" (http://www.opencalais.com/node/9501). www.opencalais.com. . Retrieved 2009-11-10. "Wikipedia has a Linked Data twin called DBpedia. DBpedia has the same structured information as Wikipedia – but translated into a machine-readable format."

[5] http://spotlight.dbpedia.org/rest/annotate?text=President%20Obama%20called%20Wednesday%20on%20Congress%20to%20extend%20a%20tax%20break%20for%20students%20included%20&confidence=0.2&support=20

[6] http://dbpedia.org/resource/Barack_Obama

[7] http://dbpedia.org/resource/United_States_Congress

[8] http://xmlns.com/foaf/0.1/Person

[9] http://dbpedia.org/class/yago/PresidentsOfTheUnitedStates

[10] Tim Berners-Lee (1998), "Relational Databases on the Semantic Web" (http://www.w3.org/DesignIssues/RDB-RDF.html). Retrieved: February 20, 2011.

[11] http://xmlns.com/foaf/spec/#term_homepage

[12] http://xmlns.com/foaf/spec/#term_Person

[13] OpenLink Software. Virtuoso RDF Views – Getting Started Guide (http://www.openlinksw.co.uk/virtuoso/Whitepapers/pdf/Virtuoso_SQL_to_RDF_Mapping.pdf). Retrieved February 20, 2011. Text posted by owner.

[14] Hu et al. (2007), “Discovering Simple Mappings Between Relational Database Schemas and Ontologies”, In Proc. of 6th International Semantic Web Conference (ISWC 2007), 2nd Asian Semantic Web Conference (ASWC 2007), LNCS 4825, pages 225-238, Busan, Korea, 11-15 November 2007. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.97.6934&rep=rep1&type=pdf

[15] R. Ghawi and N. Cullot (2007), "Database-to-Ontology Mapping Generation for Semantic Interoperability". In Third International Workshop on Database Interoperability (InterDB 2007). http://le2i.cnrs.fr/IMG/publications/InterDB07-Ghawi.pdf

[16] Li et al. (2005) "A Semi-automatic Ontology Acquisition Method for the Semantic Web", WAIM, volume 3739 of Lecture Notes in Computer Science, page 209-220. Springer. http://dx.doi.org/10.1007/11563952_19

[17] Tirmizi et al. (2008), “Translating SQL Applications to the Semantic Web”, Lecture Notes in Computer Science, Volume 5181/2008 (Database and Expert Systems Applications). http://citeseer.ist.psu.edu/viewdoc/download;jsessionid=15E8AB2A37BD06DAE59255A1AC3095F0?doi=10.1.1.140.3169&rep=rep1&type=pdf

[18] Farid Cerbah (2008). "Learning Highly Structured Semantic Repositories from Relational Databases", The Semantic Web: Research and Applications, volume 5021 of Lecture Notes in Computer Science, Springer, Berlin / Heidelberg http://www.tao-project.eu/resources/publications/cerbah-learning-highly-structured-semantic-repositories-from-relational-databases.pdf
[19] http://rhizomik.net/html/redefer/xml2rdf/
[20] http://www.w3.org/TR/rdb-direct-mapping/
[21] http://logd.tw.rpi.edu/technology/csv2rdf4lod
[22] http://www.mindswap.org/~mhgrove/ConvertToRDF/
[23] http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/
[24] http://ccnt.zju.edu.cn/projects/dartgrid
[25] http://protegewiki.stanford.edu/wiki/DataMaster
[26] http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/
[27] http://kwarc.info/projects/krextor/
[28] http://www.cs.toronto.edu/semanticweb/maponto/
[29] http://metamorphoses.sourceforge.net/
[30] http://protege.cim3.net/cgi-bin/wiki.pl?MappingMaster
[31] http://neon-toolkit.org/wiki/ODEMapster
[32] http://aksw.org/Projects/Stats2RDF
[33] http://poolparty.punkt.at/
[34] http://www.tao-project.eu/researchanddevelopment/demosanddownloads/RDBToOnto.html
[35] http://ebiquity.umbc.edu/project/html/id/82/RDF123
[36] http://sourceforge.net/projects/rdote/
[37] http://sourceforge.net/projects/relational-owl/
[38] http://ebiquity.umbc.edu/paper/html/id/480/T2LD-An-automatic-framework-for-extracting-interpreting-and-representing-tables-as-Linked-Data
[39] http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html
[40] http://www.topquadrant.com/products/TB_Composer.html
[41] http://triplify.org
[42] http://virtuoso.openlinksw.com
[43] http://www.cn.ntua.gr/~nkons/essays_en.html#t
[44] http://xlwrap.sourceforge.net/
[45] http://rhizomik.net/html/redefer/#XML2RDF
[46] Frawley William. F. et al. (1992), "Knowledge Discovery in Databases: An Overview", *AI Magazine* (Vol 13, No 3), 57-70 (online full version: http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/1011)
[47] Fayyad U. et al. (1996), "From Data Mining to Knowledge Discovery in Databases", *AI Magazine* (Vol 17, No 3), 37-54 (online full version: http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/1230

Article Sources and Contributors

Knowledge extraction *Source*: http://en.wikipedia.org/w/index.php?oldid=417596933 *Contributors*: Gregbard, KingsleyIdehen, MacTed, SebastianHellmann, Soeren1611, 1 anonymous edits

License

Chapter 3

R2RML - RDB to RDF Mapping Language

The R2RML working draft[1] specifies an RDF notation for mapping relational tables, views or queries into RDF. The primary area of applicability of this is extracting RDF from relational databases, but in special cases R2RML could lend itself to on-the-fly translation of SPARQL into SQL or to converting RDF data to a relational form. The latter application is not the primary intended use of R2RML but may be desirable for importing Linked Data into relational stores. This is possible if the constituent mappings and underlying SQL objects constitute updateable views in the SQL sense.

Data integration is often mentioned as a motivating use case for the adoption of RDF. This integration will very often be between relational databases which have logical entities in common, each with its local schema and identifiers. Thus, we expect to see relational to RDF mapping use cases involving the possibility of a triple coming from multiple sources. This does not present any problem if RDF is being extracted but does lead to complications if SPARQL queries are mapped into SQL. In specific, one will end up with potentially very long queries consisting of joins of unions. Most of the joins between terms of the unions will often be provably empty and can thus be optimized away. This capability however requires the mapping language to be able to express metadata about mappings, i.e. that IRI' s coming from one place are always disjoint from IRI's coming from another place. Without such metadata optimizing SPARQL to SQL translation is not possible, which will significantly limit the possibility of querying collections of SQL databases through a SPARQL end point without ETL-ing the mapped RDF into an RDF store. Data integration is often mentioned as a motivating use case for the adoption of RDF. This integration will very often be between relational databases which have logical entities in common,

[1] `http://www.w3.org/TR/r2rml/`

each with its local schema and identifiers. Thus, we expect to see relational to RDF mapping use cases involving the possibility of a triple coming from multiple sources. This does not present any problem if RDF is being extracted but does lead to complications if SPARQL queries are mapped into SQL. In specific, one will end up with potentially very long queries consisting of joins of unions. Most of the joins between terms of the unions will often be provably empty and can thus be optimized away. This capability however requires the mapping language to be able to express metadata about mappings, i.e. that IRI' s coming from one place are always disjoint from IRI's coming from another place. Without such metadata optimizing SPARQL to SQL translation is not possible, which will significantly limit the possibility of querying collections of SQL databases through a SPARQL end point without ETL-ing the mapped RDF into an RDF store.

RDF is emerging as a format for interoperable data publishing. This does not entail that RDF were preferable as a data warehousing model. Besides, for large warehouses, RDF is far from cost competitive with relational technology, even though LOD2 expects to narrow this gap. Thus it follows that on the fly mapping of SPARQL to SQL will be important. Regardless of the relative cost or performance of relational or RDF technology, it is not a feasible proposition to convert relational warehouses to RDF in general, rather existing investments must be protected and reused. Due to these reasons, R2RML will have to evolve in the direction of facilitating querying of federated relational resources.

Chapter 4

Knowledge Extraction with Virtuoso (Detailed Introduction)

4.1 Virtuoso RDFViews

As most of the world's data resides in relational databases and is clearly of value to the emerging Semantic Web, there is an obvious need to expose this data as RDF. Because of databases' tried and trusted strengths in terms of performance, security, maintainability and so forth, the core data should remain in the database, rather than be duplicated in RDF form outside the DBMS. Thus a key infrastructural requirement is a technology that enables the dynamic generation/mapping of RDF views of relational data. Virtuoso provides such a capability through its RDF Views support.

4.1.1 Virtuoso Meta-Schema Language

"RDF Views" is actually a moniker referring to the two key technologies at the heart of Virtuoso's RDF support – Virtuoso's **RDF Meta-Schema** and its declarative **Meta-Schema Language** for mapping SQL data to RDF ontologies.

What is a meta-schema language? A general definition might be "a declarative language for expressing relationships in abstract data models". Based on this definition, the Virtuoso Meta Schema Language is a domain-specific extension of this concept for mapping a logical data model expressed in SQL to a conceptual data model expressed in RDF. Before examining Virtuoso's RDF Views and Meta-Schema Language in detail however, we need first to look at some fundamental mapping concepts; that is: how data can be modeled conceptually through ontologies, how ontologies can be represented in RDF, and how, in the broadest conceptual terms, SQL

data can be mapped to RDF.

4.1.2 A Conceptual View of SQL to RDF Entity Mapping

At the most basic level, Virtuoso's RDF Views transform the result set of a SQL SELECT statement into a set of triples. Before describing how these transformations are defined using Virtuoso's Meta-Schema Language, it is worth considering how, in general terms, SQL data can be transformed to RDF.

When building a SQL representation of a problem domain, the starting point is frequently an entity-relationship diagram (ERD). Typically, each entity is represented as a database table, each attribute of the entity becomes a column in that table, and relationships between entities are indicated by foreign keys. Each table typically defines a particular class of entity, each column one of its attributes. Each row in the table describes an entity instance, uniquely identified by a primary key. The table rows collectively describe an entity set.

In an equivalent RDF representation of the same entity set:

- Each column in the table is an attribute (i.e. predicate)
- Each column value is an attribute value (i.e. object)
- Each row key represents an entity ID (i.e. subject)
- Each row represents an entity instance
- Each row (entity instance) is represented in RDF by a collection of triples with a common subject (entity ID).

So, to render an equivalent view in RDF, in the simplest case, a basic algorithm could be:

1. Create an RDFS class for each table
2. Convert all primary keys and foreign keys into IRI's
3. Assign a predicate IRI to each column
4. Assign an rdf:type predicate for each row, linking it to an RDFS class IRI corresponding to the table
5. For each column that is neither part of a primary or foreign key, construct a triple containing the primary key IRI as the subject, the column IRI as the predicate and the column's value as the object.

Based on the above algorithm, some of the key requirements are:

- Definition of a RDFS class and IRI for each table

- Construction of a predicate IRI for each non-key column
- Construction of an IRI for each primary key value

4.1.3 Virtuoso RDF Views Creation

Virtuoso RDF Views expose pre-existing relational data as virtual RDF graphs available for querying directly through SPARQL or via Virtuoso's in-built support for SPARQL embedded within SQL (SPASQL). The virtual RDF graphs are created without physically regenerating the relational data as RDF data sets. As indicated earlier, the key components of Virtuoso RDF Views are the Virtuoso RDF Meta-Schema and the RDF Meta-Schema Language.

4.1.4 Virtuoso RDF Meta-Schema Language

The building blocks of the meta schema are quad map patterns, IRI classes and literal classes. Other meta-schema features such as group map patterns and quad storage are essentially organizational enhancements aimed at making it easier to administer large sets of quad map patterns through the use of grouping and naming:

- **group map patterns**: group together map patterns which share a common graph
- **quad storage**: groups together group map patterns as a named set

Naming is used at three levels – **quad map patterns**, group map patterns and quad storage can all be named to facilitate altering or deleting map patterns individually, or as a group at the group map pattern or quad storage level. An additional benefit of naming is easier debugging and more readable debug output.

4.1.5 Quad Map Patterns

The basic unit of the meta schema is a **quad map pattern**. A simple quad map pattern fully defines one particular transformation from one set of relational columns into triples that match one SPARQL graph pattern. At its heart, an RDF view definition is simply a collection of quad map patterns.

The main part of a quad map pattern is four declarations of **quad map values**, with each declaration specifying how to calculate the value of the corresponding triple field from the SQL data.

4.1.6 Named Quad Map Patterns

Quad map patterns (aka quad patterns) can be named. The assigned name then acts as a logical name which identifies the combination of a named graph and its associated triple pattern.

4.1.7 Group Map Patterns

Quad map patterns for the same graph can be grouped together into a **group map pattern**. **Named Group Map Patterns** like quad patterns, group map patterns can also be named.

4.1.8 Quad Storage

Quad storage is a named set of quad patterns, used for compartimentalizing the RDF to SQL mapping. Quad patterns contained by a particular quad storage can then be manipulated en-bloc. The three statements for manipulating storages are:

- create quad storage *storage-name quad-map declarations* .
- alter quad storage *storage-name quad-map declarations or drop commands*
- drop quad storage *storage-name* .

A map pattern can only be created within a quad storage definition, as a part of **create quad storage** or **alter quad storage** statement. (Initially, the map pattern is used by only one storage but, once created, map patterns can be imported from one quad storage into another). The **drop quad storage** statement deletes the named quad storage and all contained quad patterns. Quad map patterns can be deleted individually using the **drop quad map** map-name directive. When used inside an **alter quad storage** statement it removes a map only from that quad storage, otherwise it removes the map from all storages.

4.1.9 IRI Classes – Creating IRIs from Keys

Recall in the earlier section "A Conceptual View of SQL to RDF Entity Mapping", where we presented a 1:1 Mapping of a SQL Table/View to an RDF Entity, that one of the key requirements identified in the mapping process was:

- Construction of a subject IRI for each primary key column value

An IRI class performs this 'construction'. It defines how key values (for an atomic or compound key) are combined into an IRI string and how an IRI string is decomposed back into the key value(s). When declaring that a table's primary key is converted into a IRI according to one IRI class, one usually declares that all foreign keys referring to this class also get converted into an IRI using the same class.

4.1.10 Recent Enhancements

Recent enhancements in Virtuoso RDF Views support enable the materialization and synchronization of the RDF View generated triples with the RDF Quad store using RDB2RDF Triggers. This enables standard SPARQL operations like inferencing, faceted browsing and others to be performed on the materialized triples. To perform this task the required RDB data objects must first be linked into Virtuoso, from which local incrementally snapshot replicated copies can be created and automatically kept in sync. A standard set of RDF Views can then be created of these locally replicated objects on which a set of RDB2RDF Triggers are created for converting the local RDFViews to physical triples and keeping both in sync.

4.1.11 Conclusions

We have described how we can arbitrarily map relational database schemas to RDF ontologies en route to generating virtual RDF Data Sets (Graphs) that are then accessible to SPARQL Queries from within SQL or via the SPARQL Query Protocol. All of this is achieved without compromising the inherent flexibility of the RDF data model or the SPARQL Query Language.

It should also be noted that all of the functionality demonstrated also applies to the Virtual DBMS functionality realm of Virtuoso. Thus, you can now map 3rd party ODBC or JDBC accessible SQL data to RDF on the fly. Likewise, you can also use the same Virtual DBMS layer to map data exposed via local or 3rd party SOAP or REST based Web Services to RDF.

In addition to providing immense power and flexibility at the data mapping level, we have also paid great attention to the low level optimization of Virtuoso's underlying RDF storage engine (Triple or Quad Store).

4.2 Virtuoso Sponger Cartridges (RDF'izers)

The Virtuoso Sponger is the Linked Data middleware component of Virtuoso that generates Linked Data from a variety of structured and semi-structured data sources, supporting a wide variety of data representation and serialization formats. The sponger is transparently integrated into Virtuoso's SPARQL Query Processor where it delivers URI de-referencing

within SPARQL query patterns, across disparate data spaces. It also delivers configurable smart HTTP caching services. Optionally, it can be used by the Virtuoso Content Crawler for structured data ingestion, periodically populating and replenishing data within the native RDF Quad Store.

The sponger is a fully fledged HTTP proxy service that is also directly accessible via SOAP or REST interfaces. OpenLink's broad portfolio of Linked-Data-aware products supports a number of routes for creating or consuming Linked Data. The Sponger provides a key platform for developers to generate quality data meshes from unstructured or semi-structured data sources.

Architecturally, the Sponger is comprised of two types of cartridges, Extractor and Meta Cartridges. Extractor Cartridges focus on data extraction and transformation services while the Meta Cartridges provide look ups and joins across other Linked Data spaces and Web 2.0 APIs. Both cartridge types are themselves comprised of a data extractors and RDF Schema/Ontology Mapper components. Cartridges are highly customizable. Custom cartridges can be developed using any language supported by the Virtuoso Server Extensions API enabling structured Linked Data generation from resource types not available in the default Sponger Cartridge collection bundled – as part of the Virtuoso Sponger VAD package (rdf_mappers_dav.vad).

The Virtuoso Sponger is fully extensible by virtue of its pluggable cartridge architecture. New data formats can be sponged by creating new cartridges. While OpenLink is active in adding cartridges for new data sources, third parties are free to develop their own custom cartridges. Entity extractors can be built using Virtuoso PL, C/C++, Java or any other external language supported by Virtuoso's Server Extension API. Writing a new cartridge should only be necessary to generate RDF from a REST-style Web service not supported by an existing cartridge, or to customize the output from an existing cartridge to your own requirements. Apart from these circumstances, the existing Sponger infrastructure should meet most users needs.

Figure 4.1 provides a visualization of the layering of the Virtuoso RDF Views and Sponger Cartridge RDF Data Integration Middleware components in the Virtuoso Universal Server architecture .

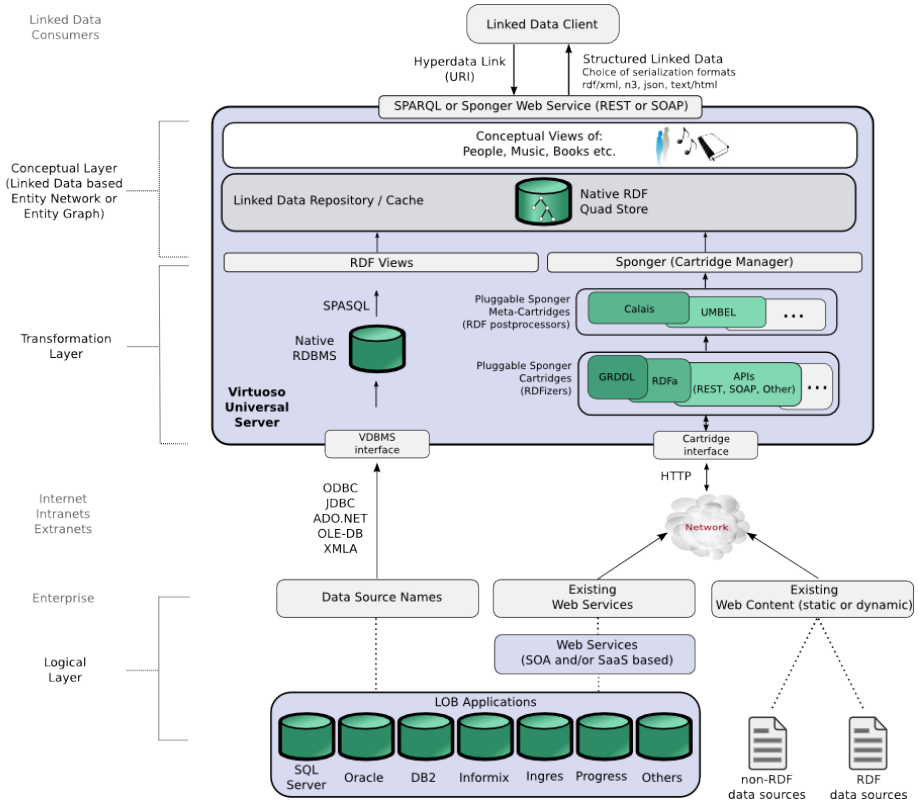


Figure 4.1: Sponger Cartridge RDF Data Integration Middleware

Chapter 5

Categorisation and Typology of Knowledge Extraction Tools

The data collected in this deliverable is available in structured form as Linked Data. An OntoWiki[1] was deployed and can be accessed and edited at `http://data.lod2.eu/2011/tools/` (note the extra / at the end). `http://tinyurl.com/KETSurvey` is the direct link to browse all Knowledge Extraction Tools. Collecting data in a structured form required the development of a schema. In Section 5.1, we will first list the general properties and features we collected about the tools. For this we re-used the *Description of a Project (DOAP)* Vocabulary[2]. As we discovered that some properties were inappropriate or missing we extended and changed certain properties. Although we undertook quite an effort to exhaustively collect all properties for each tool, certain data points could not be found and remain missing (e.g. some tools do not have a mailing list or a bug database). In the next step, we reviewed all surveys about converting relational databases to RDF (see Section 5.2) and extracted the most important features to create a Knowledge Extraction Tool Survey Schema (KET Schema). The properties of this survey ontology can be found in Section 5.3. In Section 5.4, we give a short tutorial on how to access the information online. Additionally, tools from `http://semanticweb.org/wiki/Tools` and `http://www.w3.org/2001/sw/wiki/Tools` are imported into the OntoWiki. In Chapter 6, the collected data can be found in form of a Tool Survey Table (Section 6.1) using the KET Schema and in a list giving all the other information (Section 6.2).

[1] `http://ontowiki.net/Projects/OntoWiki`
[2] `http://trac.usefulinc.com/doap`

5.1 General information about tools

bibsonomy url
`http://data.lod2.eu/2011/tools/bibsonomyUrl`
the URL of a BibSonomy entry

current release date
`http://data.lod2.eu/2011/tools/currentReleaseDate`
the date when the current (latest) release was published

current version
`http://data.lod2.eu/2011/tools/currentVersion`
a string describing the current (latest) release number

documentation page
`http://data.lod2.eu/2011/tools/documentationPage`
a link to the documentation page

maintainer
`http://data.lod2.eu/2011/tools/maintainer`
the people who are currently maintaining the tool / project

sources
`http://data.lod2.eu/2011/tools/sources`
just post some links to Subversion, Mercurial, etc.

status
`http://data.lod2.eu/2011/tools/status`
describes the current status of the tool. Can be e.g. proof-of-concept, pre-alpha, alpha, beta, stable, mature, discontinued

usage examples
`http://data.lod2.eu/2011/tools/usageExamples`
either a reference deployment of the tool or some example code

usage examples
`http://data.lod2.eu/2011/tools/supportedOS`
either a reference deployment of the tool or some example code

Title
`http://purl.org/dc/elements/1.1/title`
A name given to the resource.

bug database
`http://usefulinc.com/ns/doap#bug-database`
Bug tracker for a project.

category
`http://usefulinc.com/ns/doap#category`
A category of project.

description
`http://usefulinc.com/ns/doap#description`
Plain text description of a project, of 2-4 sentences in length.

developer
`http://usefulinc.com/ns/doap#developer`
Developer of software for the project.

download page
`http://usefulinc.com/ns/doap#download-page`
Web page from which the project software can be downloaded.

homepage
`http://usefulinc.com/ns/doap#homepage`
URL of a project's homepage, associated with exactly one project.

mailing list
`http://usefulinc.com/ns/doap#mailing-list`
Mailing list home page or email address.

name
`http://usefulinc.com/ns/doap#name`
A name of something.

programming language
`http://usefulinc.com/ns/doap#programming-language`
Programming language a project is implemented in or intended for use with.

short description
`http://usefulinc.com/ns/doap#shortdesc`
Short (8 or 9 words) plain text description of a project.

5.2 Review of existing criteria for knowledge extraction from relational databases

During our research we discovered the following sources, which discuss criteria for classifying knowledge extraction approaches from relational databases.

- In Auer et al. [3] a tool called Triplify was introduced. In the paper's Related Work Section is a table that compares several approaches with the help of 5 criteria. These criteria were based on previous work of the RDB2RDF Incubator Group, but were much more refined. We also considered the survey report [15] of the RDB2RDF Incubator Group, but could not find any new criteria to add to the material presented in [3].
- An early paper published in 2007 by Ghawi and Cullot [7].
- Another source of input is the work by Konstantinou, Spanos and Mitrou [10]. Spanos, Stavrou and Mitrou recently submitted another survey[3] to the Semantic Web Journal. Due to the open review process the information is available for consideration although it was not yet peer-reviewed.

5.2.1 Triplify and RDB2RDF Survey report

The table displayed in Figure 5.1 is taken from the Triplify WWW paper[3]. The survey report [15] furthermore contained a chart(see Figure 5.2) showing the reference framework for classifying the approaches and an extensive table classifying the approaches (see Figure 5.3).

Approach	**Automation** (a)	**Domain or database semantics-driven** (b)	**Access paradigm** (c)	**Mapping language** (d)	**Domain reliance** (e)
Dartgrid [17]	Manual	Domain	SPARQL	Visual Tool	dependent
Hu et al. [11]	Auto	Both	ETL	intern	dependent
Tirmizi et al. [16]	Auto	DB	ETL	FOL	general
Li et al. [12]	Semi	DB	ETL	n/a	general
DB2OWL[10]	Semi	DB	SPARQL	R2O	general/dependent
RDBToOnto [6]	Semi	DB+M	ETL	Visual Tool	general
Sahoo et al. [15]	Manual	Domain	ETL	XSLT	dependent
R2O[13]	Manual	DB+M	SPARQL	R2O	dependent
D2RQ[4]	Auto	DB+M	LD, SPARQL	D2RQ	general
Virtuoso RDF View [5, 9]	Semi	DB+M	SPARQL	own	general
Triplify	Manual	Domain	LD	SQL	general

Table 4: An integrated overview of mapping approaches. Criteria for classification were merged, some removed, fields were completed, when missing. DB+M means that the semi-automatic approach can later be customized manually

Figure 5.1: Table comparing relevant approaches from [3]

The following criteria can be extracted:

[3] Bringing Relational Databases into the Semantic Web: A Survey. `http://www.semantic-web-journal.net/content/new-submission-bringing-relational-databases-semantic-web-survey`

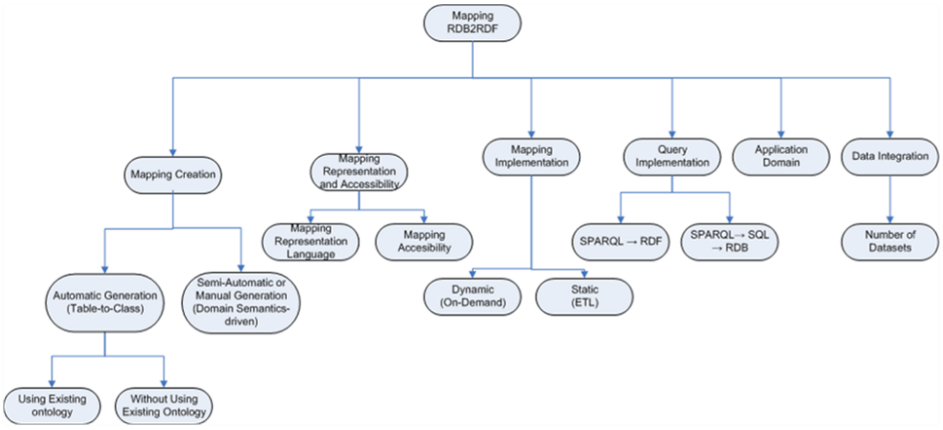


Figure 5.2: Reference Architecture for RDB2RDF Survey by [15]

Automation Degree. Degree of mapping creation automation.
Values: Manual, Automatic, Semi-Automatic.

Domain or Database Semantics Driven. Some approaches are tailored to model a domain, sometimes with the help of existing ontologies, while others attempt to extract domain information primarily from the given database schema with few other resources used (domain or database semantics-driven). The latter often results in a table-to-class, column-to-predicate mapping.Some approaches also use a (semi) automatic approach based on the database, but allow manual customization to model domain semantics.
Values: Domain, DB, DB+M, Both

Access Paradigm. Resulting access paradigm (ETL, Linked Data, SPARQL access). Note that the access paradigm also determines whether the resulting RDF model updates automatically. ETL means a one time conversion, while Linked Data and SPARQL always process queries versus the original database.
Values: SPARQL, ETL, LD

Mapping Language. The used mapping language as an important factor for reusability and initial learning cost.
Values: Visual Tool, intern, FOL, n/a, R2O, XSLT, D2RQ, proprietary, SQL

Domain reliance. Domain reliance (general or domain-dependent): requiring a pre-defined ontology is a clear indicator of domain dependency.
Values: Dependent, General

PROJECTS	MAPPING CREATION	MAPPING REPRESENTATION AND ACCESSIBILITY		MAPPING IMPLEMENTATION	QUERY IMPLEMENTATION	APPLICATION DOMAIN	DATA INTEGRATION	
	Automatic (Table-to-Class) **or Manual/Semi-Automatic** (Domain Semantics-driven)	**Representation Language**	**Mapping Access**	**Static (ETL) or Dynamic**	**SPARQL→RDF or SPARQL→SQL→RDB**		**Yes/No**	**Number of Datasets**
1. Hu et. al, 2007	Automatic (with use of existing ontology)	First Order Logic formulae or Horn Clauses	Files	None Specified	None Specified	Generic	Enables (through contextual mappings)	Potentially Multiple
2. Kashyap et al., 2007	Manual/Semi-Automatic (Domain Semantics-driven)	Mediator Framework Classes	Mapping mediator	Dynamic	SPARQL→SQL→RDB	Life Sciences	Enables	Potentially Multiple
3. DB2OWL (Cullot et al., 2007)	Automatic (Table to Class)	R2O language	R2O mapping document		SPARQL→SQL→RDB	Generic	Enables	Potentially Multiple
4. Tirmizi et. al 2008	Automatic (Table to Class, SQL-DDL to RDF)	First Order Logic	None specified	Static	None Specified	Generic	No	None
5. SOAM (Li et al., 2005)	Automatic (Table to Class) with user input	Logic Rules	Implemented as part of system	Static	Potentially SPARQL (on generated populated ontology)	Generic (Case Study: Economics)	No	None
6. Sahoo et al., 2008	Manual/Semi-Automatic (Domain Semantics-driven)	XPath expressions	XSLT document	Static	SPARQL	Life Sciences	Yes	Test included five (Gene, Biological Pathway)
7. Byrne, 2008	Manual/Semi-Automatic (Domain Semantics-driven)	SKOS vocabulary	RDF document	Static	SPARQL	Cultural Heritage	No	None
8. Green et. al, 2008	Manual/Semi-Automatic (Domain Semantics-driven)	D2RQ language	D2RQ mapping file	Dynamic	SPARQL→SQL→RDB	Ordnance Survey	Yes	Mutltiple
9. Virtuoso RDF View (Blakeley, 2007)	Both (user-specified)	SPASQL-based Meta Schema Language	Quad Storage	Both	Both	Generic	Enables	Potentially Multiple
10. D2RQ (Bizer et al., 2007)	Both (user-specified)	D2RQ language	D2RQ mapping file	Both	Both	Generic	Enables	Potentially Multiple
11. R2O (Barrasa et al., 2006)	Both (user-specified)	R2O language	R2O mapping document	Both	Both	Generic	Enables	Potentially Multiple
12. Dartgrid (Wu et al., 2006)	Automatic (Table to Class)	XML File	Visualized Mapping tool	Dynamic	SPARQL→SQL→RDB (Provide search and query interface)	Life Science (Traditional Chinese Medicine, TCM)	Yes	Test included databases for herb, compound formulas, disease, drug, TCM treatment.
13. RDBtoOnto (Cerbah, 2008)	Automatic (Table to Class, allows user intervention)	Constraint rules	Not explicitly stored	Static	Potentially SPARQL (on generated populated ontology)	Generic	No	None
14. Asio Tools	Automatic (Table to	OWL Full based language	File based	Both	SPARQL→SQL→RDB	Generic	Enables	Potentially Multiple

Figure 5.3: Comparison of approaches from [15]

Type. Although not used in the table the paper discusses four different classes:
Values: Alignment, Database Mining, Integration, Languages/Servers

5.2.2 DB2OWL by Ghawi and Cullot

In 2007 Ghawi and Cullot [7] published one of the first small surveys in the area of knowledge extraction from relational databases. They created the classification displayed in Figure 5.4 and used a total of 6 criteria grouped into three areas: Ontology, Exploitation and Automatisation (see Figure 5.5).

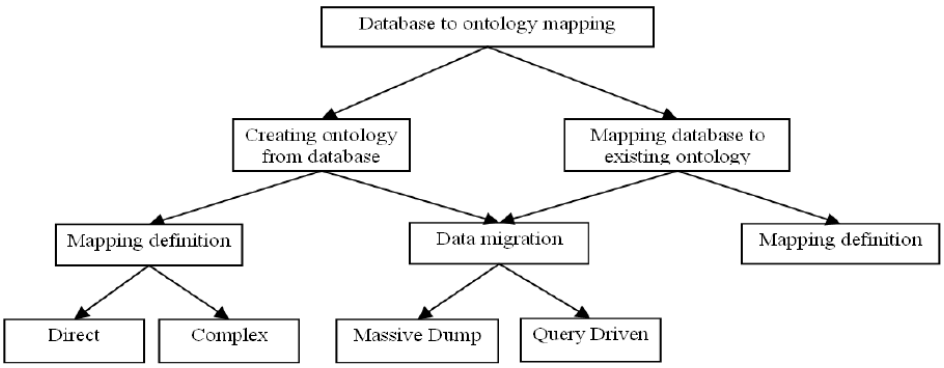


Figure 5.4: Classification of database-to-ontology mapping approaches [7].

Approach	Ontology		Exploitation		Automatisation	
	Created	Existing	Massive dump	Query driven	Mapping definition	Instance export
Volz et al.	×		×		Semi	Auto
DataGenie	×		×		Auto	Auto
Relational.OWL	×		×		Auto	Auto
KAON reverse		×		×	Semi	Auto
vis A vis		×		×	Manual	Auto
D2R map		×		×	Manual	Auto
R2O		×	×	×	Manual	Auto
DB2OWL	×			×	Auto	Auto

Figure 5.5: Table of comparison from [7]

The following criteria were used:

Ontology Binary feature deciding whether an ontology has to exist in advance or if it is created by the approach.
Values: Created, Existing

Exploitation This criteria states whether the result can only be queried or if a data dump can be extracted.
Values: Massive dump, Query driven.

Automatisation see above

Instance export As the schema mapping and the instance export were treated separate in the classification there is a criteria instance export, which is concerned with ontology population. It is always "Auto".

5.2.3 Survey by Konstantinou, Spanos and Mitrou

The first survey[4] was published in 2008 [10] and contained a chart (see Figure 5.6) showing the used classification taxonomy. They also compared 8 approaches shown in Figure 5.7. Note that before 2008 RDF/OWL and SPARQL were still quite new and other formats were still used more widely. We will not elaborate on the history of the Semantic Web in this report.

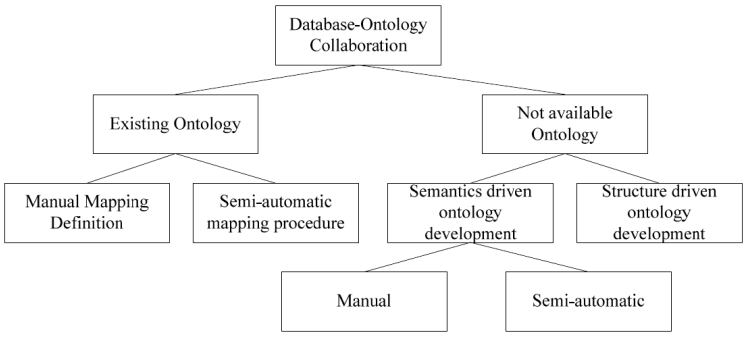


Figure 5.6: Classification taxonomy by [10]

The following criteria were used:

Ontology Language. The output language.
Values: RDF, OWL, DAML+OIL, ODL, Web-PDDL

RDBMS. The implemented database access method was analysed.
Values: Any, SQL, JDBC, ODBC, DB2, MySQL, Oracle

[4] `http://www.rintonpress.com/xjwe7/jwe-7-1/001-024.pdf`

Tool	Ontology Language	RDBMS	Semantic Query Language	Information needed/Degree of automation
D2RQ	RDF, DAML+OIL	Any RDBMS offering JDBC or ODBC access	RDQL	Both manual and automatic
D2R MAP	RDF	Any RDBMS offering JDBC or ODBC access	None	Manual
Clio	N/A	Any	N/A	Semi-automatic
MOMIS	ODL_{I3}	Any	OQL_{I3}	Semi-automatic
R_2O	RDF/OWL	Any SQL-implementing RDBMS	None	Manual
OntoGrate	Web-PDDL	Any SQL-implementing RDBMS	Web-PDDL	Manual
MAPONTO	OWL	Any SQL-implementing RDBMS	None	Semi-automatic
Relational.OWL	RDF/OWL	DB2, MySQL, Oracle	Any language that can query an OWL ontology	Automatic

Figure 5.7: Table of comparison by [10]

Semantic Query Language. If implemented this was mostly a predecessor of SPARQL.
Values: None, RDQL, OQL, Web-PDDL

Automation Degree. See above.
Values: Manual, Automatic, Semi-Automatic

Methodology Techniques. Summarizes the methodology of the approach as text. Some questions: What language is used for the mapping (SQL, XQuery, XML)? What algorithms are used (Clustering, matching)?

Components Mapped. A relational table has certain item that can be considered for a mapping. This criteria states, what items are considered by the mapping approach.
Values: DB Tables, columns, primary/foreign keys, integrity constraints, datatypes

Consistency Checks. States whether the consistency of the result is checked. Note that inconsistency can only happen, if the result is sufficiently expressive (contains e.g. disjointness)
Values: Yes, No, if yes it stated how consistency was checked (Jena API, ODB-Tools Engine, OntoEngine Reasoner)

User Interaction. How the user interacts with the tool in terms of mapping creation and querying. These criteria are quite general and do not have values, but text describing how it is done. Here are some questions: Does the tool use a graphical user interface? For what is it used? Can the user add and edit mappings in the GUI? Is the GUI used for aligning an

Ontology with the database? Does the user provide a mapping in form of a mapping language?

In the "Concluding Remarks" Section of [10] seven requirements are defined, which should be considered for further development of Database to Ontology Mapping Tools. Some of the requirements are not suitable for certain Use Cases, we, nevertheless, summarize them here:

1. **Dynamic Changing.** Changes to the Database or the Ontology should be reflected in the mapping.
2. **User-centered collaborative design** Similar to ontology engineering, the creation of the mapping and its maintenance would benefit from a tool that allows collaborative editing of the mapping.
3. **Conformity with standard formats.** This should account for the mapping language as well as the output language.
4. **Versioning and Rollback.** Coming for the same direction as requirement number 2, changes to the mapping should be versioned to support collaborative editing by allowing rollbacks (similar to a Wiki-Workflow or a VCS in Software development).
5. **Automation at design-time.** Automation reduces costs and errors and should be used to a "maximum possible extent"
6. **Completeness.** The mapping should consider all items (tables, columns, foreign keys, etc.) available in relational databases.
7. **Reusability of mappings.** A mapping standard (similar to XMI for XML or KIF for Ontologies) would improve reusability of mappings.

5.2.4 Survey by Spanos, Stavrou and Mitrou

As mentioned before the survey at the time of writing has been submitted to the Semantic Web Journal and is therefore available[5]. We will not include the tables here as the peer-reviewing process is not finished yet. There is, however, almost a full page table that uses mostly the criteria from Triplify[3]. The only criteria that were added seem to be:

Ontology Language. The output language.
Values: RDF, RDFS, OWL-Lite, OWL, OWL DL, OWL+SWIRL, F-Logic

Tool Availability.
Values: Commercial, Yes, No

[5]The submission draft is available at `http://www.semantic-web-journal.net/content/new-submission-bringing-relational-databases-semantic-web-survey`

5.3 Knowledge Extraction Tool Survey Schema

LOD2 candidate
`http://data.lod2.eu/2011/tools/ket/LOD2candidate`
This tool should be considered for the LOD2 stack. Boolean

can reuse vocabularies
`http://data.lod2.eu/2011/tools/ket/canReuseVocabularies`
The tool is able to reuse existing vocabularies in the mapping. For example the table column 'firstName' can be mapped to foaf:firstName. Some automatic approaches are not capable of reusing/mapping vocabularies. Boolean.

data exposition
`http://data.lod2.eu/2011/tools/ket/dataExposition`
Is SPARQL or another query language possible? Values can be either ETL (Dump), SPARQL (or another Query Language) or LinkedData. Note that the access paradigm also determines whether the resulting RDF model updates automatically. ETL means a one time conversion, while Linked Data and SPARQL always process queries versus the original database.

data source
`http://data.lod2.eu/2011/tools/ket/dataSource`
The data source the tool can be applied on? RDB, XML, CSV, etc.

data synchronization
`http://data.lod2.eu/2011/tools/ket/dataSynchronization`
Is a dump created once or is the data queried live from the legacy source? Static or Dynamic. If the tool writes the changes made to the RDF back to the legacy source it is bi-directional.

description of GUI
`http://data.lod2.eu/2011/tools/ket/GUIDescription`
What capabilities does the GUI have? A short text.

has GUI
`http://data.lod2.eu/2011/tools/ket/hasGUI`
Does the tool have a visual user interface? Boolean

mapping automatisation
`http://data.lod2.eu/2011/tools/ket/mappingAutomatisation`
The degree to which the mapping creation is assisted/automatised. Manual, GUI, semi-automatic, automatic.

mapping language
`http://data.lod2.eu/2011/tools/ket/mappingLanguage`

The mapping language used by the approach (e.g. SQL, R2O, D2RQ, R2RML). The used mapping language is an important factor for reusability and initial learning cost as well as flexibility and expressiveness. Most of the users are for example familiar with SQL and no additional training is neccessary. But, although SQL has extensive capabilities for selecting data in the WHERE clause, an additional mechanism for conversion and mapping is needed.

requires a Domain Ontology

`http://data.lod2.eu/2011/tools/ket/requiresDomainOntology`
A pre-existing ontology is needed to map to it. Boolean

5.4 How to access information

Go to `http://tinyurl.com/KETSurvey` and a list of all Knowledge Extraction tools is displayed. The view can be extended by selecting properties on the right hand side. Figure 5.8, 5.9, 5.10 show screenshots of the different views.

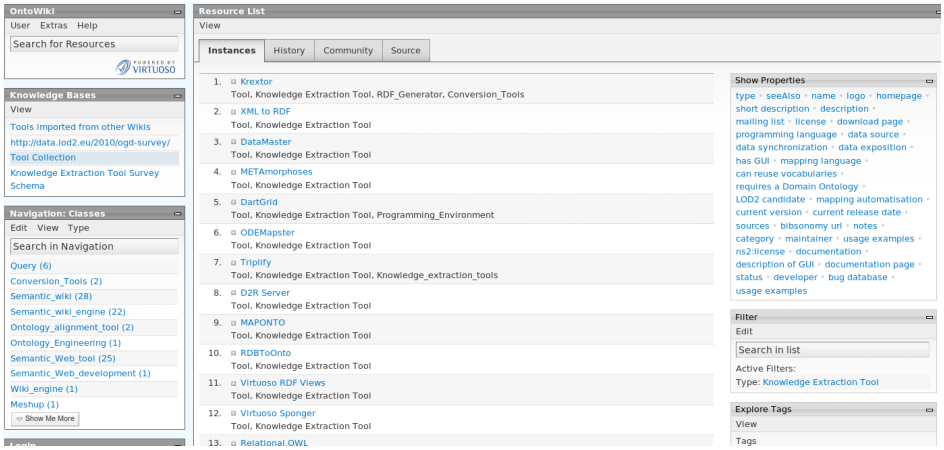


Figure 5.8: The list as found on `http://tinyurl.com/KETSurvey`

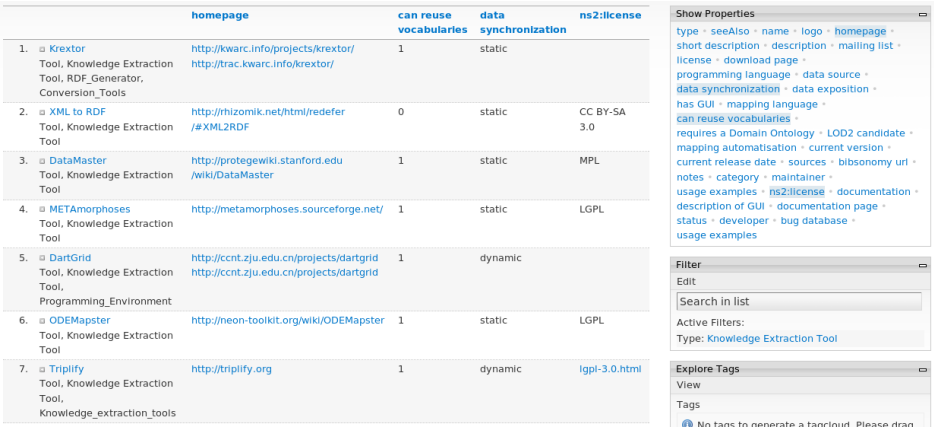


Figure 5.9: Additionally 4 properties were selected on the right and displayed in the list (see `http://tinyurl.com/KETFilterExample`)

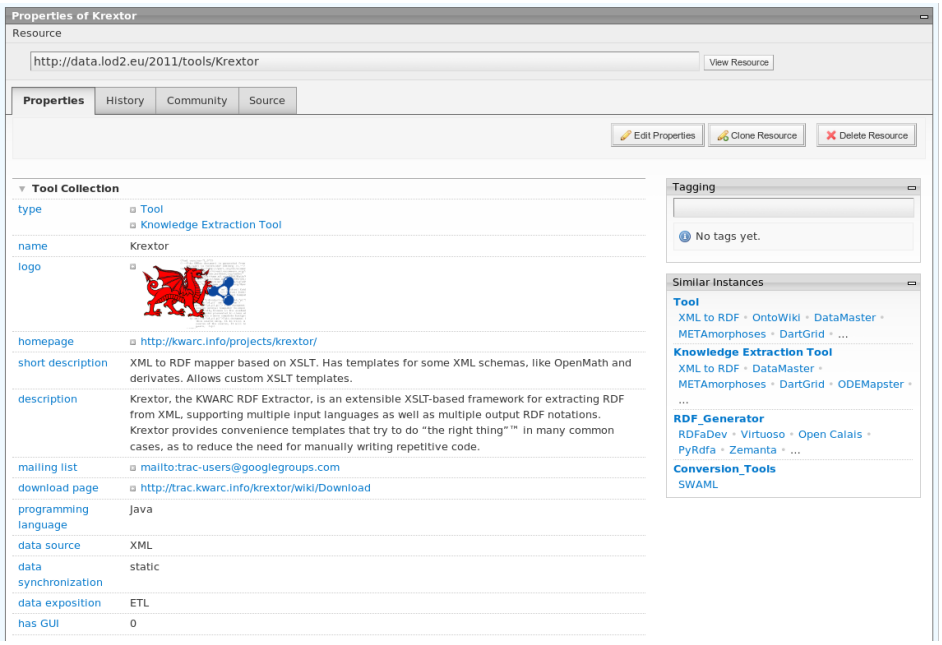


Figure 5.10: Detailed view of the Krextor tool
`http://data.lod2.eu/view/?r=Krextor&m=http://data.lod2.eu/2011/tools/`

Chapter 6

Tool Collection

6.1 Tool Survey Table

Name	Data Source	Data Extraction	Data Synchronisation	Mapping Language	Vocab. Reuse	Mapping Automat.	Req. Domain Ont.	LOD2 Candid.	Uses GUI
CSV2RDF4LOD	CSV	ETL	static	none	✓	manual	-	-	-
Convert2RDF	Delimited text file	ETL	static	RDF DAML	✓	manual	-	-	✓
D2R Server	RDB	SPARQL	bi-directional	D2R Map	✓	manual	-	✓	-
DartGrid	RDB	own query language	dynamic	Visual Tool	✓	manual	-	-	✓
DataMaster	RDB	ETL	static	proprietary	✓	manual	✓	-	✓
Google Refine's RDF Extension	CSV, XML	ETL	static	none		semi-automatic	-	-	✓
Krextor	XML	ETL	static	xslt	✓	manual	✓	✓	-
MAPONTO	RDB	ETL	static	proprietary	✓	manual	✓	-	-
METAmorphoses	RDB	ETL	static	proprietary xml	✓	manual	-	-	✓
MappingMaster	CSV	ETL	static	proprietary	✓	GUI	-	-	✓
ODEMapster	RDB	ETL	static	proprietary	✓	manual	✓	-	✓
OntoWiki CSV Importer	CSV	ETL	static	Data Cube Vocabulary	✓	semi-automatic	-	✓	✓
Poolparty Extraktor (PPX)	XML, Text	Linked Data	dynamic	RDF (SKOS)	✓	semi-automatic	✓		-

Name	Data Source	Data Exposition	Data Synchronisation	Mapping Language	Vocab. Reuse	Mapping Automat.	Req. Domain Ont.	LOD2 Candid.	Uses GUI
RDBToOnto	RDB	ETL	static	none	-	automatic, fine tunable	-	-	✓
RDF 123	CSV	ETL	static	-	-	manual	-	-	✓
RDOTE	RDB	ETL	static	SQL	✓	manual	✓		✓
Relational.OWL	RDB	ETL	static	none	-	automatic	-	-	-
T2LD	CSV	ETL	static	-	-	automatic	-	-	-
The RDF Data Cube Vocabulary	spreadsheets			Data Cube Vocabulary	✓	manual	-	-	
TopBraid Composer	CSV	ETL	static	SKOS	-	semi-automatic	-	-	✓
Triplify	RDB	LinkedData	dynamic	SQL	✓	manual	-	✓	-
Virtuoso RDF Views	RDB	SPARQL	dynamic	Meta Schema Laguage	✓	semi-automatic	-	✓	✓
Virtuoso Sponger	structured and semi-structured	SPARQL	dynamic	Virtuoso PL & XSLT	✓	semi-automatic	-	✓	-
VisAVis	RDB	RDQL	dynamic	SQL	✓	manual	✓	-	✓
XLWrap: Spreadsheet to RDF	CSV	ETL	static	TriG Syntax	✓	manual	-	-	-
XML to RDF	XML	ETL	static	-	-	manual	-	-	-

6.2 Tool List

The following sections give the detailed information collected about the tools.

6.2.1 CSV2RDF4LOD

Short Description

In its simplest form, csv2rdf4lod is a quick and easy way to produce an RDF encoding of data available in Comma-Separated-Values (CSV). In its advanced form, csv2rdf4lod is a custom reasoner tailored for some heavy-duty data integration.

Status

mature

Homepage

`http://logd.tw.rpi.edu/technology/csv2rdf4lod`

Description

In its advanced form, csv2rdf4lod is a custom reasoner tailored for some heavy-duty data integration. Although csv2rdf4lod can handle tabular data from well-structured RDBMS dumps, its forte is in handling "messier" tabular data created manually or using less rigorous information modeling strategies – perfect for handling real data that evolved "in the wild". In either case, csv2rdf4lod is designed to aggregate and integrate multiple versions of multiple datasets of multiple source organizations in an incremental and backward-compatible way. conversion:Enhancement is the most frequently used page for those up and running with csv2rdf4lod. It lists the enhancements that the converter recognizes, along with the beginning and end states of the RDF produced when performing the enhancement. It even cites datasets that benefited from the enhancements, so you can go check out the results "in the wild". (BTW, the enhancements are encoded in RDF using our conversion vocabulary , effortlessly decoupling them from the specific converter implementation and enabling cool things like querying for datasets according to how they were enhanced.)

Notes

Source: `http://logd.tw.rpi.edu/technology/csv2rdf4lod`

Supported OS

Windows, Unix

Maintainer

Johanna Flores, Tim Lebo

License

`http://creativecommons.org/publicdomain/zero/1.0/`

Mailing List

`http://opengovernmentdata.org/mailing-list/`

Usage Examples

`https://github.com/timrdf/csv2rdf4lod-automation/wiki/Real-world-examples`

Download Page

`https://github.com/timrdf/csv2rdf4lod-automation/wiki/Installing-csv2rdf4l`

6.2.2 Convert2RDF

Short Description

The ConvertToRDF tool is designed to take plain-text delimited files, like .csv files dumped from Microsoft Excel, and convert them to RDF. To use it all you need to write is a file to map from one form to the other.

Homepage

`http://www.mindswap.org/~mhgrove/ConvertToRDF/`

Description

Writing RDF by hand is difficult and often the data that needs to be converted to an RDF format is huge; entirely too much to do by hand, or too complex to be worth doing. Convert To RDF is a tool for automatically converting delimited text data into RDF via a simple mapping mechanism. The original ConvertToRDF tool worked from the commandline taking in a map file which defined how to perform the conversion. Writing map files by hand was sometimes a complicated task so a GUI version of the program has been designed. The new Convert to RDF provides the user with a table layout. When a delimited text file, or an Excel file, is opened in Convert to RDF, the data is shown in the main table of the program. From this point, creating a mapping is just a matter of a few clicks and drags.

Supported OS

Windows

Maintainer

Michael Grove

Mailing List

mhgrove@hotmail.com

Download Page

`http://www.mindswap.org/~mhgrove/ConvertToRDF/ConvertToRDF-v1.2.zip`

6.2.3 D2R Server

Short Description

D2R Server is a tool for publishing relational databases on the Semantic Web. It enables RDF and HTML browsers to navigate the content of the database, and allows applications to query the database using the SPARQL query language.

Status

mature

Homepage

`http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/`

Description

D2R Server is a tool for publishing the content of relational databases on the Semantic Web, a global information space consisting of linked data.

Data on the Semantic Web is modelled and represented in RDF. D2R Server uses a customizable D2RQ mapping to map database content into this format, and allows the RDF data to be browsed and searched - the two main access paradigms to the Semantic Web.

D2R Server's Linked Data interface makes RDF descriptions of individual resources available over the HTTP protocol. An RDF description can be retrieved simply by accessing the resource's URI over the Web. Using a Semantic Web browser like Tabulator (slides) or Disco, you can follow links from one resource to the next, surfing the Web of Data.

The SPARQL interface enables applications to search and query the database using the SPARQL query language over the SPARQL protocol.

A traditional HTML interface offers access to the familiar Web browsers.

D2R Server architecture diagram6.1

Requests from the Web are rewritten into SQL queries via the mapping. This on-the-fly translation allows publishing of RDF from large live databases and eliminates the need for replicating the data into a dedicated RDF triple store.

Read more about the interfaces offered by D2R Server, including example HTTP requests and responses, in the Technical Note Publishing Databases on the Semantic Web.

Maintainer

Chris Bizer, Richard Cyganiak

Mailing List
d2rq-map-devel@lists.sourceforge.net

Current Version
v0.7 (alpha) from 2009-08-11

Usage Examples
Live Demo: `http://www4.wiwiss.fu-berlin.de/is-group/`

Download Page
`http://sourceforge.net/projects/d2rq-map/files/D2R%20Server/`

Literature
[4]

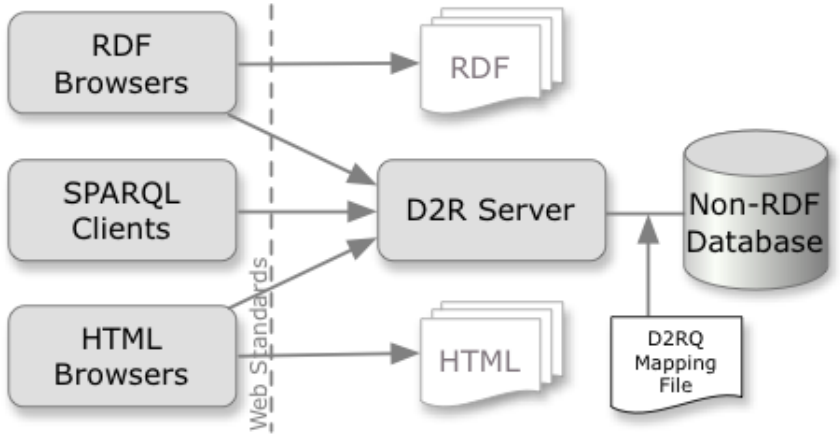


Figure 6.1: D2R Server architecture diagram (Source: `http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/images/architecture.png`).

6.2.4 DartGrid

Short Description

DartGrid is a data integration framework using semantic web technologies. It features in visuliazed semantic mapping tools (Currently support relational-RDF/OWL mapping, XML-RDF/OWL mapping is still under development), and a SPARQL-SQL query component based on rewriting-query-using-view approach.

Status

not retrievable

Homepage

`http://ccnt.zju.edu.cn/projects/dartgrid`

Description

The Dartgrid Semantic Web toolkit offers tools for the mapping and querying of RDF generated from RDB. The mapping is basically a manual table to class mapping where the user is provided with a visual tool to define the mappings. The mappings are then stored and used for the conversion. The construction of SPARQL queries is assisted by the visual tool and the queries are translated to SQL queries based on the previously defined mappings. A full?text search is also provided.

Literature

[19]

6.2.5 DataMaster

Short Description
Protege plugin for importing databases into ontologies.

Status
mature

Homepage
`http://protegewiki.stanford.edu/wiki/DataMaster`

Description
DataMaster is a Protege plug-in for importing schema structure and data from relational databases into Protege. DataMaster supports both OWL and frame-based ontologies and can be used with any relational database with JDBC/ODBC drivers.

Part of the rationale for developing DataMaster was that existing Protege plug-ins such as DataGenie do not support OWL ontologies or schema-only imports.

This plug-in is NOT a database back-end. The typical use-case for this plug-in is importing legacy data into Protege before doing additional knowledge acquisition or knowledge modeling. This plug-in currently does not include any capability for moving data in the opposite direction, i.e., from Protege classes and instances into a relational database. Another use-case for this plug-in might be to import a database schema as classes or instances in the ontology which may be later used to dynamically query the content of the database using SQWRL queries. DataMaster could be also used as a database viewer. For efficiency, a database might be stored as a set of custom-designed database tables, but then DataMaster could be used to view portions of the schema from within Protege user interface.

Notes
Description was obtained from homepage

GUI Description
Protege plugin

Maintainer
Csongor Nyulas, `http://protegewiki.stanford.edu/wiki/Csongor_Nyulas`

License
MPL

Current Version
1.3.2 from December 17, 2009

Download Page

`http://protegewiki.stanford.edu/wiki/DataMaster`

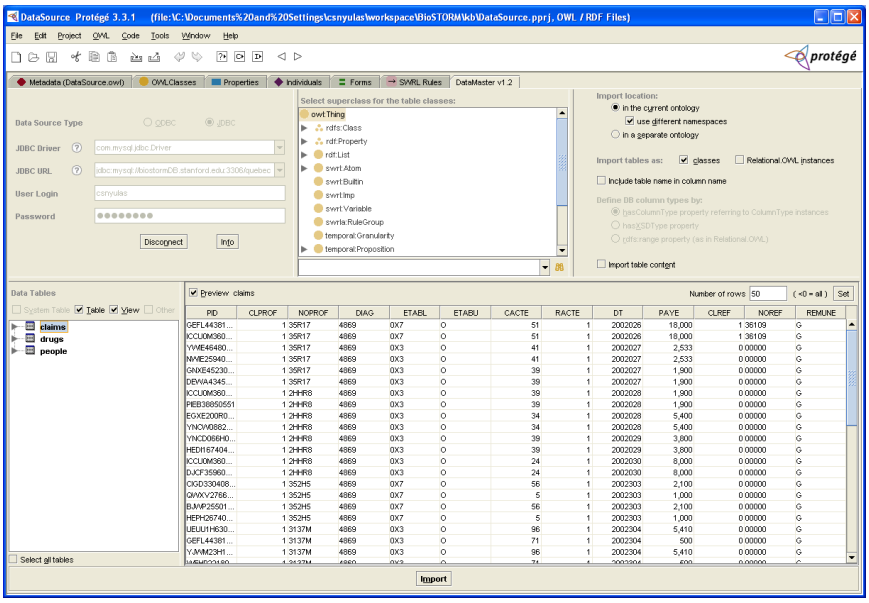


Figure 6.2: DataMaster screenshot (Source: `http://protegewiki.stanford.edu/images/3/3c/DataMaster_screenshot.png`).

6.2.6 Google Refine's RDF Extension

Short Description
This project adds a graphical user interface(GUI) for exporting data of Google Refine projects in RDF format. The export is based on mapping the data to a template graph using the GUI.

Status
alpha

Homepage
`http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/`

Description
1. Graphical interface for defining the mapping 2. Use of arbitrary vocabularies/ Ontologies 3. Import existing vocabularies/ Ontologies 4. Autocomplete for property and class names 5. RDF/XML and Turtle export

Notes
`http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/`

GUI Description
`http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/#example`

Supported OS
Linux, Widnows, MAC

Maintainer
Fadi Maali, Richard Cyganiak

License
`http://www.opensource.org/licenses/bsd-license.php`

Mailing List
`http://github.com/fadmaa/grefine-rdf-extension/issues`

Current Version
0.2.1 from 11/11/2010

Usage Examples
`http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/#example`

Download Page
`http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/#quick_start`

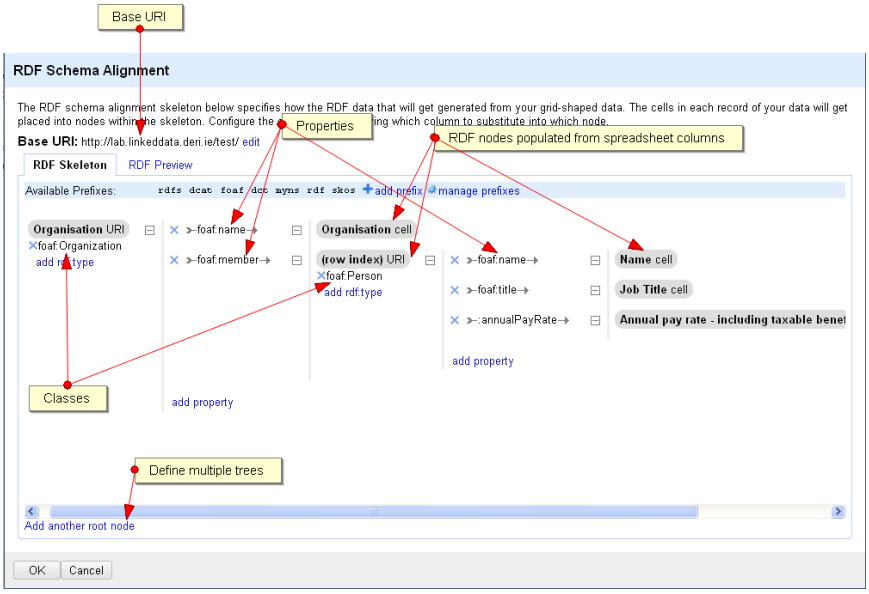
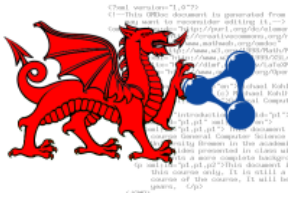


Figure 6.3: Screenshot of Google Refine with RDF Extension (Source: `http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/imgs/skeleton_example_annotated.png`).

6.2.7 Krextor



Short Description

XML to RDF mapper based on XSLT. Has templates for some XML schemas, like OpenMath and derivates. Allows custom XSLT templates.

Homepage

`http://kwarc.info/projects/krextor/`

Description

Krextor, the KWARC RDF Extractor, is an extensible XSLT-based framework for extracting RDF from XML, supporting multiple input languages as well as multiple output RDF notations. Krextor provides convenience templates that try to do ?the right thing?? in many

common cases, as to reduce the need for manually writing repetitive code.

Notes
Description was taken from homepage.

Mailing List
mailto:trac-users@googlegroups.com

Current Version
0.3 from 2008-12-31

Download Page
`http://trac.kwarc.info/krextor/wiki/Download`

Literature
[11]

6.2.8 MAPONTO

Short Description
Is a semi-automatic mapping tool for finding formal realtions between data models.

Status
inactive

Homepage
`http://www.cs.toronto.edu/semanticweb/maponto/`

Description
MapOnto is a research project aiming at discovering semantic mappings between different data models, e.g, database schemas, conceptual schemas, and ontologies. So far, we have developed tools for discovering semantic mappings between database schemas and ontologies as well as between different database schemas

Notes
description was taken from homepage

GUI Description
Plug in for Protege

License
`http://www.mozilla.org/MPL/MPL-1.1.html`

Current Version
prealpha from 2007-07-05

Usage Examples
`http://www.cs.toronto.edu/semanticweb/maponto/demo.content.html`

Literature
[1]

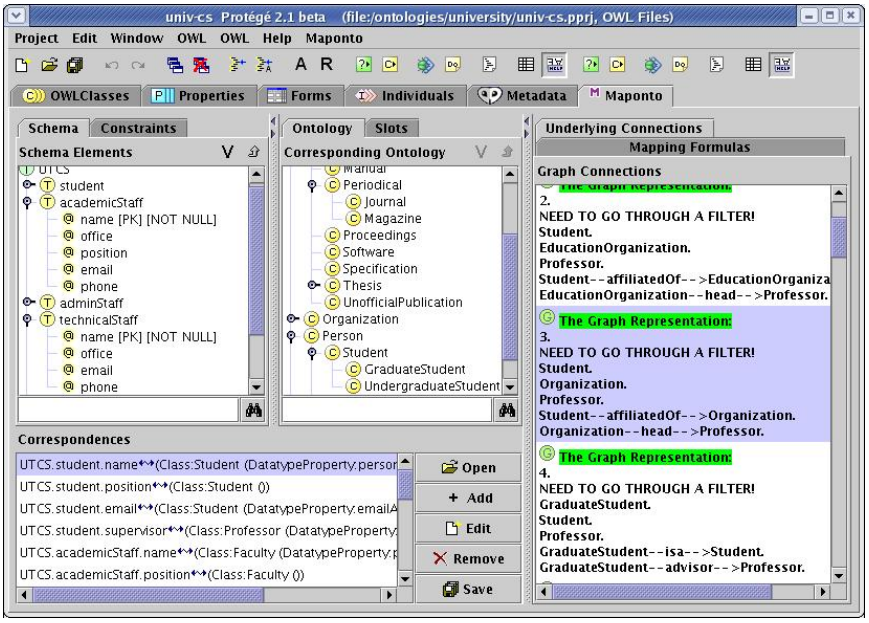


Figure 6.4: Maponto screenshot (Source: `http://www.cs.toronto.edu/semanticweb/maponto/images/mapping-connection-screen.jpg`).

6.2.9 METAmorphoses

Short Description

METAmorphoses allows mapping of relational data to RDF. It is configured in two steps, by firts defining a mapping file which is applied to the data base using a template definition.

Status

inactive

Homepage

`http://metamorphoses.sourceforge.net/`

Description

METAmorphoses is a set of tools for flexible and easy-to-use generation of RDF metadata directly from a relational database. Metadata are generated according to the mapping from an existing database schema to a particular ontology. The METAmorposes package contains the following tools: 1. MMPHP (METAmorphoses for PHP): a PHP tool for flexible and easy-to-use generation of RDF metadata directly from a relational database. Metadata are genereated according to the mapping from an existing database schema to a particular ontology. 2. METAmorphoses (processor): The Java library that processes DB-to-ontology mapping and tranforms relational data to RDF.

3. RDF Shout:The simple Java Servlet application that uses METAmorphoses Processor to publish data from a relational database as RDF documents on the web. 4. METAmorphoses Editor: The drag-and-drop editor for a DB-to-ontology mapping creation. The resulting mapping documents can be used in the METAmorphoses processor.

Notes

description was taken from homepage

GUI Description

stand alone mapping editor with drag and drop capabilities

License

LGPL

Current Version

0.2.5 from 2007-02-07

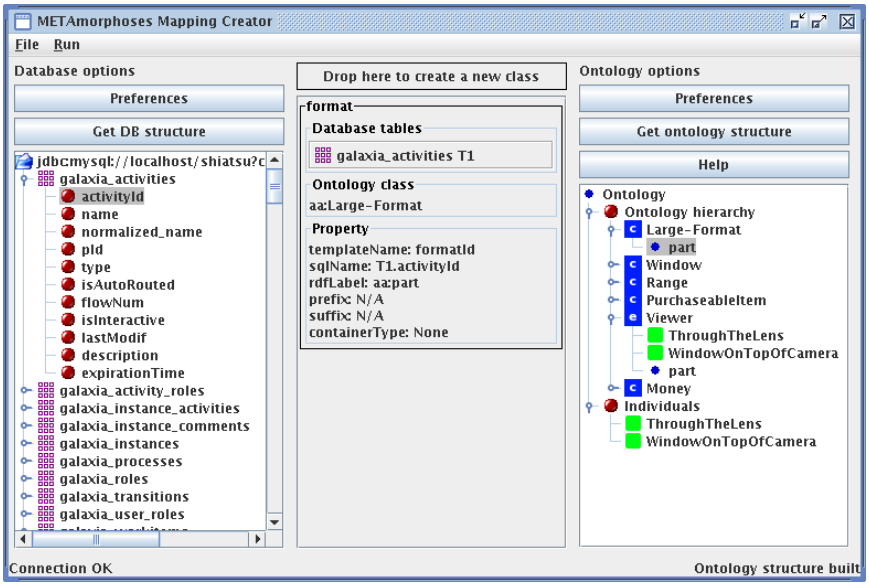


Figure 6.5: METAmorphoses screenshot (Source: `http://metamorphoses.sourceforge.net/img/editor.png`).

6.2.10 MappingMaster

Short Description

MappingMaster is an open source Protege-OWL plugin that can be used to transform the content of spreadsheets in to OWL ontologies. It has two primary components: (1) Domain Specific Language: Mappings in MappingMaster are specified using a domain specific language (DSL). (2) MappingMaster Tab: A graphical user interface for defining, managing, and executing mappings defined using this DSL is also provided.

Status

mature

Homepage

`http://protege.cim3.net/cgi-bin/wiki.pl?MappingMaster`

Description

One of the hurdles that new and existing users of Semantic Web standards continue to face is converting preexisting, non-Semantic Web encoded information into one of the many Semantic Web languages (e.g., RDF, OWL). In some domains, a large deal of this information is represented in spreadsheets (e.g., financial services), which has motivated both academia and industry to develop a variety of general-purpose spreadsheet mapping techniques to avoid manually encoding spreadsheet content in OWL or writing custom extraction programs. Existing mapping approaches, however, suffer from a variety of limitations. First, many mapping techniques assume very simple data models within spreadsheets. Typically, it is assumed that each table in a spreadsheet adheres to a relational model where each row in the table describes a different entity and each column describes an attribute for that entity; we refer to this as the ?entity-per-row? assumption. Unfortunately, there are numerous real-world spreadsheets that do not adhere to this simple data model, as many spreadsheet-authoring tools are extremely flexible and do not restrict the manner in which users author tabular structures. Common examples of complex layouts can be found in the financial domain. Here, analysts or companies publish sales forecasts or results, which are typically represented by tables that have products or market segments listed in a column, quarters or years listed in a row, and sales figures specified for each product/market segment and date. Recently, there have been efforts to overcome the entity-per-row limitation and to support mappings for arbitrary spreadsheets. However, to the best of our knowledge, these approaches use an RDF triples-based approach to encode mapping rules. They can be effective when mapping spreadsheet content to

RDF, but are very cumbersome when encoding content in OWL due to its verbose RDF serialization. To overcome these limitations, we propose a new declarative OWL-centric mapping language that supports arbitrary spreadsheet-to-OWL mappings. The language also supports syntactic transformations of cell contents, as well as inline OWL axioms involving classes, properties and individuals extracted from cell contents. In the end, the mapping language enables mapping information from complex spreadsheets to OWL using a compact, user-friendly syntax.

Notes
Paper: `http://iswc2010.semanticweb.org/accepted-papers/414`

GUI Description
`http://protege.cim3.net/cgi-bin/wiki.pl?MappingMasterGUI`

Supported OS
Platform independant

Maintainer
Martin O'Connor at Stanford Center for Biomedical Informatics Research.

License
Open Source

Mailing List
protege-owl@lists.stanford.edu

Current Version
0.8 from 2010

Usage Examples
`http://protege.cim3.net/cgi-bin/wiki.pl?MappingMasterGUI`

Download Page
`http://protege.cim3.net/cgi-bin/wiki.pl?MappingMaster`

Literature
[13]

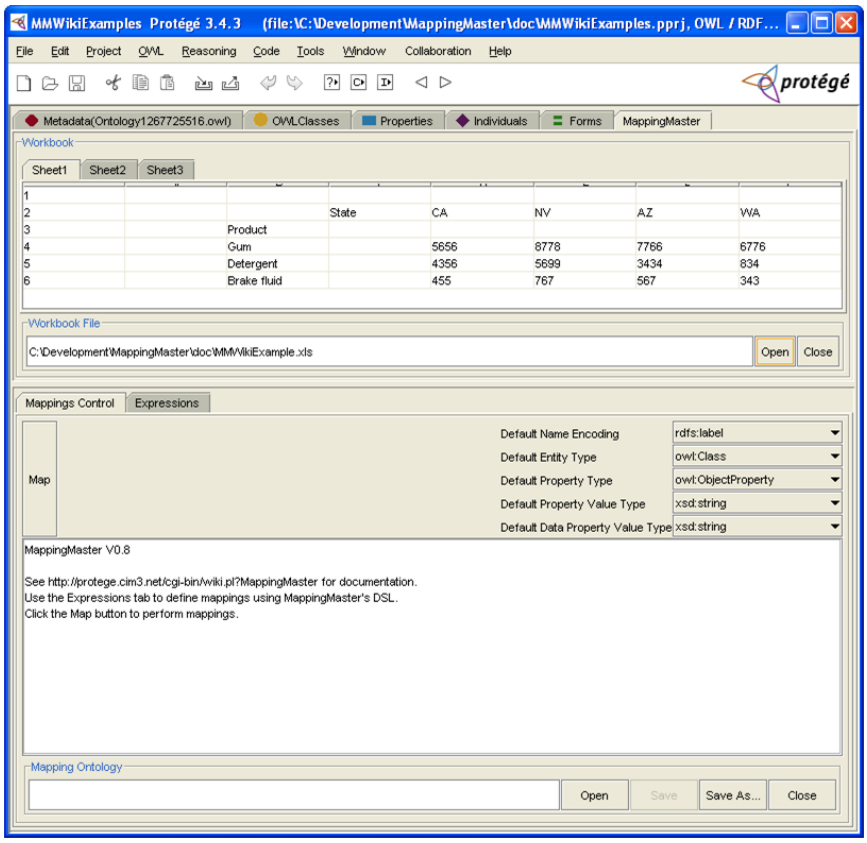


Figure 6.6: MappingMaster screenshot (Source: `http://swrl.stanford.edu/MappingMaster/1.0/ScreenShots/MMWorkbookView.png`).

6.2.11 ODEMapster

Short Description

ODEMapster is a plugin for the NeON toolkit for mapping relational data to ontologies.

Status

active

Homepage

`http://neon-toolkit.org/wiki/ODEMapster`

Description

ODEMapster processor, which generates Semantic Web instances from relational instances based on the mapping description expressed in an R2O document. ODEMapster offers two modes of execution: Query driven upgrade (on-the-fly query translation) and massive upgrade

batch process that generates all possible Semantic Web individuals from the data repository.

Notes
description was taken from homepage

GUI Description
plug in for the commmercial NeON toolkit

License
LGPL

Current Version
2.2.7 from 2010-07-02

Usage Examples
`http://www.cs.toronto.edu/semanticweb/maponto/demo.content.html`

Literature
[14]

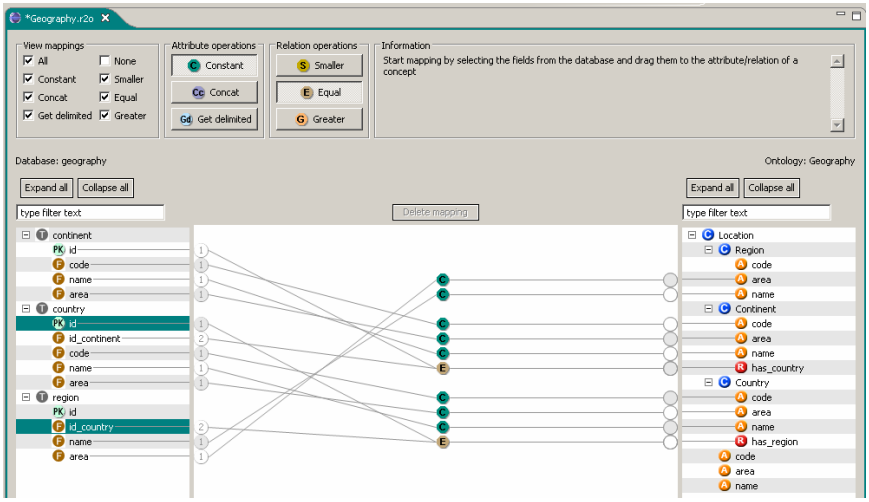


Figure 6.7: ODEMapster screenshot (Source: `http://neon-toolkit.org/w/images/ODEMApsterGeneral.png`).

6.2.12 OntoWiki CSV Importer

Short Description

Biomedical statistical data is often published as Excel sheets. Thus they have the advantage of being easily readable by humans. However, they cannot be queried efficiently. Also it is difficult to integrate with other datasets, which may be in different formats. Our approach is to convert the data into a single data model ? RDF. But in these biomedical datasets, a single statistical value is described in several dimensions. Thus a simple row-based transformation is not possible. Therefore, we used The RDF Data Cube vocabulary for the conversion as it is designed particularly to represent multidimensional statistical data using RDF.

Status

developing

Homepage

`http://aksw.org/Projects/Stats2RDF`

Description

Transforming CSV to RDF in a fully automated way is not feasible as there may be dimensions encoded in the heading or label of a sheet. Therefore, we introduce a semi-automated approach as a plug-in in OntoWiki. Using this plug-in, a CSV file can be converted to RDF using the Data Cube Vocabulary. We used the WHO's Global Health Observatory dataset as a first use case. It is primarily available as Excel sheets. We converted them to CSV files and then transformed them into RDF.

Notes

`http://aksw.org/Projects/Stats2RDF`

GUI Description

`https://docs.google.com/leaf?id=0B8Mh-RR0aBWQNDRjM2UyMmEtOTNlZC00ZDIwLTg5ZDMtMGZkYT`
`en&authkey=CIr8_MYN`

Supported OS

Windows, Linux, Mac

Maintainer

Amrapali Zaveri

License

GNU GPL v2, CC BY-SA 3.0

Usage Examples

`http://aksw.org/Projects/Stats2RDF`

Download Page

`http://code.google.com/p/ontowiki/downloads/list`

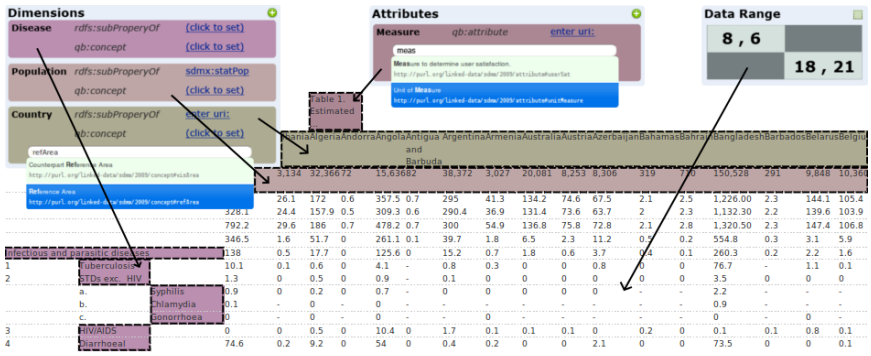


Figure 6.8: OntoWiki CSV Importer screenshot (Source: `http://redd.aksw.org/images/stats2rdf.png`).

6.2.13 Poolparty Extraktor (PPX)

Short Description

The PoolParty Extractor (PPX) is responsible for enhancing (XML-) documents by mapping metadata values ((semi)structured information) to concepts in a thesaurus and by extracting additional metadata from document data (unstructured information) itself and mapping this additional metadata again to concepts in a thesaurus.

Status

mature

Homepage

`http://poolparty.punkt.at/`

Description

PPX is interpreting explicitly provided metadata as (semi-)structured information ready to be mapped to thesaurus concepts. As a basic configuration a mapping scheme between predefined metadata fields of documents on the one side and collections of concepts (concept schemes) in thesauri on the other side is provided. Upon document processing PPX is receiving RDF formatted metadata from the collector which is then processes by looking up values in the thesauri. In addition to already (semi-)structured metadata explicitly provided by document authors, PPX is also constructed for finding new metadata from unstructured document text. It therefore uses a mixed approach of NLP techniques (natural language processing) and statistics based heuristics. As a first step, document text is analysed and single words and multi-word phrases are collected from it, which are also weighted according to their position and prominence in the text. In a second step these words and phrases are looked up in a special index constructed from the thesauri.

Supported OS

Linux

Maintainer

punkt. netServices

License

Commercial

Current Version
2 from 1/31/2011

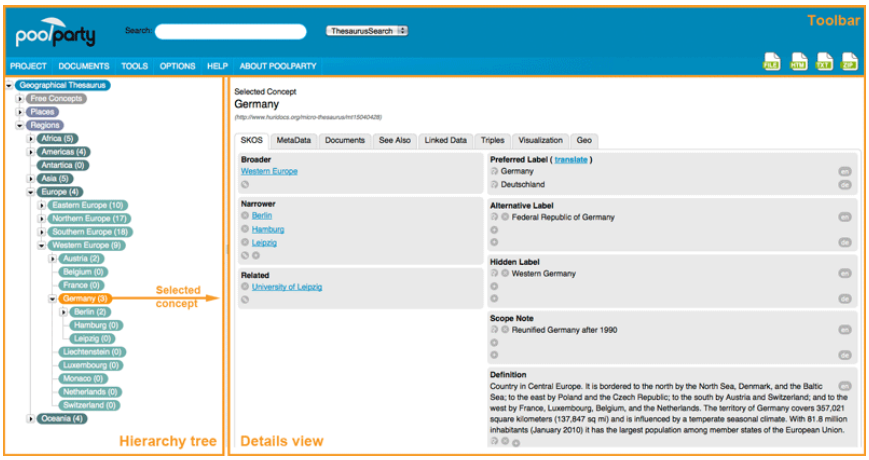


Figure 6.9: PoolParty screenshot (Source: `http://poolparty.punkt.at/wp-content/gallery/test/gui.png`).

6.2.14 RDBToOnto

Short Description

RDBToOnto is a tool that allows to automatically generate fine-tuned populated ontologies from relational databases (in RDFS/OWL).

Status

mature

Homepage

`http://www.tao-project.eu/researchanddevelopment/demosanddownloads/RDBToOnto.html`

Description

A major feature of this tool is the ability to produce highly structured ontologies by exploiting both the database schema and structuring patterns hidden in the data (see publications for details on the RTAXON learning method, including its formal description). Though automated to a large extent, the process can be constrained in many ways through a friendly user interface. It also provides a framework that eases the development and integration of new learning methods and database readers. A database optimization module allows to enhance the input database before ontology generation. , A major feature of this tool is the ability to produce highly structured ontologies by exploiting both the database schema and structuring patterns hidden in the data (see publications for details on the RTAXON learning method, including its formal description). Though automated to a large extent, the process can be constrained in many ways through a friendly user interface. It also provides a framework that eases the development and integration of new learning methods and database readers. A database optimization module allows to enhance the input database before ontology generation.

Notes

Description was copied from homepage and adjusted.

GUI Description

The GUI is separated in three parts and the user has to configure all three parts from top to bottom. 1. Input sources are selected and configured, 2. The methods for learning the ontology from the source (RTAXON is provided, see bibsonomy URL.) 3. Local constraints can be created with specifying rules.

Maintainer

Farid Cerbah ¡farid.cerbah@dassault-aviation.fr¿, TAO European Project (IST-2004-026460)

License
proprietary (closed)

Download Page
`http://www.tao-project.eu/researchanddevelopment/demosanddownloads/RDBToOnto.html`

Literature
[5]

6.2.15 RDF 123

Short Description
RDF123 is an application and web service for converting data in simple spreadsheets to an RDF graph.

Status
mature

Homepage
`http://ebiquity.umbc.edu/project/html/id/82/RDF123`

Description
Users control how the spreadsheet's data is converted to RDF by constructing a graphical RDF123 template that specifies how each row in the spreadsheet is converted as well as metadata for the spreadsheet and its RDF translation. The template can map spreadsheet cells to a new RDF node or to a literal value. Labels on the nodes in the map can be used to create blank nodes or labeled nodes, attach a XSD datatype, and invoke simple functions (e.g., string concatenation). The graph produced for the spreadsheet is the union of the sub-graphs created for each row. The template itself is stored as a valid RDF document encouraging reuse and extensibility.

Notes
`http://rdf123.umbc.edu/`

GUI Description
`http://ebiquity.umbc.edu/resource/html/id/237/RDF123-java-application-v1-0`

Supported OS
Windows, Linux, Web service, Java application

Maintainer
Lushan Han

License
GPL, CC BY 2.0, MIT License

Mailing List
`http://groups.google.com/group/rdf123?hl=en&pli=1`

Current Version
true from April 2007

Usage Examples
`http://rdf123.umbc.edu/examples/`

Download Page

`http://ebiquity.umbc.edu/project/html/id/82/RDF123?res=on#res`

Literature

[8]

6.2.16 RDOTE

Short Description
R. is a GNU/GPL licensed framework for transporting data residing in RDB into the Semantic Web, which provides a friendly GUI, as well as enough expressivity for creating RDF dumps of RBD data.

Homepage
`http://sourceforge.net/projects/rdote/`

Maintainer
`https://sourceforge.net/projects/rdote/`

License
GPL

Current Version
no version from 5/27/2010

Usage Examples
Video: `http://www.youtube.com/watch?v=pk7izhFeuf0`

Download Page
`http://sourceforge.net/projects/rdote/files`

6.2.17 Relational.OWL

Short Description
Relational.OWL automatically extracts the semantics of virtually any relational database and transforms this information automatically into RDF/OWL, processable by a wide majority of Semantic Web applications.

Status
alpha, discontinued

Homepage
`http://sourceforge.net/projects/relational-owl/`

Description
Relational.OWL converts a database directly to OWL. This means that a vocabulary was created to represent databases, i.e. an Ontology (Relational.OWL Ontology), that has classes such as 'Table'. or 'Column'.

Notes
This is a Sourceforge project.

Maintainer
Cristian Perez de Laborda, Stefan Conrad, perezdel, conrad@cs.uni-duesseldorf.de

License
GPL

Download Page
`http://sourceforge.net/projects/relational-owl/files/`

Literature
[6]

6.2.18 T2LD

Short Description

An automatic framework for extracting, interpreting and representing tables as Linked Data

Homepage

http://ebiquity.umbc.edu/paper/html/id/480/T2LD-An-automatic-framework-for-extracti

Description

T2LD is an automatic framework for extracting, interpreting and generating linked data from tables. In the process of representing tables as linked data, we assign every column header a class label from an appropriate ontology, link table cells (if appropriate) to an entity from the Linked Open Data cloud and identify relations between various columns in the table, which helps us to build an overall interpretation of the table. Using the limited evidence provided by a table in the form of table headers and table data in rows and columns, we adopt a novel approach of querying existing knowledge bases such as Wikitology, DBpedia etc. to figure the class labels for table headers. In the process of entity linking, besides querying knowledgebases, we use machine learning algorithms like support vector machine and algorithms which can learn to rank entities within a given set to link a table cell to entity. We further use the class labels, linked entities and information from the knowledge bases to identify relations between columns. We prototyped a system to evaluate our approach against tables obtained from Google Squared, Wikipedia and set of tables obtained from a dataset which Google shared with us.

Notes

Masters thesis, acceptes as a poster at ISWC 2010, http://ebiquity.umbc.edu/_file_directory_/papers/513.pdf

Maintainer

Varish Mulwad

6.2.19 The RDF Data Cube Vocabulary

Short Description

There are many situations where it would be useful to be able to publish multi-dimensional data, such as statistics, on the web in such a way that it can be linked to related data sets and concepts. The Data Cube vocabulary provides a means to do this using the W3C RDF (Resource Description Framework) standard. The model underpinning the Data Cube vocabulary is compatible with the cube model that underlies SDMX (Statistical Data and Metadata eXchange), an ISO standard for exchanging and sharing statistical data and metadata among organizations. The Data Cube vocabulary is a core foundation which supports extension vocabularies to enable publication of other aspects of statistical data flows.

Status

mature

Homepage

`http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html`

Description

Statistical data is a foundation for policy prediction, planning and adjustments and underpins many of the mash-ups and visualisations we see on the web. There is strong interest in being able to publish statistical data in a web-friendly format to enable it to be linked and combined with related information. At the heart of a statistical dataset is a set of observed values organized along a group of dimensions, together with associated metadata. The Data Cube vocabulary enables such information to be represented using the the W3C RDF (Resource Description Framework) standard and published following the principles of linked data. The vocabulary is based upon the approach used by the SDMX ISO standard for statistical data exchange. This cube model is very general and so the Data Cube vocabulary can be used for other data sets such as survey data, spreadsheets and OLAP data cubes [OLAP]. The Data Cube vocabulary is focused purely on the publication of multi-dimensional data on the web. We envisage a series of modular vocabularies being developed which extend this core foundation. In particular, we see the need for an SDMX extension vocabulary to support the publication of additional context to statistical data (such as the encompassing Data Flows and associated Provision Agreements). Other extensions are possible to support metadata for surveys (so called "micro-data", as encompassed by DDI) or publication of statistical reference metadata.

Notes

`http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html`

Maintainer

Richard Cyganiak, Dave Reynolds, Jeni Tennison (TSO)

Mailing List

`http://lists.w3.org/Archives/Public/public-rdb2rdf-comments/`

Usage Examples

`http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/cube.html#example`

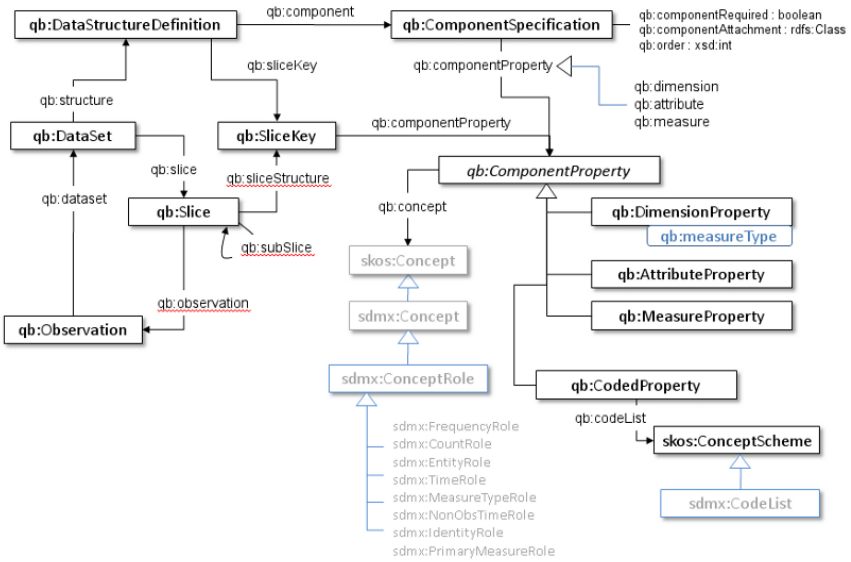


Figure 6.10: RDF Data Cube Vocabulary overview (Source: `http://publishing-statistical-data.googlecode.com/svn/trunk/specs/src/main/html/qb-fig1.png`).

6.2.20 TopBraid Composer

Short Description
TopBraid Composer is a professional development environment for the W3C's Semantic Web standards RDF Schema, the OWL Web Ontology Language and the SPARQL Query Language. The Free Edition is an entry-level tool for creating and editing RDF/OWL files and running SPARQL queries over them.

Status
mature

Homepage
`http://www.topquadrant.com/products/TB_Composer.html`

Description
Using a SPARQLMotion script, the basic steps of converting a spreadsheet like this to SKOS are: (1) Read in the spreadsheet as a set of RDF triples. ((2) Use a CONSTRUCT query to convert the spreadsheet triples to SKOS triples. This is the step that varies the most from one conversion to another, because people can arrange spreadsheets any way they want, so the logic of the CONSTRUCT query has to infer the correct relationships between the values on the spreadsheet. (3) Save the SKOS triples as an RDF file or in whatever format is appropriate to your applications that will use this data.

Notes
Sources: `http://www.topquadrant.com/products/TB_Composer.html`, `http://topquadrantblog.blogspot.com/2010/12/how-to-convert-spreadsheet-to-html`

GUI Description
`http://www.topquadrant.com/products/TB_Composer.html`

Supported OS
Windows, Linux, Mac

Maintainer
TopQuadrant

License

Mailing List
composersupport@topquadrant.com.

Current Version
3.4.2 from May, 2006

Usage Examples
`http://www.topquadrant.com/composer/videos/tutorials/spreadsheets/import.html`

Download Page
`http://www.topquadrant.com/products/TB_install.php`

Literature
[16]

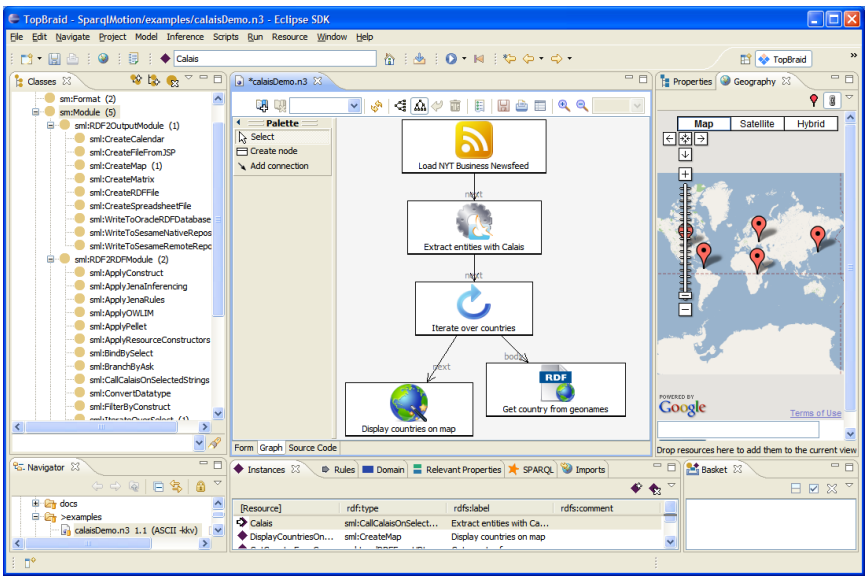


Figure 6.11: TopBraid screenshot (Source: `http://www.topquadrant.com/topbraid/composer/images/blog/Calais-SPARQLMotion-TopBraidComposer.png`).

6.2.21 Triplify

Short Description

Triplify is a simple approach to publish RDF and Linked Data from relational databases. Triplify is based on mapping HTTP-URI requests onto relational database queries expressed in SQL with some additions. Triplify transforms the resulting relations into RDF statements and publishes the data on the Web in various RDF serializations, in particular as Linked Data. Triplify as a light?weight software component, which can be easily integrated and deployed with the numerous widely installed Web applications. The approach does not support SPARQL, includes a method for publishing update logs to enable incremental crawling of linked data sources. Triplify is complemented by a library of configurations for common relational schemata and a REST?enabled datasource registry. Despite its lightweight architecture Triplify is usable to publish very large datasets, such as 160 GB of geo data from the OpenStreetMap project.

Status

mature

Homepage

`http://triplify.org`

Description

Despite significant research and development efforts the vision of the Semantic Web has not yet become reality. The growth of semantic representations is probably still outpaced by the growth of traditional Web pages and one might remain skeptical about the potential success of the Semantic Web at all. But are there alternatives? From our point of view: Not really! We think that the missing spark for starting the Semantic Web is to overcome the chicken-and-egg dilemma in the simultaneous lack of semantic representations and semantics-conscious search facilities on the Web. Triplify tackles this dilemma by leveraging relational representations behind existing Web applications. A large part of Web content is generated by database-driven Web applications. However, the structure and semantics encoded in relational database schemes is unfortunately inaccessible to Web search engines, mashups, etc. Imagine the wealth of content available for semantic searches and mashups, if the structured content of these Web applications would be accessible on the Web. Within the Semantic Web

initiative a number of standards and techniques have been developed to support the encoding and exchange of structured information and knowledge on the Web. That's the core of the Triplify approach - exploiting structured relational representations behind Web applications to create a critical mass of semantic representations on the Web. Triplify is based on the definition of relational database queries for a specific Web application in order to retrieve valuable information and to convert the results of these queries into RDF, JSON and Linked Data. Experience has shown that for most web-applications a relatively small number of queries (usually between 3-7) is sufficient to extract the important information. After generating such database views, the Triplify software can be used to convert the views into an RDF, JSON or Linked Data representation, which can be shared and accessed on the (Semantic) Web.

Notes
Description is taken from the paper and the homepage.

Maintainer
AKSW, in particular Soeren Auer, Sebastian Tramp, Jens Lehmann, Sebastian Hellmann

License
`http://gnu.org/licenses/lgpl-3.0.html`

Mailing List
mailto:triplify-discussion@lists.sourceforge.net

Current Version
0.8 from 2010-05-03

Usage Examples
Open Street maps was converted with Triplify: `http://linkedgeodata.org`

Download Page
`http://sourceforge.net/projects/triplify/files/`

Literature
[2]

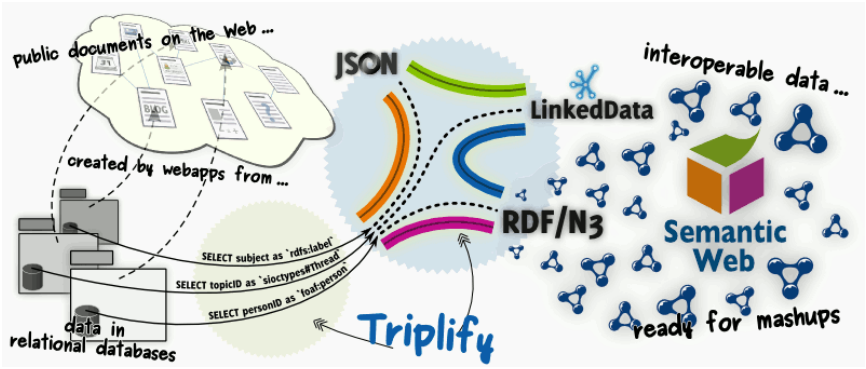


Figure 6.12: Triplify overview (Source: `http://triplify.org/files/triplify-overview.png`).

6.2.22 Virtuoso RDF Views

Short Description

Virtuoso's RDF Views map relational data into RDF and allow the RDF representation of the relational data to be customised.

Status

mature

Homepage

`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSSQLRDF`

Description

Virtuoso's RDF Views map relational data into RDF and allow the RDF representation of the relational data to be customised. Virtuoso includes a declarative Meta Schema Language for defining the mapping of SQL data to RDF ontologies. The mapping is dynamic; consequently changes to the underlying data are reflected immediately in the RDF representation. No changes are required to the underlying relational schema - so minimising disruption to a critical company asset.

GUI Description
Virtuoso Conductor Wizard for creating RDF Views, step by step or one click geenration and deployment as Linked Data

Supported OS
Windows, Linux, Mac, Unix

Maintainer
OpenLink Software

License
GPL v2, Commercial

Mailing List
`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSMailingLists`

Current Version
6.1.2 from Sept 2010

Usage Examples
`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSSQL2RDF`

Download Page
`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSDownload`

Literature
[17]

6.2.23 Virtuoso Sponger

Short Description
The Virtuoso Sponger is the Linked Data middleware component of Virtuoso that generates Linked Data from a variety of data sources, supporting a wide variety of data representation and serialization formats.

Status
mature

Homepage
`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VirtSponger`

Description
The Virtuoso Sponger is the Linked Data middleware component of Virtuoso that generates Linked Data from a variety of data sources, supporting a wide variety of data representation and serialization formats. The sponger is transparently integrated into Virtuoso's SPARQL Query Processor where it delivers URI de-referencing within SPARQL query patterns, across disparate data spaces. It also delivers configurable smart HTTP caching services. Optionally, it can be used by the Virtuoso Content Crawler to periodically populate and replenish data within the native RDF Quad Store.

GUI Description
Virtuoso Conductor Content Crawler Interface

Supported OS
Windows, Linux, Mac, Unix

Maintainer
OpenLink Software

License
GPL v2, Commercial

Mailing List
mailto:http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSMailingLists

Current Version

6.1.2 from Sept 2010

Usage Examples

`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VirtSponger#Sponger%20Usage%20Examples`

Download Page

`http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSDownload`

Literature

[18]

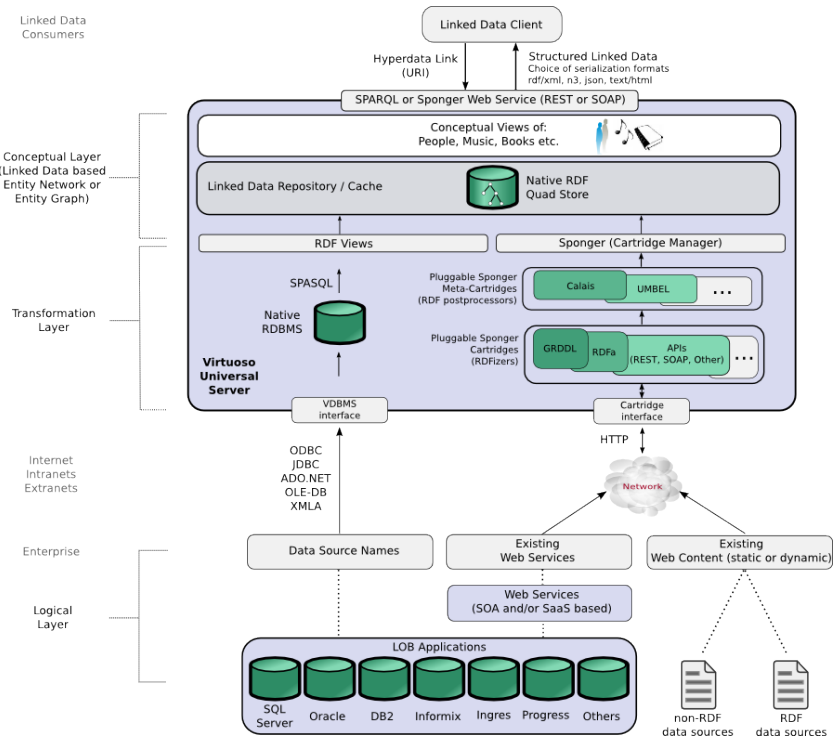


Figure 6.13: Virtuoso Sponger overview (Source: `http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VirtSponger/linked_data_gen_opts3a.png`).

6.2.24 VisAVis

Short Description

VisAVis or VisAVisTab is a Protege Plugin, that allows to populate the current Ontology with manually specifying SQL queries for OWL classes. With the help of these SQL queries, entries in the database table are mapped to OWL classes. This seems to be the predecessor to RDOTE.

Status

discontinued

Homepage

`http://www.cn.ntua.gr/~nkons/essays_en.html#t`

GUI Description

Protege Plugin

License

unknown (source code available)

Usage Examples

Hiking -¿ Field(s): activities.description with condition(s): (activities_types.name=?Hiking?). The corresponding SQL query, which is automatically generated by our SQL builder, is shown below in a snapshot of the enhanced ontology: ¡owl:Class rdf:about="#Hiking"¿ ¡queryString¿SELECT activities.description FROM activities, activities_types WHERE (activities.activity_type_id=activities_types.id) AND (activities_types.name = "Hiking") ¡/queryString¿ ¡rdfs:subClassOf¿ ¡owl:Class rdf:about="#Sports"/¿ ¡/rdfs:subClassOf¿ ¡/owl:Class¿

Download Page

`http://www.cn.ntua.gr/~nkons/essays_en.html#t`

Literature

[9]

6.2.25 XLWrap: Spreadsheet to RDF

Short Description
XLWrap is a spreadsheet-to-RDF wrapper which is capable of transforming spreadsheets to arbitrary RDF graphs based on a mapping specification. It supports Microsoft Excel and OpenDocument spreadsheets such as comma- (and tab-) separated value (CSV) files and it can load local files or download remote files via HTTP.

Status
developing

Homepage
`http://xlwrap.sourceforge.net/`

Description
Application Areas:(1) Integration of spreadsheets from distributed locations based on ontologies (together with SemWIQ, it is possible to integrate spreadsheets with other data sources such as relational databases), (2) Publication of data from spreadsheets on the Web of Data as RDF (and via SPARQL), (3) Quick setup of an editable data source for the rapid prototyping of Semantic Web and Linked Data applications.

Notes
`http://xlwrap.sourceforge.net/`

Maintainer
Andreas Langegger

License
Apache License

Mailing List
`https://sourceforge.net/mail/?group_id=267519`

Usage Examples
see `http://xlwrap.sourceforge.net/#example`, `http://xlwrap.sourceforge.net/patterns.html`

Download Page
`http://xlwrap.sourceforge.net/#download`

Literature
[12]

6.2.26 XML to RDF

Short Description

The XML2RDF mapping is part of the ReDeFer project. It allows moving metadata from the XML to the Semantic Web world in a transparent way. XML instances are mapped to RDF ones that are semantically enriched. The semantics are those previously explicited by the XSD to OWL mappings of the involved XSDs using the XSD2OWL tool.

Homepage

`http://rhizomik.net/html/redefer/#XML2RDF`

Description

It is possible to perform semantic queries on the resulting RDF that take into account the semantics of the substitutionGroup. If we use XQuery in order to retrieve MPEG-7 SegmentType descriptions from an XML database with MPEG-7 metadata, we must be aware of the hierarchy of segment types and implement an XQuery that has to cover any kind of multimedia segment, i.e. VideoSegmentType, AnalyticClipType, AudiSegmentType, etc. Once the hierarchy of segments types is available in Web Ontology Language (OWL) form, semantic queries benefit from the, now, explicit semantics. Therefore, a semantic query for SegmentType will retrieve all subclasses without requiring additional developing efforts. This is necessary because, although XML Schemas capture some semantics of the domain they model, XML tools are based on syntax. The captured semantics remain implicit from XML processing tools point of view. Therefore, when an XQuery searches for a SegmentType, the XQuery processor has no way to know that there are many other kinds of segment types that can appear in its place, i.e. they are more concrete kinds of segments.

Notes

`http://rhizomik.net/html/redefer/#XML2RDF`

Maintainer

GRIHO (Human-Computer Interaction and data integration) research group.

License

CC BY-SA 3.0

Mailing List

contact@rhizomik.net

Usage Examples

`http://rhizomik.net/html/redefer/#XML2RDF`

Download Page

`http://rhizomik.net/html/redefer/#XML2RDF`

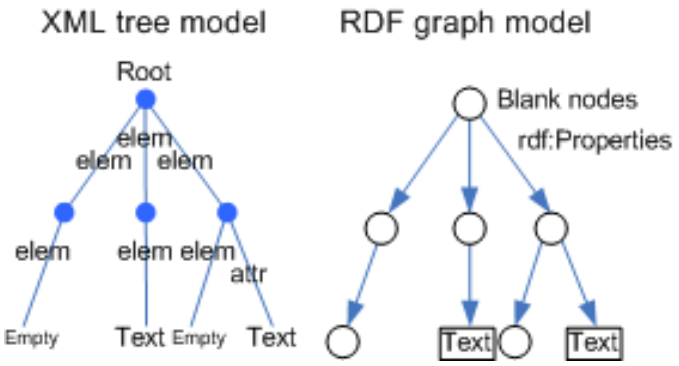


Figure 6.14: XML to RDF mapping (Source: `http://rhizomik.net/html/redefer/images/models.png`).

6.3 Conclusion and Outlook

As indicated in Chapter 2, we aided in filling a gap by providing a definition of "Knowledge Extraction". The resulting Wikipedia article was immediately extended by other users. Although, we collected and analised almost 30 tools, we found that many were only academic proof-of-concept implementations. Only 13 tools were classified as "mature" [1] and suitable for productive use. 5 of these mature tools (D2R, Triplify, Virtuoso RDF Views, Virtuoso Sponger, Pool Party Extractor) were developed by consortium members and will be extended further during the course of this project.

Out of the many options, the Google Refine RDF Extension[2] seemed especially promising. The component is currently in alpha status, but under constant development. As Google Refine[3] is considered a "power tool" and has proven its effectiveness as one of the tools used in the creation of Freebase, the consequential extension to produce RDF is almost guaranteed to be highly useful for a multitude of use cases. Zemanta[4] - who awaits to join the LOD2 consortium - plans to extend and generalize Google Refine further upon acceptance.

Although Knowledge Extraction plays an important part in LOD2, a deep integration of software into the LOD2 Stack is not strictly necessary. Knowledge Extraction serves as the basis for other methods such as interlink or enrichment and repair. In an initial step an extraction tool can be employed to create RDF out of legacy sources. An ad-hoc integration seems the most viable solution as the resulting RDF can be loaded easily into a LOD2 knowledge base and is then available to the LOD2 stack for publishing and further refinement. Such a shallow integration is necessary as legacy sources are likely to be manifold, distributed and including very diverse use cases. The conclusion is that there can not be "one" suitable Knowledge Extraction tool, but many tools that can be easily adapted to the specific use case. The integration is then done over the creation and reuse of RDF Vocabularies.

[1] browse the tools: `http://tinyurl.com/6hg8uuy`

[2] `http://data.lod2.eu/2011/tools/Google_Refine_RDF_Extension` or `http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/`

[3] `http://code.google.com/p/google-refine/`

[4] `http://www.zemanta.com/`

Bibliography

[1] Y. An, A. Borgida, and J. Mylopoulos. Discovering the semantics of relational tables through mappings. *Journal of Data Semantics*, 7:1–32, 2006.

[2] Soeren Auer, Sebastian Dietzold, Jens Lehmann, Sebastian Hellmann, and David Aumueller. Triplify: Light-weight linked data publication from relational databases. In Juan Quemada, Gonzalo Leon, Yoelle S. Maarek, and Wolfgang Nejdl, editors, *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, pages 621–630. ACM, 2009.

[3] Sören Auer, Sebastian Dietzold, Jens Lehmann, Sebastian Hellmann, and David Aumueller. Triplify: Light-weight linked data publication from relational databases. In Juan Quemada, Gonzalo León, Yoëlle S. Maarek, and Wolfgang Nejdl, editors, *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, pages 621–630. ACM, 2009.

[4] Christian Bizer and Richard Cyganiak. D2r server – publishing relational databases on the semantic web. Poster at the 5th International Semantic Web Conference (ISWC2006), 2006.

[5] Farid Cerbah. Mining the content of relational databases to learn ontologies with deeper taxonomies. In *Web Intelligence*, pages 553–557. IEEE, 2008.

[6] Cristian Perez de Laborda and Stefan Conrad. Relational.owl: a data and schema representation format based on owl. In *APCCM '05: Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling*, pages 89–96, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.

[7] R. Ghawi and N. Cullot. Database-to-Ontology Mapping Generation for Semantic Interoperability. In *Third International Workshop on Database Interoperability (InterDB 2007)*, 2007.

[8] Lushan Han, Tim Finin, Cynthia Sims Parr, Joel Sachs, and Anupam Joshi. Rdf123: From spreadsheets to rdf. In Amit P. Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy W. Finin, and Krishnaprasad Thirunarayan, editors, *International Semantic Web Conference*, volume 5318 of *Lecture Notes in Computer Science*, pages 451–466. Springer, 2008.

[9] Nikolaos Konstantinou, Dimitrios-Emmanuel Spanos, Michael Chalas, Emmanuel Solidakis, and Nikolaos Mitrou. Visavis: An approach to an intermediate layer between ontologies and relational database contents. In Flavius Frasincar, Geert-Jan Houben, and Philippe Thiran, editors, *WISM*, volume 239 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

[10] Nikolaos Konstantinou, Dimitrios-Emmanuel Spanos, and Nikolas Mitrou. Ontology and database mapping: A survey of current implementations and future directions. *J. Web Eng.*, 7(1):1–24, 2008.

[11] Christoph Lange. Krextor – an extensible XML-¿RDF extraction framework. volume 449 of *CEUR Workshop Proceedings ISSN 1613-0073*, June 2009.

[12] Andreas Langegger and Wolfram Wöß. Xlwrap - querying and integrating arbitrary spreadsheets with sparql. In Abraham Bernstein, David R. Karger, Tom Heath, Lee Feigenbaum, Diana Maynard, Enrico Motta, and Krishnaprasad Thirunarayan, editors, *International Semantic Web Conference*, volume 5823 of *Lecture Notes in Computer Science*, pages 359–374. Springer, 2009.

[13] Martin J. O'Connor, Christian Halaschek-Wiener, and Mark A. Musen. Mapping master: A flexible approach for mapping spreadsheets to owl. In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks, and Birte Glimm, editors, *International Semantic Web Conference (2)*, volume 6497 of *Lecture Notes in Computer Science*, pages 194–208. Springer, 2010.

[14] Freddy Priyatna. Rdf-based access to multiple relational data sources. Master's thesis, Universidad Politécnica de Madrid, 2009.

[15] Satya S. Sahoo, Wolfgang Halb, Sebastian Hellmann, Kingsley Idehen, Ted Thibodeau Jr, Sören Auer, Juan Sequeda, and Ahmed Ezzat. A survey of current approaches for mapping of relational databases to rdf, 01 2009.

[16] TopQuadrant. Topbraid composer. http://www.topbraidcomposer.com/, 2007. (accessed 2007-02-27).

[17] W3C. Openlink universal integration middleware - virtuoso product family. Online `http://virtuoso.openlinksw.com/`, September 2009.

[18] W3C. Openlink universal integration middleware - virtuoso product family. Online `http://virtuoso.openlinksw.com/`, September 2009.

[19] Zhaohui Wu, Huajun Chen, ShuiGuang Deng, and Yuxing Mao. Dartgrid: Rdf-mediated database integration and process coordination using grid as the platform. In Yanchun Zhang, Katsumi Tanaka, Jeffrey Xu Yu, Shan Wang, and Minglu Li, editors, *APWeb*, volume 3399 of *Lecture Notes in Computer Science*, pages 351–363. Springer, 2005.