

Instance Transformation for Semantic Data Mediation

Semantic Web and Web Services Conference

Keywords: Ontology mediation, mapping, alignment, Schema Matching, Data Mediation

François Scharffe

Digital Enterprise Research Institute,
University of Innsbruck, Austria
Email: francois.scharffe@deri.org

Abstract—Ontologies formally specify the terminology used to describe semantic web services functionalities and behavior. Mediators are used to allow interoperability between heterogeneously described web services, a particular type of mediator being the data mediator. A data mediator is used to solve terminological mismatches that arise when two different ontologies are used to describe two services. When a service interacts with another the queries and the resulting instance data must be translated from terms of service into terms of the other. These processes are known as query rewriting and instance transformation. In this short paper we try to address the problem of specifying transformation of the instances as part of the mapping specification between two ontologies. We study the different necessary transformations and give a classification of those.

I. INTRODUCTION

The semantic web and semantic web services have in common the semantic description of the the knowledge resources. The multiplicity of ontologies expected to appear, carried along by the difficulty to agree on common models to represent a particular domain, leads to the ontology mediation research field. This field take advantage of the work already done in the database community in integrating heterogeneous databases to provide transparent query answering. We distinguish two phases in the ontology mediation area. At design-time correspondences are determined between the two ontologies to be aligned. This task is usually realized by hand, using a graphical tool and assisted by an algorithm proposing matching candidates. Once the alignment established, it can be used during run-time to mediate between two ontologies. When the ontologies are describing web services, the alignment is part of a data mediator, which role is to rewrite queries and transform instances, providing interoperability between the services. Query

rewriting is the task that uses the ontology alignment to rewrite a query addressed to a source ontology O_s to a query in terms of a target ontology O_t . Once the rewritten query is executed, the resulting instance data must be converted in terms of O_s . This is the instance transformation task. We present in this paper a number of instance transformations organised in a taxonomy. This classification of the different transformation is necessary to build an effective ontology mediation tool or semantic web service data mediation component.

II. RELATED WORK

A few approaches deal with the instance transformation problem in the ontology mediation field. The schema matching techniques coming from the database research deal with the transformation of the schema instances. In [1] the transformations are given in the form of a *Matching Expression* which may be an operation adding taxes to a price for example. Another kind of possible expressions may be a particular SQL query. In [2] a *generic ontology of mapping relations* is defined. A mapping instance of this ontology linking the entities of two other ontologies is created. This instance may be specified using a *functional-slot-mapping* and then assigning a transformation function to a mapping. This function may be used to express aggregation between the source and the target components, or on the contrary to realize one-to-many decompositions. It also include lexical, numerical and functional transformations. The paper also discuss the possibility of calculating more complex transformations such as date interval from outbound and inbound flight dates. In [3] the mapping functions are represented in a so-called *conversion function network* which aims at relating ontologies having the same base but different contextualizations. The functions perform translation of currencies, aggregate or

split string instances, and dynamically determine values. The transformations are also covered by the Mapping Framework [4]. In this last approach the authors have implemented their framework to effectively cope with the mappings. Instance transformations are represented accordingly to the framework using *Semantic Bridges*. In these bridges can be used different mapping relations to specify the transformations of the instances. Additionally to the basic copy of ontological entities where no transformation is operated, the authors identify the following transformations. "Count Relations" is to count the number of times a particular relation is instantiated for a particular instance (see below the hasParent/noChildren example). The "Split" and "Concat" functions are used to manipulate the attributes values strings to erase the modeling differences. It is interesting to note that in this framework, the instance transformation functions are represented in a service layer accessed by the mapping interface, new services may then easily be added.

III. INSTANCE TRANSFORMATION

In this section we present and classify the instance transformation functionalities a data mediator or an ontology mapping tool must have. We also give insights into the dynamic aspects of instance transformation as well as its possible uses for automating the similarity finding step.

A. Transformation functions classification

We distinguish two types of transformation functions, the structural transformation is linked to the difference of granularity in the two ontologies: a source ontology may have a concept 'Name' which has two subconcepts 'FirstName' and 'LastName', while in the target ontology only the concept 'Name' exists. Supposing that these concepts all have an attribute 'hasString', the first ontology instances of the concepts 'FirstName' and 'LastName' need to be merged in one instance of the 'Name' concept in the target ontology. To realize the merge operation, the strings corresponding to the attributes values need to be concatenated. When defining the mapping, not only it has to be specified that the concepts are equivalents, but also how the instances will be transformed. In the last examples two strings have to be merged. 'FirstName' and 'LastName' may become 'FirstName LastName' or 'FirstName.LastName' or even 'F.LastName', 'F' being the initial letter of the 'FirstName'. The mapping specification should then introduce a set of string operators to help specifying how to concatenate, and conversely to split instances attributes.

Another requirement comes from the different encodings of the values in the source and target ontologies. A real in the source ontology may be represented as an integer in the target ontology or the other way around. Different strategies of conversion may be used and must be specified in the mapping: the real numbers may be truncated, rounded up or down in order to be transformed into integers. Another special type is the date format, where durations may be expressed as start and end dates or as number of days for example. A function is then needed to convert from one format to the other.

The next requirement is related to the ontology structure. An ontology O_s might have a concept 'Parent' with a property 'hasChild', whereas the ontology O_t might also have a class 'Parent', but in this case only with the property 'nrOfChildren'. An aggregate function is required to count the number of children in O_s in order to come with a suitable property filler for 'nrOfChildren'. In the same category we add functions to check the values or the presence of others attributes, instances or relations and functions to compare different attributes values.

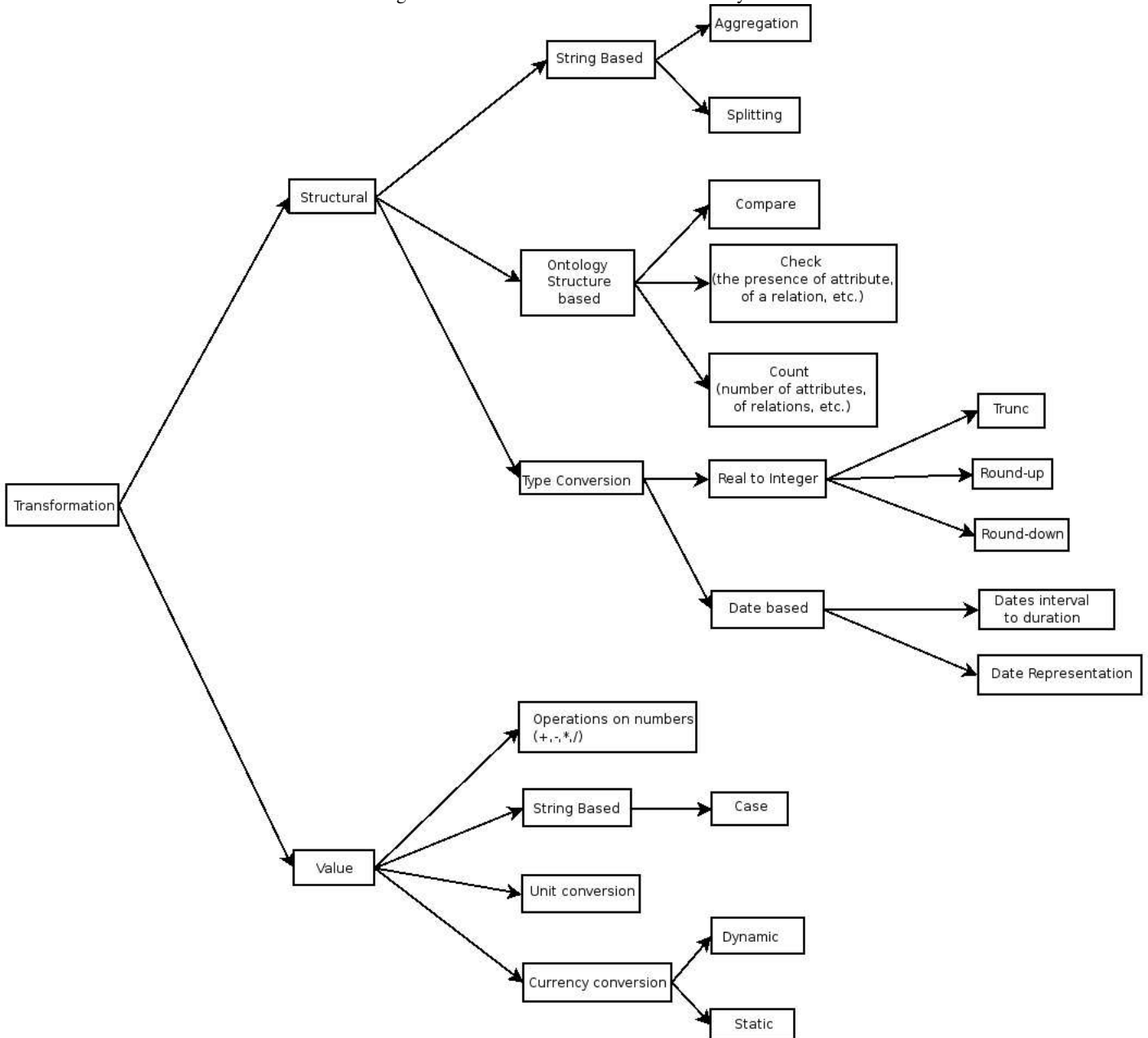
On the other side of the classification tree we find the transformations based on the values of the instances. In this category we include conversions between different currencies and units, and the operations that may have to be done to convert one value into another. The mapping specification must include a set of common mathematical operators expressive enough to correctly transform the values.

The figure 1 presents a taxonomy of transformation functions the mapping specification must deal with.

B. Dynamic aspects

In the dynamic environments that are likely to be modeled by the ontologies, it can be surely expected that the values of the instances will evolve, as the environment change. The price of a product will be likely to change over the time, for example. This dynamicity of the world should be taken into account in an instance transformation framework. In that scope, some of the conversion functions may not be hard-encoded in the mediation system, but rather making calls to remote services following the dynamics of the environment. For the example of the currency conversion functions, the rate of a currency evolves so fast that an hard-coded function will quickly become obsolete. It comes as a requirements that these functions are linked to remote services, for

Fig. 1. Transformation Functions Taxonomy



example web-services, following the evolution of the different currency rates.

C. Automated mapping

The ontology mapping task may be facilitated using automation algorithms, see [5] for a classification of the mapping techniques. These algorithms use different techniques to detect similarities between the ontologies. It is however difficult to detect complex mappings such as those designed using transformation functions. The use

of such functions in common recurring mapping patterns may help to detect such complex relations between the ontological entities to be mapped. For example [4] use mapping services to discover similarities between two attributes in an ontology.

IV. CONCLUSION

We have given an overview of the few approaches dealing with complex instance transformations. We hope the presentation of the different scenarios of application as well as the classification of the different aspects of instance transformation will help to include such functionalities in the ontology mapping tools. The mappings specified using this tools will then be used at run-time by the data mediator in order to perform mediation between semantically enabled web-services. We are actually including the specification of these transformations as mapping patterns modeled using the ontology mapping language[6] developed in the European funded project SEKT¹. We plan to use these patterns as helper to build and discover mappings between two ontologies.

V. ACKNOWLEDGEMENTS

This material is based upon works supported by the EU funding under the DIP and SEKT projects (FP6 - 507483, 506826)

REFERENCES

- [1] E. Rahm and P. A. Bernstein, "A survey of approaches to automatic schema matching," *VLDB Journal: Very Large Data Bases*, vol. 10, no. 4, pp. 334–350, 2001. [Online]. Available: citeseer.nj.nec.com/rahm01survey.html
- [2] M. Crubzy, Z. Pincus, and M. A. Musen, "Mediating knowledge between application components," in *Proceedings of the Semantic Integration Workshop of the Second International Semantic Web Conference (ISWC-03)*, Sanibel Island, Florida, 2003.
- [3] A. Firat, S. E. Madnick, and B. Grosz, "Contextual alignment of ontologies for semantic interoperability," MIT Sloan, MIT Sloan Working Paper 4515-04, 2004.
- [4] N. Silva and J. ao Rocha, "Service-oriented ontology mapping system," in *Proceedings of the Workshop on Semantic Integration of the International Semantic Web Conference (ISWC2003)*, Sanibel Island, USA, 2003.
- [5] P. Shvaiko and J. Euzenat, "A survey of schema-based matching approaches," University of Trento, Tech. Rep. DIT-04-087, 2004.
- [6] F. Scharffe and J. de Bruijn, "A language to specify mappings between ontologies," in *Proc. of the Internet Based Systems IEEE Conference (SITIS05)*. [Online]. Available: <http://sw.deri.org/francois/>

¹<http://www.sekt-project.org>