Mosto: Generating SPARQL Executable Mappings Between Ontologies ⋆ ⋆⋆

Carlos R. Rivero, Inma Hernández, David Ruiz, and Rafael Corchuelo

University of Sevilla, Spain
{carlosrivero, inmahernandez, druiz, corchu}@us.es

Abstract. Data translation is an integration task that aims at populating a target model with data of a source model, which is usually performed by means of mappings. To reduce costs, there are some techniques to automatically generate executable mappings in a given query language, which are executed using a query engine to perform the data translation task. Unfortunately, current approaches to automatically generate executable mappings are based on nested relational models, which cannot be straightforwardly applied to semantic-web ontologies due to some differences between both models. In this paper, we present Mosto, a tool to perform the data translation using automatically generated SPARQL executable mappings. In this demo, ER attendees will have an opportunity to test this automatic generation when performing the data translation task between two different versions of the DBpedia ontology.

Keywords: Information Integration, Data Translation, Semantic-web Ontologies, SPARQL executable mappings

1 Introduction

Data translation is an integration task that aims at populating a target model with data of a source model, which is becoming a major research task in the semantic-web context [5, 12]. Mediators are pieces of software that help perform this task, which rely on mappings that relate source and target models [4].

To reduce integration costs, some techniques automatically generate a set of uninterpreted mappings, a.k.a. correspondences, which must be interpreted to perform the data translation task [4]. They are hints that usually relate a source entity with a target entity, although they may be more complex [4]. The main issue regarding correspondences is that there is not a unique interpretation of them, i.e., different approaches interpret correspondences in different ways [2].

⋆ Supported by the European Commission (FEDER), the Spanish and the Andalusian R&D&I programmes (grants TIN2007-64119, P07-TIC-2602, P08-TIC-4100, TIN2008-04718-E, TIN2010-21744, TIN2010-09809-E, TIN2010-10811-E, and TIN2010-09988-E).

⋆⋆ An implementation and examples regarding this paper are available at: http://tdg-seville.info/carlosrivero/Mosto

Executable mappings encode an interpretation of correspondences in a given query language [6, 11]. These mappings are executed by means of a query engine to perform the data translation task. The main benefit of using these mappings is that, instead of relying on ad-hoc programs that are difficult to create and maintain, a query engine performs the data translation task [6].

In the bibliography, Ressler et al. [10] devised a visual tool to specify handcrafted SPARQL executable mappings. However, it is well-known that handcrafted executable mappings increase integration costs [8]. Furthermore, there are a number of visual tools to specify correspondences between source and target models, such as Clio, Muse, or Clip [1, 6, 9]. After specifying the correspondences, these tools automatically generate a set of executable mappings based on them. Unfortunately, these tools focus on nested relational models, and they are not straightforwardly applicable to semantic-web ontologies due to a number of inherent differences between them [7, 11].

In this paper, we present Mosto, a tool to perform the data translation task between OWL ontologies using automatically generated SPARQL executable mappings. To the best of our knowledge, this is the first tool to automatically generate executable mappings in the semantic-web context. To describe it, we use a demo scenario integrating two different versions of the DBpedia ontology [3].

This paper is organised as follows: Section 2 describes the system, and Section 3 deals with the demo that ER attendees will have the opportunity to test.

2 Mosto

In this section, we present Mosto, our tool to perform the data translation task between two OWL ontologies by means of SPARQL executable mappings. Performing this task in our tool comprises four steps, namely: 1) Selecting source and target ontologies; 2) Specifying restrictions and correspondences; 3) Generating SPARQL executable mappings; and 4) Executing SPARQL executable mappings. These steps are described in the rest of the section.

The first step deals with the selection of source and target ontologies to be integrated. In our demo scenario (cf. Figure 1), we integrate DBpedia ontology v3.2 with DBpedia ontology v3.6, which are shown in a tree-based notation in which classes, data properties and object properties are represented by circles, squares and pentagons, respectively. Note that subclasses are represented between brackets, e.g., *dbp:Artist* is subclass of *dbp:Person* is represented as "*dbp:Artist [dbp:Person]*". In addition, the domain of a property is represented by nesting the property in a class, and the range is represented between '<' and '>', e.g., the domain of *dbp:director* is *dbp:Film* and its range is *dbp:Person*. After selecting the ontologies, Mosto extracts a number of implicit restrictions, which are restrictions that are due to the modelling language of source and target ontologies, in our case, the OWL ontology language.

In the second step, the user specifies explicit restrictions in the source and target ontologies, and correspondences between them. Explicit restrictions are

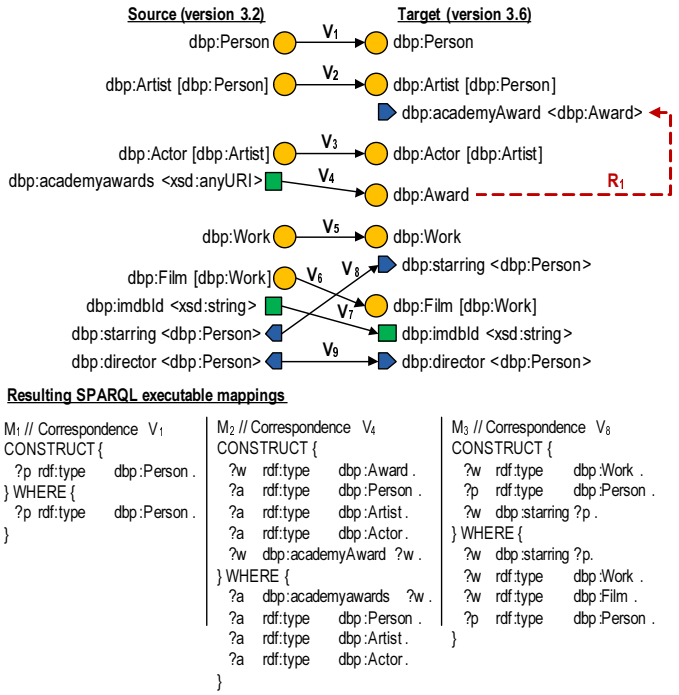

Fig. 1. Evolution in DBpedia (demo scenario)

necessary to adapt existing ontologies to the requirements of a specific scenario, e.g., R_1 is an explicit restriction by which *dbp:academyAward* has a minimal cardinality of one with respect to *dbp:Award*. Correspondences are represented as arrows in Figure 1, e.g., V_8 is a class correspondence that relates *dbp:starring* object property in both source and target ontology versions. Note that Mosto allows to load previously defined restrictions and/or correspondences, or to visually generate new restrictions and/or correspondences according to user preferences.

In the third step, Mosto generates a set of SPARQL executable mappings using (implicit and explicit) restrictions and correspondences. The technique to automatically generate these mappings is described in [11], which is based on clustering those source restrictions, target restrictions and other correspondences that we must take into account to produce coherent target data when performing the data translation task. Note that, if we use each correspondence in isolation to translate data, we may produce incoherent target data, e.g., correspondence V_8 cannot be used in isolation since we do not know how to translate the domain and range of *dbp:starring*. Therefore, our technique clusters V_1, V_5 and V_8, which translate the domain and range of *dbp:starring*, respectively. In addition, every cluster is transformed into a SPARQL executable mapping that encode an interpretation of correspondences. Figure 1 shows three examples of SPARQL

executable mappings generated with our tool: M_1, M_2 and M_3 are the resulting executable mappings of correspondences V_1, V_4 and V_8, respectively.

Finally, in the fourth step, Mosto is able to perform the data translation task by executing the previously generated SPARQL executable mappings over the source ontology to produce instances of the target ontology. Note that, thanks to our SPARQL executable mappings, we are able to automatically translate the data from a previous version of an ontology to a new version.

3 The Demo

In this demo, ER attendees will have an opportunity to use Mosto to test the automatic generation of SPARQL executable mappings using our demo scenario, which integrates different versions of the DBpedia ontology. We will show how the addition or removal of correspondences and restrictions affect the resulting executable mappings. Furthermore, we will perform the data translation task using these mappings, and check whether resulting target data are as expected.

Expected evidences in our demo scenario are the following, namely: 1) the time to generate executable mappings is less than one second; 2) Mosto facilitates the specification of restrictions and correspondences in complex scenarios; and 3) the resulting target data are coherent with expected results.

References

1. B. Alexe, L. Chiticariu, R. J. Miller, and W. C. Tan. Muse: Mapping understanding and design by example. In *ICDE*, pages 10–19, 2008.
2. P. A. Bernstein and S. Melnik. Model management 2.0: manipulating richer mappings. In *SIGMOD*, pages 1–12, 2007.
3. C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia - a crystallization point for the web of data. *J. Web Sem.*, 2009.
4. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, 2007.
5. R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.
6. L. M. Haas, M. A. Hernández, H. Ho, L. Popa, and M. Roth. Clio grows up: from research prototype to industrial tool. In *SIGMOD*, pages 805–810, 2005.
7. B. Motik, I. Horrocks, and U. Sattler. Bridging the gap between OWL and relational databases. *J. Web Sem.*, 7(2):74–89, 2009.
8. M. Petropoulos, A. Deutsch, Y. Papakonstantinou, and Y. Katsis. Exporting and interactively querying web service-accessed sources: The CLIDE system. *ACM Trans. Database Syst.*, 32(4), 2007.
9. A. Raffio, D. Braga, S. Ceri, P. Papotti, and M. A. Hernández. Clip: a tool for mapping hierarchical schemas. In *SIGMOD*, pages 1271–1274, 2008.
10. J. Ressler, M. Dean, E. Benson, E. Dorner, and C. Morris. Application of ontology translation. In *ISWC/ASWC*, pages 830–842, 2007.
11. C. R. Rivero, I. Hernández, D. Ruiz, and R. Corchuelo. Generating SPARQL executable mappings to integrate ontologies. In *ER*, 2011.
12. N. Shadbolt, T. Berners-Lee, and W. Hall. The semantic web revisited. *IEEE Int. Sys.*, 21(3):96–101, 2006.