LOVER: Support for Modeling Data Using Linked Open Vocabularies

Johann Schaible
GESIS - Leibniz Institute for the Social Sciences
Germany
johann.schaible@gesis.org

Thomas Gottron; Stefan Scheglmann
WeST - Institute for Web Science and Technologies
Germany
{gottron,schegi}@uni-koblenz.org

Ansgar Scherp
Institute for Business Informatics and Mathematics
Germany
ansgar@informatik.uni-mannheim.de

ABSTRACT

Various best practices and principles are provided to guide an ontology engineer when modeling Linked Data. The choice of appropriate vocabularies is one essential aspect in the guidelines, as it leads to better interpretation, querying, and consumption of the data by Linked Data applications and users. In this paper, we propose LOVER: a novel approach to support the ontology engineer in modeling a Linked Data dataset. We illustrate the concept of LOVER, which supports the engineer by recommending appropriate classes and properties from existing and actively used vocabularies. The recommendations are made on the basis of on an iterative multimodal search. It uses different, orthogonal information sources for finding vocabulary terms, e.g. based on a best string match or schema information on other datasets published in the Linked Open Data cloud. We describe LOVER's recommendation mechanism in general and illustrate it along a real-life example from the social sciences domain.

Categories and Subject Descriptors

E.2 [**Data Storage Representations**]: Linked representations; H.3.3 [**Information Search and Retrieval**]: Search process, Selection process

General Terms

Design, Measurement

Keywords

Linked Data Modeling, Vocabulary Mapping, Support System

1. INTRODUCTION

The Linked Open Data (LOD) cloud comprises data from diverse domains. The data itself describes resources, is provided in RDF format and interlinks many different and independent data sources. To publish Linked Data, Bizer et al. provided a set of Linked Data guidelines [1], which were updated a few years later by Heath and Bizer [2]. This includes that the data modeler should rather reuse classes and properties from existing vocabularies than re-invent them, and mix several vocabularies where appropriate. Modeling Linked Data generally requires an ontology engineer and a domain expert. The domain expert, on the one hand, has to verify the semantic correctness and integrity of the properties and classes used to represent the data. The aim of the domain expert is to represent resources completely and accurately. The ontology engineer on the other hand has to verify the quality of the dataset with respect to the Linked Data guidelines. The aim of the ontology engineer is to make it as easy as possible for client applications to process the data.

Hogan et al. [3] have shown that most data providers mix only the most prominent vocabularies and re-use only the most common terms. There are tools and services which promote diverse existing vocabularies, but the ontology engineer still has to do a huge part of the modeling process manually with no or minimal support. This is very time consuming but directly affects the semantic richness of the dataset. Investing too little effort in this task might decrease the interoperability of the data, i.e. it makes it difficult for Linked Data applications to consume the data. One reason for this is the syntax of SPARQL queries, as it requires the user to specify the precise details of the structure of the RDF graph being queried. Thus, the user has to be familiar with the dataset, and if several datasets from the same domain are modeled differently, the user or the Linked Data application has to deal with the problem of schema heterogeneity [4].

Therefore, it is a crucial task to support the ontology engineer in modeling Linked Data in the best possible way. Such support can be categorized into eight types of support:

(1) Providing a search mechanism for classes and properties.
(2) Providing meta information about vocabularies.
(3) Providing information about the semantically correct usage of a term.
(4) Providing meta information on terms.
(5) Enabling to complement existing vocabularies.
(6) Enabling to mix a reasonable amount of vocabularies.
(7) Enabling the ontology engineer to model the Linked Data dataset according to the ontology engineering principles as provided by [6].
(8) Providing the support in a semi-automatic way.

EDBT/ICDT '13, March 18 - 22, 2013, Genoa, Italy.

A more detailed discussion of these types of support can be found in the extended version of this paper[1]. It also comprises an enumeration of existing tools and approaches which are used for modeling Linked Data, and an evaluation of these tools and approaches with respect to the different types of support. It is shown that no existing solution includes all types of support. Thus, the different tools stand in isolation and there is no holistic solution to properly support the ontology engineer in modeling Linked Data.

To alleviate this situation and to take a further step towards supporting an ontology engineer in modeling Linked Data, we present LOVER (short for: Linked Open Vocabulary EngineeRing). LOVER is a generic approach, which uses an iterative multimodal search mechanism to recommend the re-use of classes and properties from existing and actively used vocabularies in the LOD cloud. Hereby, LOVER integrates different information sources such as Swoogle for string match based search and the SchemEX index [5] for schema based search. LOVER is not restricted to these information sources and can be extended to incorporate additional sources such as LOV or Wordnet, if the service provides an API. A multimodal search includes specific contextual information, such as the vocabularies already used in the model, which is used to obtain better fitting results. Using this information, LOVER iterates through all schema elements, and during each iteration, it recommends a set of terms for every schema element, adapts, and updates the recommendations using the context information. This way, LOVER is most likely to achieve an optimized mapping. By this, it supports increasing the re-use of existing vocabularies and mixing an appropriate amount of different vocabularies. Furthermore, the search for vocabularies is integrated in the modeling system, which implies that the ontology engineer does not have to incorporate the terms manually.

The remainder of the paper is structured as follows: Section 2 describes the LOVER approach. We illustrate the general concept of our approach and demonstrate it in Section 3 on a real-life example from the social sciences domain. Simultaneously, we provide a preliminary proof of concept of the LOVER approach based on the results from the example. In Section 4, we we conclude our work and discuss the purpose of LOVER in comparison to existing approaches.

2. THE LOVER APPROACH

Regarding the eight types of support in more detail, a system for modeling Linked Data should (1) provide a search mechanism, which supports the ontology engineer to search for classes and properties. It should (2) provide meta information on vocabularies, such as its specific domain and the relative and the absolute number of occurrences in the LOD cloud and (3) provide information on the semantic correct usage of a term as it retrieves the information from Swoogle or other information sources. In addition, it should (4) provide further meta information on terms using the same information sources, (5) support the engineer in complementing existing vocabularies, and also update the recommendation for a mapping after defining a class hierarchy, as this way, it is possible to invent a class and set a subclass relationship between it and another external class. It should (6) support the ontology engineer to mix a reasonable amount of vocabularies, (7) allow the ontology engineer to design a first draft and then refine the data model by adding further details, which is supported by letting him enumerate schema elements, define classes and their hierarchy, and finally object and annotation properties, and (8) allow the ontology engineer to verify the recommendations. This way, each type of support is in every aspect semi-automatic.

LOVER follows an iterative multimodal search mechanism to recommend the ontology engineer classes and properties from existing and actively used vocabularies in the LOD cloud. This way, the ontology engineer is able to re-use such terms to represent data in RDF, instead of re-inventing them. Hereby, LOVER integrates different information sources such as Swoogle or the SchemEX index to gather relevant information on vocabularies mentioned in the previous section. Therefore, LOVER is able to support the ontology engineer regarding every type of support stated above. In following, we provide a more detailed illustration of the multimodal search functionality from different sources in general and applied to our context of searching for re-usable classes and properties. We explain how an iterative approach is advantageous and provide support for deciding for the "best possible way" to publish a data set using Linked Data. Finally, we illustrate how LOVER's iterative approach supports the ontology engineer in developing a Linked Data dataset according to the Ontology Engineering Principles. Hereby, we will illustrate where LOVER provides the different types of support for the ontology engineer.

A multimodal search utilizes several methodologies to retrieve information. Such methodologies include a search by keywords and a search by concepts, where every concept is derived from a specific context. This allows the user to specify his information need more precisely. The multimodal search component in LOVER includes context information about the dataset which has to be modeled as Linked Data. These context information may provide better search results with respect to the types of support we have identified. It covers information on the vocabularies already used in the dataset, information whether the search is for a property or for a class, as well as whether the schema element is a domain, range, object property, or annotation property. By using the Swoogle API for a search based on exact string match, LOVER is able to retrieve information provided by Swoogle on specific terms and vocabularies. This includes for example the number of a specific usage of a term, e.g. `dc:creator` is used 386 times as annotation property (as retrieved on 22/1/2013). The SchemEX index provides information on effective schema design in the LOD cloud, i.e. how properties and classes are used in practice. This information can be used to recommend, e.g., the most common object property between two given concepts. Such a search mechanism can also be very helpful to complement existing vocabularies by defining own subclasses or sub-properties. For example, the ontology engineer models an schema element `ex:MalePerson`, which represents a person, whose gender is male. If some context information was available that it is a "Person", the recommendation might suggest that `ex:MalePerson` might be a subclass of `foaf:Person`. LOVER can incorporate this information from a manual input of the ontology engineer or from previous iterations.

The LOVER approach is an iterative mechanism, meaning that the recommendation uses the context information from every iteration to recommend the most appropriate terms. One iteration is complete, if the user models a schema element by either reusing a term or specifying an own one. This way LOVER recommends but also adapts and updates the recommendations of classes and properties, even for schema elements which have already been mapped to specific terms. Therefore, LOVER is most likely to achieve an optimized mapping according to the specification of an greedy algorithm. This means, to recommend an optimal term for a mapping is an optimization problem. A greedy algorithm utilizes a specific heuristic to provide the optimal choice at each iteration of a pro-

[1]LOVER: Support for Modeling Data Using Linked Open Vocabularies (Work Report) `www.uni-koblenz-landau.de/koblenz/fb4/forschung/publications/Reports`

cess. In the case of modeling a Linked Data dataset, defining one or several mappings at a time is considered to be one iteration. The heuristic comprises an update of the context information after each iteration and performing a further search for appropriate term. To find a global optimum is a NP-complete problem, but the underlying heuristic may result in a local optimum, which approximates a reasonable solution. By this, it supports increasing the reuse of existing vocabularies and mixing a reasonable amount of different vocabularies with respect to the Ontology Engineering Principles. Summarizing, LOVER provides a rough draft of the model which is filled up with more and more details after each further iteration.

At the beginning of the data publishing process, the ontology engineer defines keywords, which determine domain and scope of the Linked Data dataset he intends to model and to publish. This is the first step of (7). Hereby, LOVER incorporates the specified information from the ontology engineer to use it as context in its first search for an appropriate vocabulary. For example, the ontology engineer intends to model a dataset containing data on people and their relationships to each other. This information can be used to allocate the first re-usable vocabularies, in this case FOAF as the data is about people. As next step, the ontology engineer enumerates all schema elements which describe the dataset. This is the also the second step regarding (7), as it is important to get the list of elements which are supposed to be mapped to classes and properties from existing vocabularies. LOVER performs a first keyword-based search to provide and recommend the ontology engineer terms from already published vocabularies for each enumerated schema element. This is support type (1). As part of the recommendation, LOVER displays several meta information we have mention before on the vocabulary and its terms. This supports the engineer regarding the types (2), (3), and (4). This meta information is also used in accordance with the collected context information to rank the results of the term search. Hereby, terms from vocabularies which are already taken into account are ranked higher than terms from other vocabularies. This enables the ontology engineer to specify several mappings to one vocabulary, which results in a mixing of a reasonable amount of vocabularies. This supporting feature is executed with respect to the type (6). Furthermore, if the search involves several information sources, the ontology engineer is presented several result lists from which he may choose the preferred term. This semi-automatic support, which refers to type (8), enables the engineer to make each decision. To this end, the ontology engineer performs a first draft of a mapping. Then, he defines which schema elements are classes within his dataset, which is the further step of (7). LOVER uses this information to perform another search in order to recommend classes from established vocabularies for the schema elements the engineer has already marked as mapping classes. This supports the engineer with respect to type (3). As next step, he defines the hierarchy between the classes. This enables the ontology engineer to select a mapping for the parent class. If the subclass is not mappable to any existing term, LOVER is at least able to provide a recommendation to complement an existing vocabulary, which is a support with respect to type (5). Once the classes and their hierarchy has been modeled, the ontology engineer has to define the object properties, in order to refine their representation. This is again a further step of (7). As the engineer defines an object property between two specific classes, LOVER uses this information as input to search in the SchemEX index. It retrieves the possible object properties and recommends these to the ontology engineer. As last step, the ontology engineer has to define a datatype and annotation properties. LOVER then searches for such datatype and annotation properties of the already modeled class. In addition it displays the best keyword-based search results

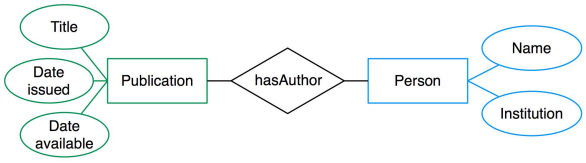

Figure 1: The running example's database scheme

from another information source. The ontology engineer chooses the terms according to (8), and finishes modeling his dataset. When having completed these several iterations, LOVER has made it possible to provide support for the ontology engineer for modeling a Linked Data set including all types of support stated above.

3. MODELING AN EXAMPLE DATASET

To illustrate the LOVER approach and to provide a first preliminary and conceptual evaluation of it, we define the following scenario with real world data from the domain of social sciences: An ontology engineer intends to publish a subset of the data of the GESIS Social Science Open Access Repository (SSOAR)[2] as Linked Data. The Entity Relationship (ER) model in Figure 1 illustrates the schema of the dataset in our scenario. The goal is to model this schema as Linked Data according to the best practices, especially regarding the re-use and mix of existing and actively used vocabularies and their terms. The ontology engineer either uses terms from external vocabularies directly or defines his own vocabulary but links its terms to equal classes or properties from external vocabularies with `owl:equivalentClass` or `owl:equivalentProperty`.

Again, while illustrating the modeling process using the LOVER approach, we explain where LOVER incorporates the types of support presented earlier. Initially we determine the domain and the scope of the dataset, as it is the first step of (7). The domain is *social sciences* and the scope comprises the elements *Person* and *Documents*. LOVER uses this information for a first retrieval of some vocabularies the ontology engineer might be able to re-use. In our example this would be the Dublin Core and the FOAF ontology, as these are the primary results of searching for *Documents* and *Person*. As next step, the ontology engineer enumerates all schema elements within the dataset, i.e. *Publication*, *Title*, *Date issued*, *Date available*, *hasAuthor*, *Person*, *Name*, and *Institution*. LOVER performs a keyword-based search using the Swoogle API and provides the ontology engineer a set of terms for each schema element, which is a support with respect to (1). In our case we can assume that all provided terms are from FOAF and Dublin Core. Then, the recommendations include (2) meta information on FOAF and Dublin Core, (3) the semantic usage of each term within these two vocabularies, and (4) other meta information on each term, such as domain and range of properties and class hierarchies for classes. These information are provided by the vocabulary publishers and can be retrieved via the Swoogle API. The engineer chooses the terms `foaf:publication` for *Publication*, `dcterms:URI` for *URI*, `dcterms:title` for *Title* `dcterms:issued` for *Date issued*, `dcterms:available` for *Date available*, `foaf:Person` for *Person*, and `foaf:name` for *Name*. This is by all means a semi-automatic mapping and thus is provided according to support type (8). Let us say that for *hasAuthor* and *Institution* LOVER was not able to suggest a useful recommendation. The engineer now enumerates the

[2] `http://www.ssoar.info/` retr. 2012/12/20

classes, i.e. *Person* and *Publication* like it is provided by (7). In this scenario, there is no need to define a class hierarchy. If the ontology engineer defined such a hierarchy and provided the parent class with a mapping, LOVER would support to complement the existing vocabulary according to (5). In our scenario, the enumerated classes are used by LOVER to perform another multimodal search for classes only to represent these schema elements. Again, LOVER provides support with respect to type (3). In the case of *Person* no changes occur, but in the case of *Publication* LOVER adapts and updates its recommendation to `swrc:Publication`, since this is the best fitting string-based result for a class. The ontology engineer realizes that this is a better semantic mapping than before and changes the mapping to `swrc:Publication`. Now, the SWRC ontology is integrated in the Linked Data dataset. Thus, LOVER performs a routine to examine if other schema elements can be mapped to terms from the SWRC ontology. This way, it provides support according to type (6). LOVER can yet again update a recommendation to achieve a better mapping. In our case, LOVER has found a suitable mapping for *hasAuthor* with `swrc:author` and `swrc:institution` for *Institution*. The ontology engineer chooses this mapping. The next step is to refine the object properties. The engineer specifies that the property `swrc:author` has the domain `swrc:Publication` and the range `foaf:Person`. LOVER uses this information to search the SchemEX index for the number of occurrences of such a `swrc:Publication swrc:author foaf:Person` triple. In this step, LOVER queries SchemEX several times leaving out in turn the domain, the range, and the object property, to search for the number of occurrences without restricting the the subject type, object type, and concrete property. This way, LOVER retrieves and presents further meta information on the actual usage of types/properties in the Linked Data cloud and provides support for the ontology engineer with respect to type (4). In example, LOVER finds a few occurrences of such a triple. But what if the object property is withdrawn from the search? By doing this, LOVER is able to find more triples including other object properties, such as dcterms:creator between the two classes. This triple has occurred way more often in the Linked Open Data cloud. The ontology engineer might want to change the object property to `dcterms:creator`, but both would be semantically correct according to (3). Finally the ontology engineer specifies that the properties `swrc:institution` and `foaf:name` are annotation properties of `faof:Person` and `dcterms:URI`, `dcterms:title`, `dcterms:issued`, and `dcterms:available` are annotation properties of `swrc:Publication`.

After applying LOVER, the resulting data scheme looks like as depicted in Figure 2. It is described in turtle format representing an example instance. Overall, the use of the LOVER approach has a positive effect on finding and applying appropriate vocabularies and terms for re-use.

4. CONCLUSION

In this paper, we have proposed LOVER as novel approach for incorporating eight different types of support to enable an ontology engineer to model Linked Data according to the best practices. We have illustrated how LOVER supports the ontology engineer to comply to the best practices such as re-use of established vocabularies and mixing a reasonable amount of different vocabularies. Its mechanism complies to the iterative approach of generating Linked Data starting from a rough draft and adding details afterwards, similar to the process of developing an ontology. In addition, it provides a multimodal search functionality, which approximates an optimal mapping of schema elements to terms from established vocabularies, and recommends these mappings to the engineer.

```
@prefix dc: <http://purl.org/dc/elements/1.1/>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns
    #>.
@prefix swrc: <http://swrc.ontoware.org/ontology#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.

<http://ex1/001> rdf:type swrc:Publication;
        dc:title "Example Title";
        dc:issued "Example Issued Date";
        dc:available "Example Available Date";
        dc:creator <http://ex1/name/xyz>.

<http://ex1/name/xyz> a foaf:person;
        foaf:name "xyz";
        swrc:institution "Example Institution".
```

Figure 2: RDF for the Example

After defining different heuristics for the multimodal search, the future work will comprise the implementation of a first prototype of LOVER. We will conduct an evaluation of the quantitative performance of the multimodal search mechanisms, where we examine different variants of recommendations.

5. REFERENCES

[1] BIZER, C., CYGANIAK, R., AND HEATH, T. How to Publish Linked Data on the Web, July 2008.

[2] HEATH, T., AND BIZER, C. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web. Morgan & Claypool Publishers, 2011.

[3] HOGAN, A., UMBRICH, J., HARTH, A., CYGANIAK, R., POLLERES, A., AND DECKER, S. An empirical survey of linked data conformance. *Web Semantics: Science, Services and Agents on the World Wide Web 14* (2012), 14 – 44.

[4] JAIN, P., HITZLER, P., YEH, P. Z., VERMA, K., AND SHETH, A. P. A.p.: Linked data is merely more data. In *In: AAAI Spring Symposium 'Linked Data Meets Artificial Intelligence', AAAI* (2010), Press, pp. 82–86.

[5] KONRATH, M., GOTTRON, T., STAAB, S., AND SCHERP, A. Schemex - efficient construction of a data catalogue by stream-based indexing of linked data. *Web Semantics: Science, Services and Agents on the World Wide Web* (2012).

[6] NOY, N. F., AND MCGUINNESS, D. L. *Ontology development 101: A guide to creating your first ontology*. Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880, 2001.