The R2R Framework: Publishing and Discovering Mappings on the Web

Christian Bizer, Andreas Schultz
Web-based Systems Group
Freie Universität Berlin, Germany
chris@bizer.de, a.schultz@fu-berlin.de

Abstract: The promise of the Web of Linked Data is to enable client applications to discover new data sources by following RDF links at run-time and to smoothly integrate data from these sources. Linked Data sources use different vocabularies to describe the same type of objects. It is also common practice to mix terms from different widely used vocabularies with proprietary terms. Thus Linked Data applications need to apply mappings to translate Web data to their local schema before doing any sophisticated data processing. Maintaining a local or central set of mappings that cover all Linked Data sources is likely to be impossible due to the size and dynamics of the Web of Linked Data. Thus this paper propagates a distributed, pay-as-you-go integration approach where data publishers, vocabulary maintainers and third parties may publish expressive mappings on the Web. A client application which discovers data that is represented using terms that are unknown to the application may search the Web for mappings and apply the discovered mappings to translate data to its local schema. We propose a language for publishing expressive, named mappings on the Web and a composition method for chaining partial mappings from different sources based on a mapping quality assessment heuristic. The composition method is implemented within the R2R Mapping Engine which can be used by Linked Data applications to translate Web data to their local schema.

Keywords: Linked data, dataspaces, schema mapping, self-descriptive data, pay-as-you-go data integration

1 Introduction

The Web of Linked Data [1] has grown considerably over the last three years and covers a wide range of different domains today [2]. Linked Data sources use different vocabularies to represent data about a specific type of object. For instance, DBpedia, Freebase and LinkedMDB all use their own proprietary vocabularies to represent data about movies (see Fig. 1). GeoNames, LinkedGeoData, and the UK Ordnance Survey all use a different terms to refer to the concept *Administrative District*. For other types of objects, vocabularies have emerged that are used by multiple data sources but usually also not by all data sources that provide data about these objects. For instance,

FOAF[1] is widely used to represent data about people. As commonly used vocabularies[2] often do not provide all terms that a data source needs to publish its content on the Web of Linked Data, data sources often mix terms from multiple commonly used vocabularies with proprietary terms.

The resulting heterogeneity is a major obstacle to building useful Linked Data applications and thus to realizing the promise of the Web of Linked Data: To enable applications to work on top of a single global dataspace which allows them to discover and integrate new data sources at run-time.

Translating data from a potentially endless set of Linked Data sources to the target vocabulary that is expected by an application requires a large number of mappings. Maintaining a local or central set of mappings that covers all Linked Data sources is likely to be impossible, or at least very costly. Thus this paper propagates a distributed, pay-as-you-go data integration approach: Distributed as we build on data publishers, vocabulary maintainers as well as third parties to publish mappings on the Web; pay-as-you-go [3][5] as Linked Data applications are assumed to display Web Data in a rather un-integrated fashion in the absence of mappings, just as Linked Data browsers like Tabulator or Marbles and Linked Data search engines like Sindice or FalconS do today. As more effort is invested over time into generating and publishing mappings on the Web (= pay-as-you-go), Linked Data applications can discover these mappings and use them to further integrate Web data in order to be able to deliver more sophisticated functionality.

This paper proposes a language for publishing expressive, named mappings on the Web and a composition method for chaining partial mappings from different sources based on a mapping quality assessment heuristic. The *R2R Mapping Language* is designed to fulfill the following requirements:

1. *Vocabulary cherry-picking*: As data sources mix terms from different vocabularies, the mapping language has to support fine-grained, self-contained term mappings which can be flexibly combined.
2. *Interlinking and discovery*: Every term mapping must be identified with its own dereferenceable URI in order to enable mappings to be interlinked with RDFS or OWL vocabulary term definitions [7][8] and voiD dataset descriptions [18], and to allow clients applications to discover and retrieve mappings by following RDF links.
3. *Expressivity:* The language needs to provide for structural transformations in order to overcome differing publishing patterns and for property value transformations, for instance in order to normalize different units of measurement.
4. *Dataset-level and vocabulary-level mappings:* Different data sources use different value formats to represent values of the same property. For instance, they provide a distance either in meters or kilometers or names either as *first name family name* or *family name, first name*. Therefore the language must provide for dataset-level mappings as well as for more generic vocabulary-level mappings.

[1] http://www.foaf-project.org/
[2] http://esw.w3.org/TaskForces/CommunityProjects/LinkingOpenData/CommonVocabularies

The mapping composition method is designed to fulfill the following requirements:

1. *Term-level composition*: The composition method should apply a best-effort approach to generate executable transformations based on all mappings that have been discovered so far. If no direct mapping is available for a term, the method should compose mappings into a mapping chain.
2. *Mapping quality assessment*: As the quality of the mappings that are published on the Web may vary widely, the method should apply a heuristic to assess the quality of mappings and prefer mappings that are likely to deliver better results.

The paper is structured as follows: Section 2 discusses related work. Section 3 describes the publication of mappings on the Web using the *R2R Mapping Language*. Section 4 describes how applications can use the mappings to translate Web data to a target vocabulary. Section 5 presents the evaluation of *R2R Mapping Language*.

2. Related Work

There is a large body of related work on ontology alignment in the knowledge representation community [20][11] as well as a large body of related work on schema matching in the database community [21]. Both communities have also developed formalisms to represent correspondences in the form of mappings. For this paper, we consider the task of generating correspondences out of scope and focus solely on publishing existing correspondences on the Web and on discovering and combining mappings from different sources.

For publishing mappings on the Web, the RDF Schema [7], OWL [8], and SKOS [9] recommendations provide the terms *rdfs:subClassOf* / *rdfs:subPropertyOf*, *owl:equivalentClass* / *owl:equivalentProperty*, as well as the SKOS mapping properties. These constructs can be used to represent simple term correspondences. But they are not expressive enough to represent more complex correspondences which involve structural transformations or property value transformations like normalizing different units of measurement. They are thus not sufficient to achieve deeper integration. More expressive mapping languages for RDF data have been developed by Euzenat [19] and Haslhofer [22]. These languages provide a high expressivity but do not provide for interlinking mappings with other Web resources. Thus, they do not allow applications to discover mappings in a follow-your-nose fashion [13] as the R2R Framework does.

This paper builds on the dataspace paradigm that is being developed within the database community [3]. We adopt a pay-as-you-go data integration approach [3][5], as for Web-scale data integration it does not make sense to apply a schema-first integration approach which relies on a unifying schema to be modeled over all data sources before the dataspace can be used. Thus we consider a step-by-step integration approach, which decreases heterogeneity over time, as more likely to succeed [5]. Existing pay-as-you-go data integration systems [4], like PayGo, iMeMex or SEMEX, assume that a single authority controls a dataspace and that this authority also administrates the mappings that are used to incrementally decrease heterogeneity. In contrast, by building on mappings that are published and interlinked on the Web,

we propagate a distributed, community-based approach to mapping provision. A dataspace system that builds on similarly fine-grained term mappings as the approach presented in the paper is iTrails [11]. In contrast to our work which uses mappings for data translation, iTrails uses mappings (hints) for query expansion.

3. Publishing Mappings on the Web

This chapter introduces the *R2R Mapping Language* and explains how mappings are interlinked with RDFS [7] and OWL [8] vocabulary term definitions as well as with voiD dataset descriptions [18]. The mapping language is explained along the use case of integrating data about movies from DBpedia, Freebase and LinkedMDB. Figure 1 shows a subset of the data provided by these sources for the movie *The Shining*. Figure 2 contains two R2R mappings which are published as Linked Data by DBpedia and which map the *linkedmdb:director* property to the *dbpedia-owl:director* property and the *freebase:runtime* property to the *dbpedia-owl:runtime* property. Figure 3 contains a mapping between the terms *dbpedia-owl:Person* and *foaf:Person* and a mapping between the *linkedmdb:runtime* property and the *freebase:runtime* property. The mappings are assumed to be published on the Web by a third party.

```
01: # Data from LinkedMDB
02: <http://data.linkedmdb.org/resource/film/2014> rdf:type movie:film ;
03:  linkedmdb:director
04:       <http://data.linkedmdb.org/resource/director/8476> ;
05:  linkedmdb:runtime "146" .
06:
07: # Data from DBpedia
08: <http://dbpedia.org/resource/The_Shining_%28film%29> a dbpedia:Film
09:  dbpedia-owl:director dbpedia:Stanley_Kubrick ;
10:  dbpedia-owl:runtime "8760.000000"^^xsd:double .
11:
12: # Data from Freebase
13: <rdf.freebase.com/ns/guid.9202a8c04000641f800000000046c3da>
14:  freebase:film.film.directed_by freebase:en.stanley_kubrick ;
15:  freebase:film.film.runtime freebase:m.0k6ftd .
16:  freebase::m.0k6ftd
17:  freebase:film.film_cut.runtime "146.0"^^xsd:float .
```

Fig. 1: Data about the movie The Shining published by LinkedMDB, DBpedia and Freebase (namespace declarations are omitted).

3.1. Representing Correspondences

There are two W3C standards that can be employed as the basis for designing a language for publishing fine grained term correspondences on the Web: The SPARQL query language [14] and the Rules Interchange Format (RIF) [15]. We have chosen to base the R2R Mapping Language on SPARQL because many developers are already familiar with using SPARQL CONSTRUCT queries to express data transformations and as there are several performant SPARQL stores [16] that are known to scale to the multi-billion triples use cases that arise from the Web of Linked Data. In the

following we will give an overview of the *R2R Mapping Language*. The complete specification of the language is found on the R2R website[3].

```
01: <http://mappings.dbpedia.org/r2r/LinkedMDBDirectorToDirector>
02:  rdf:type r2r:Mapping ;
03:  r2r:prefixDefinitions "dbpedia-owl: <http://dbpedia.org/ontology/>.
04:           lmdb: <http://data.linkedmdb.org/resource/movie/>" ;
05:  r2r:sourcePattern "?SUBJ lmdb:director ?director" ;
06:  r2r:targetPattern "?SUBJ dbpedia-owl:director ?director" ;
07:  dc:creator www4:is-group/resource/persons/Person4 ;
08:  dc:date "2010-06-23"^^xsd:date .
09:
10: <http://mappings.dbpedia.org/r2r/FreebaseFilmRuntimeToRuntime>
11:  rdf:type r2r:Mapping ;
12:  r2r:prefixDefinitions "dbpedia-owl: <http://dbpedia.org/ontology/>
13:           . fb: <http://rdf.freebase.com/ns/>" ;
14:  r2r:sourcePattern "?SUBJ fb:film.film.runtime ?ro .
15:           ?ro fb:film.film_cut.runtime ?runtimeInMinutes" ;
16:  r2r:targetPattern "?SUBJ dbpedia-owl:runtime
17:           ?'runtimeInSeconds'^^xsd:double" ;
18:  r2r:transformation "?runtimeInSeconds = ?runtimeInMinutes * 60" ;
19:  r2r:sourceDataset <http://mappings.dbpedia.org/r2r/freebaseVOID> ;
20:  r2r:targetDataset <http://dbpedia.org/DBpediaVOID> ;
21:  dc:creator www4:is-group/resource/persons/Person4;
22:  dc:date "2010-06-23"^^xsd:date.
23:
24: <http://dbpedia.org/ontology/runtime> r2r:hasMapping
25:    <http://mappings.dbpedia.org/r2r/FreebaseFilmRuntimeToRuntime>
```

Fig. 2: Two R2R mappings which are published by the DBpedia project as Linked Data.

The R2R mapping language is designed for publishing mappings as Linked Data on the Web. Thus mappings are represented as RDF and each mapping is assigned its own dereferenceable URI. Similar to the SPARQL query language, the R2R mapping language operates on RDF level and does not rely on any further assumptions about the semantics of Web data. The main construct of the R2R mapping language are *r2r:Mappings*. A *r2r:Mapping* is a self-contained unit that represents a correspondence between terms in two vocabularies. Similar to a SPARQL CONSTRUCT clause, a *r2r:Mapping* has a *r2r:sourcePattern* and a *r2r:targetPattern*.

The source pattern is matched against Web data and binds values to a set of variables. The source pattern may consist of all expressions that are allowed inside a SPARQL WHERE clause with two restrictions: (i) variables are not allowed in the predicate position of a triple pattern; (ii) a special variable ?SUBJ must be used to identify the subject URI of the resource being mapped. This variable is later used to combine triple patterns from different mappings.

The target pattern is used to produce triples in the target vocabulary. The target pattern may consist of multiple triples patterns of the form *VarOrIRIref IRIref VarOrIRIrefOrModifier*. As syntactic shorthand, the target patterns may also use property paths of the form *VarOrIRIref (IRIref VarOrIRIref)+ IRIref VarOrIRIrefOrLiteral OrModifier*. Lines 5 and 6 within Figure 2 contain a source and a target pattern which defines how *lmdb:director* triples are translated into *dbpedia-*

[3] http://www4.wiwiss.fu-berlin.de/bizer/r2r/spec/

owl:director triples. Lines 16 and 17 in Fig. 3 contain a path expression that generates triples according to the Freebase publishing pattern.

```
01: <http://thirdparty.org/mappingDbpediaPersonToFoafPerson>
02:  rdf:type r2r:Mapping ;
03:  r2r:prefixDefinitions "dbpedia-owl: <http://dbpedia.org/ontology/>.
04:           foaf: <http://xmlns.com/foaf/0.1/>" ;
05:  r2r:sourcePattern "?SUBJ rdf:type dbpedia-owl:Person" ;
06:  r2r:targetPattern "?SUBJ rdf:type foaf:Person" ;
07:  dc:creator <http://thirdparty.org/andreas> ;
08:  dc:date "2010-06-11"^^xsd:date .
09:
10: <http://thirdparty.org/mappingRuntimeLinkedmdbToFreebase>
11:  rdf:type r2r:Mapping ;
12:  r2r:prefixDefinitions
13:    "movie: <http://data.linkedmdb.org/resource/movie/> .
14:     fb: <http://rdf.freebase.com/ns/>" ;
15:  r2r:sourcePattern "?SUBJ movie:runtime ?runtime" ;
16:  r2r:targetPattern "?SUBJ fb:film.film.runtime ?generatedURI
17:     fb:film.film_cut.runtime ?'runtime'^^xsd:float" ;
18:  r2r:transformation "?generatedURI = concat(?SUBJ, 'Runtime')" ;
19:  r2r:sourceDataset <http://mappings.dbpedia.org/r2r/linkedmdbVOID> ;
20:  r2r:targetDataset <http://mappings.dbpedia.org/r2r/freebaseVOID> ;
21:  dc:creator <http://thirdparty.org/andreas> ;
22:  dc:date "2010-06-11"^^xsd:date .
```

Fig. 3: R2R mappings for translating *rdf:type dbpedia:Person* triples into *rdf:type foaf:Person* triples and for mapping between the *linkedmdb:runtime* property and the *freebase:runtime* property.

r2r:Mappings may contain *r2r:transformation* clauses. Transformations define how variable bindings are transformed before being inserted into the target pattern. Line 18 within Figure 2 contains a transformation which calculates the *?runtimeInSeconds* from the *?runtimeInMinutes.* A *r2r:transformation* always consists of a result variable followed by the equals sign and a transformation definition represented by nest-able functions. The R2R mapping language provides a set of common string functions, such as concat() or split(), arithmetic functions, as well as list functions. Line 18 within Figure 3 shows a transformation which concatenates 'Runtime' to the subject URI in order to generate a URI for the new intermediate resource that is generated by the target pattern in Line 16. In addition to transformation functions, which only modify the lexical value of a variable binding, R2R also offers modifiers that can be used to modify further aspects of the binding. There are modifiers for assigning data types and language tags and for converting a literal into a URI reference using a pattern. Line 17 within Figure 2 shows how a data type modifier is used to assign *xsd:double* to the runtime in seconds. The modifier is expressed by using single quotes around the variable name preceding the data type definition.

3.2. Interlinking Mappings with Vocabulary Terms and Dataset Descriptions

In order to enable Linked Data applications to discover mappings on the Web of Linked Data, R2R mappings are interlinked with RDFS or OWL term definitions that are published according to the best-practices provided by Berrueta and Phipps in [10]

as well as with voiD dataset descriptions [18]. The R2R mapping language defines the link type *r2r:hasMapping* to interlink mappings with RDFS or OWL term definitions and voiD dataset descriptions. Lines 24 and 25 contain a *r2r:hasMapping* link pointing from the vocabulary term *dbpedia-owl:runtime* to the *FreebaseFilm RuntimeToRuntime* mapping. Linked Data applications that dereference the vocabulary term *dbpedia-owl:runtime* receive this link together with the definition of the term and can follow it to discover the mapping.

As different data sources use different value formats to represent values of the same RDF property, the R2R Mapping Language distinguishes between *vocabulary-level mappings* and *dataset-level mappings*. Vocabulary-level mappings are usually more generic and might be applied to transform data from and to all data sources that use a specific vocabulary term. Dataset-level mappings specify how data should be translated between two specific data sources. They usually define more detailed transformations to overcome property value heterogeneity, for instance by normalizing different units of measurement or by adding language tags or data types to property values.

If a mapping applies property value transformations or structural transformations that only make sense on specific input data, the mapping publisher can restrict the scope of the mapping to be used only with input data that conforms to the publishing pattern of a specific data source by adding a *r2r:sourceDataset* triple to the mapping pointing at the voiD dataset description of the data source. For instance, a data source might provide people's names in the form *last name, first name* and exactly such input is required by a mapping which splits the name property into a *firstname* and *lastname* property. Thus the publisher restricts the scope of the mapping to data having the same form as data from this data source. Similarly, if a mapping applies value transformations or structural transformations to produce data according to the publishing pattern of a specific data source, the mapping publisher can annotate the mapping to produce specific output by interlinking it with the voiD description of the target dataset using the *r2r:targetDataset* link type. Line 19 in Figure 2 uses the *r2r:sourceDataset* property to restrict the mapping to be applied to data that is represented according to the Freebase publishing pattern. Line 20 annotates the mapping to produce values according to the DBpedia publishing pattern, meaning that the runtime is represented in seconds and that the literal has the datatype *xsd:double*.

In addition to interlinking mappings, the publishers of mappings should also provide metadata about the mappings in order to give Linked Data applications hints for assessing the quality of published mappings. Lines 7 and 8 in Figure 2 contain such meta information in the form of a *dc:creator* link to data about the publisher of the mapping and a *dc:date* triple declaring the creation date of the mapping.

4. Using the R2R Framework

We have implemented a *R2R Mapping Engine* that can be used within applications that consume Linked Data from the Web to translate Web data to an application-specific target schema. The *R2R Mapping Engine* is written in Java and can be

downloaded from the R2R website[4] under the terms of the Apache license. This chapter describes how the *R2R Mapping Engine* is used within Linked Data applications.

4.1. Architecture

Figure 4 gives an overview of the architecture of a Linked Data application that uses R2R mappings to translate Web data to the applications-specific target schema. The architecture consists of a *Web Data Access Module* which retrieves RDF data from the Web by following RDF links. The data access module can for instance be realized using ldSpider[5]. The access module stores Web data in a *Temporal Store*. The data is represented as a set of Named Graphs where all data from one data source is contained in its own Named Graph. These graphs are called dataset graphs (DSG_n) in the proceeding. The URI of the corresponding voiD description is attached to each DSG_n.

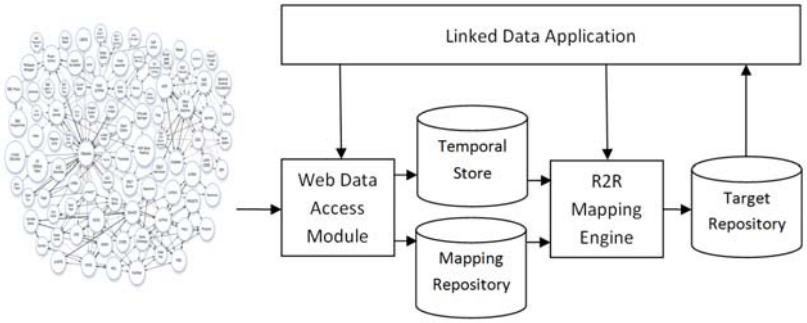

Fig. 4: Architecture of a Linked Data application that employs R2R mappings to translate Web data to its local target schema.

R2R mappings that are discovered on the Web are stored in a *Mapping Repository*. In addition to discovering mappings by following RDF links, the *Web Data Access Module* also queries the Semantic Web Search engines Sindice and FalconS for further R2R mappings. This ensures that third party mappings are also discovered. The application provides the *R2R Mapping Engine* with a description of the target vocabulary. The description consists of a simple set of URIs identifying the terms (properties as well as classes) of the target vocabulary. This set will be called target vocabulary terms (*TVT)* in the proceeding. The mapping engine translates the data from the temporal store into the target vocabulary and stores the resulting triples in the target repository. Afterwards, it deletes the data in the temporal store. The application can now issue queries using the target vocabulary against the target repository.

[4] http://www4.wiwiss.fu-berlin.de/bizer/r2r/
[5] http://code.google.com/p/ldspider/

4.2. Overview of the Data Translation Process

The *R2R Mapping Engine* applies a mapping composition method for selecting and chaining partial mappings from different sources based on a mapping quality assessment heuristic. The mapping chains that are most likely to produce high-quality output data are then employed to translate data to the target vocabulary. The complete mapping composition and data translation process is described in [16]. In the following, we give a brief overview of the process:

1. For each DSG_n, the engine determines the set of vocabulary terms (SVT_n) that are used within the graph.
2. For each term TVT_n in *TVT* and each DSG_n, the engine builds a mapping search graph which contains the mappings that can be chained to connect TVT_n to term in SVT_n.
3. The engine constructs the mapping chain $MC_{TVTnDSGn}$ from the search graph which is most likely to produce high quality translations based on the mapping quality assessment heuristics described below.
4. The engine executes each mapping chain and writes the resulting triples into the target repository.

The quality of any content that is published on the Web is uncertain due to the open nature of the Web [23]. In the Linked Data context uncertainty applies to instance data, links between data sources as well as to mappings. Thus before mappings should be used, a Linked Data application needs to assess their quality and decide whether it wants to trust discovered mappings. The mapping quality assessment heuristic used by the R2R Mapping Engine is based on the following assumptions:

1. As we expect the quality of vocabulary-level mappings provided by vocabulary maintainers themselves to be higher than the quality of mappings provided by third parties, the engine prefers mappings which are published in the same domain as the vocabulary itself.
2. As we expect the quality of dataset-level mappings provided by data publishers to be higher than the quality of mappings provided by third parties, the engine prefers mappings which are published in the same domain as the target dataset, afterwards it considers mappings published in the same domain as the source dataset, and then mapping published in any domain.
3. Because every mapping is a potential source of failure, we expect the quality of data translations to decrease with the length of the mapping chains. The mapping engine therefore prefers short mapping chains wherever possible.

The details on how these heuristics are applied within the mapping quality assessment function used by the R2R Mapping Engine are described in [16].

5. Evaluation

We have tested the expressivity of the R2R Mapping Language by formulating mappings between DBpedia and 11 data sources that are interlinked with DBpedia[6]. The mappings are published as Linked Data on the Web. In addition all mappings can be downloaded as a single file from http://mappings.dbpedia.org/r2r/DBpediaToX.n3. Table 1 gives an overview of the R2R features that were used to formulate the mappings. The abbreviations in the table headings refer to the following R2R features that were required for formulating the mappings: *URI replace* = Simple translation by replacing URIs of the source vocabulary with URIs of the target vocabulary; *Struct trans 1:1* = Structural transformation of the RDF graph describing an instance; *Struct trans 1:n* = Structural transformation where one value in the source dataset results in the creation of multiple values in the target dataset or vice versa; *Val trans* = Literal value transformation for instance using a string function; *UoM trans* = Unit of measurement normalization; *DT mod* = Data type modifier applied to literal value; *LG mod* = Language modifier applied to literal value; L2U mod = Modifier applied to create a URI from a literal value.

Class : Data sources	URI repl-ace	Struc trans 1:1	Struc trans 1:n	Val trans	UoM trans	DT mod	LG mod	L2U mod
Place : GeoNames / DBpedia	x					x		
Artist : MusicBrainz / DBpedia	x	x						
Place : NYT / DBpedia	x							
Country : Factbook / DBpedia	x	x	x		x	x		
Book : BookMashup / DBpedia	x	x		x		x	x	x
Author : Gutenberg / DBpedia	x	x	x	x			x	
County : US Census / DBpedia	x			x		x		
Organiza. : Dailymed /DBpedia	x							
Film : Linkedmdb / DBpedia	x	x			x	x		
Drug : Drugbank / DBpedia	x	x	x	x		x	x	x
Film : Freebase / DBpedia	x	x		x	x	x		
Musician : Freebase / DBpedia	x	x		x	x	x		

Table 1: Overview of the R2R features that were required to formulate mappings between DBpedia and datasets that are interlinked with DBpedia.

The R2R Mapping Language proved to be expressive enough in this experiment to represent all mappings that were required to translate data between the representations used by the sources. The experiment also showed that far more expressivity is required to properly translate data to a target schema than currently provided by standard terms such as owl*:equivalentClass / owl:equivalentProperty* [8] or *rdfs:subClassOf / rdfs:subPropertyOf* [7].

[6] http://wiki.dbpedia.org/Interlinking

6. Conclusion

A key difference that distinguishes Linked Data technologies from other approaches to sharing data on the Web, like Web 2.0 APIs, is that the Linked Data principles enable data to be published in a self-descriptive manner [13] and thus ease the integration of data from different sources. An important aspect of self-descriptiveness is the reuse of terms from common vocabularies that are made accessible on the Web via dereferenceable URIs and thus allow client applications to retrieve term definitions [10].

But as common vocabularies often do not cover all aspects of published data and as choosing terms from common vocabularies increases the effort involved in Linked Data publication, we experience that a lot of data on the Web is represented using proprietary terms.

This paper takes the self-descriptiveness [13] of Web data a step further by proposing a framework to enable data publishers, vocabulary maintainers and third parties to publish fine-grained vocabulary mappings on the Web and to interlink these mappings with other web resources so that they can be discovered by client applications in a follow-your-nose fashion.

This approach decomposes the Web-scale data integration problem along two dimensions: Time and effort allocation. Data integration can be realized in a pay-as-you-go fashion over time, meaning that data providers can follow Tim Berners-Lee's *raw data now* practice and start publishing Linked Data using any vocabulary. Later, they (or third parties) can invest effort into reusing terms from common vocabularies and/or invest effort into creating mappings which tie the data to related data sources. This lowers the entry barrier for publishing Linked Data to a large extent. For applications, this means that they can initially provide only basic services on top of very large dataspace. As more mappings become available in the dataspace, larger parts of the dataspace can be integrated more deeply, and the quality of the provided services can be increased accordingly. The second dimension is effort allocation. In classic data integration settings as well as within the context of Web 2.0 mashups, the data integration effort is solely shouldered by the data consumer. By publishing mappings on the Web, data publishers and third parties - like industry consortia - may provide integration hints [12]. This enables the data integration effort to be split between data publishers, third parties and the data consumer.

As the amount of instance data on the Web of Linked Data is growing rapidly and as more and more data providers start setting RDF links between instances, we think that it is time for the Linked Data research community to focus its attention on overcoming the vocabulary-level heterogeneity which we observe on the Web of Linked Data. We see the R2R Framework as a first step into this direction and hope that the R2R Framework will interest further groups to work on solutions for integrating Web data based on mappings that are published by various parties on the Web of Linked Data and which are thus inherently of uncertain quality [24].

References

1. Berners-Lee, T.: Design Issues: Linked Data. http://www.w3.org/DesignIssues/LinkedData.html (2006)
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web & Information Systems, Vol. 5, Issue 3, pp 1-22 (2009)
3. Franklin, M.J., Halevy, A.Y., Maier, D.: From databases to dataspaces: A new abstraction for information management. SIGMOD Record 34(4), pp. 27–33 (2005)
4. Hedeler, C., et al.: Dimensions of Dataspaces. In: Proceedings of the 26th British National Conference on Databases, pp. 55-66 (2009)
5. Madhavan, J., Shawn, J. R., Cohen, S., Dong, X., Ko, D., Yu, C., Halevy, A.: Web-scale Data Integration: You can only afford to Pay As You Go. Proceedings of the Conference on Innovative Data Systems Research (2007)
6. Das Sarma, A., Dong, X., Halevy, A.: Bootstrapping pay-as-you-go data integration systems. Proceedings of the Conference on Management of Data, SIGMOD (2008)
7. Brickley, D., Guha, R.V.: RDF Vocabulary Description Language 1.0: RDF Schema - W3C Recommendation. http://www.w3.org/TR/rdf-schema/ (2004)
8. McGuinness, D., van Harmelen, F.: OWL Web Ontology Language - W3C Recommendation. http://www.w3.org/TR/owl-features/ (2004)
9. Miles, A., Bechhofer, S.: SKOS Simple Knowledge Organization System Reference – W3C Recommendation. http://www.w3.org/TR/2009/REC-skos-reference-20090818/ (2009)
10. Berrueta, D., Phipps, J.: Best Practice Recipes for Publishing RDF Vocabularies - W3C Working Group Note. http://www.w3.org/TR/swbp-vocab-pub/ (2008)
11. Shvaiko, P., Euzenat, J.: Ontology Matching. http://www.ontologymatching.org/ (2010)
12. Vaz Salles, M.A., Dittrich, J., Karakashian, S.K., Girard, O.R., Blunschi, L.: iTrails: Pay-as-you-go Information Integration in Dataspaces. In: Conference of Very large Data Bases (VLDB 2007), 663-674 (2007)
13. Mendelsohn, N.: The Self-Describing Web. W3C TAG Finding. http://www.w3.org/2001/tag/doc/selfDescribingDocuments.html (2009)
14. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF - W3C Recommendation. http://www.w3.org/TR/rdf-sparql-query/ (2008)
15. Kifer, M.; Boley, H.: RIF Overview - W3C Working Group Note. http://www.w3.org//TR/2010/NOTE-rif-overview-20100622/ (2010)
16. Bizer, C., Schultz, A.: The R2R Data Translation Process. FUB Technical Report. http://www4.wiwiss.fu-berlin.de/bizer/r2r/tr/TR-R2R-DataTranlationProcess.pdf
17. Bizer, C., Schultz, A.: The Berlin SPARQL Benchmark. International Journal on Semantic Web & Information Systems, Vol. 5, Issue 2, pp 1-24 (2009)
18. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets. Proceedings of the 2nd Workshop on Linked Data on the Web (2009)
19. Euzenat, J., Scharffe, F., Zimmermann A.: Expressive alignment language and implementation. Knowledge Web project report, KWEB/2004/D2.2.10/1.0 (2007)
20. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Heidelberg (2007)
21. Rahm, E. and Bernstein, P.: A survey of approaches to automatic schema matching. *The VLDB Journal* 10, 4, pp. 334-350 (2001)
22. Haslhofer, B.: A Web-based Mapping Technique for Establishing Metadata Interoperability. PhD thesis, Universität Wien (2008)
23. Bizer, C., Cyganiak, R.: Quality-driven information filtering using the WIQA policy framework. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Volume 7, Issue 1, pp. 1-10 (2009)
24. Dong, X., Halevy, A. Y., and Yu, C.: Data integration with uncertainty. In: Conference of Very large Data Bases (VLDB 2007), 687-698 (2007)