User-friendly Visual Creation of R2RML Mappings in SQuaRE

Jarosław Bąk[1], Michał Blinkiewicz[1] and Agnieszka Ławrynowicz[2]

[1] Institute of Control and Information Engineering,
Poznan University of Technology,
Piotrowo 3a, 60-965 Poznan, Poland
[2] Faculty of Computing,
Poznan University of Technology,
Piotrowo 3, 60-965 Poznan, Poland
`{firstname.lastname}@put.poznan.pl`

Abstract. We present the recent progress of SQuaRE (SPARQL Queries and R2RML mappings Environment) which is aimed at providing support for ontology-based data access. The latest SQuaRE's progress is based on the results of an evaluation conducted in order to gather user experience about using the tool. We describe the evaluation, results, newly implemented features as well as our future development plans.

Keywords: visual R2RML mappings, ontology-based data access, ontology visualization

1 Introduction

Ontology-Based Data Access (OBDA) is a technique [9] that allows to view relational or tabular-based data as RDF[3] triples. In general, the technique provides methods for a seamless translation of tables into RDF graphs. Moreover, OBDA supports ontology reasoning which is important when we want to utilize the full potential of an ontology. As a result, a user that applies an OBDA tool is able (or should be able) to perform the same operations on tabular data as they were RDF triples stored in a triple store (infer, query, update etc.). However, it is a very simplified description of the OBDA approach since the process of integrating relational data with ontologies is complicated.

Usually, an OBDA user needs to create a set of mappings between an ontology and relational data in order to be able to execute a SPARQL query. As a result the OBDA approach requires that the user is familiar with ontologies (OWL[4]), semantic data representation (RDF), mappings (R2RML [3]), a database language (SQL) and a semantic query language (SPARQL[5]). According to this, a lot of skills is required to begin the process of integrating legacy data with new

[3] Resource Description Framework: `https://www.w3.org/RDF/`
[4] Web Ontology Language: `https://www.w3.org/OWL/`
[5] `https://www.w3.org/TR/sparql11-overview/`

semantic technologies. As a result we are strongly motivated to provide an easy-to-use solution that supports an OBDA user in creating R2RML mappings and SPARQL queries. Therefore, even an inexperienced user will be able to create mappings, execute queries and obtain results using OBDA.

The aforementioned issues and our motivation brought the development of SQuaRE (SPARQL Queries and R2RML mappings Environment) [2]. The main goal of the tool is to provide an easy to use interface that supports creation of R2RML mappings as well as creation and execution of SPARQL queries. In order to verify the usability and easiness of SQuaRE's usage we conducted an evaluation in which we focused on user experience. The results of this evaluation helped us to provide new features and to improve tool's interface.

In this paper we present the recent progress of SQuaRE [1,2]. We describe the results of our evaluation as well as new features which were highly anticipated by our testers. The main contributions include: (i) a wizard that helps to begin work with SQuaRE, (ii) a support of domain and range axioms when creating mappings and (iii) the addition of special conditional nodes in order to filter database data. Next sections present related work, description and results of the conducted evaluation, new features in SQuaRE and the next steps in development and research.

2 Related Work

Several tools have been implemented to support a user in defining mappings between data sources and ontologies. We provide the comparison table of the most similar tools to SQuaRE (see Table 1). SQuaRE, Map-On [10], ODEMap-

Table 1: Comparison of main features of SQuaRE and related tools

Features	SQuaRE	OntopPro	Map-On	ODEMapster	Karma	RBA	RMLEditor
Visual mappings editor	✓	–	✓	✓	✓	–	✓
Visual SPARQL queries creator	–	–	–	–	–	–	–
SPARQL queries executor	✓	✓	–	–	–	–	–
Relational database support	✓	✓	✓	✓	✓	✓	✓
Other data formats support	–	–	–	–	✓	–	✓
Ontology browser	✓	✓	✓	✓	✓	✓	–
Database schema browser	✓	–	✓	✓	✓	✓	✓
Web-based	✓	–	✓	–	✓	–	✓

ster[6], Karma [7] and RMLEditor [5] are equipped with a visual mapping editor. Map-On provides a graph layout for creating mappings as well as viewing ontologies and databases. ODEMapster supports a tree graphical layout for database schema and an ontology. Karma provides a table-like interface for representing data sources and a tree layout for visualising an ontology. RMLEditor supports

[6] `http://neon-toolkit.org/wiki/ODEMapster`

a graph layout for mappings creation with a tree layout for visualising data sources and a table-based layout for viewing results of mappings execution on the data source. RBA (R2RML By Assertion) [8] supports a tree layout for displaying databases and ontologies but a user is not able to see a graphical form of the mappings. None of the compared tools likewise SQuaRE do not have a visual SPARQL query creator. However, only SQuaRE and OntopPro[7] enable SPARQL queries execution against the created mappings. All of the selected tools are capable to map relational databases. However, Karma and RMLEditor also support other data formats (like JSON, CSV, etc.). Ontology and database schema browsers are built in almost all aforementioned tools. OntopPro, which is a Protégé plugin, does not allow to browse database schema. The similar situation occurs in RMLEditor which does not provide ontology browser (user is able to find ontology entities but without the overall view at an ontology). Furthermore, only SQuaRE, Map-On, Karma and RMLEditor are accessible via a Web-based interface.

The aforementioned tools provide different features that overlap in some cases. However, none of them provide the comprehensive functionality for an OBDA-based scenario. SQuaRE provides features for creating and managing of both: R2RML mappings and SPARQL queries. Moreover, it supports users in the execution of queries and presents results in a table-like or a graph-based way depending on the form of a SPARQL query [2]. Moreover, we are going to implement support for a graphical creation of SWRL rules [6], which will be another difference to the mentioned tools.

SQuaRE is aimed at providing a simple user interface and easy to use methodology. Nevertheless, it should be perceived as a tool that tries to acquire the best features of other applications and provide them in a graphical way with an easy-to-use interface. The most similar tool at this stage of development is Karma, which provides more mapping methods than SQuaRE and more features regarding data integration, but, in turn, does not handle SPARQL queries and results in a graphical manner. Another very similar tool is RMLEditor[8] which also provides more mapping methods than SQuaRE. Moreover, it is the only tool that employs RML[9] as a mapping language.

It is worth to notice that SQuaRE is still at the early stage of development whereas most of the tools from the list were being developed in the last few years. Some of them are even discontinued, like ODEMapster or RBA.

3 Evaluation of SQuaRE

In order to evaluate SQuaRE we conducted a test in which we focused on the easiness of creating R2RML mappings. In the evaluation we employed the MovieOntology scenario[10] in which the IMDB Movie Ontology constitutes a semantic

[7] `http://ontop.inf.unibz.it/components/sample-page/`
[8] `http://rml.io/RMLeditor`
[9] `http://rml.io/spec.html`
[10] `https://github.com/ontop/ontop/wiki/Example_MovieOntology`

description of IMDB[11] data. As a result a user needs to create appropriate mappings in order to access IMDB data with an OBDA approach.

The main goal of our experiment was to check whether users are satisfied with SQuaRE's interface and its process of creating R2RML mappings. As a result we expected to obtain a number of improvement suggestions as well as gather user experience about using SQuaRE.

The protocol of our evaluation was the following:

1. First 10 minutes of the evaluation:
   – Users were introduced with basic features of SQuaRE.
   – Users were familiarized with the process of creating mappings for classes and properties.
   – Users were guided through our method of creating SPARQL queries, executing them and obtaining results in a tabular- or graph-based way.
2. Next 30 minutes of the evaluation:
   – Users were familiarized with the MovieOntology scenario and they connected to a PostgreSQL database that contains IMDB data.
   – Users were asked to create 10 mappings: 2 for classes, 3 for object properties and 5 for datatype properties.
   – After creation of each mapping (or two of them) users were asked to execute a SPARQL query (queries were provided) to check whether mapping is working or not. In the evaluation each user was asked to execute 7 queries in total.
3. In the last 10 minutes of the evaluation users were requested to answer the following open questions:
   – Do you think that a web-based interface is convenient?
   – Is the process of creating mappings comfortable?
   – Did you have any problem with mappings creation?
   – Do you think that connections between elements on the graphs are clear?
   – Which elements of SQuaRE need improvements and which graphical elements are unclear?

Six users took part in our SQuaRE's evaluation. Five of them were Master students introduced with semantic technologies (especially with OWL ontologies and SPARQL) whereas one user with PhD degree had only general knowledge in this domain.

The evaluation took 50 minutes. During that time none of the users was able to finish all the mappings. Two of them finished 6 mappings, 3 of them finished 4 mappings, and one of them was able to finish 3 mappings. However, we obtained a very positive feedback with numerous suggestions about the improvements of SQuaRE. The main results of the evaluation are the following:

1. Users like a web-based interface, however, they pointed out that a desktop version should be also available.

[11] `http://imdb.com`

2. Users like our method of drag and drop when creating mappings; moreover, they emphasized that this kind of creation provides clear and easy way of constructing mappings. However, they pointed out that filtering data by writing SQL statements is not a convenient way.
3. Generally, users did not have any problems with mappings creation. However, at some points they had problems with understanding the IMDB ontology as well as provided database schema. Thus, issues occurred when users were creating mappings. However, they were not connected with the mappings creation method but with reflecting the database elements in the ontology entities.
4. Users appreciated different colors for elements but some of them did not understand what is the meaning of the colors – a short legend or documentation is needed.
5. Users emphasized that SQuaRE's user interface is clear and easy to understand but the navigation and work flow need improvements. Especially, when a user starts using SQuaRE a few options need to be set. Those settings confused users. Particularly, when the user starts using SQuaRE he/she is confound where to start and what to do first.

According to the conducted evaluation we developed and implemented new features that resolve most of the aforementioned issues. New functionality as well as SQuaRE's progress are described in Section 4.

4 New User-friendly Features

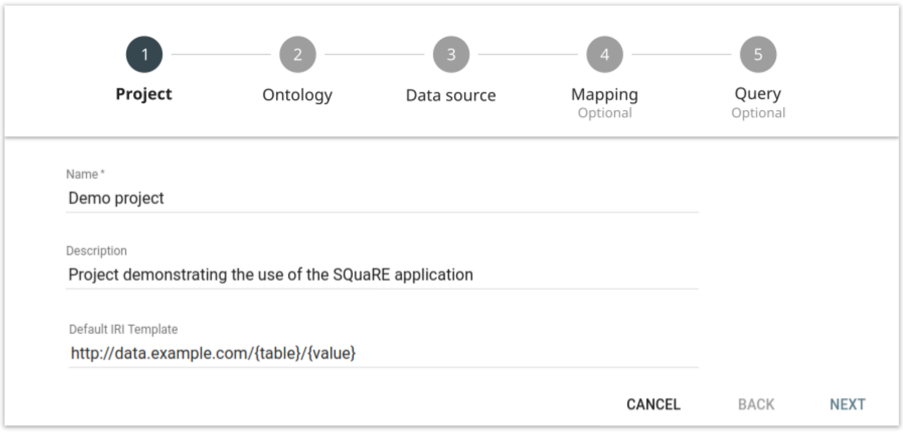
4.1 First Steps Wizard

The results of the evaluation described in Section 3 show that users may be confused when they use SQuaRE for the first time. The solution to this problem could be a wizard assisting a user in a process of environment configuration and usage which means configuring data source connection, loading appropriate ontology, providing basic settings (like a default IRI[12] template), creating the first mapping and SPARQL queries. In the current version of the presented tool we have incorporated such a wizard in order to support users in their first steps when using SQuaRE. When the user creates a new project he/she may choose to start with the provided wizard or without it.

The wizard consists of five ordered cards (tabs) which are switchable back and forth. First card allows a user to fill up information about the project itself, i.e., project name, its description and default IRI template (shown in Fig. 1).

The next card is related to ontology selection. This card enables to upload an OWL ontology and browse class and property hierarchies. When the user is convinced about the chosen ontology he/she may go to the next card, which is dedicated to data source configuration. This step contains a form, which has to be filled with correct values related to a database connection configuration.

[12] Internationalized Resource Identifiers: `https://tools.ietf.org/html/rfc3987`



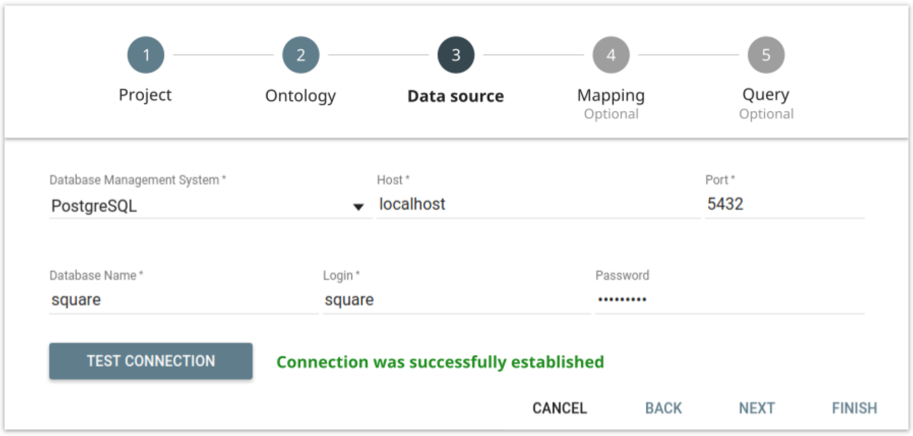
Name *
Demo project

Description
Project demonstrating the use of the SQuaRE application

Default IRI Template
http://data.example.com/{table}/{value}

Fig. 1: Wizard card related to project configuration.

It includes the database host location, a port number, credentials etc. After providing all necessary values the database connection is verified. In case of any errors the user needs to correct them before moving on. The card containing database configuration is shown in Fig. 2.

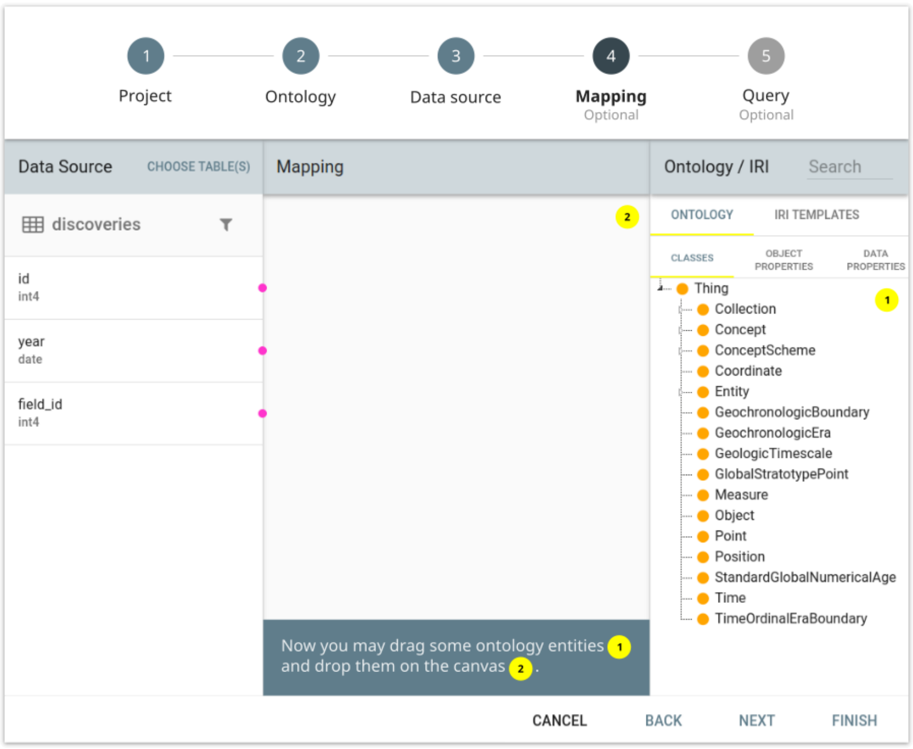


Database Management System *	Host *	Port *
PostgreSQL	localhost	5432
Database Name *	Login *	Password
square	square	•••••••••

Connection was successfully established

Fig. 2: Wizard card related to database configuration.

The next card is dedicated to mappings. During the mapping process the user is assisted by many supportive descriptions along with indications of what should be done next (shown in Fig. 3). At the center bottom part of the screen the suggestion of what to do next is located. The suggestion text may contain references to some parts of the screen in the form of yellow circles with numbers.



Now you may drag some ontology entities 1 and drop them on the canvas 2.

Fig. 3: Mapping card containing helpful tooltips.

The same yellow circles are located somewhere in the screen. In Fig. 3 the upper right or upper left corner contain such circles. In this tab the user may create a mapping or a set of mappings, save them and go to the next tab to verify them by creating a SPARQL query.

The final card provides an interface to query a virtual graph (formed on the basis of previously created mappings) using SPARQL. The last two cards are optional which means that the wizard may be finished earlier.

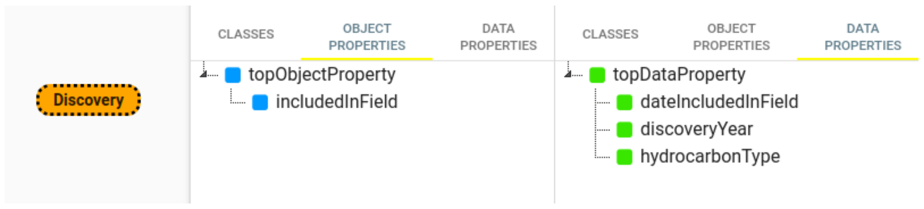
At every state the user may switch to a previous card (besides the first one). However, the next card may be chosen only when the present one is validated.

The presented first steps wizard should be perceived as an additional feature in SQuaRE in contrast to the main mapping and querying functionality that does not contain a wizard-related user interface components (like a top bar which may occupy the working area). However, we think that this may improve users experience in case of using the SQuaRE framework.

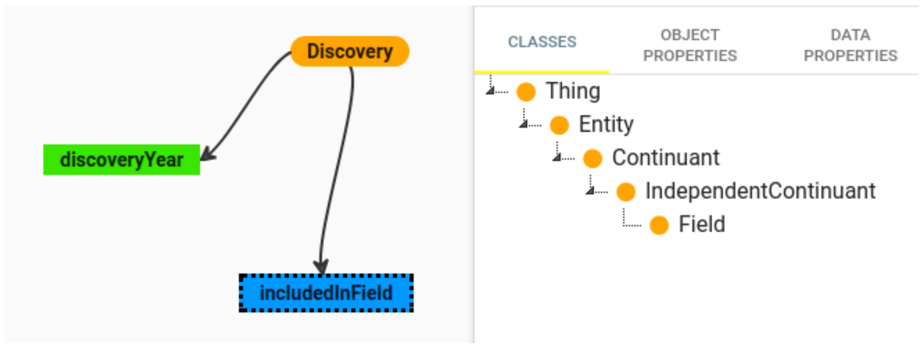
4.2 Utilization of Domain and Range Axioms

One of the most boring and time consuming task during mappings creation may be finding proper ontology entities (e.g. what properties can be connected with an already mapped OWL class) corresponding to the already mapped entity or a selected database column. So far, SQuaRE allows a user to search by a keyword through available ontology entities. However, we think that taking into account domain and range axioms would be an another improvement. Especially, during the evaluation users pointed out it is hard to find appropriate connections between ontology entities.

As a result the new version of SQuaRE allows a user to filter available ontology entities using domain and range axioms. The filtering is executed when the user selects an ontology entity dropped earlier into the mapping canvas. The selected entity is visually highlighted and lists of available classes and properties are narrowed down to only those that fulfill domain and range axioms (examples are shown in Fig. 4(a) and (b)).



(a) Properties filtered by domain axioms



(b) Classes filtered by property's range axioms

Fig. 4: Ontology entities filtered by domain or range axioms.

Thus, when a class is selected a filter is applied to datatype and object properties which have that class in its domain set. However, when a datatype or object property is selected and it is not connected to any other entity on the canvas the classes tree is narrowed to only those that are in the selected property's domain set.

The similar approach is applied when a user selects an object property that is linked to a class. According to the existing connection (subject-property or property-object) appropriate filtering is applied. In subject-property case the classes tree contains only classes in the property's range set. In the opposite property-object case the classes tree contains only classes in the property's domain set.

Datatype properties may be also linked to a column from a database table. If such a property is selected columns that do not fulfill data range axioms describing permitted data types are greyed out on the database columns list.

Furthermore, any link between ontology entities which is incorrect from the point of view of domain and range axioms is visually marked by a suitable icon placed near the incorrect link (as shown in Fig. 5). In case when errors are connected with domain axioms the letter D occurs near the icon. Otherwise, if range axioms do not contain the mapped class or its superclasses the letter R will appear. In Fig. 5 both cases are presented.

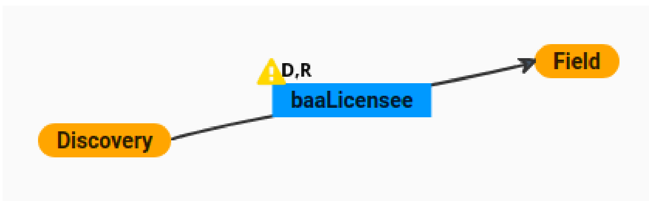


Fig. 5: Connected entities are not in the property's domain and range sets.

The utilization of domain and range axioms uses a filtering method which is executed when an ontology is loaded into SQuaRE. The method works in the following way:

– For each OWL class all object and datatype properties are searched for the occurrence of the class in domain and range axioms. When a property contains the class in its domain or range field the property is added to a list of valid properties for this class. Moreover, each superproperty of each valid property is also added to the list.
– For each object property all classes in its domain and range fields are added to lists of valid classes for this property: one list for domain classes and one list for range classes, respectively. Moreover, superclasses of those classes are also added to the list.
– For each datatype property all classes in its domain field are added to a list of valid domain classes for this property. Once again, superclasses of those classes are also added to the list. However, for the range field all datatypes are added to a list of valid datatypes.
– The lists are used every time when a user wants to create a mapping. SQuaRE analyses the currently selected element on the canvas and displays available classes or properties. However, if necessary, the filtering method can be disabled.

We are convinced that the proposed utilization of domain and range axioms may significantly speed up mappings creation because users will obtain a way of getting the most appropriate classes and properties that correspond to the already mapped ontology's entity. However, it is worth noting that the described filtering may only work when an applied ontology makes use of domain and/or range restrictions.

4.3 Conditional Nodes

Last but not least change has been also dictated by the evaluation which we have performed. The problem has arisen when there was a need to filter data, stored in a database, by some condition. Formerly, SQuaRE allowed to filter a whole chosen table by providing SQL *WHERE* clause. It turned out to be insufficient. Users would prefer to add conditions to particular mapping parts. Therefore, we considered to add the ability to enrich mapping nodes, especially ontology entities with conditions related to data stored in a database.

An exemplary mapping containing those conditional nodes, marked by the white funnel icon inside the red circle, is shown in Fig. 6. When the mouse cursor

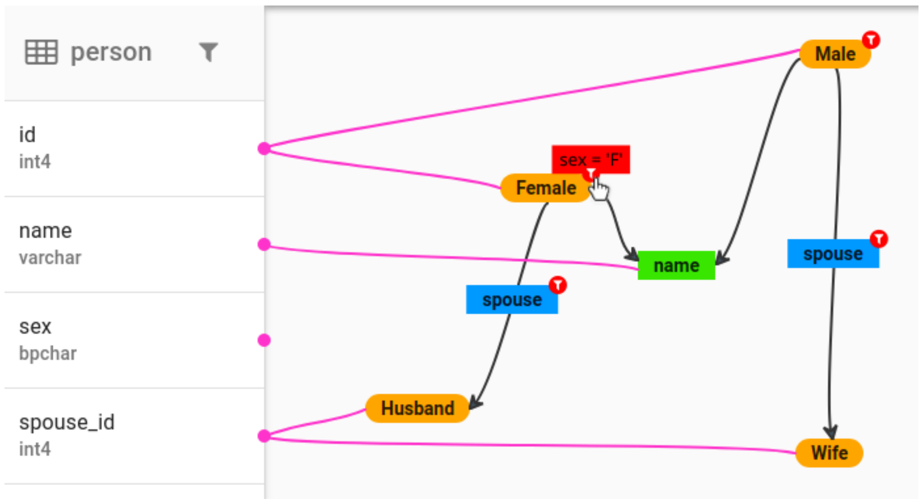


Fig. 6: Mapping with conditional nodes

hovers over any of red circles then a label containing the conditional expression appears. In the example shown in Fig. 6 an instance of the relationship *is-a Female* will be created only for rows which value of *sex* column is equal to *'F'*. Similarly, *Male* class has condition *sex = 'M'* and only rows with value *'M'* in the *sex* column will produce instances of the relationship *is-a Male*. Both *Female* and *Male* have *name* property unconditionally. However, the *spouse* property is taken into account only when *spouse_id is not null*.

Those conditions allow filtering on any step of the mapping process. This possibility improves user capabilities but also increases the complexity of generated R2RML mappings.

5 Summary and Future Work

In this paper we presented the latest progress of SQuaRE's development. We described an evaluation of the tool as well as new features that were suggested by users in the conducted experiment. Those features include: a special wizard to start working with SQuaRE, utilization of domain and range axioms in mappings creation as well as the addition of conditional nodes to the current visual representation of R2RML mappings. Moreover, we fixed some minor issues regarding ontology loading, custom IRI templates and other implementation bugs.

In the next version of SQuaRE we plan to finalize a graph-based method for creating and executing SPARQL queries as well as to make the tool available online.

Another feature that we want to develop is to provide support for the VOWL[13] notation. Therefore, switchable notations will be useful for users familiarized with different visual languages.

Moreover, the long term plans are to support the RML language [4]. As a result we will be able to map different data sources like CSV, JSON and others.

Furthermore, we are going to prepare another evaluation in which we compare SQuaRE with other tools in the aspect of creating and managing R2RML mappings and SPARQL queries.

Acknowledgments. The work presented in this paper was supported by 04/45/DSMK/0174 project.

References

1. Michał Blinkiewicz and Jarosław Bąk. Square: A visual approach for ontology-based data access. In Yuan-Fang Li, Wei Hu, Jin Song Dong, Grigoris Antoniou, Zhe Wang, Jun Sun, and Yang Liu, editors, *Semantic Technology: 6th Joint International Conference, JIST 2016, Singapore, Singapore, November 2-4, 2016, Revised Selected Papers*, pages 47–55, Cham, 2016. Springer International Publishing.
2. Michal Blinkiewicz and Jaroslaw Bak. Square: A visual support for OBDA approach. In *Proceedings of the Second International Workshop on Visualization and Interaction for Ontologies and Linked Data co-located with the 15th International Semantic Web Conference, VOILA@ISWC 2016, Kobe, Japan, October 17, 2016.*, pages 41–53, 2016.
3. Souripriya Das, Richard Cyganiak, and Seema Sundara. R2RML: RDB to RDF mapping language. W3C recommendation, W3C, September 2012. https://www.w3.org/TR/r2rml/.

[13] `http://vowl.visualdataweb.org/v2/`

4. Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. RML: a generic language for integrated RDF mappings of heterogeneous data. In *Proceedings of the 7th Workshop on Linked Data on the Web*, April 2014.
5. Pieter Heyvaert, Anastasia Dimou, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, Erik Mannens, and Rik Van de Walle. Rmleditor: A graph-based mapping editor for linked data mappings. In Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange, editors, *The Semantic Web. Latest Advances and New Domains: 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 – June 2, 2016, Proceedings*, pages 709–723, Cham, 2016. Springer International Publishing.
6. Ian Horrocks, Peter F. Patel-schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A semantic web rule language combining OWL and RuleML. 2004. Accessed: 04/04/2013.
7. Craig A. Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan, and Parag Mallick. *Semi-automatically Mapping Structured Sources into the Semantic Web*, pages 375–390. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
8. Luís Eufrasio T. Neto, Vânia Maria P. Vidal, Marco A. Casanova, and José Maria Monteiro. *R2RML by Assertion: A Semi-automatic Tool for Generating Customised R2RML Mappings*, pages 248–252. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
9. Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. *Linking Data to Ontologies*, pages 133–173. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
10. Álvaro Siciliaa, German Nemirovskib, and Andreas Nolleb. Map-on: A web-based editor for visual ontology mapping. *Semantic Web Journal*, (Preprint):1–12.