



Combining and Uniting Business Intelligence with Semantic Technologies

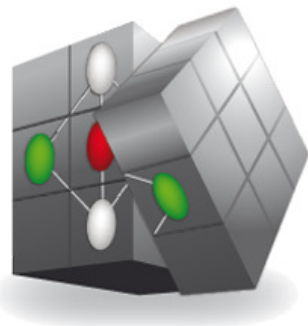
Acronym: CUBIST

Project No: 257403

Small or Medium-scale Focused Research Project

FP7-ICT-2009-5

Duration: 2010/10/01-2013/09/30



cubist

Your Business Intelligence

Semantic ETL from structured data sources Report v.1

Abstract: This deliverable will provide approaches for automated or semi-automated mapping of structured enterprise data sources (databases) to the RDF conceptual model, so that they can be integrated with the RDF triple store.

Type:	Report
Document ID:	CUBIST D2.2.1
Workpackage:	WP5
Leading partner:	SAP
Author(s):	Frithjof Dau (SAP)
Dissemination level:	PU
Status:	Final
Date:	20 October 2011
Version:	1.0



Versioning and contribution history

Version	Description	Contributors
0.1	draft	Frithjof Dau (SAP)
0.2	Done chapter on RDB2RDF	Frithjof Dau (SAP)
0.3	Introduction	Frithjof Dau (SAP)
0.4	Chapters on CSV and XML	Robert Rieger (SAP)
0.5	Extended section about W3C RDB2RDF	Marin Dimitrov (Ontotext)
0.6	Incorporate corrections from Emre Sevinç	Frithjof Dau (SAP)
0.7	Inserted four new/updated tool sections from Ontotext	Alex Simov (Ontotext)
0.8	Added short summary, minor corrections	Frithjof Dau (SAP)
0.9	Changed according to review from Emre Sevinç	Frithjof Dau (SAP)
1.0	Changed according to review from Marc Kirchhoff	Frithjof Dau (SAP)

Reviewers

Name	Affiliation
Emre Sevinç	SAS
Marc Kirchhoff	SAP



1	INTRODUCTION	4
2	CSV AND SPREADSHEET CONVERTERS.....	10
2.1	INTRODUCTION	10
2.2	TOOLS	10
3	XML CONVERTERS	26
3.1	INTRODUCTION	26
3.2	TOOLS	26
4	RDB CONVERTERS.....	30
4.1	INTRODUCTION	30
4.2	FIRST EXAMPLE APPROACH: TRIPLIFY	31
4.3	SECOND EXAMPLE APPROACH: D2R SERVER.....	33
4.4	THIRD EXAMPLE APPROACH: W3C RDB2RDF OVERVIEW.....	37
4.5	TOOLS	43
5	SUMMARY.....	65
6	REFERENCES	66



1 Introduction

The main aspect of CUBIST is to enable BI over data and information which comes from a broad variety of sources, both unstructured and structured. Data from different sources has to be digested, semantically enriched and stored in the CUBIST Information Warehouse, so that it then can be analysed by business users with the help of FCA-based Visual Analytics features provided by CUBIST.

In “classical” BI, the process of federating data from various sources into a data warehouse is well-known and referred to as ETL, standing for “extract” (i.e. the data is pulled from the sources), “transform” (i.e. the data is cleansed, enriched –e.g. with time stamps- and transformed in order to suit the overall schema of the data warehouse) and “load” (i.e. finally storing the transformed data in the data warehouse). In the style of “classical” BI, we call the process of federating data in CUBIST “Semantic ETL” (SETL). There are two main differences from the traditional ETL:

- The federated data is not stored in a Data Warehouse, but in a Triple Store instead.
- Traditional ETL only targets structured sources, whereas in CUBIST, unstructured sources are targeted as well.

Due to its importance, a work package on its own, namely “WP2: Semantic ETL and Data Integration” is dedicated to SETL. As it has already been discussed in D3.1.1. “Requirements analysis (RDF triple stores for BI)”, federating data from unstructured and structured sources is fundamentally different. For this reason, one task in WP2 (Task 2.1) is dedicated to SETL from unstructured sources, and another task (Task 2.2) to SETL from structured sources. In the diagram given next, an informal architecture of CUBIST is provided, where the task of SETL from structured sources is highlighted.

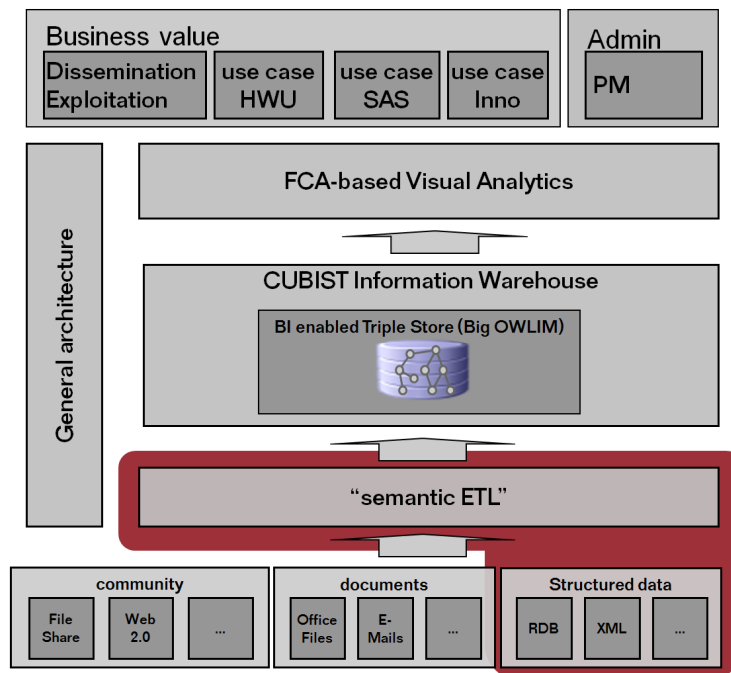


Figure 1 Semantic ETL from structured sources

In D3.1.1 (chapter 2), we have already distinguished and described the following more fine-grained types of data sources:

- sources from WWW
- unstructured text documents (word, txt, pdf)
- structured text documents (word, txt, pdf)
- spreadsheets
- XML
- databases
- existing ontologies

Among these sources, the first three need information retrieval technologies from the field of information extraction and are thus considered unstructured sources, and the latter four are structured sources. Existing ontologies can be simply reused, so we target in this deliverable spreadsheets, XML-files, and relational databases. For spreadsheets, we precisely cover their materialization in the form of csv- or tsv-files (comma-separated files and tabulator-separated files).

As argued in D3.1.1 as well, different kinds of data sources require different approaches for federating the data in them. We recap the broad approaches, as provided in D3.1.1.

- **Spreadsheets:** Spreadsheets vary to a large extent in terms of complexity, and the higher the complexity of a spreadsheet is, the more complicated it is to federate the data of the spreadsheet into a triple store. For less complex spreadsheets which are essentially tables, a standard approach to import the contained data is to model the spreadsheet rows as instances and the columns as attributes of the instances. Technically, this can be done by exporting the



spreadsheet to a csv-file, which is then imported similarly to structured text documents, or by excel-macros which directly import the data in the spreadsheet into the triple store.

- **XML:** XML is a textual, yet semi-structured data format which serves the purpose to encode documents in machine-readable form. Due to the structured form of XML, it is essentially possible to transform all information in an XML document into RDF triples and thus making it semantically searchable. For XML-documents, there exists the well-known distinction between being well-formed and being valid, the latter w.r.t. to a given schema document. Being well-formed means that the XML-document is syntactically correct, whereas being a valid document moreover conforms to a given schema document (a DTD or XSD file), thus so-to-speak being semantically well-formed (if the schema is understood to provide some semantic information of the data in the XML-file). Obviously, if a schema definition for an XML-document exists, it is easier to import the data in the xml-file into the triple store in a correct manner.
- **Databases:** Importing data from relational databases into triple stores is a well investigated field, and there exists already a variety of tools for supporting this task, e.g. *Triplify*¹ or *D2R Server*². Databases contain both schema information (structures of tables) as well as instance data (the entries in the tables). Schema information of the database can be mapped to schema information in the ontology (e.g., tables can be mapped to classes or properties, table columns can be mapped to attributes or properties), and entities can then be imported and accordingly assigned to classes or properties. The essence of importing a database is thus the mapping from the schema of the database to the ontological schema. Depending on the quality of this mapping, the entities in the database can be queried from the triple store. Moreover, some of the tools allow on-the-fly data translation of the data, i.e. there is no need to replicate data in triple stores, instead queries to the triple store are translated to SQL-queries to the relational database in real-time.

For each of these three kinds of data sources, there exist tools which enable the federation of the data in these sources into a triple store. This deliverable is a state-of-the-art-analysis of existing tools, which serves as a basis for choosing tools in the further development of the integrated CUBIST prototype, which will start after the second CUBIST milestone in M12. This deliverable is complemented by D2.1.1 “Semantic ETL from unstructured data sources”, having the same goal as D2.2.1 for unstructured sources. Both D2.1.1 and D2.2.2 will have a counterpart in M24, namely D2.1.2 “Semantic ETL from unstructured data sources v2” and D2.2.2 “Semantic ETL from structured data sources v2”, where we then will provide information about which tools are actually used in CUBIST, and how they are incorporated in the integrated prototype.

Unsurprisingly, there already exist overviews about tools for federating data from structured sources. Instead of conducting our own research from scratch, it is quite natural to reuse existing evaluations. A

¹ <http://triplify.org>

² <http://www4.wiiss.fu-berlin.de/bizer/d2rq/index.htm>



recent and decent scrutiny has been carried out by the EC-funded Integrated Project LOD2 “Creating Knowledge out of Interlinked Data”³, which has some very renowned universities and companies in the field of Semantic Technologies among their partners (e.g. the group of Chris Bizer of FU Berlin, the group of Sören Auer of the University of Leipzig, and OpenLink Software). In March 2011, they released their deliverable “D3.1.1 – State-of-the-Art Report for Extraction from Structured Sources” [LOD2]. The authors state in the deliverable that “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 [...] a snapshot of these online resources at the time of the creation of this PDF.” The lod2-consortium maintains a “Knowledge Extraction Tools Survey” (KET Survey) in an Ontowiki [KETSurvey]. This deliverable is based on the evaluation of LOD2, particularly on the data in the OntoWiki. We have adopted their list of tools and some of their data. Some data have been updated for this deliverable (e.g. we have checked whether the release dates and release versions of tools as stated in the OntoWiki are still up-to-date), and additionally for each tool we have estimated whether the tool is a candidate to be considered for CUBIST.

The tools (and the kind of data sources they target) considered in LOD2 are:

1. XML: Krextor
2. XML: XML to RDF
3. RDB: DataMaster
4. RDB: METAmorphoses
5. RDB: DartGrid
6. RDB: ODEMapster
7. RDB: Triplify
8. RDB: D2R Server
9. RDB: MAPONTO
10. RDB: RDBToOnto
11. RDB: Virtuoso RDF Views
12. structured, semi-structured: Virtuoso Sponger
13. RDB: Relational.OWL
14. RDB: VisAVis
15. XML. Text: Poolparty Extractor (PPX)
16. RDB: RDOTE
17. CSV: MappingMaster
18. CSV: RDF 123
19. CSV: XLWrap: Spreadsheet to RDF
20. CSV: CSV2RDF4LOD
21. CSV: Convert2RDF

³ www.lod2.eu



22. CSV: TopBraid Composer
23. CSV, XML: Google Refine's RDF Extension
24. CSV: OntoWiki CSV Importer
25. CSV: T2LD
26. Spreadsheets: The RDF Data Cube Vocabulary

In addition to these tools, we moreover evaluated:

1. CSV: RDF extensions for Talend Open Studio
2. CSV: CUBIST simple TSV2TTL Converter
3. RDB: Revelytix's Spyder
4. RBB: DB2Triples

LOD2's Knowledge Extraction Tools Survey provides for each tool a list of attributes which describe specific aspects of the tool. Amongst these, for CUBIST, we have selected a set of attributes relevant to CUBIST. The considered fields and their descriptions, taken from [LOD2], are provided next.

- **data source:** The data source the tool can be applied on? *RDB, XML, CSV*, etc.
- **programming language:** Programming language a project is implemented in or intended for use with.
- **data synchronization:** 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*.
- **data exposition:** 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.
- **has GUI:** Does the tool have a visual user interface? *Boolean*
- **mapping language:** The mapping language used by the approach (e.g. *SQL, R2O, D2RQ, R2RML*). The mapping language used 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 necessary. But, although SQL has extensive capabilities for selecting data in the WHERE clause, an additional mechanism for conversion and mapping is needed.
- **can reuse vocabularies:** 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*.



- **requires a Domain Ontology:** A pre-existing ontology is needed to map to it. *Boolean*
- **mapping automatisation:** The degree to which the mapping creation is assisted/automatised. *Manual, GU⁴I, semi-automatic, automatic.*
- **current version:** a string describing the current (latest) release number
- **current release date:** the date when the current (latest) release was published

The next chapters go into details for tools converting Excel/CSV, XML and Relational databases, respectively. For each tool, we first provide a table with core data of the tool. This information is mainly taken from the KET Survey. For data which might be meanwhile outdated (particularly the “current version” and “current release date” fields), we have conducted our own research in order to ensure that the information provided here is up-to-date. Moreover, compared to the KET Survey, we have sometimes altered the description of the tools.

After this core data, we provide for each tool a short estimation whether the usage of the tool is feasible in CUBIST. Main criteria applied are:

- 1) The tool must be freely (i.e. non-commercial) available
- 2) The tool must not have a viral (i.e. GPL) open-source licence: When the tool is to be integrated in the CUBIST prototype, the prototype must not be forced to be licenced under the same license terms. If a tool is a stand-alone tool which is merely used, but not integrated in the prototype, then a viral licence does not infect the prototype, and such a tool can be still considered for CUBIST.
- 3) The tool must be stable and mature to a large extent and well documented
- 4) The tool must not be outdated.
- 5) The tool should be still maintained, or there should be a community where questions can be posed.

⁴ “GUI” means here that the modelling of the mapping is conducted in a GUI.



2 CSV and Spreadsheet Converters

2.1 Introduction

Comma-separated value (CSV) files contain data that -- unlike XML -- is structured only on a syntactical level and has no explicit associated schema information. CSV is used to tabulate data using plain-text format and a delimiting symbol to distinguish one entry from the other.

Spreadsheet converters translate the tabular data into RDF by creating a temporary translation ontology which then might be mapped to existing target ontologies. The translation process is similar to the RDB2RDF translation, where rows are interpreted as entities and columns as attributes. The only drawback is that missing schema information must be provided either manually or must be guessed from the data elements.

2.2 Tools

In this section, an overview of existing tools for the conversion of CSV and spreadsheet data to RDF is provided. The list of tools as well as core information about the tools is adopted from the Knowledge Extraction Survey [KETSurvey] of the LOD2 project.



2.2.1 CSV2RDF4LOD

Name: CSV2RDF4LOD	
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. See https://github.com/timrdf/csv2rdf4lod-automation/wiki for the source code, documentation, examples, and issues tracking.
Prog. Language	Java
Data Sync	static
Data Exposition	ETL
hasGUI	No
Mapping Language	RDF Vocabulary
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manually
Current Version	06.10.2011
Current Rel. Date	October 2011
Licence	Apache License
Homepage	http://logd.tw.rpi.edu/technology/csv2rdf4lod

CUBIST Comments:

Mainly developed by one person, this tool in the current version is quite mature, though a bit cumbersome to use, since it does not provide a Graphical User Interface for the end user but works on the command line.

Estimation: Although maintained by one person only, this tool should be considered in CUBIST.



2.2.2 ConvertToRDF

Name:	ConvertToRDF
Description	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 command line, 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.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	RDF DAML
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	1.21
Current Rel. Date	n/a
Licence	n/a
Homepage	http://www.mindswap.org/~mhgrove/convert

CUBIST Comments:

Convert2RDF in the current version is an easy to use tool to transform CSV files via a mapping file into RDF. It comes with a detailed documentation on how to use it.

The licence of the tool was not retrievable. In the zip-file of the tool, there are no licence conditions. If the tool is to be used, this has to be clarified with the author of the tool.

Estimation: This tool is a CUBIST candidate.



2.2.3 Google Refine's RDF Extension

Name: Google Refine's RDF Extension	
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. Another essential feature is providing a reconciliation service extension for mapping textual data from the input data to concepts in the LOD cloud (accessible via SPARQL endpoint). This enables flexible enrichment of data on the base of semantic knowledge from LOD.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	GUI
Can reuse Voc	Yes
Req. Domain Ont	Not required, but benefits ETL
Map. Automat.	Semi-Automatic
Current Version	0.2.1
Current Rel. Date	2010-11-11
Licence	BSD
Homepage	http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/

CUBIST Comments:

This project in the current version depends on the Google Refine, a power tool for working with messy data, cleaning it up and transforming it from one format into another. Since you may import spreadsheet data into Google Refine, you may then use the RDF extension to provide the CSV data as RDF.

The RDF mapping GUI (Figure) provides an intuitive approach for mapping table columns to graph structures, reusing existing vocabularies/ontologies.



Base URI: <http://example.org/data#> [edit](#)

RDF Skeleton **RDF Preview**

Available Prefixes: dc rdfs foaf dbo owl xsd rdf [+ add prefix](#) [manage prefixes](#)

Country Name URI	<input type="checkbox"/>	X rdfs:label →	<input type="checkbox"/>	Country Name cell
X:Country		X dc:identifier →	<input type="checkbox"/>	Country Code cell
add rdf:type		X :gdp →	<input type="checkbox"/>	(blank) cell
			<input type="checkbox"/>	X rdf:value
				X :year →
				add property
		add property		

Figure 2 RDF Mapping (fragment) in Google Refine's RDF extensions

The resulting content may contain values from the input data (directly or processed with certain transformations) as well as results from a reconciliation service usage. The reconciliation itself can be executed against Freebase⁵, arbitrary SPARQL endpoints or RDF dump files.

Estimation: Although the project is still in an alpha state, it seems to be very suitable for processing noisy data by mapping it to existing semantic models (to be developed in CUBIST)

⁵ <http://www.freebase.com/>



2.2.4 MappingMaster

Name: Mapping Master	
Description	MappingMaster is an open source Protege-OWL plugin that can be used to transform the content of spreadsheets into 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.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	proprietary
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	GUI
Current Version	0.8
Current Rel. Date	2010
Licence	n/a
Homepage	http://protege.cim3.net/cgi-bin/wiki.pl?MappingMaster

CUBIST Comments:

MappingMaster in the current version is part of Protégé-OWL and provides a GUI to import and transform CSV data to an OWL representation.

Estimation: As MappingMaster is a Protege Plugin, where Protege is not a standard tool for CUBIST, this tool is not a CUBIST candidate.



2.2.5 Stats2RDF

Name:	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.
Prog. Language	PHP, javascript
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	Data Cube Vocabulary
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Semi-automatic
Current Version	n/a
Current Rel. Date	n/a
Licence	GNU GPL v2, CC BY-SA 3.0
Homepage	http://aksw.org/Projects/Stats2RDF

CUBIST Comments:

Stats2RDF in the current version depends on OntoWiki and can only be used with it.

Estimation: As Stats2RDF is not a standalone tool or component to be included in a CUBIST prototype, it is not a CUBIST candidate.



2.2.6 RDF 123

Name: RDF 123	
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 labelled nodes, attach a XSD data type, 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.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	None
Can reuse Voc	No
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	1
Current Rel. Date	April 2007
Licence	GPL, CC BY 2.0, MIT License
Homepage	http://ebiquity.umbc.edu/project/html/id/82/RDF123

CUBIST Comments:

RDF123 in the current version is a quite basic tool to transform given CSV data into RDF using manual mapping definitions. It comes as desktop application for different operation systems. It is cumbersome to use.

Estimation: As RDF123 is a quite basic tool and cumbersome to use, it is not a CUBIST candidate.



2.2.7 T2LD

Name: T2LD	
Description	T2LD is an automatic framework for extracting, interpreting and generating linked data from tables. In the process of representing tables as linked data, the tool assigns 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 identifies relations between various columns in the table, which helps 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, the tool adopts 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, it uses 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.
Prog. Language	
Data Sync	Static
Data Exposition	ETL
hasGUI	No
Mapping Language	No
Can reuse Voc	No
Req. Domain Ont	No
Map. Automat.	Automatic
Current Version	n/a
Current Rel. Date	August 2010
Licence	n/a
Homepage	http://ebiquity.umbc.edu/paper/html/id/480/T2LD-An-automatic-framework-for-extracting-interpreting-and-representing-tables-as-Linked-Data



CUBIST Comments:

T2LD particularly addresses the generation of *Linked Data*. The automatic mapping generation focuses on Linked Data sets (e.g. T2LD predict classes from four vocabularies – DBpedia Ontology, Freebase, WordNet and Yago). T2LD is the master thesis result of one student.

Estimation: Since T2LD seems not to be maintained anymore and CUBIST is not focusing on LinkedData sets, this tool is not a CUBIST candidate.



2.2.8 TopBraid Composer

Name: TopBraid Composer	
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.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	SKOS
Can reuse Voc	No
Req. Domain Ont	No
Map. Automat.	Semi-automatic
Current Version	3.4.2
Current Rel. Date	May, 2006
Licence	commercial
Homepage	http://www.topbraidcomposer.com

CUBIST Comments:

TopBraid Composer comes in a free, standard and maestro edition, where only the commercial standard edition supports spreadsheet data import.

Estimation: Due to the commercial license, this is not a CUBIST candidate.



2.2.9 XLWrap

Name: XLWrap	
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.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	No
Mapping Language	TriG Syntax
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	0.3 RC
Current Rel. Date	n/a
Licence	Apache License
Homepage	http://xlwrap.sourceforge.net/

CUBIST Comments:

- XLWrap is a standalone tool to manually map spreadsheet data to RDF. It seems sufficiently documented though still in an early stage of development.
- The mapping language of XLWrap seems quite powerful.

Estimation: XLWrap is a candidate for CUBIST.



2.2.10 RDF extensions for Talend Open Studio

Name: RDF Extensions for Talend Open Studio	
Description	Talend Open Studio ⁶ is a powerful and versatile open source solution for data integration. It comprises three major applications (Business Modeler, Job Designer, and Metadata Manager) within a single graphical development environment based on Eclipse, which is easily adapted to corporate needs.
Prog. Language	Java/Eclipse
Data Sync	Static and Dynamic
Data Exposition	ETL
hasGUI	Yes
Mapping Language	GUI
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	1
Current Rel. Date	October 2011
Licence	GPL
Homepage	http://confluence.ontotext.com/display/LIFESKIM/Custom+components+documentation

CUBIST Comments:

The Studio itself comes with variety of components dealing with: all popular types of RDBMS; different types of local files (cvs, xml, txt); different kinds of data processing (aggregation, filtering, sorting, mapping, etc.). All together, they provide environment for very sophisticated data transformations and management.

⁶ <http://www.talend.com/products-data-integration/talend-open-studio.php>



With the addition of RDF components to the environment, the whole ETL machinery can be applied directly to the semantic modelling area. The RDF extension components can be grouped as:

- RDF Parsers and Serializers
- Connectors to remote triple stores, allowing querying and updates
- Semantic annotation of unstructured documents (GATE⁷ based)

The following Figure 1 gives a feeling about the modelling environment of the Studio

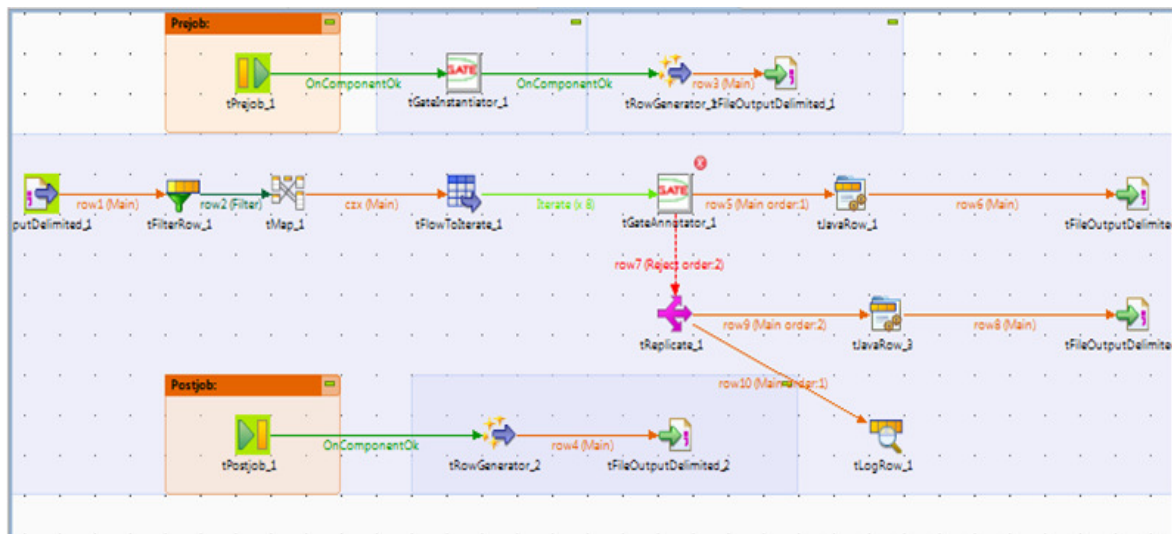


Figure 1. Semantic annotation modelling in Talend Open Studio

Estimation: In spite of its viral GPL license, the Studio is still a good candidate for using in CUBIST. The license does not imply any restrictions on the usage and the outcomes of the tool.

⁷ <http://gate.ac.uk/>



2.2.11 CUBIST Simple TSV2TTL Converter

Name: CUBIST Simple TSV2TTL Converter	
Description	A simple conversion tool which is capable to import comma, semicolon- or tab-separated textfiles and, based on a template, produce a Turtle (.ttl) ⁸ file out of them.
Prog. Language	Java/Eclipse
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	Proprietary (simple text substitution with wildcards)
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	1
Current Rel. Date	October 2011
Licence	n/a
Homepage	n/a

CUBIST Comments:

For the first conversion task in CUBIST, a simple own tool has been developed. It can load comma, semicolon- or tab-separated text files and, based on a conversion-template, generates a Turtle (.ttl) file out of the imported data. A screenshot is provided below.

Evaluation: As this is a CUBIST-internal prototype, this is certainly a candidate for CUBIST

⁸ Turtle - Terse RDF Triple Language, <http://www.w3.org/TeamSubmission/turtle/>

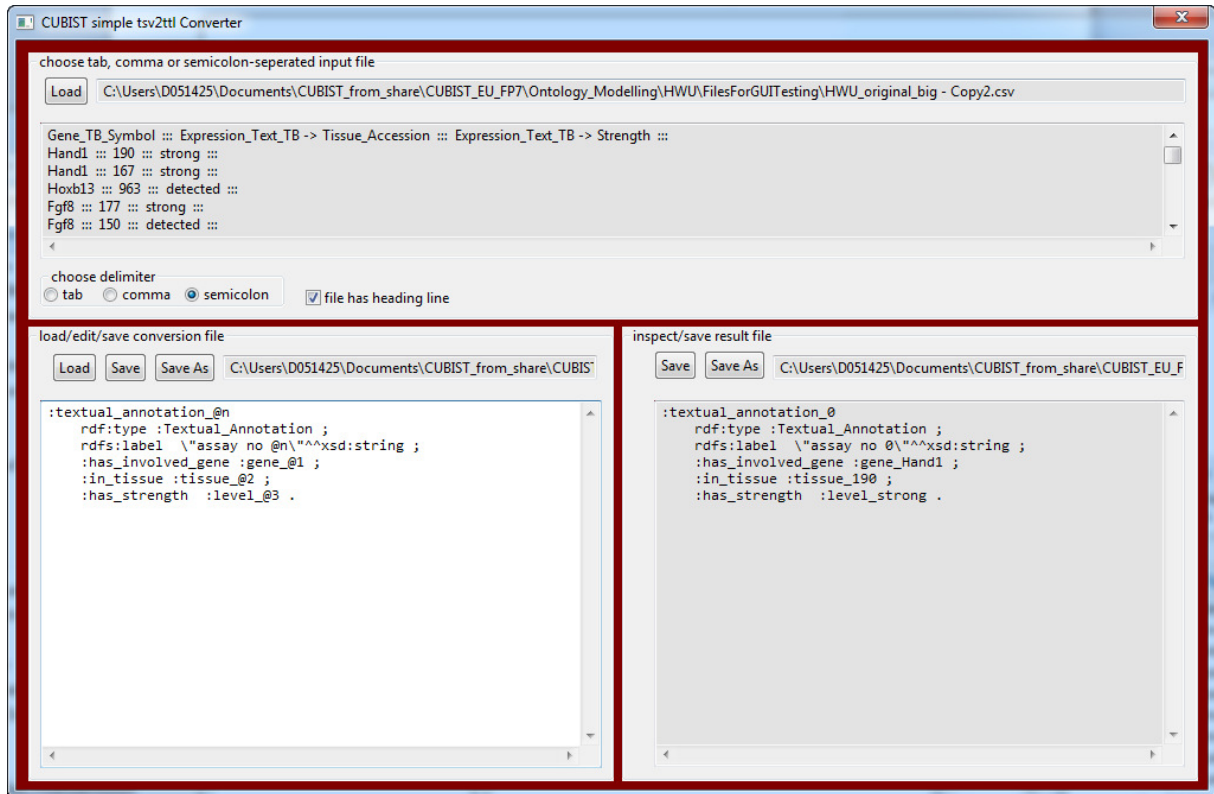


Figure 3: CUBIST internal csv2ttl converter



3 XML Converters

3.1 Introduction

Documents represented using the eXtensible Markup Language (XML) usually reference also a schema document (XSD) describing the higher level model of instance files. XML is structured as a tree with edges associating element nodes with attributes or other elements. Thus, it can be easily represented in RDF, which is structured as a graph. Usually, approaches that transform XML into RDF use the eXtensible Stylesheet Language Transformation (XSLT) to map from a source schema to a target one. This process however, is more complex than converting relational databases into RDF. In the XML world, users have to design, whether an XML element represents a subject, predicate or an object in terms of RDF.

3.2 Tools

In this section, an overview of existing tools for the conversion of XML data to RDF is provided. The list of tools as well as core information about the tools is adopted from the Knowledge Extraction Survey [KETSurvey] of the LOD2 project.



3.2.1 Krestor

Name:	Krestor
Description	Krestor, 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. Krestor provides convenience templates that try to do “the right thing”™ in many common cases, as to reduce the need for manually writing repetitive code.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	no
Mapping Language	XSLT
Can reuse Voc	yes
Req. Domain Ont	yes
Map. Automat.	Manual
Current Version	0.3
Current Rel. Date	2008-12-31
Licence	LGPL
Homepage	http://trac.kwarc.info/krestor/

CUBIST Comments:

Krestor relies on a default approach of XSLT transformation to generate RDF out of XML data. It seems not to be heavily maintained, since the latest version is from 2008. In the project’s issue tracker, there are currently 71 tickets containing 11 defects stemming from 2008 to 2010 with only one of them assigned to a project member.

Estimation: Due to the comments given above, Krestor is not a CUBIST candidate.



3.2.2 Poolparty Extractor PPX

Name:	
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 processed 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 thus 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.
Prog. Language	Java
Data Sync	Dynamic
Data Exposition	Linked Data
hasGUI	No
Mapping Language	RDF (SKOS)
Can reuse Voc	Yes
Req. Domain Ont	Yes
Map. Automat.	Semi-automatic
Current Version	2
Current Rel. Date	2011-01-31
Licence	Commercial
Homepage	http://poolparty.punkt.at/

CUBIST Comments:

PPX is a commercial tool provided by the Semantic Web Company.

Estimation: Due to the commercial license, PPX is not a CUBIST candidate.



3.2.3 XML2RDF

Name: XML2RDF	
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 explicated by the XSD to OWL mappings of the involved XSDs using the XSD2OWL tool.
Prog. Language	n/a
Data Sync	Static
Data Exposition	ETL
hasGUI	No
Mapping Language	None
Can reuse Voc	No
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	n/a
Current Rel. Date	n/a
Licence	n/a
Homepage	http://rhizomik.net/html/redefer/#XML2RDF

CUBIST Comments:

XML2RDF seems to be no reusable standalone tool or component. The homepage offers a web form to submit XML data and returns a RDF document. Version and license information could not be found during the compilation of this document.

Estimation: Due to the comments given above, XML2RDF is not a CUBIST candidate.



4 RDB Converters

In this chapter, converters from relational databases to RDFS are investigated. We provide a short description of the basic idea of such converters, followed by two tools which realize such converters, as well as a section describing the current W3C effort to provide a standardized mapping language for converters. Finally, in the last section, fourteen existing tools are listed. For each tool, based on the information the KET Survey from the LOD2 project and our own research, a description as well as an estimation whether the tool should be further considered for CUBIST is provided.

4.1 Introduction

The basic idea of transforming a relational database into an RDFS ontology is well-described in the whitepaper “OpenLink Software. Virtuoso RDF Views – Getting Started Guide” [RDF Views]. The next four paragraphs are taken from that whitepaper (and can be found on http://en.wikipedia.org/wiki/Knowledge_extraction as well).

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 (Internationalized Resource Identifiers, being a generalization of the Uniform Resource Identifiers). assign a predicate IRI to each column
3. assign an `rdf:type` predicate for each row, linking it to an RDFS class IRI corresponding to the table
4. 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.



4.2 First Example Approach: Triplify

Triplify is a small, lightweight (less than 500 lines of code) plugin for Web applications, programmed in PHP5. According to triplify.org, “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.” Moreover, the authors of Triplify estimate that “for a typical Web application a configuration for Triplify can be created in less than one hour”.

The mappings from databases to RDFS are expressed in a PHP script with an SQL-based language. They generate class instances, object- and data type properties, but it should be noted that they have only limited data transformation capabilities (apart from SQL).

We demonstrate Triplify with the Innovantage Use Case of CUBIST. In this use case, we deal with structured & textual representation of on-line advertised vacant job positions in UK. Relevant data types are job adverts, company records and job boards, which are currently stored in a relational database. In Figure 4 a small fraction of the database schema and the targeted ontology are depicted.

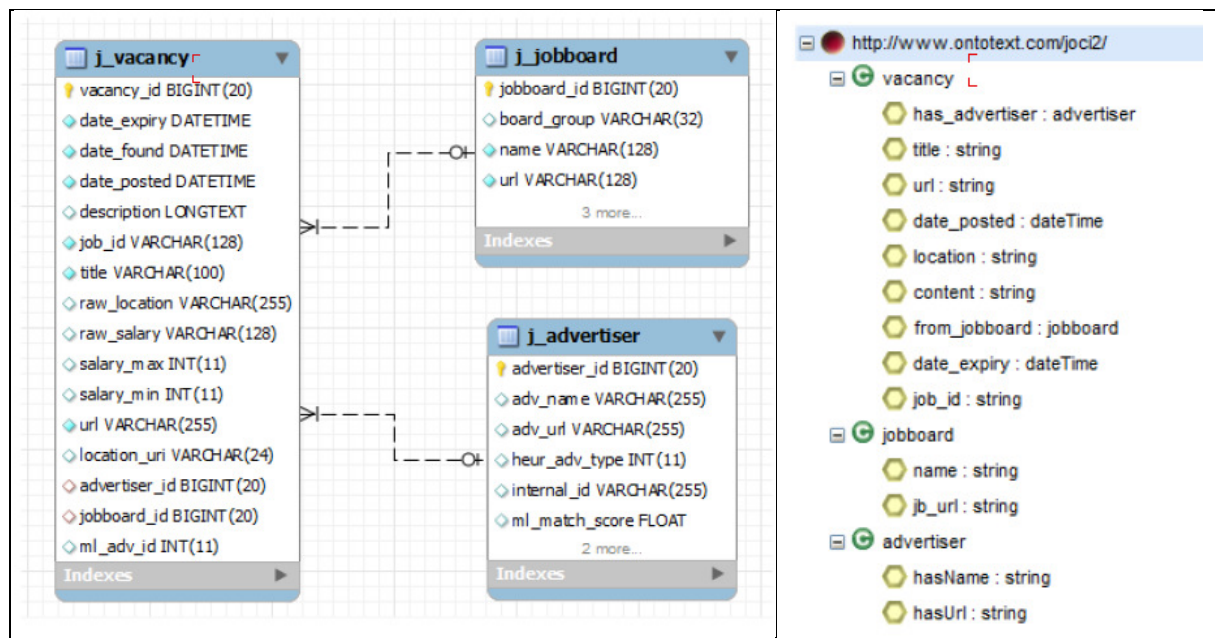


Figure 4 Simplified database schema (3 of 31 tables) and target triple-set ontology

Note that

- not all attributes of the table are used (e.g. “date_found”)
- attributes might be renamed during the transition (e.g. “description” is renamed to “content”, or “raw_location” it renamed to “location”)



- foreign key conditions between the tables have to be translated to object-properties in the ontology

Next, the formal mapping description of Triplify is provided.

```
$triplify['queries']=array(  
  'vacancy'=>  
    "SELECT vac.vacancy_id AS id,  
            vac.date_expiry AS 'date_expiry',  
            vac.date_posted AS 'date_posted',  
            vac.description AS 'sioc:content',  
            vac.job_id AS 'job_id',  
            vac.title AS 'dc:title',  
            vac.raw_location AS 'location',  
            vac.url AS 'vacancy url',  
            vac.advertiser_id AS 'has_advertiser',  
            vac.jobboard_id AS 'from_jobboard'  
    FROM joci2.j_vacancy vac",  
  'advertiser'=>  
    "SELECT advertiser_id AS id,  
            adv_name AS 'hasName',  
            adv_url AS 'hasUrl'  
    FROM joci2.j_advertiser",  
  'jobboard'=>  
    "SELECT jb.jobboard_id AS id,  
            jb.formatted_name AS 'name',  
            jb.url AS 'jb_url'  
    FROM joci2.j_jobboard jb"  
);  
$triplify['objectProperties']=array(  
  'has_advertiser'=>'advertiser',  
  'from_jobboard'=>'jobboard'  
);
```

Figure 5 Mapping description (fragment) for Triplify

As one can see:

- We have for each table a query which translates and renames selected attributes of the table to RDFS-properties (both data-properties and object-properties)
- The first column for each query represents the ID of the object and has to be named "id", all other columns represent characteristics.
- The data types of attributes are not specified. Indeed, “Triplify uses SQL introspection to retrieve automatically the data type of a certain column and create RDF literals with matching XSD datatypes” [Triplify]. It is anyhow possible to instruct Triplify to generate RDF literals of a certain datatype. For example, it is possible to write as 'date_posted ^^xsd:dateTime'
- We have dedicated instructions which generate the object-properties between instances (see the highlighted parts of the mapping)

In the next figure, the resulting triples for one job posting, as they appear in Ontotext’s BIGOwlim Triple Store, are shown.

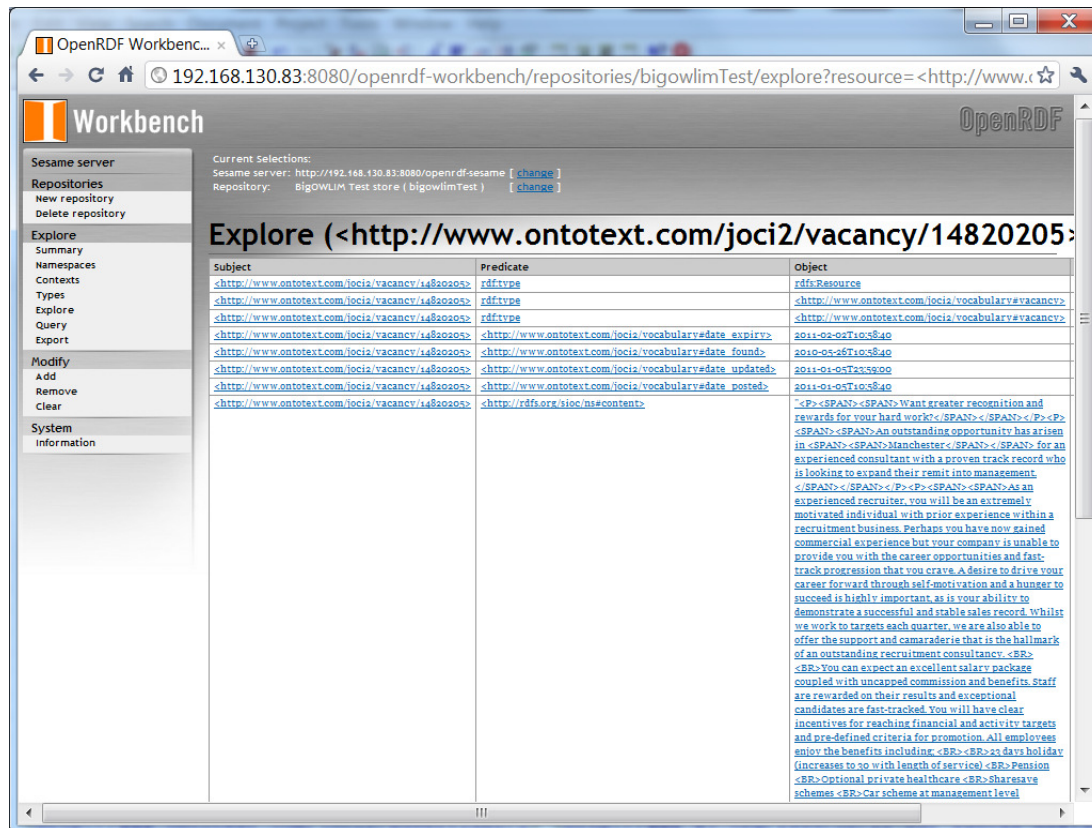


Figure 6 Triples generated with Triplify

4.3 Second Example Approach: D2R Server

D2RServer is a tool which is able to turn relational databases into RDF data or SPARQL endpoints. It runs either as stand-alone web server or inside an existing servlet container. It is a quite old and famous tool: Its development started in 2003 at FU Berlin, Germany [D2R Map] and as of Dec. 2010 has been downloaded 13.300 times⁹.

D2R provides different means to transform relational data into RDF data:

- It is possible to materialize the data
- D2R can create a SPARQL endpoint without transforming the database, i.e. it can provide semantic access to the relational data on the fly.
- Finally, it can also provide on-demand RDF/HTML pages following the LOD protocol

The overall structure of D2R is depicted below.

⁹ <http://precedings.nature.com/documents/5660/version/1/files/npre20115660-1.pdf>, slide 8

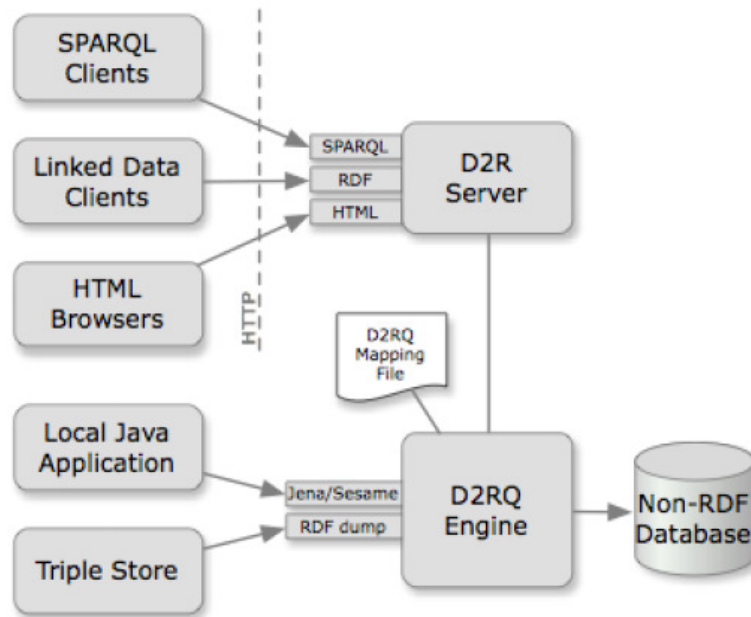


Figure 7 D2R

D2R provides a declarative language¹⁰ for mapping databases to RDF. The mapping itself is expressed as an RDF document (the mapping language is very similar to R2RML –being described in the next section- and to some extent can be considered as a predecessor to R2RML). Mappings can be manually created or automatically generated, latter allowing an easy bootstrapping. According to <http://www4.wiwiss.fu-berlin.de/bizer/d2rmap/d2rmap.htm>, the mapping process executed by the D2R processor has four logical steps:

1. Selection of a record set from the database using SQL
2. Grouping of the record set by the `d2r:groupBy` columns.
3. Creation of class instances and identifier construction.
4. Mapping of the grouped record set data to instance properties.

A diagram for this process is given below.

¹⁰ <http://www4.wiwiss.fu-berlin.de/bizer/d2rq/spec/#specification>

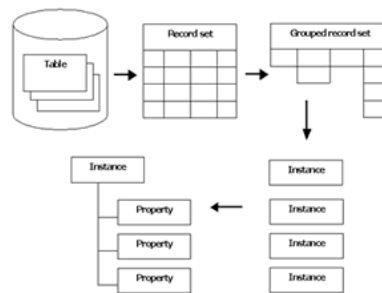


Figure 8 Mapping process in D2R

Mappings from databases to ontologies in D2R are described using `d2rq:ClassMaps` and `d2rq:PropertyBridges`. A `ClassMap` represents a class or a group of similar classes of an ontology, and it specifies how instances of the class are identified. A `ClassMap` in turn has a set of `PropertyBridges`, which specify how the properties of an instance are created. `Property Bridges` relate database table columns to RDF properties. The values of these properties are often literals, but can also be URIs or blank nodes that relate the resource to other resources.

To see an example, consider the following database and the targeted triples to be generated.¹¹

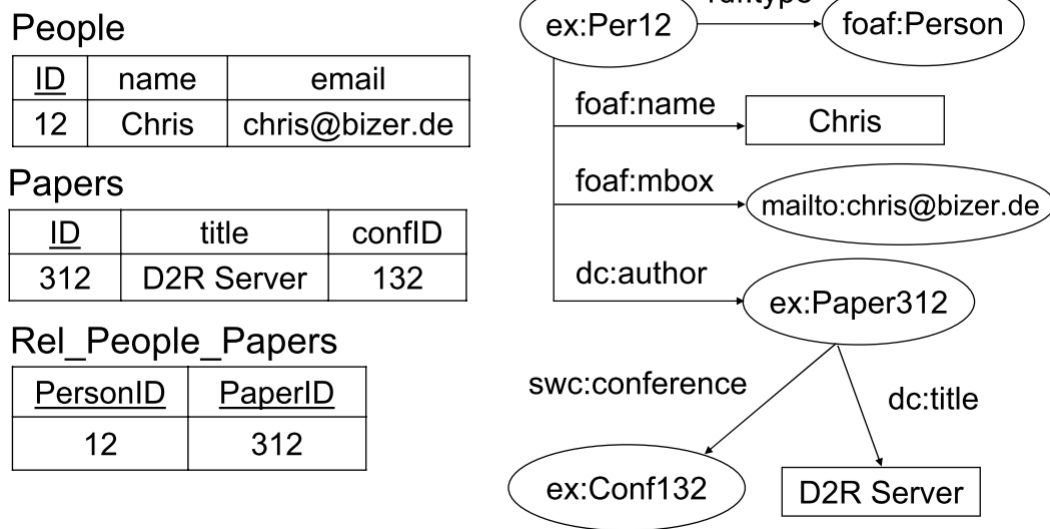


Figure 9 An example DB and targeted triples for D2R

¹¹ The example is adopted from <http://www4.wiwiw.fu-berlin.de/bizer/pub/Bizer-HUBerlin-Talk.pdf>



A graph-based depiction of the corresponding mapping is provided below.

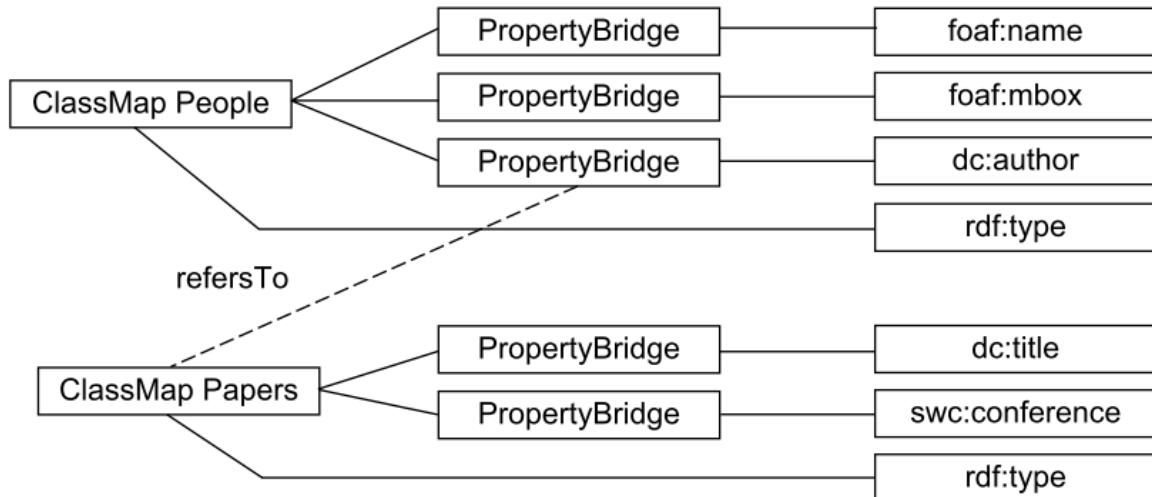


Figure 10: a D2RQ mapping example

Finally, it should be noted that D2R is already used in the field of gene expression research, i.e. in a setting similar to the HWU use case of CUBIST.

- On <http://www.bioontology.org/wiki/index.php/OBD:SPARQL-GO>, a preliminary pilot/experimental project for providing a SPARQL endpoint to the 6 million or so annotations in the Gene Ontology database is described. D2RQ is used to map the GO DB schema to RDF and d2r-server as a SPARQL endpoint. The D2R mappings are provided there as well.
- <http://www.bioontology.org/wiki/index.php/OBD:SPARQL-InSitu> describes a SPARQL endpoint for a database containing annotated images of gene expression in fruit fly embryogenesis. Though HWU does not deal with fruit flies (but with mouse embryos), it is worth looking at their approach for incorporating pictures and their annotations into an ontology.



4.4 Third Example Approach: W3C RDB2RDF overview

The RDB2RDF Working Group at W3C [RDB2RDF] was established in 2010 in order to standardize a language (called R2RML) for mapping relational databases into RDF/OWL. While tools and approaches for such mappings have been available for several years already, the lack of a standard recommendation has resulted in custom and incompatible ETL approaches for RDF-izing legacy relational data.

The RDB2RDF WG focuses on three use cases for the design of the R2RML mapping language (Figure):

1. *Integrate a legacy RDBMS with existing RDF data* (which requires a way to convert the relational database into RDF)
2. *Integrate a legacy RDBMS with another structured or unstructured data source* (similar to the above with the assumption that the other data sources can also be converted into RDF)
3. *Provide a SPARQL endpoint on top of a legacy RDBMS*

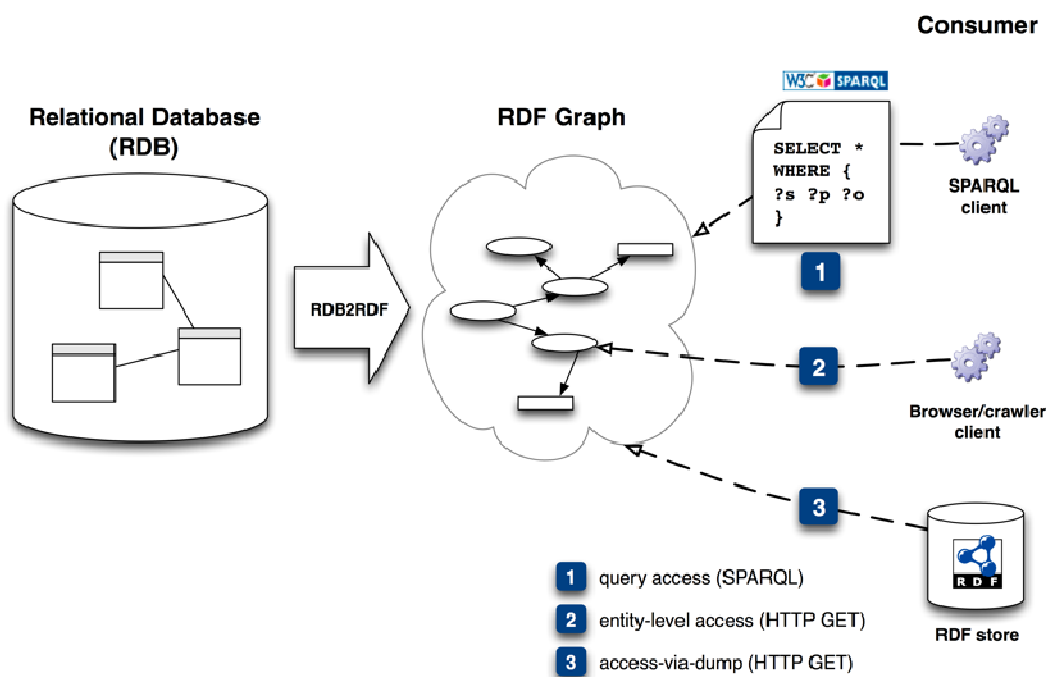


Figure 11 RDB2RDF principle, (c) W3C

The RDB2RDF WG will provide two main specifications:

- “A Direct Mapping of Relational Data to RDF” [RDB-direct], which defines a simple direct mapping of a relational database where the target RDF vocabulary directly reflects the schema names in the database;
- “R2RML: RDB to RDF Mapping Language” [R2RML], which makes it possible for highly customized mappings over relational schemata to be defined.



4.4.1 Direct mapping

The direct mapping will generate an RDF graph (also called *direct graph*) based directly from the schema of the particular relational database. The structure or the names and identifiers in the resulting RDF graph cannot be customized and directly reflect the structure of the source relational schema.

The direct graph generated by the direct mapping of a relational schema is the union of all the *table graphs* which are generated for each table in the source schema. Each table graph itself contains all the triples generated for the rows in that table. For each table row in the source database three types of triples can be generated:

- *Row type triple* – only one per row, it simply states that the respective row node is of the class (*rdf:type*) corresponding to the class representing the table. The row type triple contains: subject is the row node itself, predicate is *rdf:type*, object is IRI generated for the table
- *Literal triple* – one for each column in the table, it has the following parts: subject is the row node itself, predicate is IRI generated for the respective column in the table, object is the value in the respective (row, column) cell of the table
- *Reference triple* – one for each FK (foreign key) column in the table, it has the following structure: subject is the row node itself, predicate is IRI generated for the respective column in the table, object is the row node of the referenced row

Each element of the source schema (table, column, row) has a unique identifier in the direct graph, which is generated from its name in the source relational schema as follows:

- *Table IRIs* are generated from the percent-encoded table names
- *Column IRIs* are the concatenation of the percent-encoded table name, the hash symbol (“#”) and the percent-encoded column name
- *Row IRIs* are the concatenation of the percent-encoded table name, the solidus symbol (“/”), and *for each column* part of the PK (primary key): 1) the percent-encoded column name, 2) the minus character (“-“), and 3) the percent-encoded value of the PK cell.

The direct mapping also will map the values in a table cell (row, column) from the SQL data types to the corresponding XML Schema data types as follows:

SQL data type	XSD data type
BINARY, BINARY LARGE OBJECT	xsd:base64Binary
NUMERIC, DECIMAL	xsd:decimal
SMALLINT, BIGINT, INTEGER	xsd:integer
FLOAT, REAL, DOUBLE PRECISION	xsd:double
BOOLEAN	xsd:boolean
DATE	xsd:date
TIME	xsd:time
TIMESTAMP	xsd:dateTime



4.4.2 R2RML mapping language

W3C works on a language called R2RML for “expressing customized mappings from relational databases to RDF datasets. Such mappings provide the ability to view existing relational data in the RDF data model, expressed in a structure and target vocabulary of the mapping author's choice. R2RML mappings are themselves RDF graphs and written down in Turtle syntax. R2RML enables different types of mapping implementations. [...]The mapping is conceptual; R2RML processors are free to materialize the output data, or to offer virtual access through an interface that queries the underlying database, or to offer any other means of providing access to the output RDF dataset. (see <http://www.w3.org/TR/r2rml/>).

As of October 2011, the website <http://www.w3.org/TR/r2rml/> is a “Last Call Working Draft of the "R2RML: RDB to RDF Mapping Language". Publication as a Last Call Working Draft indicates that the RDB2RDF Working Group believes it has addressed all substantive issues and that the document is stable. The Working Group expects to advance this specification to Recommendation Status”.

In the following, core features of R2RML are explained, based on the examples and explanations of <http://www.w3.org/TR/r2rml/>.

First of all, R2RML mappings refer not only to physical, but logical tables in a database, as there are tables, views, or arbitrary SQL queries (which could be understood as views as well). This allows to translate expressions which can be calculated with SQL to be mapped to RDFS. An example for this will be given later.

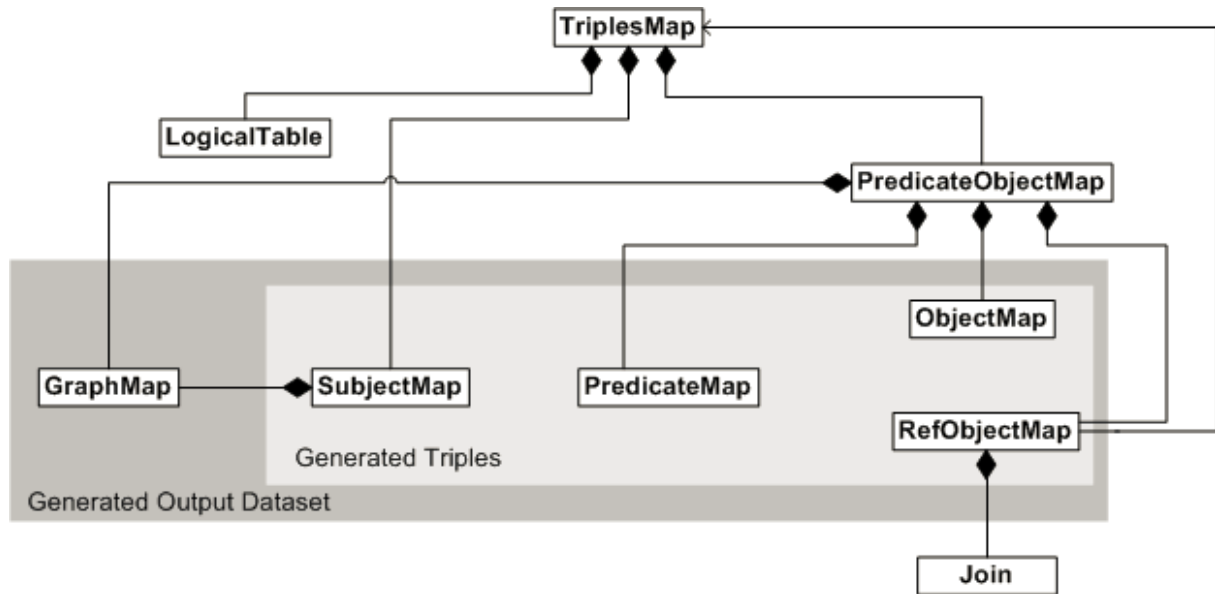
Mappings are realized via triples maps, which are rules that map each row in a (logical) table to a set of RDF triples. They have two main parts:

1. subject maps that generate the subjects of all RDF triples (i.e. IRIs)
2. Multiple predicate-object maps that in turn consist of predicate maps and object maps, which generate predicates and objects of triples.

Triples are produced by combining the subject map with a predicate map and object map, and applying these three to each logical table row.

R2RML can generate named graphs. The output dataset of an R2RML mapping contains triples of the default graphs as well as triples in other named graphs. For mapping triples to a named graph, graph maps are used.

In the next figure, an overview of R2RML is provided.



We consider now an example database with two tables EMP and DEPT, and one data row (only) in each table.

DEPT		
DEPTNO	DNAME	LOC
INTEGER PRIMARY KEY	VARCHAR(30)	VARCHAR(100)
10	APPSERVER	NEW YORK

EMP			
EMPNO	ENAME	JOB	DEPTNO
INTEGER PRIMARY KEY	VARCHAR(100)	VARCHAR(20)	INTEGER REFERENCES DEPT (DEPTNO)
7369	SMITH	CLERK	10

The desired RDF triples to be produced from this database are as follows:

```

<http://data.example.com/employee/7369> rdf:type ex:Employee.
<http://data.example.com/employee/7369> ex:name "SMITH".
<http://data.example.com/employee/7369> ex:department
<http://data.example.com/department/10>.

<http://data.example.com/department/10> rdf:type ex:Department.
<http://data.example.com/department/10> ex:name "APPSERVER".
<http://data.example.com/department/10> ex:location "NEW YORK".
<http://data.example.com/department/10> ex:staff 1.
  
```

We start with a mapping for the instances of the EMP table only. The mapping is given next, both in turtle-notation and with a graph-based depiction.

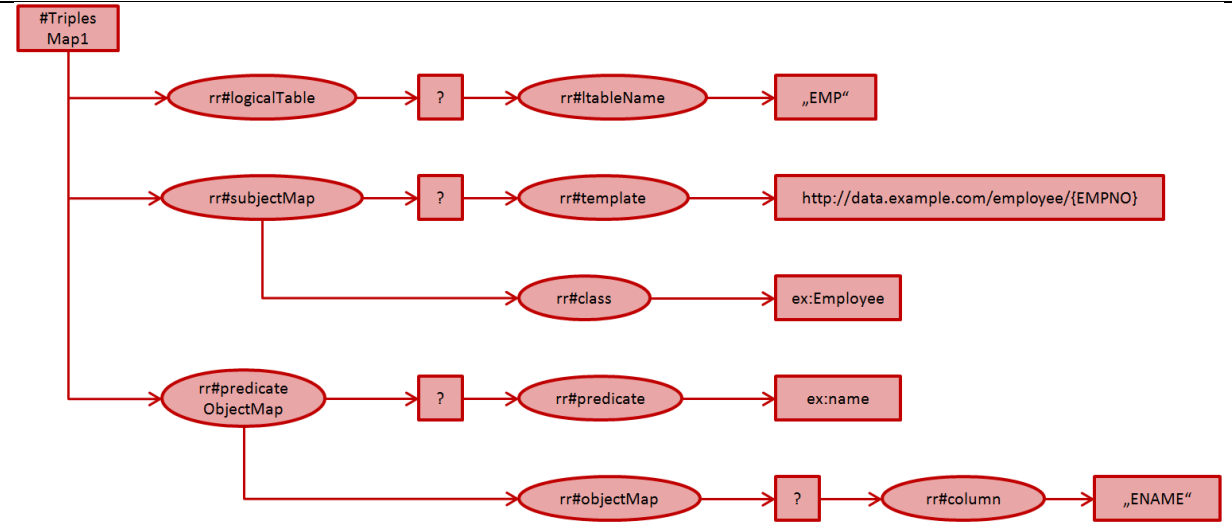


```

@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <@prefix rr: <http://example.com/ns#>.

<#TriplesMap1>
  rr:logicalTable [ rr:tableName "EMP" ];
  rr:subjectMap [
    rr:template "http://data.example.com/employee/{EMPNO}";
    rr:class ex:Employee;
  ];
  rr:predicateObjectMap [
    rr:predicate ex:name;
    rr:objectMap [ rr:column "ENAME" ];
  ].

```



This mapping yields the triples for the EMP, table, i.e.

```

<http://data.example.com/employee/7369> rdf:type ex:Employee.
<http://data.example.com/employee/7369> ex:name "SMITH".

```

Next we consider the DEPT table. Note that in the desired output triples, we have a triple

```

<http://data.example.com/department/10> ex:staff 1.

```

i.e. for a department the number of employees of that department has to be computed. This is not done by R2RML: Instead, we define a R2RML view, based on a SQL-query, which does the job, and use this view then. The SQL-query is as follows:

```

<#DeptTableView> rr:sqlQuery """
SELECT DEPTNO,
       DNAME,
       LOC,
       (SELECT COUNT(*) FROM EMP WHERE EMP.DEPTNO=DEPT.DEPTNO) AS STAFF
FROM DEPT;
""".

```



The triples map that generates the desired DEPT triples based on this R2RML view is as follows:

```
<#TriplesMap2>
  rr:logicalTable <#DeptTableView>;
  rr:subjectMap [
    rr:template "http://data.example.com/department/{DEPTNO}";
    rr:class ex:Department;
  ];
  rr:predicateObjectMap [
    rr:predicate ex:name;
    rr:objectMap [ rr:column "DNAME" ];
  ];
  rr:predicateObjectMap [
    rr:predicate ex:location;
    rr:objectMap [ rr:column "LOC" ];
  ];
  rr:predicateObjectMap [
    rr:predicate ex:staff;
    rr:objectMap [ rr:column "STAFF" ];
  ].
```

This map generates the following triples:

```
<http://data.example.com/department/10> rdf:type ex:Department.
<http://data.example.com/department/10> ex:name "APPSERVER".
<http://data.example.com/department/10> ex:location "NEW YORK".
<http://data.example.com/department/10> ex:staff 1.
```

To this end, we have for each table generated the subjects and the corresponding triples, but we still have to link the two tables. I.e. we have to generate triples where the subjects come from the first triples map (<#TriplesMap1>), the objects come from the second triples map (<#TriplesMap2>). This can be achieved by adding another `rr:predicateObjectMap` to <#TriplesMap1>, using <#TriplesMap2>, as a so-called parent triples map:

```
<#TriplesMap1>
  rr:predicateObjectMap [
    rr:predicate ex:department;
    rr:objectMap [
      rr:parentTriplesMap <#TriplesMap2>;
      rr:joinCondition [
        rr:child "DEPTNO";
        rr:parent "DEPTNO";
      ];
    ];
  ].
```

This performs a join between the EMP table and the R2RML view, on the DEPTNO columns. The objects will be generated from the subject map of the parent triples map, yielding the desired triple:

```
<http://data.example.com/employee/7369> ex:department
<http://data.example.com/department/10>.
```



This concludes the example from <http://www.w3.org/TR/r2rml/>. As stated in the beginning, R2RML is in its final draft, and it is expected to become a W3C-recommendation. The CUBIST consortium is not aware of tools which are already capable of dealing with R2RML. For the time being, there exist only partial, experimental implementations. For some existing tools there are plans to incorporate R2RML. For example, OpenLink Software is "currently working on an R2RML to RDF Views translator" (statement from Garry Biggs from OpenLink Software on 2011-09-26, see <http://answers.semanticweb.com/questions/11781/strategies-for-mapping-rdb-to-rdf>). With respect to R2RML there is no current need for action within CUBIST, but of course we will follow the further development of R2RML. It is indeed expected that full implementations will come, and replace D2R.¹²

4.5 Tools

In this section, an overview of existing tools for the conversion of data in relational databases to RDF is provided. The list of tools as well as core information about the tools is adopted from the Knowledge Extraction Survey [KETSurvey] of the LOD2 project.

It turned out that six out of fourteen tools can be considered for CUBIST. The most important ones are Triplify and D2R, which already have been described in previous sections.

¹² See Martin Giese :Semantic Days 2011 TutorialTutorial, Presenting Relational Databases as RDF. June 2011, <http://sws.ifi.uio.no/project/semdays2011/lectures/lecture05.pdf>



4.5.1 DataMaster

Name: DataMaster	
Description	<p>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.</p> <p>Part of the rationale for developing DataMaster was that existing Protege plug-ins do not support OWL ontologies or schema-only imports.</p> <p>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.</p>
Prog. Language	Java
Data Sync	static
Data Exposition	ETL
hasGUI	Yes (Protege Plugin)
Mapping Language	proprietary
Can reuse Voc	yes
Req. Domain Ont	yes
Map. Automat.	manual
Current Version	1.3.2
Current Rel. Date	2009/12/17
Licence	MPL
Homepage	http://protegewiki.stanford.edu/wiki/DataMaster



CUBIST Comments:

DataMaster is a Protege Plugin. Protege is not a standard tool for CUBIST, and a federation of a database with DataMaster would need the manual start of Protege and the manual import of the RDB data to the ontology. This causes too much overhead and inhibits an integration of DataMaster in the CUBIST prototype.

Estimation: For this reason, DataMaster is not considered for CUBIST.



4.5.2 METAmorphoses

Name: METAmorphoses	
Description	<p>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.</p> <p>The METAmorphoses package contains the following tools:</p> <ol style="list-style-type: none">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 generated 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 transforms 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.
Prog. Language	Java
Data Sync	static
Data Exposition	ETL
hasGUI	Yes: standalone mapping editor with drag and drop capabilities
Mapping Language	proprietary xml
Can reuse Voc	yes
Req. Domain Ont	no
Map. Automat.	manual
Current Version	0.2.5
Current Rel. Date	<ul style="list-style-type: none">• MMPHP: 2008-01-21• METAmorphoses processor: 2007-02-07• RDF-Shout: 2007-02-07



	<ul style="list-style-type: none">• MM Mapping Editor: 2006-10-16
Licence	LGPL (The author writes: “Hence our interpretation of the LGPL is that the use of the unmodified METAmorphoses source or binary, or the rebundling of unmodified METAmorphoses classes into your program's .jar file, does not affect the license of your application code. If you modify METAmorphoses and redistribute your modifications, the LGPL applies.”)
Homepage	http://metamorphoses.sourceforge.net/

CUBIST Comments:

- This tool has only one author, who is not actively working on the tool anymore.
- Moreover (together with Maponto), this tool is not able to reuse existing vocabularies.

Estimation: For these reasons, METAmorphoses is not considered for CUBIST.



4.5.3 DartGrid

Name: 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.
Prog. Language	Java
Data Sync	dynamic
Data Exposition	Own query language
hasGUI	yes
Mapping Language	Visual tool
Can reuse Voc	yes
Req. Domain Ont	no
Map. Automat.	Manual
Current Version	?
Current Rel. Date	?
Licence	?
Homepage	http://ccnt.zju.edu.cn/projects/dartgrid

CUBIST Comments:

The homepage URL is taken from <http://www.w3.org/wiki/DartGrid>, but for the time of writing this CUBIST deliverable, the homepage was not accessible. We have not found recent documentation, even the current version and current release date is not retrievable.

Estimation: For this reason, DartGrid is not considered for CUBIST.



4.5.4 ODEMapster

Name: ODEMapster	
Description	<p>ODEMapster plugin provides users a Graphical User Interface that allows to create, execute, or query mappings between ontologies and databases. The mappings are expressed in R2O language, which is a mapping language between ontologies and databases featuring fully declarative, DBMS independent, and extensible set of primitive operations. This plugin works with OWL/RDF(S) ontologies and with MySQL or Oracle databases. Multiple R2O mappings per ontology can be created. ODEMapster is the processor in charge of carrying out the exploitation of the mappings defined using R2O, performing both massive and query driven data upgrade.</p> <p>Ontology engineers perform the ontology population, annotation or aggregation from unstructured information sources activities with various manual or (semi)automatic methods to transform unstructured, semistructured and/or structured data sources into ontology instances. Thus, ODEMapster plays the part where ontology population is performed from structured data sources (RDBMS).</p>
Prog. Language	Java
Data Sync	static
Data Exposition	ETL
hasGUI	Yes: plug in for the commercial NeON toolkit.
Mapping Language	Proprietary (R2O)
Can reuse Voc	yes
Req. Domain Ont	
Map. Automat.	manual
Current Version	2.2.7
Current Rel. Date	2010-07-02
Licence	LGPL
Homepage	http://neon-toolkit.org/wiki/ODEMapster



CUBIST Comments:

ODEMapster is a Plugin for the NeON toolkit. This toolkit is not a standard tool for CUBIST, and a federation of a database with ODEMapster would require the manual start of NeON and the manual import of the RDB data to the ontology. This causes too much overhead and inhibits an integration of ODEMapster in the CUBIST prototype.

Estimation: For this reason, ODEMapster is not considered for CUBIST.



4.5.5 Triplify

Name:	Triplify
Description	<p>Triplify is a small plugin for Web applications, which reveals the semantic structures encoded in relational databases by making database content available as RDF, JSON or Linked Data.</p> <p>Triplify is very lightweight: It consists of only a few files with less than 500 lines of code. For a typical Web application a configuration for Triplify can be created in less than one hour and if this Web application is deployed multiple times (as most open-source Web applications are), the configuration can be reused without modifications.</p> <p>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.</p>
Prog. Language	PHP, MySQL
Data Sync	dynamic
Data Exposition	LinkedData
hasGUI	no
Mapping Language	SQL
Can reuse Voc	yes
Req. Domain Ont	no
Map. Automat.	manual
Current Version	0.8
Current Rel. Date	2010-03-05
Licence	LGPL
Homepage	http://triplify.org



CUBIST Comments:

1. The LGPL-licence of Triplify is harmless for CUBIST, as the following statement from the authors of Triplify makes explicit: “Triplify is licensed under the terms of the GNU Lesser General Public License. You are free to copy, modify or redistribute it, even together with commercial software.” (see <http://triplify.org/About#h1-9>).
2. Though still being in its 0.8 version, Triplify is already a mature tool, and it is well-documented.
3. It originates from leading researchers in the field of Semantic Web Technologies and Linked Data, mainly Sören Auer (e.g. founder of DBpedia.org, see <http://www.informatik.uni-leipzig.de/~auer/>)
4. On <http://triplify.org/About> the following issues are mentioned:
 - Triplify is still beta grade software. There might be issues with exotic encodings, databases, PHP/Apache configurations. We would appreciate your feedback in order to solve these issues.
 - Performance: Triplify is currently aimed at small to medium Web applications (i.e. less than 100MB database content). However, Triplify supports the caching of the triplification results and can hence also be used with large Web applications.¹³
 - Privacy: Please be cautious not to reveal any sensitive information. Email addresses should be SHA1 hashed; password hashes and information, which is not meant to be publicly accessible, should be omitted in the Triplify SQL queries. As a rule of thumb, you should only make information available through Triplify, which is also publicly accessible on Web pages.

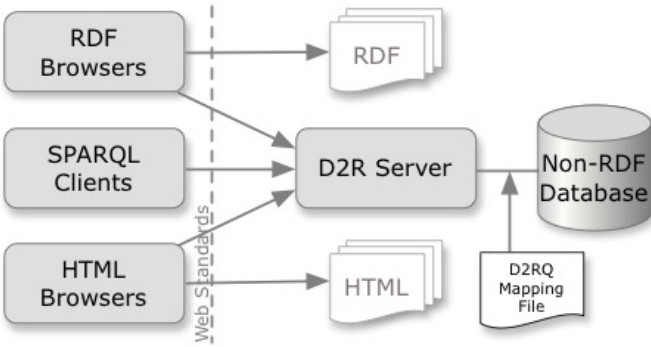
The current estimation is that these issues do not pose problems to CUBIST.

Estimation: For these reasons, Triplify is a candidate for CUBIST.

¹³ The W3C Survey of Current Approaches for mapping of relational databases to TDF (2009, see http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport.pdf) states that “despite its lightweight architecture Triplify is usable to publish very large datasets, such as 160 GB of geo data from the OpenStreetMap project.



4.5.6 D2R Server

Name:	D2R Server
Description	<p>D2R Server is a tool for publishing the content of relational databases on the Semantic Web, a global information space consisting of linked data.</p> <p>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 <i>browsed</i> and <i>searched</i> – the two main access paradigms to the Semantic Web.</p> <p>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.</p> <p>The SPARQL interface enables applications to search and query the database using the SPARQL query language over the SPARQL protocol.</p> <p>A traditional HTML interface offers access to the familiar Web browsers.</p>  <p>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.</p>
Prog. Language	Java
Data Sync	Bi-directional
Data Exposition	SPARQL
hasGUI	no



Mapping Language	D2R Map
Can reuse Voc	yes
Req. Domain Ont	no
Map. Automat.	manual
Current Version	v0.7, released 2009-08-10 (alpha)
Current Rel. Date	v0.7, released 2009-08-10
Licence	Apache License V2.0 ¹⁴
Homepage	http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/

CUBIST Comments:

- The Apache Licence v2.0 is not viral, thus the tool can be used in CUBIST.
- D2R, though still being in its 0.7 version, is already a mature and well-documented tool.
- Similarly to Triplify, it originates from leading researchers in the field of Semantic Web Technologies and Linked Data, mainly Chris Bizer.
- D2R is a very powerful tool for federating data from databases, e.g. it is one of very few tools which allow for a bi-directional synchronization.

Estimation: For these reasons, D2R Server is a candidate for CUBIST.

¹⁴ The package which can be downloaded from sourceforge still contains a file "licence.txt" referring to the (viral) GPL-licence. According to Richard Cyfniak, one of the developers, "Apache 2.0 is correct. Earlier versions were GPL licensed. The license file in the release is an oversight." Personal email-communication with Frithjof Dau, 2011/10/11.



4.5.7 MAPONTO

Name: MAPONTO	
Description	<p>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</p> <p>Inspired by the Clio project, we build the tool to work in an interactive and semi-automatic manner. Starting with a set of simple attribute-to-attribute correspondences, the tool analyzes semantics in the two input data models (e.g., a schema and an ontology, or two schemas), and then generates a set of logical formulas representing a semantic relationship between the two data models. The final list of logical formulas is ordered by the tool with the most "reasonable" mappings between the two models on top. Finally, the user could choose the expected mapping from the list.</p> <p>Our ultimate goal is to establish a semantic continuum between different schemas and ontologies. Database practices suggest that data semantics amounts to establish and maintain a semantic correspondence from one model/schema to another model/schema. Correspondences form a semantic continuum in which the semantics of each model/schema encapsulated in the mappings to other models. Once the continuum has been established, more complex issues arise, such as mapping composition and query reformulation. These issues are worth investigating in the future, but are beyond the scope of this project.</p>
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	No (plugin for Protégé)
Mapping Language	proprietary
Can reuse Voc	yes
Req. Domain Ont	yes
Map. Automat.	manual



Current Version	prealpha
Current Rel. Date	2011-01-20
Licence	Mozilla Public License 1.1 (MPL 1.1)
Homepage	http://www.cs.toronto.edu/semanticweb/maponto/

CUBIST Comments:

- MAPONTO is a Protege Plugin. Protege is not a standard tool for CUBIST, and a federation of a database with MAPONTO would need the manual start of Protege and the manual import of the RDB data to the ontology. This causes too much overhead and inhibits an integration of MAPONTO in the CUBIST prototype.
- Moreover (together with Metamorphoses), this tool is not able to reuse existing vocabularies.

Estimation: For these reasons, MAPONTO is not considered for CUBIST.



4.5.8 RDBToOnto

Name: RDBToOnto	
Description	<p>RDBToOnto is a tool that allows to automatically generate fine-tuned populated ontologies from relational databases (in RDFS/OWL).</p> <p>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).</p> <p>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 enhancing the input database before ontology generation.</p>
Prog. Language	Java
Data Sync	static
Data Exposition	ETL
hasGUI	<p>Yes. The GUI is separated in three parts and the user has to configure all three parts from top to bottom.</p> <ul style="list-style-type: none">• Input sources are selected and configured,• the methods for learning the ontology from the source,• local constraints can be created with specifying rules
Mapping Language	none
Can reuse Voc	no
Req. Domain Ont	No
Map. Automat.	automatic, fine tunable
Current Version	V1.2, beta
Current Rel. Date	2008
Licence	proprietary (closed)
Homepage	http://www.tao-project.eu/researchanddevelopment/demosanddownloads/RDBToOnto.html



CUBIST Comments:

- Though not maintained anymore and officially in a Beta-state, RDBToOnto can be considered a mature tool.
- Similar to RDOE, this is a standalone-tool which can be used for importing a bulk of data from databases into the CUBIST semantic warehouse, thus the licence is not negatively effecting the CUBIST prototype.
- The capability to create highly “structured ontologies” (e.g. including hierarchies of concepts) might render RDBToOnto as a good tool for creating an (initial) ontology for the use cases in CUBIST.

Estimation: For these reasons, RDBToOnto is considered a candidate for CUBIST.



4.5.9 Virtuoso RDF Views

Name: Virtuoso RDF Views	
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.
Prog. Language	C
Data Sync	dynamic
Data Exposition	SPARQL
hasGUI	Yes. Virtuoso Conductor Wizard for creating RDF Views, step by step or one click generation and deployment as Linked Data
Mapping Language	Meta Schema Language
Can reuse Voc	yes
Req. Domain Ont	no
Map. Automat.	Semi-automatic
Current Version	6.2
Current Rel. Date	2010-09
Licence	Commercial or GPL
Homepage	http://docs.openlinksw.com/virtuoso

CUBIST Comments:

This tool is certainly mature and well maintained. But the licence is either commercial or GPL (“the original GNU General Public License Version 2, dated June 1991”, and as it is stated on <http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSLicense>: “If your application directly uses Virtuoso's sources or libraries (e.g. libvirtuoso), then the GPL forces your application to be released under the GPL too”).

Estimation: For this reason, Virtuoso RDF Views is not considered for CUBIST.



4.5.10 Relational.OWL

Name:	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'.
Prog. Language	Java
Data Sync	static
Data Exposition	ETL
hasGUI	no
Mapping Language	none
Can reuse Voc	no
Req. Domain Ont	no
Map. Automat.	automatic
Current Version	0.1 Alpha (discontinued)
Current Rel. Date	2009-07-17
Licence	GPL
Homepage	http://sourceforge.net/projects/relational-owl/

CUBIST Comments:

- 1) The licence of Relational.OWL is the viral GPL licence.
- 2) A direct one-to-one translation of a database schema to an ontology is too simple and not suited for CUBIST.
- 3) The tool is in a very early stage and discontinued.

Estimation: For these reasons, Relational.OWL is not considered for CUBIST.



4.5.11 VisAVis

Name: VisAVis	
Description	VisAVis or VisAVisTab is a Protégé 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 RDOE
Prog. Language	Java
Data Sync	dynamic
Data Exposition	RDQL
hasGUI	Yes (Protege Plugin)
Mapping Language	SQL
Can reuse Voc	yes
Req. Domain Ont	yes
Map. Automat.	manual
Current Version	?
Current Rel. Date	?
Licence	unknown
Homepage	?

CUBIST Comments:

The homepage mentioned in the KET Survey is not active anymore; the tool is not listed on the plugin-site of Protégé. The tool seems to be not retrievable anymore and is discontinued.

Estimation: For this reason, VisaVis is not considered for CUBIST.



4.5.12 RDO TE

Name:	RDO TE
Description	RDO TE 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 RDB data.
Prog. Language	Java
Data Sync	Static
Data Exposition	ETL
hasGUI	Yes
Mapping Language	SQL
Can reuse Voc	Yes
Req. Domain Ont	Yes
Map. Automat.	Manual
Current Version	V2.0
Current Rel. Date	2011-09-22
Licence	GPL
Homepage	http://sourceforge.net/projects/rdote/

CUBIST Comments:

- Though the licence of RDO TE is the viral GPL licence, a plain application of RDO TE as a stand-alone tool for importing data from databases into the CUBIST semantic warehouse is not infecting the CUBIST prototype.
- The GUI of RDO TE is appealing (see <http://www.youtube.com/watch?v=pk7izhFeuf0>)
- RDO TE is mature and still maintained (see current release date).

Estimation: For these reasons, RDO TE is considered as a candidate for CUBIST.



4.5.13 DB2Triples

Name: DB2Triples	
Description	RDB2RDF Antidot implementation
Prog. Language	Java
Data Sync	Dynamic
Data Exposition	SPARQL
hasGUI	No
Mapping Language	<ul style="list-style-type: none">- R2RML: RDB to RDF Mapping Language (http://www.w3.org/TR/2011/WD-r2rml-20110324/)- A Direct Mapping of Relational Data to RDF (http://www.w3.org/TR/2011/WD-rdb-direct-mapping-20110324/)
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	0.9
Current Rel. Date	2011
Licence	LGPL
Homepage	https://github.com/antidot/db2triples

CUBIST Comments:

This is one of the few open source R2RML implementation available at the moment, being actively developed.

Estimation: Due to the expected importance of R2RML, this is a prospective CUBIST candidate.



4.5.14 Revelytix's Spyder

Name:	Revelytix's Spyder
Description	Spyder is a free tool for converting relational data stores to RDF. It implements R2RML, an emerging standard under development by the W3C R2RML Working Group. Spyder creates a SPARQL 1.1 endpoint for any relational database, providing the ability to merge data from any relational store, regardless of its location.
Prog. Language	Java
Data Sync	Dynamic
Data Exposition	SPARQL
hasGUI	No
Mapping Language	R2RML
Can reuse Voc	Yes
Req. Domain Ont	No
Map. Automat.	Manual
Current Version	0.9.1
Current Rel. Date	29/09/2011
Licence	proprietary license (free evaluation copies available)
Homepage	http://www.revelytix.com/content/spyder

CUBIST Comments:

The free evaluation copies are not restricted for usage for certain time period. On the other hand, the distribution package contains a good quality documentation and comprehensive examples which is a good indication and attracts the attention of the CUBIST project.

Estimation: Revelytix's Spyder is considered a candidate for CUBIST.



5 Summary

Amongst the CSV and Spreadsheet Converters, we identified the following candidates:

- CSV2RDF4LOD
- Convert2RDF
- Google Refine's RDF Extension
- XLWrap
- RDF extensions for Talend Open Studio
- CUBIST Simple TSV2TTL Converter

Amongst the XML Converters, no candidate was identified. Thus further investigations have to be conducted, or our own XML Converter has to be developed.

Amongst the RDB Converters, we identified the following candidates:

- Triplify
- D2R Server
- RDBToOnto
- RDOE
- DB2Triples (prospective)
- Revelytix's Spyder



6 References

- [D2R MAP] Christian Bizer: D2R MAP - A Database to RDF Mapping Language. WWW Posters, (2003)
- [KETSurvey] LOD2 Knowledge Extraction Tools Survey. See <http://data.lod2.eu/2011/tools/> (note the extra / at the end) or <http://tinyurl.com/KETSurvey>
- [LOD2] D3.1.1 – State-of-the-Art Report for Extraction from Structured Sources. LOD2 Deliverable. (<http://lod2.eu/Deliverable/D3.1.1.html> and <http://static.lod2.eu/Deliverables/deliverable-3.1.1.pdf>).
- [RDB2RDF] RDB2RDF Working Group website, <http://www.w3.org/2001/sw/rdb2rdf/>
- [RDB-direct] M. Arenas, A. Bertails, E. Prud'hommeaux and J. Sequeda. A Direct Mapping of Relational Data to RDF. W3C Working Draft (20 Sep 2011)
- [RDF Views] OpenLink Software. Virtuoso RDF Views – Getting Started Guide. Whitepaper. http://www.openlinksw.co.uk/virtuoso/Whitepapers/pdf/Virtuoso_SQL_to_RDF_Mapping.pdf (retrieved October 2011)
- [R2RML] S. Das, S. Sundara and R. Cyganiak. R2RML: RDB to RDF Mapping Language. W3C Working Draft (20 Sep 2011)
- [Triplify] Sören Auer, Sebastian Dietzold, Jens Lehmann, Sebastian Hellmann, David Aumueller: Triplify – Light-Weight Linked Data Publication from Relational Databases. See <http://dbs.uni-leipzig.de/file/triplify.pdf>
- [Turtle] Turtle - Terse RDF Triple Language, D. Beckett and T. Berners-Lee. W3C, 14 January 2008.