Fine-tuning triplification with Semion

Andrea Giovanni Nuzzolese[1], Aldo Gangemi[1],
Valentina Presutti[1], Paolo Ciancarini[2]

[1] Semantic Technology Lab, ISTC-CNR, Rome. Italy
[2] Dep. of Computer Science, Università di Bologna. Italy

Abstract The Web of Data is fed mainly by "triplifiers (or RDFizers)", tools able to transform content (usually from databases) to linked data. Current triplifiers implement diverse methods, and are usually based on bulk recipes, which make fixed assumptions on the domain semantics. They focus more on syntactic than on semantic transformation, and allow for limited (sometimes no) customization of the process. We present Semion, a method and a tool for triplifying content sources that overcomes such limitations. It focuses on applying good practices of design, provides high customizability of the transformation process, and exploits OWL expressivity for describing the domain of interest.

1 Introduction

In the traditional hypertext Web, which can be identified by the expression "Web of documents", the nature of the relationships between two linked documents is implicit: the usual encoding language used i.e. HTML, is not sufficiently expressive to enable individual entities, described in a particular document, to be connected by typed links to other related entities [10].
In recent years, the Web has evolved from a global information space of linked documents to one where both documents and data are linked. Underpinning this evolution is a set of best practices for publishing and connecting structured data on the Web known as Linked Data [8]. The aim of Linked Data is to bootstrap the Web of Data by identifying existing data sets that are available under open licenses, converting them to RDF according to [8], and publishing them on the Web. There are commonly accepted solutions for transforming non-RDF data sources into RDF datasets. They rely on predetermined implicit assumptions on the domain semantics of the non-RDF data source. For example, relational databases are associated with ontologies where each table is a `rdfs:Class`, each table record is an `owl:Individual` of that class, and each table column is a `rdf:Property`, regardless the intensional semantics of the database tables, records, and columns as it was conceived in the original conceptual model of the database. Such implicit assumptions imply:

- limited customization of the transformation process (e.g. a user cannot map a table to a property)
- difficulty in adopting good practices of knowledge reengineering and ontology design (e.g. ontology design patterns [19])

– limited exploitation of OWL [14] expressivity for describing the domain, so that the services of OWL inference engines result to be sometimes limited.

The tool described here, Semion, implements a method that overcomes the above issues. The Semion method allows to reengineer any data source to RDF triples, without fixing assumptions on the domain semantics, which can be customized by the user. It is based on three main steps: (i) a **syntactic transformation** of the data source to RDF datasets according to an OWL ontology that represents the data source structure i.e. the source meta-model. For example, the OWL ontology for a relational database would include the classes “table”, “column”, and “row”. The ontology can be either provided by the user, or reused from a repository of existing ones. The transformation is therefore independent from any assumption about the domain semantics. (ii) A **first refactoring** step that allows to transform the obtained RDF dataset according to a so called “mediator”. A mediator is any ontology that represents the **informal semantics** that we use for organizing our knowledge, i.e. the semantics of human semiotic processes. The value of a semiotic representation is its ability to support the representation of different knowledge sources developed according to different underlying semiotic perspectives. In [11] some examples are provided of knowledge representation schemata, either formal or informal, which can be aligned to semiotic concepts and relations. However, there can be many ways of expressing such informal semantics. A popular example of a mediator is SKOS [16], which addresses the organization of knowledge by means of narrower/broader “concepts”. This step is analogous to a reverse engineering action performed in order to get the knowledge out of the constraints of a specific data structure. In this paper, we focus on the use of the Linguistic Meta-Model (LMM) [11] as mediator ontology, an OWL-DL ontology that formalizes some semiotic relations. (iii) A **second refactoring** step that maps the RDF model expressed in terms of the mediator, to a formal language, e.g. OWL, also enabling custom semantic choices (e.g. relation as a logical class or relation). (iv) A **third refactoring** step that allows to express the resulting OWL model according to specific domain ontologies e.g. DOLCE, FOAF, the Gene Ontology, indicated by the user. This last action results in a RDF dataset, which expresses the knowledge stored in the original data source, according to a set of assumptions on the domain semantics selected and customized by the user.
The contribution of this paper is threefold: the Semion method, a tool that implements such method, and the tool evaluation. The evaluation has been performed by comparing Semion to existing tools that address similar requirements, and by applying it to a use cases i.e. triplification of WordNet database.
The paper is organized as follows: Section 2 discusses related work. Section 3 describes the Semion method, while Section 4 contains details regarding the Semion tool and its application to two use cases. Section 5 describes the results of a feature-based comparison of Semion with other related tools. Finally section 6 discusses conclusion and future work.

2 Related work

The wide spreading of Linked Data led to the development of several methods and tools for transforming non-RDF (legacy) data sources to linked data, and publish them on the Web of data. In this section we briefly describe the most popular and used ones.

D2R Server [9] is a tool for publishing relational databases on the Semantic Web. It enables RDF and HTML browsers to navigate the content of a database, and allows other applications to query a database through SPARQL. D2R Server uses the D2RQ Mapping Language to map the content of a relational database to RDF. A D2RQ mapping rule specifies how to assign URIs to resources, and which properties are used to describe them. The **Talis Platform** [3] provides Linked Data-compliant hosting for content, and its associated RDF data. Data held in the platform are organized into "stores" that can be individually secured if needed. The content and metadata become immediately accessible over the Web and discoverable using both SPARQL [20], and a keyword-based search engine. **Triplify** [7] is a PHP plug-in for Web applications. It implements a black-box recipe for making database content available on the Web as RDF, JSON or Linked Data. **Virtuoso** [1] combines functionalities of traditional RDBMS, ORDBMS, virtual database, RDF, XML, free-text, Web application server, and file server in a single system. It enables a single multithreaded server process that implements multiple protocols. **QuOnto** [6] is a Java-based tool for ontology representation and reasoning. It implements the DL-Lite family of ontology representation languages, and uses relational DBMSs to store the extensional level of the ontologies. It relies on the Ontology Based Data Access (OBDA), which provides access to heterogeneous data sources through an intermediate ontology. **METAmorphoses** [21] 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. **Krextor** [15] is an extensible XSLT-based framework for extracting RDF from XML, supporting multiple input languages as well as multiple output RDF notations. A relevant project related to the topic of reengineering legacy data sources to RDF is the **RDFizer** [2] project, an on-line directory that collects accepted tools for converting data from various formats, i.e. BibTEX, Java, Javadoc, etc., to RDF.

Although such existing tools served successfully the requirement of bootstrapping the Web of Data, they share two limitations: *adaptability* to heterogeneous data source structures, and *customizability* of the transformation process.

Semion implements a transformation method aiming at triplifying heterogeneous content sources without such limitations. It aims at obtaining high-quality (in the sense of task-based, relevant, and semantically well-founded) data and ontologies, through a transformation process with explicit, customizable, and incremental steps.

3 Semion method

Figure 1 depicts the two key processes composing the Semion method. The first process i.e. the reengineering process, performs a syntactic transformation of the data source to RDF without making any choice with respect to neither the formal semantics to be used for expressing the domain knowledge (encoded by the datasource), nor the formal language to be used for encoding the (final) resulting dataset. RDF at this stage is only used as a serialization format. The second process i.e. the refactoring process, performs semantic-based transformations of the RDF dataset. During this process it is possible to choose a specific formal semantics e.g. model theory, and a specific formal language e.g. OWL2-EL, to be used for modeling the dataset (and associated ontology), which will be the result of the whole Semion triplification procedure. Finally, the refactoring process can be iterated once more in order to align the resulting ontology to existing ones that the user might want to use for his/her specific application.

3.1 The reengineering process

The reengineering process aims at producing a RDF dataset starting from a content source encoded in any possible format[1]. The goal of this process is to express, through RDF triples, the same (or a selection of) data that are stored in the data source. The approach followed by this process is mainly syntactic. In order to achieve such goal Semion requires, as input, a RDF vocabulary describing the data source meta-model e.g. for a relational database, a vocabulary containing concepts like table, field, query, result set, etc[2]. Additionally, it requires, as input, a set of rules mapping the vocabulary entities to the data source entities e.g. DB tables map to `rdfs:Resource` with `rdf:type mydb:Table`. The result of the reengineering process is a RDF dataset including both the data and the schema of the data source, encoded according to the adopted database vocabulary[3]. The domain semantics of the dataset is at this point implicit.
Depending on the amount of data to be reengineered, this process can be performed with incremental iterations. At the moment, once the RDF dataset is produced, it is independent on the original data source, i.e. any change applied to the original source after the reengineering step is not reflected (automatically) into the RDF dataset.

[1] Although Semion's current implementation supports relational database and XML sources, the theoretical method here described is designed in order to be applicable to any possible data source.

[2] An example of a relational database vocabulary used by Semion implementation can be downloaded at http://ontologydesignpatterns.org/ont/iks/dbs_l1.owl

[3] http://stlab.istc.cnr.it/software/semion/tool/samples/customerProductSchema.rdf and http://stlab.istc.cnr.it/software/semion/tool/samples/customerProductData.rdf are RDF datasets resulting from the reengineering of a sample database for storing and managing data about customers and products. Respectively, they express the schema and the data of the sample database according to the Semion default database vocabulary available at http://ontologydesignpatterns.org/ont/iks/dbs_l1.owl

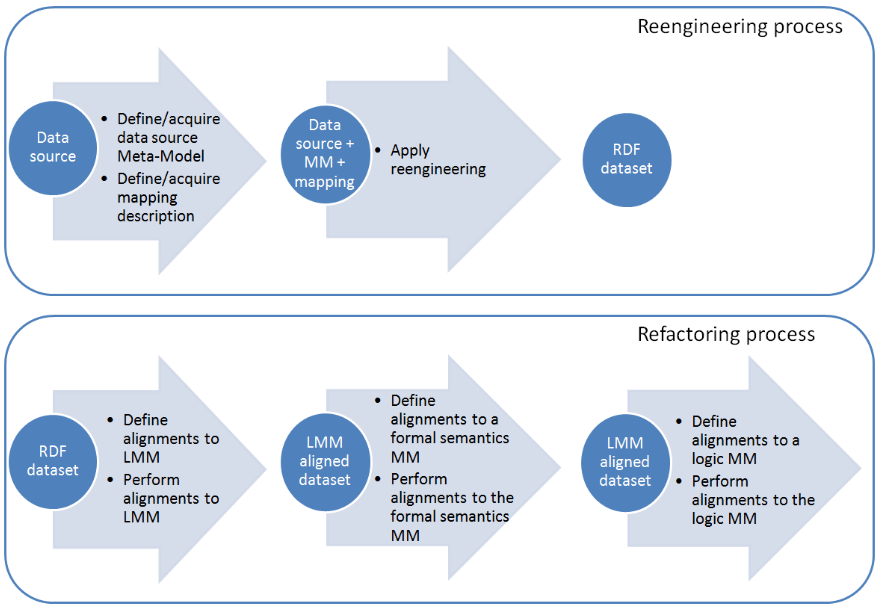

Reengineering process
Data source
Define/acquire data source Meta-Model
Define/acquire mapping description
Data source + MM + mapping
Apply reengineering
RDF dataset
Refactoring process
RDF dataset
Define alignments to LMM
Perform alignments to LMM
LMM aligned dataset
Define alignments to a formal semantics MM
Perform alignments to the formal semantics MM
LMM aligned dataset
Define alignments to a logic MM
Perform alignments to the logic MM

Figure 1. Tranforming method: key concepts.

3.2 The refactoring process

The final goal of the whole Semion triplification method is to obtain a RDF dataset modeled according to a certain formal semantics, encoded in a specific logic language, with its knowledge expressed according to axioms defined in a domain ontology. The reengineering process, explained above, produces a dataset containing the data of the original data source encoded in RDF format. The refactoring process allows us to further transform such dataset, by introducing formal and domain semantics through a number of steps in which formal and domain design choices are made explicit. The first step of the refactoring process, as depicted in Figure 1, consists of defining a set of rules that map the available RDF dataset to a so called “mediator” ontology such as the Linguistic-Meta Model (LMM) [18]. LMM is an OWL ontology describing the entities of the informal semantics that we use for organizing our knowledge in a semiotic-cognitive way. The core of LMM represents the main concepts of semiotics according to [17], namely:

– the **lmm:Expression**[4] class includes social objects produced by agents in the context of communication acts. They are natural language terms, symbols in formal languages, icons, and whatever can be used as a vehicle for

[4] lmm: is the prefix for http://www.ontologydesignpatterns.org/ont/lmm/LMM_L2.owl

communication. Expressions have a content (or meaning) and possibly a reference. For example, the word *dog* can have the meaning of a collection of animals in the utterance: *dogs are usually friendly to children*, and can refer to a specific dog in the utterance: *Viggo dog has played with my daughter this morning.* In this case, an additional semiotic relation holds, i.e. that Viggo dog is *interpreted by* the meaning *dog*;
- the **lmm:Meaning** class includes any entity that is supposed to be the content of an expression in a communication act, e.g. other expressions that explain an expression as in dictionaries, the cognitive processes corresponding to the use of an expression in context, the concepts in a classification scheme, are all examples of meanings;
- the **lmm:Reference** class includes any possible individual that is referred to by an expression, under a certain meaning;
- the **lmm:LinguisticAct** class includes the actual communication events, in which some agents use expressions to either convey meanings or refer to entities.

Now consider a relational database as an example data source. In this case, the refactoring to LMM could be performed according to the following mapping assertions:

Table(?x) → `lmm:Meaning`(?x)
Record(?x) → `lmm:Reference`(?x)

The previous assertions state that, starting from the database meta-model, and in the scope of the refactoring rules applied to a database *db1*,

- a table t_1 is the meaning of the table structure and vocabulary (that are expressions) defined for that table in *db1*
- a record r_1 in t_1 is the reference of the record structure and vocabulary (that are expressions) defined for that record within the table t_1. r_1 is the reference of the same expression that has t_1 as meaning, so that r_1 is *interpreted by* t_1.

Although Semion uses LMM as built-in mediator ontology, it allows the user to customize such choice. For example, the Simple Knowledge Organization System (SKOS) [16] can be used as a mediator. SKOS is a common data model for knowledge organization systems such as thesauri, classification schemes, subject heading systems and taxonomies. SKOS describes the typical informal (semiotic) semantics used by communities of practices familiar with classification schemes, for organizing their knowledge.
This refactoring step can be seen as a reverse engineering action that gets the knowledge out of the constraints of a specific data structure: the obtained RDF dataset is expressed in terms of semiotic entities.
The next refactoring step consists in aligning such dataset to a formal semantics i.e. semiotic entities are mapped to formal entities complying to a specific reference formal semantics e.g. model theory. In the previous example,

`lmm:Meaning` can be mapped to e.g. `forsem:Class`[5] or `forsem:Relation`, while `lmm:Reference` can be mapped to e.g. `forsem:SetElement` or (one or more) `forsem:Proposition`. The next iteration allows to choose a specific logic language, which complies to the specified formal semantics e.g. OWL (e.g. `owl:Class`, `owl:ObjectProperty`, `owl:NamedIndividual`, `owl:PropertyAssertion`, etc.). The user might want to merge the last two steps into one if the need is only to choose a logical language without making it explicit the formal semantics behind it. Nevertheless, Semion method allows a higher degree of customization in order to express the same formal semantics into different logical languages.

4 Semion tool

The method described in the previous section is implemented in a tool called Semion[6]. Semion is available both as reusable components organized in two main Java libraries (called SemionCore), and as a standalone graphical tool based on the Standard Widget Toolkit(SWT) [5] and JFace [4]. The tool has been designed according to the Model-View-Controller pattern, in which view and controller are part of the user interface, while the models are provided by the SemionCore libraries. Currently the tool provides support, and has been tested for transforming relational databases to RDF, however it has been designed in order to be easily extensible for supporting transformations of any kind of data source to RDF. Figure 2 shows the reengineering perspective of the Semion tool, according to the method terminology. In this perspective, the user is supported to transform a database to a RDF dataset according to a vocabulary describing the database meta-model; Semion provides a built-in vocabulary for such reengineering process[7]. The user can choose whether to reengineer the whole database by generating a single dump of RDF triples, or to reengineer only a selection of database entities e.g. a subset of tables and their records. Additionally, the interface provides a SPARQL [20] view, which allows to query the resulting RDF dataset.
The refactoring perspective supports the user in performing the refactoring process. Semion performs refactoring transformations according to a set of user-defined rules. Such rules can be expressed in terms of a simple syntax of the form:

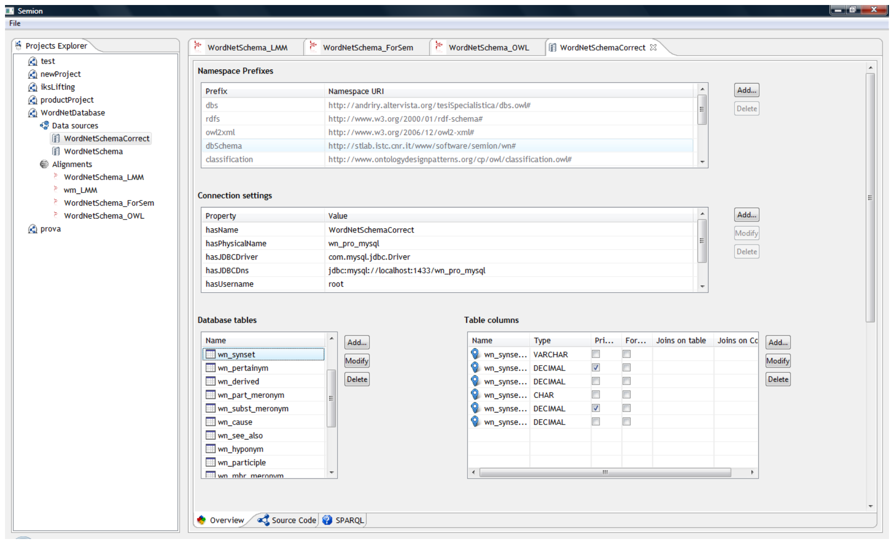
$$antecedent \rightarrow consequent$$

Technically, such rules are interpreted and executed as SPARQL CONSTRUCT queries. For example, the following rule cause the transformation of resources of type *dbs:Table* to instances of the class *dul:Concept* of DOLCE Ultra Light [12]:

$$dbs : Table(?x) \rightarrow DUL : Concept(?x)$$

This rule is interpreted and executed as the SPARQL query:

[5] forsem: is the prefix for http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl
[6] http://stlab.istc.cnr.it/software/semion/tool
[7] http://ontologydesignpatterns.org/ont/iks/dbs_l1.owl



Namespace Prefixes

Prefix	Namespace URI
dbs	http://andriy.altervista.org/tesiSpecialistica/dbs.owl#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl2xml	http://www.w3.org/2006/12/owl2-xml#
dbSchema	http://stlab.istc.cnr.it/www/software/semion/wn#
classification	http://www.ontologydesignpatterns.org/cp/owl/classification.owl#

Connection settings

Property	Value
hasName	WordNetSchemaCorrect
hasPhysicalName	wn_pro_mysql
hasJDBCDriver	com.mysql.jdbc.Driver
hasJDBCDns	jdbc:mysql://localhost:1433/wn_pro_mysql
hasUsername	root

Database tables

Name
wn_synset
wn_pertainym
wn_derived
wn_part_meronym
wn_subst_meronym
wn_cause
wn_see_also
wn_hyponym
wn_participle
wn_mbr_meronym

Table columns

Name	Type	Pri...	For...	Joins on table	Joins on Co
wn_synse...	VARCHAR	☐	☐		
wn_synse...	DECIMAL	✓	☐		
wn_synse...	DECIMAL	☐	☐		
wn_synse...	CHAR	☐	☐		
wn_synse...	DECIMAL	✓	☐		
wn_synse...	DECIMAL	☐	☐		

Figure 2. Semion tool: view of the reengineering interface.

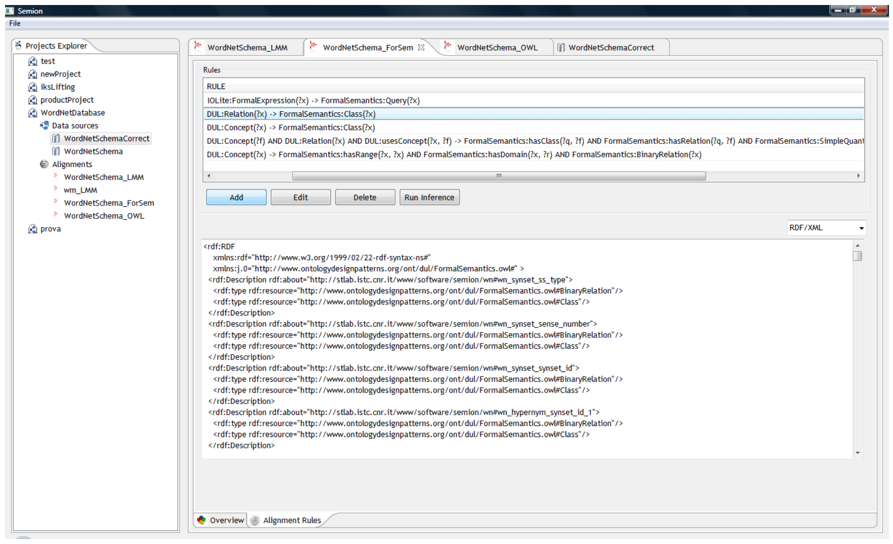
```
CONSTRUCT { ?x rdf:type DUL:Concept. }
WHERE { ?x rdf:type dbs:Table. }
```

Semion tool has been applied for triplifying the WordNet database [8]. According to the Semion method, WordNet has been first transformed to RDF triples according to a defined vocabulary for RDB. The resulting dataset has been then transformed to a new one based on LMM by defining specific refactoring rules. Next, a refactoring step for fixing the formal semantics has been performed. Figures 3 and 4 show two screenshots of the tool interface during refactoring steps performed for the WordNet use case. Figure 3 deals with mapping WordNet LMM-based dataset to a vocabulary expressing a formal semantics, Figure 4 shows how the resulting dataset is mapped to the OWL vocabulary for obtaining an OWL ontology (including its individuals) for WordNet. Finally Figure 4 shows the screenshot of the tool while performing the last refactoring step, which transforms the dataset according to the OWL vocabulary.

5 Evaluation

Semion aims at maximizing flexibility of the transformation process: it wants to support both a user who wants to customize reengineering and refactoring of data sources, and a user who wants to reuse "good practices" such as recipes to convert databases, thesauri, etc., and does not want to know anything about the reengineering and refactoring clockwork. In addition to the WordNet use

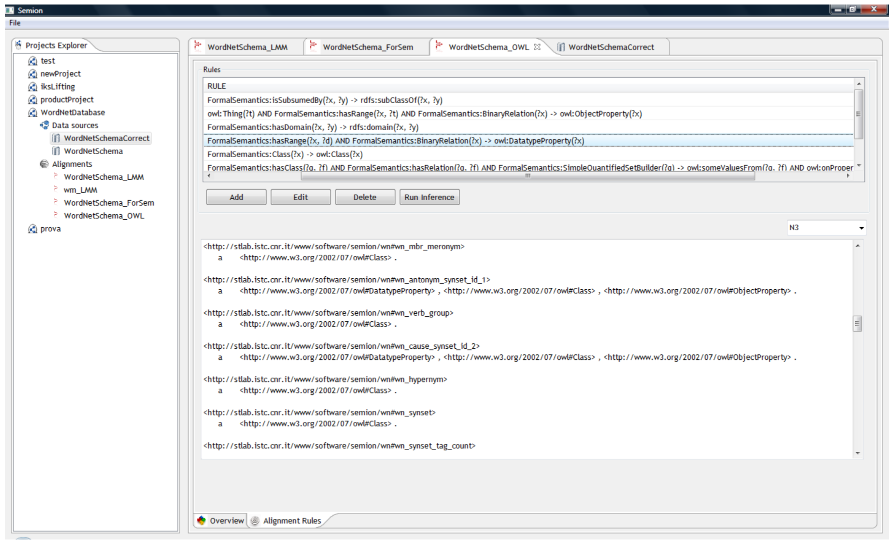
[8] MySQL version of the WordNet database.



```
RULE
IOLite:FormalExpression(?x) -> FormalSemantics:Query(?x)
DUL:Relation(?x) -> FormalSemantics:Class(?x)
DUL:Concept(?x) -> FormalSemantics:Class(?x)
DUL:Concept(?f) AND DUL:Relation(?x) AND DUL:usesConcept(?x, ?f) -> FormalSemantics:hasClass(?q, ?f) AND FormalSemantics:hasRelation(?q, ?f) AND FormalSemantics:SimpleQuant
DUL:Concept(?x) -> FormalSemantics:hasRange(?x, ?x) AND FormalSemantics:hasDomain(?x, ?r) AND FormalSemantics:BinaryRelation(?x)
```

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:j.0="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#" >
 <rdf:Description rdf:about="http://stlab.istc.cnr.it/www/software/semion/wn#wn_synset_ss_type">
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#BinaryRelation"/>
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#Class"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://stlab.istc.cnr.it/www/software/semion/wn#wn_synset_sense_number">
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#BinaryRelation"/>
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#Class"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://stlab.istc.cnr.it/www/software/semion/wn#wn_synset_synset_id">
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#BinaryRelation"/>
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#Class"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://stlab.istc.cnr.it/www/software/semion/wn#wn_hypernym_synset_id_1">
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#BinaryRelation"/>
  <rdf:type rdf:resource="http://www.ontologydesignpatterns.org/ont/dul/FormalSemantics.owl#Class"/>
 </rdf:Description>
```

Figure 3. Alignment to the FormalSemantics vocabulary.



```
RULE
FormalSemantics:isSubsumedBy(?x, ?y) -> rdfs:subClassOf(?x, ?y)
owl:Thing(?t) AND FormalSemantics:hasRange(?x, ?t) AND FormalSemantics:BinaryRelation(?x) -> owl:ObjectProperty(?x)
FormalSemantics:hasDomain(?x, ?y) -> rdfs:domain(?x, ?y)
FormalSemantics:hasRange(?x, ?d) AND FormalSemantics:BinaryRelation(?x) -> owl:DatatypeProperty(?x)
FormalSemantics:Class(?x) -> owl:Class(?x)
FormalSemantics:hasClass(?q, ?f) AND FormalSemantics:hasRelation(?q, ?f) AND FormalSemantics:SimpleQuantifiedSetBuilder(?q) -> owl:someValuesFrom(?q, ?f) AND owl:onProper
```

```
<http://stlab.istc.cnr.it/www/software/semion/wn#wn_mbr_meronym>
    a     <http://www.w3.org/2002/07/owl#Class> .

<http://stlab.istc.cnr.it/www/software/semion/wn#wn_antonym_synset_id_1>
    a     <http://www.w3.org/2002/07/owl#DatatypeProperty> , <http://www.w3.org/2002/07/owl#Class> , <http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://stlab.istc.cnr.it/www/software/semion/wn#wn_verb_group>
    a     <http://www.w3.org/2002/07/owl#Class> .

<http://stlab.istc.cnr.it/www/software/semion/wn#wn_cause_synset_id_2>
    a     <http://www.w3.org/2002/07/owl#DatatypeProperty> , <http://www.w3.org/2002/07/owl#Class> , <http://www.w3.org/2002/07/owl#ObjectProperty> .

<http://stlab.istc.cnr.it/www/software/semion/wn#wn_hypernym>
    a     <http://www.w3.org/2002/07/owl#Class> .

<http://stlab.istc.cnr.it/www/software/semion/wn#wn_synset>
    a     <http://www.w3.org/2002/07/owl#Class> .

<http://stlab.istc.cnr.it/www/software/semion/wn#wn_synset_tag_count>
```

Figure 4. Alignment to the OWL vocabulary.

case presented in the previous section, we have performed a comparison of the Semion tool to other related tools according to a defined set of functionalities. Such functionalities are not meant to constitute an evaluation framework, they have been selected for emphasizing the characteristics that distinguish triplification methods, based on the issues that we have addressed in this paper i.e.

customization of the transformation process, adoption of good practices for data reengineering and domain modeling, exploitation of OWL expressivity. In the future, Semion will be subject of a more rigorous evaluation. The tools involved in such comparison are D2R [9], Triplify [7] and METAMorphoses [21]. The other tools mentioned in Section 2 have not been included because of the lack of availability of details about their design. We are also missing many other tools that provide support for reengineering data sources to ontologies, e.g. from XML databases, for HTML scraping, etc. This is because we could evaluate only the current implementation of Semion, which supports transformation of only relational database, hence we leave this aspect to future work.
D2R transforms relational databases in a different way with respect to Semion, since it allows to access the source as it was an RDF graph translating, through a mapping file, SPARQL queries into SQL queries. The mapping file can be configured, but the transforming choices are made implicit in the "bridges" that it realizes. Triplify reveals the semantic structures encoded in relational databases, but the transformation and the domain semantics is not configurable as it is in Semion. METAMorphoses has some similarity to Semion, since it allows to configure the mapping from the data based on a metamodel, and to map the result to another ontology. However, it is not clear if the mapping can be made by means of any (even customized) metamodel for the data source and if the second mapping can be applied iteratively to other ontologies.
Table 1 shows the functionalities selected for comparing Semion performances with the other tools'. The first two functionalities deal with the core business of triplifiers i.e. transforming legacy content to RDF datasets. Although the approaches are different, all tools are able to tranform non-RDF data sources to RDF. Nevertheless, it must be noticed that while Semion is transforms also the schema (and produces an ontology) of the source, the other tools do not. Alternatively, they keep a reference to the original schema e.g. a mapping, in order to access and extract data.
The transformation process is highly customizable in Semion, while is fixed in Triplify. D2R allows some degree of customization when defining the mapping rules, and METAMorphoses allows only customization of the domain ontology the transformation has to comply with.
Extensibility to support other source type is a Semion feature, it is implemented by providing the source metamodel ontology. The other tools, at the moment, do not consider support for other types of legacy sources but relational databases.
Triplification of the original data structure is a Semion' specific feature. It refers to the result of the reengineering process, which do not impose any semantic assumptions at domain level when triplifying the datasource i.e. using a metamodel approach such as SKOS' one, for any possible data source. The other tools include always some implicit semantic choices at domain level during the transformation e.g. the bridge approach described in Section 3.
Another Semion' specific feature is the support for incremental, iterative mapping of the dataset to custom ontologies. The definition and reuse of existing practices for reengineering i.e. transformation recipes, is supported by both

Semion and METAMorphoses. D2R partially supports this functionality because it is possible to specify references to a target ontology in the mapping definition. Furthermore Semion is compliant with the good reengineering practices as the Ontology Design Patterns (ODP) [13].
Finally, there are two Semion's special features. Semion is integrated with an inference engine i.e. it provides OWL reasoning support, and its implementation relies completely on semantic web standards and technologies. In other words Semion is itself a semantic web-based tool, everything that could be done by using standard technologies was implemented with such approach e.g. SPARQL, SWRL, etc., in order to minimize the amount of developed ad-hoc code for reasoning purposes.

Table 1. Functionalities implemented by Semion and comparison to other tools

Functionality	**Semion**	**D2R**	**Triplify**	**METAMorphoses**
Transform non-RDF data to RDF	yes	yes	yes	yes
Transform non-RDF data source schemata to RDF	yes	partly	partly	partly
Customization of the transformation process	yes	partly	no	partly
Extensibility to support other source types	yes	no	no	no
Triplification of original data structure	yes	no	no	no
Incremental, iterative mapping to custom ontologies	yes	no	no	no
Support for translation recipes definition and reuse	yes	partly	no	yes
OWL(2) and reasoning support	yes	no	no	no
Based on semantic web standards and technologies	yes	no	no	no

6 Conclusion and future work

We have presented Semion, a method and a smart triplification tool that is designed for transforming any non-RDF sources to RDF datasets. At the moment Semion supports only transformation of relational databases. The Semion method is divided into two processes. The first one allows a syntactic transformation of the datasource to RDF without making assumptions on the domain semantics. The second is an iterative process, and allows for a highly customizable transformation of the dataset based on formal and domain semantics. We have also applied Semion to a use case, and shown a feature-based comparison of it with related tools. Current and future effort is focused on extending the types of legacy sources supported e.g. XML, latex, PNG, etc., on making it available under various forms e.g. restful services, on both experimental and user-based evaluation, and its usage in large use cases.

Acknowledgements

This work has been part-funded by the European Commission under grant agreement FP7-ICT-2007-3/ No. 231527 (IKS - Interactive Knowledge Stack).

References

1. OpenLink Software. Virtuoso Universal Server 4.5 Data Management and Integration Reviewer's Guide. http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/. Visited on January 2010.
2. Simile. RDFizers. http://simile.mit.edu/wiki/RDFizers. Visited on August 2010.
3. Talis Platform Data Connected. http://www.talis.com/platform/developers/. Visited on January 2010.
4. Eclipse. JFace. http://wiki.eclipse.org/index.php/JFace, Visited on January 2010.
5. Eclipse. SWT: The Standard Widget Toolkit. http://www.eclipse.org/swt/, Visited on January 2010.
6. A. Acciarri, D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, M. Palmieri, and R. Rosati. QuOnto: Querying Ontologies. In M. M. Veloso and S. Kambhampati, editors, *AAAI*, pages 1670–1671. AAAI Press / The MIT Press, 2005.
7. S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller. Triplify: Light-Weight Linked Data Publication from Relational Databases. In *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, pages 621–630. ACM, 2009.
8. T. Berners-Lee. Linked Data. World wide web design issues, July 2006. Visited on August 2010.
9. C. Bizer and R. Cyganiak. D2RQ - Lessons Learned. *W3C Workshop on RDF Access to Relational Databases*, October 2007.
10. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
11. A. Gangemi. What's in a Schema? In C.-R. Huang, N. Calzolari, A. Gangemi, A. Lenci, A. Oltramari, and L. Prevot, editors, *Ontology and the Lexicon: A Natural Language Processing Perspective*, pages 144–182. Cambridge University Press, Cambridge, UK, 2010.
12. A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening Ontologies with DOLCE. In *Proceedings of 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, volume 2473 of Lecture Notes in Computer Science, page 166 ff, Sigünza, Spain, Oct. 1–4 2002.
13. A. Gangemi and V. Presutti. Ontology Design Pattern portal. Available at: http://www.ontologydesign-patterns.org, 2008.
14. F. v. Harmelen and D. L. McGuinness. OWL Web Ontology Language Overview. W3C recommendation, W3C, Feb. 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/.

15. C. Lange. Krextor An Extensible XML to RDF Extraction Framework. In S. Auer, C. Bizer, and G. A. Grimnes, editors, *Proc. of 5th Workshop on Scripting and Development for the Semantic Web at ESWC 2009*, volume 449 of *CEUR Workshop Proceedings ISSN 1613-0073*, June 2009.
16. A. Miles and S. Bechhofer. SKOS Simple Knowledge Organization System Reference. W3C working draft, W3C, June 2008. http://www.w3.org/TR/2008/WD-skos-reference-20080609/.
17. C. S. Peirce. *Hartshorne (Eds.) Collected Papers of Charles Sanders Peirce*. Harvard University Press, 1931.
18. D. Picca, A. M. Gliozzo, and A. Gangemi. LMM: an OWL-DL MetaModel to Represent Heterogeneous Lexical Knowledge. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.
19. V. Presutti and A. Gangemi. Content ontology design patterns as practical building blocks for web ontologies. In *ER '08: Proceedings of the 27th International Conference on Conceptual Modeling*, pages 128–141, Berlin, Heidelberg, 2008. Springer-Verlag.
20. E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. W3C recommendation, W3C, Jan. 2008. http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/.
21. M. Svihla and I. Jelinek. Benchmarking RDF Production Tools. In R. Wagner, N. Revell, and G. Pernul, editors, *DEXA*, volume 4653 of *Lecture Notes in Computer Science*, pages 700–709. Springer, 2007.