ShExML: An heterogeneous data mapping language based on ShEx

Herminio Garcia-Gonzalez[1,2], Daniel Fernandez-Alvarez[1], and Jose Emilio Labra-Gayo[1]

[1] Department of Computer Science, University of Oviedo, Oviedo, Asturias, Spain
herminiogg@gmail.com, danifdezalvarez@gmail.com, labra@uniovi.es
[2] Inria Lille Nord Europe, Villeneuve-d'Ascq, France
herminio.garcia-gonzalez@inria.fr

Abstract. Data interoperability is currently a problem that we are facing more intensely due to the appearance of fields like Big Data or IoT. Many data is persisted in information silos with neither interconnection nor format homogenisation. Our proposal to alleviate this problem is ShExML, a language based on ShEx that can map and merge heterogeneous data formats into a single RDF representation. We advocate the creation of this type of tools that can facilitate the migration of non-semantic data to the Semantic Web.

Keywords: data · interoperability · RDF · ShEx · ShExML

1 Introduction

Mapping and merging heterogeneous data sources is a task that has gained in importance throughout the last years. With the improvement of hardware support, the development of new technological areas—such as Big Data or Internet of Things (IoT)—and the deeper interconnection between heterogeneous devices, a huge amount of data is generated every second. However, this data is created in various formats and persisted using different technologies. Therefore, understanding and exploitation of this data becomes a hard work due to the information silos model.

One of the goals of the Semantic Web was the interconnection of data sources and the avoidance of the aforementioned information silos. Therefore, many technologies were proposed to accompany that objective. However, the migration of non-semantic data to the new semantic technologies is a hard task that many individuals and companies are not able to face due to the time or resources consumption. Migrating all databases in a company to their counterpart in Semantic Web world will carry not only the migration of the platforms, but also the data with the development of *ad-hoc* solutions for every dataset. Therefore, solutions that alleviate this translation can contribute to the adoption of semantic technologies or, at least, facilitate it.

We propose a language to map and merge heterogeneous data into its Resource Description Framework (RDF) counterpart. But also taking into account usability and easiness of use.

2 Related work

Many mapping languages and tools were proposed to perform a mapping between a non-semantic format to its RDF counterpart. This is the case of XSPARQL [1] which converts from XML to RDF based on XQuery and SPARQL queries, R2RML [2] which allows to define mappings from relational databases to RDF graphs, or CSV2RDF [4] which permits to convert from CSV to RDF.

However, none of these works tackle the mapping and the merging of heterogeneous datasets in the same solution. This is addressed by RML [3] which extends R2RML language to support formats like JSON, CSV or XML in addition to relational databases. Other alternative is YARRRML [5] a text-based language which is intended to be easy-readable by humans. YARRRML is based on YAML and can be used to represent RML and R2RML rules.

ShExML shares the same goal as RML and YARRRML. However, as being based on ShEx, validation of generated data can be done faster, i.e., the gap between ShExML and ShEx is small. Moreover, it is designed to keep the same simplicity and easiness of use that ShEx has.

3 ShExML at a glance

ShExML[3] is based on ShEx [6] which means that language constructions of ShExML are similar to ShEx. Therefore, it uses the shape as the main foundation for every transformation.

Listing 1.1. ShExML example for films

```
PREFIX : <http://example.com/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dbr: <http://dbpedia.org/resource/>
SOURCE films_xml <https://example.com/films.xml>
SOURCE films_json <https://example.com/films.json>
QUERY film_ids_xml <//film/@id>
QUERY film_names_xml <//film/name>
QUERY film_years_xml <//film/year>
QUERY film_directors_xml <//film/director>
QUERY film_ids_json <$.films[*].id>
QUERY film_names_json <$.films[*].name>
QUERY film_years_json <$.films[*].year>
QUERY film_directors_json <$.films[*].director>
EXPRESSION film_ids <$films_xml.film_ids_xml UNION $films_json.film_ids_json>
EXPRESSION film_names <$films_xml.film_names_xml UNION $films_json.film_names_json>
EXPRESSION film_years <$films_xml.film_years_xml UNION $films_json.film_years_json>
EXPRESSION film_directors <$films_xml.film_directors_xml UNION $films_json.film_directors_json>

:Films :[film_ids] {
    foaf:name [film_names] ;
    dbo:year dbr:[film_years] ;
    dbo:director [film_directors] ;
}
```

We can see ShExML as a combination of declarations followed by a set of shapes. Being the declarations a collection of variable definitions and the shapes the core procedure to define and execute the mappings.

[3] ShExML on Github: https://github.com/herminiogg/ShExML

Inside the set of declarations there are prefixes, sources, queries and expressions. Prefixes work as Turtle prefixes; sources allow to define a URL in which the file is hosted; queries are intended to define reusable queries for the previously defined sources (which normally are defined in a query language, e.g., JSONPath or XMLPath); and expressions which are used to perform the queries over a source, make unions among queries and transform them.

Listing 1.2. JSON films file

```
{
  "films": [
    {
      "id": 3,
      "name": "Inception",
      "year": "2010",
      "director":
        "Christopher Nolan"
    }, {
      "id": 4,
      "name": "The Prestige",
      "year": "2006",
      "director":
        "Christopher Nolan"
    }
  ]
}
```

Listing 1.3. XML films file

```
<films>
  <film id="1">
    <name>Dunkirk</name>
    <year>2017</year>
    <director>
      Christopher Nolan
    </director>
  </film>
  <film id="2">
    <name>Interstellar</name>
    <year>2014</year>
    <director>
      Christopher Nolan
    </director>
  </film>
</films>
```

Thus, imagine that we want to make the transformation of two lists of films: one in JSON and the other in XML (see Listings 1.2 and 1.3). We define a ShExML which can convert both files to RDF and merge them into a single RDF file (see Listing 1.1). This conversion has a single shape called `:Films` which has the main conversion for the films. In order to construct each triple a name is defined under the `:[films_ids]` directive which will match with the subject of every triple generated by this shape. Then, predicates and objects are generated, based on the previous ids, using the expressions enclosed between braces. For example, `foaf:name [films_name]` will generate a triple in the form of `subject foaf:name :object`. Notice that every expression enclosed between square brackets allows a prefix definition which tells the compiler if this expression will be a node or a literal. Moreover, if a query produces a list of results, instead of a single one, the ShExML engine performs the mapping taking into account the relation of them with each entity. Hence, making it possible to merge files with various entities. Finally, the result of this example is showed in Listing 1.4.

Listing 1.4. Result of mapping with ShExML in Turtle format

```
@prefix dbo:   <http://dbpedia.org/ontology/> .
@prefix :      <http://example.com/> .
@prefix dbr:   <http://dbpedia.org/resource/> .
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

:4      dbo:director  "Christopher Nolan" ;
        dbo:year      dbr:2006 ;
        foaf:name     "The Prestige" .

:3      dbo:director  "Christopher Nolan" ;
        dbo:year      dbr:2010 ;
```

```
        foaf:name     "Inception" .

:2      dbo:director  "Christopher Nolan" ;
        dbo:year      dbr:2014 ;
        foaf:name     "Interstellar" .

:1      dbo:director  "Christopher Nolan" ;
        dbo:year      dbr:2017 ;
        foaf:name     "Dunkirk" .
```

4 Conclusions

In this work, we have presented ShExML, a language that allows to map and merge heterogeneous data into its RDF counterpart. This tool helps the migration of semi-structured data to a semantic data format, improving its interoperability and searchability. With the development of this solution, the integration of data into the Semantic Web is an easier task and it can be adapted to different scenarios. We are planning to include some extra features in future versions, such as: the unification of URIs between different representations, the matching between generated URIs and existing ones in the Linked Open Data cloud and the conversion of streaming sources.

Acknowledgments This work has been partially funded by the Vicerectorate for Research of the University of Oviedo under the call of "Plan de Apoyo y Promoción de la Investigación" and by the Ministerio de Economía, Industria y Competitividad under the call of "Programa Estatal de I+D+i Orientada a los Retos de la Sociedad" (project TIN2017-88877-R).

References

1. Bischof, S., Decker, S., Krennwallner, T., Lopes, N., Polleres, A.: Mapping between RDF and XML with XSPARQL. Journal on Data Semantics **1**(3), 147–185 (2012)
2. Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF Mapping Language. https://www.w3.org/TR/r2rml/ (2012), W3C Recommendation 27 September 2012
3. Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van de Walle, R.: RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In: LDOW. Seoul, Korea (2014)
4. Ermilov, I., Auer, S., Stadler, C.: CSV2RDF: User-driven CSV to RDF mass conversion framework. In: Proceedings of the ISEM. vol. 13, pp. 04–06. Graz, Austria (2013)
5. Heyvaert, P., De Meester, B., Dimou, A., Verborgh, R.: Declarative Rules for Linked Data Generation at your Fingertips! In: Proceedings of the 15th ESWC: Posters and Demos. Heraklion, Greece (2018)
6. Prud'hommeaux, E., Labra Gayo, J.E., Solbrig, H.: Shape Expressions: An RDF Validation and Transformation Language. In: Proceedings of the 10th International Conference on Semantic Systems. pp. 32–40. SEM '14, ACM, New York, NY, USA (2014)