Using a Block Metaphor for Representing R2RML Mappings

Ademar Crotti Junior, Christophe Debruyne, Declan O'Sullivan

ADAPT Centre, Trinity College Dublin, Dublin 2, Ireland
`{crottija,debruync,declan.osullivan}@scss.tcd.ie`

Abstract. R2RML is a W3C Recommendation that provides for the declaration of mappings to generate RDF datasets from relational databases. One issue that hampers its adoption is the manual effort needed in the creation and maintenance of such mappings. To tackle this problem, various initiatives have started to emerge. One of the directions is to investigate how different representations can facilitate the creation and maintenance of such mappings for a wider set of stakeholders. In prior work, we proposed a visual representation based on the block metaphor for R2RML mappings that is compliant with this specification. This representation has been integrated within a tool for creating and managing R2RML mappings. In this paper, we report on a user study to evaluate the proposed visual representation considering stakeholders with different background knowledge. Preliminary findings indicate that participants were able to create accurate mappings and that the visual representation achieves good results in standard usability evaluations.

Keywords. R2RML; Visual Representation; Data Mapping.

1 Introduction

A huge part of the Linked Data web is achieved by converting non-RDF resources into RDF. This conversion process is typically called *uplift*. For relational databases, one can rely on the W3C Recommendation R2RML [4] for creating mappings from relational databases into RDF datasets. Though useful, some problems with its adoption can be observed. Firstly, R2RML mappings are stored as RDF. We argue that writing any RDF graph by hand can be troublesome and prone to error. Secondly, the R2RML mapping language has a steep learning curve, where the creation of mappings can be time consuming, and syntactically heavy in various cases [15].

Initiatives have emerged to address these problems and make the technology more accessible ranging from step-by-step wizards [17] and plugins [15] to visual *graph* representations [8, 11]. These approaches, however, focus on Knowledge Engineers, not being as intuitive for other types of users. We will discuss the advantages and disadvantages of these initiatives in Section 2.

In previous work, we have proposed a visual representation for mappings, Juma [9], and applied it to the R2RML mapping language. This representation is based on the block (or jigsaw) metaphor that has become popular with visual programming languages – where it is called the block paradigm – such as Scratch[1]. In this metaphor, concepts are represented as blocks that can only be combined with other compatible blocks. In this sense, the block metaphor targets different types of users, allowing them to focus on the logic instead of the language's syntax. In addition, it has been used successfully in other domains, such as programming [7].

In this paper, we present a user study that evaluates our visual representation of mappings applied to the R2RML mapping language considering different types of stakeholders. Our intuition is that the visual representation would be useful for non-experts and experts in Linked Data publishing from relational databases. This user study evaluated the mappings created by participants and the usability of the visual representation through a standard usability test.

The remainder of this paper is structured as follows: Section 2 reviews the related work. In Section 3 we discuss the R2RML mapping language. Section 4 describes Juma. Section 5 presents a user study used to evaluate our visual representation. Results and analysis are presented in Section 6. Section 7 concludes the paper.

2 Related Work

In this section, we discuss the state-of-the-art in mapping representation for R2RML. We have characterized these into applications with or without visual representations. Applications without visual representations have an interface that guides users in the creation of R2RML mappings. These tools, however, do not provide any visual representation for mappings. Applications with a visual representation offer a graphical view of the mapping.

No visual representation. The fluidOps editor [17] is a web-based application that relies on a step-by-step workflow. Each step focuses on the creation of one part of the mapping. The mapping is only available at the end of this process and changes in the mapping restart the workflow. Furthermore, complex mappings used to interlink subjects are not supported through the interface. To create these, one needs to define a new resource with an existing URI. In [15], an extension of this editor was proposed. In this extension, the mapping process starts based on an existing ontology. The next step is to define their relations with the source data. In this sense, changes do not restart the workflow. However, complex mappings are still not supported. OntopPro[2] [16] is a Protégé [14] plugin that uses a proprietary mapping language internally to create mappings. The tool, however, allows users to import and export mappings in R2RML. The Virtuoso Universal Server[3] has an extension where data can be converted into RDF by creating R2RML mappings or using a wizard that guides users in the creation such mappings, similar to fluidOps. R2RML By Assertion (RBA) [13] uses a tree table structure to represent ontologies and RDF vocabularies, side by side with the input data, also as a tree. In this sense, one needs to match classes and properties to attributes. The assertion of these matches generates an R2RML mapping. The adoption of these tools by non-experts is limited, since they do not provide a visual representation for mappings.

[1] `https://scratch.mit.edu/`, accessed in August 2017.
[2] `http://ontop.inf.unibz.it`, accessed in August 2017.
[3] `https://virtuoso.openlinksw.com`, accessed in August 2017.

Visual representation. Karma [10] is a web-based application where the data is loaded before it can be mapped into RDF. The ontologies used during the mapping process are represented in a tree structure and the data as a table. A graph visualization of the mapping is available. The creation of mappings using Karma can be troublesome because of the data centric approach, where every input is shown in a different table. This makes the interlinking between tables unnecessarily complex. Lembo et al. [11] uses a graph representation for R2RML mappings. However, the creation and/or editing of mappings are undertaken through text editing, which make the mapping process prone to errors. RMLeditor [8] has support for R2RML and RML [6] mapping languages. RML is an extension of R2RML to support multiple data formats such as CSV, XML and so on. The RMLeditor also uses a graph representation for the mapping. The input data and RDF output are shown as tables. MapOn [18] is yet another graph representation tool for R2RML mappings. MapOn's visual representation does not support complex mappings – having the same issue as fluidOps editor. SQuaRE [1] is a tool that provides a visual environment for the creation of R2RML mappings. This tool also uses a graph visual representation for mappings. In a first step, users need to select the tables that are going to be mapped. Ontologies and RDF vocabularies that will be used in the mapping process are shown as trees.

Visual representations are especially helpful to non-expert users. However these usually focus on Knowledge Engineers, representing uplift mappings as graphs, since the RDF data model is itself one. This representation, nonetheless, is not as intuitive for other types of user. As mentioned before, the block metaphor has been successfully used in other domains to attract a range of stakeholders (e.g. programming). Furthermore, in [2], the authors proposed the use of this metaphor for SPARQL queries. In Section 4, we describe a method, that uses the block metaphor, and how we have applied it to the R2RML mapping language.

3 R2RML

In this section, we briefly explain the main concepts related to the W3C Recommendation R2RML for the purpose of this paper. For more information, we refer the reader to the W3C Recommendation [4]. Each R2RML mapping definition consists of one or more *triples maps*. Looking at **Listing 1**, we can see that a triples map has (1) one *logical table*, (2) one *subject map* and (3) zero or more *predicate object maps*, where:

1. **Logical Table**: the table or a SQL query from which RDF will be generated.
2. **Subject Map**: subject maps define the subjects of the RDF triples. These subjects can be IRIs' or blank nodes. You may also specify zero or more URI class types.
3. **Predicate Object Map**: each predicate object map defines the predicates, using predicate maps, and objects, using object maps, of the RDF triples. Each predicate object map must have at least one predicate map and one object map. Predicates must be valid IRI's. Objects can be IRI's, blank nodes or literals. For literal values, it is possible to define a data type or a language. You may link triples maps using parent triples map. A parent triples map can have zero or more join conditions.

```
<#TripleMap1>
  rr:logicalTable [ rr:tableName "students"; ];
  rr:subjectMap [
    rr:template "http://example.org/student/{id}"; rr:class foaf:Person;];
  rr:predicateObjectMap [
    rr:predicate foaf:name; rr:objectMap [ rr:column "name"; ]; ];.
```

Listing 1. R2RML mapping definition

In this mapping, we map the table (or view) “students”. A triples map defines subjects to have the IRI `http://example.org/student/{id}`. We also declare subjects to be instances of the class `foaf:Person`. A predicate object map relates the subjects with the predicate `foaf:name` to values in the column “name” .

4 Juma: Jigsaw Puzzles for Representing Mappings

In previous work we have presented a method called **J**igsaw p**u**zzles for representing **ma**ppings, Juma [9], applied to the R2RML mapping language. As outlined in Section 1, there are a number of issues with how R2RML mappings are generated. For example, creating mappings by hand is a time consuming process, being syntactically heavy even for simple mappings. Moreover, it has a steep learning curve on various aspects, such as the R2RML vocabulary and algorithm, the RDF data model and others. Juma focus on facilitating the creation, management, and understandability of mappings, by making the technology available to a wider set of stakeholders. A tool that applies the Juma method to the R2RML mapping language is also available[4].

Our implementation uses Google’s Blockly API[5]. Google Blockly is a visual programming language that uses blocks to facilitate code creation. These blocks are shaped like jigsaw puzzle pieces that show how the language works by abstracting the language’s syntax. Furthermore, Blockly has been successfully used in many projects, such as code.org’s[6] introduction courses to Computer Science.

In our tool, each block has been designed to represent an R2RML statement that automatically generates a correspondent R2RML construct. The visual representation abstracts the R2RML vocabulary’s syntax for users. The visual representation also guides users in the creation of valid mappings by highlighting and only allowing the connection of blocks that would create a valid mapping. The menu options provide one with all possibilities within the R2RML mapping language, from which new blocks may be dragged into the workspace. The menu is also defined using a tree structure, so as to gives users a hint of how the blocks connect to each other. The visual representation also uses colors to identity the type of structure that is being created. For example, all constructs for subjects are in green, predicates are blue and so on. Furthermore, the template menu option shows a complete triple map and a predicate object map, with the default R2RML options, that can be used to bootstrap mapping creation.

For each mapping there are 3 tabs. In the first tab, Mapping, we show the menu and the visual representation. In Configuration, one can define the properties of the configuration file. The configuration file is used as input to an R2RML processor together with the R2RML mapping file. In the R2RML-Mapping tab the user can see the actual R2RML mapping generated from the visual representation. **Fig. 1** shows the R2RML mapping from **Listing 1** represented in Juma.

[4] `https://www.scss.tcd.ie/~crottija/juma/` accessed in August 2017.
[5] `https://developers.google.com/blockly/` accessed in August 2017.
[6] `https://code.org/about` accessed in August 2017.

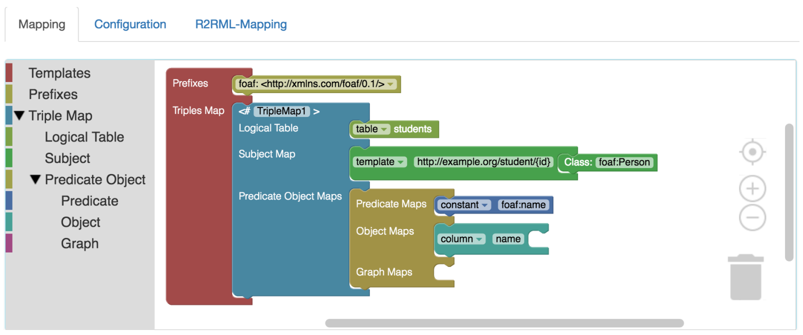

Fig. 1. Juma visual representation of an R2RML mapping

5 User Study

This section describes the user study performed to evaluate Juma applied to the R2RML mapping language.

5.1 Participants

Participants were recruited based on their background knowledge. These groups were chosen as to evaluate how different types of stakeholders engage with our visual representation in the creation of uplift mappings. We decided to focus on users that are likely to need to publish Linked Data datasets. Our intuition is that the visual representation would be useful for non-experts and experts in Linked Data publishing from relational databases. We consider that these users would be one of three types: non Semantic Web experts with background in computer science but no knowledge of R2RML; Semantic Web experts with no knowledge of R2RML; and Semantic Web experts with experience of the R2RML mapping language for Linked Data publishing. The groups were defined as follows:

- **Web Developers (WD)**: these participants had background in computer science with experience on web development, but not in Semantic Web technologies or on the R2RML mapping language;
- **Knowledge Engineers (KE)**: these participants had knowledge in Semantic Web technologies such as RDF, OWL and so on. Furthermore, these users would also not be familiar with R2RML;
- **R2RML familiar (RF)**: participants in this group had experience with R2RML.

5.2 Procedure

The study was structured in four parts:

1. **Pre-task questionnaire**: participants were asked to evaluate their knowledge in relevant fields (Semantic Web technologies and more specifically about the R2RML mapping language). Participants evaluated their familiarity using a 7-point Likert scale from 1 (strongly agree) to 7 (strongly disagree). If the participant's response was in the range 1 to 4, they were considered familiar with the technology.

2. **Technical debriefing**: after filling out the pre-questionnaire, participants had the opportunity to watch videos about RDF, R2RML and our tool. If they felt comfortable with these technologies, they could skip the videos. A presentation explaining how the tool works was also available to be used during the experiment[7]. They could watch the videos and use the presentation during the task.
3. **Mapping task**: in the main part of this study, we asked participants to create one R2RML mapping using the tool. The task is described in next section. Participants could ask questions to clarify any doubts about the experiment. In addition, they were advised to use the material provided. Any help needed to solve the task was recorded.
4. **Post-task questionnaire**: after completion of the task, we asked participants to fill out a questionnaire about the use of the tool. At this stage, we have also conducted an informal interview with participants.

5.3 Task

This user study was built on top of the Microsoft Access 2010 Northwind sample database for MySQL[8].

Participants were asked to create one R2RML mapping in three parts. For each part, a sample RDF output was shown to participants. In addition, they could run the mapping and compare the output from the tool and the sample provided. In this sense, for the purpose of this experiment, we integrated an R2RML processor [5] to the tool. Every time a mapping was executed, the current mapping and output were saved. The table diagram was shown, together with instructions, in each part of the experiment. The task was divided in three parts:

- **Part 1**: in this part, participants had to define a mapping with one subject per row of the table *employees*. The subject URI for the triples should be `http://data.example.org/employee/{id}`. These subject should also have the URI type class `foaf:Person` from the FOAF[9] vocabulary. The mapping definition should also create, for these subjects, the predicate `foaf:givenName` with object from the column *first_name*. The predicate `foaf:familyName` with object from the column *last_name*. Finally, the predicate `foaf:name` should have the concatenation of the columns *last_name* and *first_*name separated by comma as object;
- **Part 2**: in the same mapping, participants were asked to define another subject from the table *employees*. The subject URI should be `http://data.example.org/city/{city}`. These subjects should have the URI type class `foaf:Spatial_Thing`. The mapping should generate the predicate `rdfs:label`, from the RDFS[10] vocabulary, with object from the column *city* for each subject;

[7] Experiment material, expected mapping, expected output and questionnaires are available at `https://www.scss.tcd.ie/~crottija/juma/material/` .
[8] `https://github.com/dalers/mywind` accessed in August 2017.
[9] `http://xmlns.com/foaf/0.1/`
[10] `http://www.w3.org/2000/01/rdf-schema#`

- **Part 3**: in the last part, participants were asked to interlink the subject from Part 1 with the subject from Part 2 using the predicate `foaf:based_near`.

The task involved the use of different R2RML constructs, such as parent triples maps and others, as to explore their visual representation within the tool. Some elements of the task could be achieved in different ways. For example, since not all attributes are mapped, participants could use a SQL query instead of mapping the whole table, especially for Part 2, which maps only one column. Concatenating could be implemented using a template construct or an SQL query. The template construct would be the expected solution to concatenating. Part 3 of the experiment asked participants to relate the subjects created in Part 1 and Part 2. This could be achieved with a parent triples map or a template construct, since this value comes from the same table. For Part 3, parent triples map would be the expected solution. **Table 1** shows the challenges associated to the task.

Table 1. Challenges associated to the task

Part	Short description	Challenge/Non-trivial aspects
#1	Map and type entities to a class with three attributes	One attribute mapping is the concatenation of other two attributes. This requires mapping using a SQL query or the use of a template construct.
#2	Map and type another entity with one attribute	Map cities as a second entity from the same table using another triples map.
#3	Linking	Linking subjects created in Parts 1 and 2. This requires the use of a SQL query with a SQL join, or the R2RML parent triples map construct.

6 Results and Analysis

The study was executed with 15 participants, 5 in each group, thus 10 participants have no knowledge of R2RML, as defined in Section 5.1. The experiment was executed individually with each participant in a 13" MacBook Pro (2560 x 1600 resolution). This section will discuss the results and analysis arising from the experiment under two headings: the execution of the task and the usability of the visual representation.

6.1 Task Execution

In this section we show the data collected during the execution of the task. These include the mappings created, the time taken to execute the task, and any help needed by participants. **Table 2** shows accuracy and time by participant.

- Accuracy: the mappings created by the participants were executed and compared against an expected output. We calculate accuracy by the number of correct triples in the RDF output. In this sense, a correct mapping would have 53 triples over 9 records. Part 1 has 36 triples. Part 2 has 8 triples and Part 3, 9 triples. The total number of correct triples is indicated in the header of the table. We used Jena API to compare the RDF models and count the number of triples. Syntactic mistakes, such as missing slashes and extra/missing spaces, were not considered errors;

- Time: the time taken to execute the task in minutes for each part and for the task. We recorded the time manually as participants indicated that they have finished each part of the task;
- Help: during the experiment, some participants needed help in the execution of the task. In Part 1, 2 Web developers and 1 Knowledge engineer needed help. The only help needed in Part 1 was on concatenating two columns. This can be done using an R2RML template construct or by mapping using an SQL query. In Part 2, none of the participants needed help. In Part 3, 4 Web developers and 2 Knowledge engineers needed help to interlink the subjects created in Part 1 and Part 2.

Table 2. Experiment results by participant

Group	**#**	**Part 1 (36)**			**Part 2 (8)**			**Part 3 (9)**			**Total (53)**		
		Total	**%**	**Time**	**Total**	**%**	**Time**	**Total**	**%**	**Time**	**Total**	**%**	**Time**
WD	1	36	100	25	8	100	5	9	100	10	53	100	40
	2	36	100	18	8	100	10	0	0	10	44	83.1	38
	3	36	100	16	8	100	6	9	100	7	53	100	29
	4	27	75	25	8	100	7	0	0	8	40	75.5	40
	5	36	100	17	8	100	7	9	100	10	53	100	34
KE	6	36	100	13	8	100	4	9	100	8	53	100	25
	7	36	100	6	8	100	3	9	100	3	53	100	12
	8	36	100	21	8	100	9	0	0	11	44	83.1	41
	9	36	100	18	8	100	7	9	100	11	53	100	36
	10	36	100	14	8	100	7	9	100	6	53	100	27
RF	11	36	100	15	8	100	3	9	100	5	53	100	23
	12	36	100	9	8	100	4	9	100	7	53	100	20
	13	36	100	17	8	100	6	9	100	5	53	100	28
	14	36	100	13	8	100	4	9	100	10	53	100	27
	15	36	100	5	8	100	4	9	100	4	53	100	13

Table 3. Experiment results by group

Group	**Part 1 (36)**			**Part 2 (8)**			**Part 3 (9)**			**Total (53)**		
	Total	**%**	**Time**	**Total**	**%**	**Time**	**Total**	**%**	**Time**	**Total**	**%**	**Time**
WD	34.2	95	20	8	100	7	5.4	60	9	47.6	89.8	36
KE	36	100	14	8	100	6	7.2	80	7	51.2	96.6	27
RF	36	100	11	8	100	4	9	100	6	53	100	21
All	35.4	98.3	15	8	100	6	7.2	80	7	50.6	95.5	28

6.1.1 Task Execution Analysis

The mapping accuracy between all participants and within their own groups was high. The R2RML familiar group had the highest score. Moreover, participants from the

R2RML familiar group did not need help to complete the task. This may be explained by the naming conventions used in the tool, following R2RML's mapping language.

Considering the time taken to execute the task, Web developers spent significantly more time than the other groups. The biggest difference is in the execution of Part 1, which indicates a higher learning curve for participants that are not familiar with Semantic Web technologies.

The most common help was on how to interlink triples maps with the use of the parent triples map construct. Participants were able to create the R2RML construct using the tool but had difficulties defining the parent and child values for the join condition, which requires knowledge on SQL joins. In this sense, the tool offered some support for the creation of these constructs, but some participants struggled with the conceptualization of it.

6.2 PSSUQ Questionnaire

Participants were asked to fill in the Post-Study System Usability Questionnaire (PSSUQ) questionnaire [12] after finishing the task. PSSUQ was designed to assess overall satisfaction with system usability and was chosen over other questionnaires, like the System Usability Scale (SUS) [3], as it explicitly assesses other aspects of a system beyond usability, such as usefulness. Furthermore, PSSUQ was designed for scenario-based usability studies, where some questions are more targeted, such as *I was able to complete the tasks and scenarios quickly using this system*. PSSUQ also has high reliability and it allows for more nuanced responses by using a 1-7-point Likert scale.

The PSSUQ is a 19-item questionnaire with a 7-point Likert scale from 1 (strongly agree) to 7 (strongly disagree), with a not applicable option (N/A) and a comment area per question. PSSUQ gives scores in four categories: System Usefulness (SysUse), Information Quality (InfoQua), Interface Quality (IntQua) and Overall [12]. **Table 4** shows the average scores per group (all responses and scores by participant are also available[11]).

Table 4. PSSUQ average scores

PSSUQ	WD	KE	RF	All
SysUse	2.6	3.2	1.8	2.5
InfoQual	2.4	3.6	2.3	2.8
IntQual	1.7	2.4	1.5	1.9
Overall	2.3	3.2	1.9	2.5

[11] `https://www.scss.tcd.ie/~crottija/juma/experiment-data/`

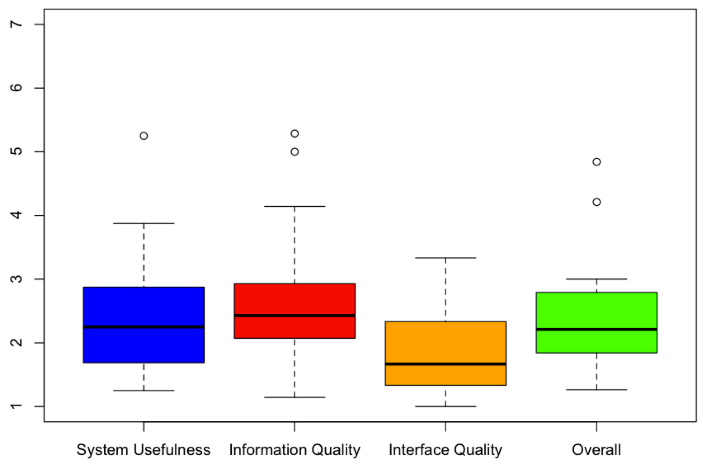

Fig. 2. Boxplot of PSSUQs' responses

Fig. 2 shows a boxplot of all PSSUQs' responses. We can see that we have one outlier in System Usefulness and two in Information Quality and Overall scores. These outliers are 2 participants from the Knowledge Engineer group. We also used the Welch Two Sample t-test between the groups in every aspect of the PSSUQ questionnaire to check for any significant differences. **Table 5** shows the p-values for this test.

Table 5. PSSUQ t-test p-values between groups

T-Test	WD vs. KE	WD vs. RF	KE vs. RF
SysUse	0.3999	0.1489	0.07196
InfoQual	0.1516	0.8489	0.1652
IntQual	0.1392	0.76	0.1456
Overall	0.2041	0.2934	0.0824

All p-values are above 0.05. This suggests that the differences between the groups are not significant. Due to the sample size, we have also applied the Friedman non-parametric test, from which the same conclusion was drawn.

Most participants did not leave any comments. Three of the participants suggested tool improvements such as showing the relational database diagram, and minimizing the typing required by users within the tool (e.g. auto-completion).

6.2.1 PSSUQ Questionnaire Analysis

We can see that the best average scores are in the R2RML familiar group, as it is shown in **Table 4**. As mentioned before, this may be explained the naming conventions adopted by the tool.

The scores in the Web developers group are more similar to the R2RML familiar group than to the Knowledge Engineer group. This may be explained by the group's expectations. In an informal interview made with participants after finishing the task, some mentioned that R2RML is complex and that abstractions in such technologies help with its adoption, especially for non-experts. Knowledge Engineers commented that they expected more information within the visual representation, while participants in the R2RML familiar group were comfortable with the concepts used.

It was also mentioned that the tool works as a template, as one does not need to know all classes and properties of the R2RML vocabulary to create mappings. Moreover, that the system quickly shows the possible constructs, only allowing blocks to connect with others as to create a valid R2RML mapping. In this group, one participant showed concern about the visual representation for large R2RML mappings. The tool offers ways of focusing on smaller parts of the mapping, by collapsing and/or expanding blocks. However, this needs to be evaluated.

In general, the tool received good usability results, within each group and overall, as can be seen in the average scores shown in **Table 4**. In this table, we can see that interface quality had the best score (1.5); followed by system usefulness and overall (1.9 each); and finally information quality (2.8). In addition, we applied statistical tests to compare the PSSUQ scores between the different groups. As can be seen in **Table 5**, these differences were not statistically significant. Furthermore, the p-values nearest to the threshold (0.05) involved the outliers identified in **Fig. 2**.

7 Conclusions and Future Work

In this paper, we have presented a user experiment to evaluate Juma, a visual representation based on the block metaphor applied to the R2RML mapping language.

We have shown that the visual representation was beneficial in the creation of accurate R2RML mappings for participants with different background knowledge. Furthermore, the most common help needed to complete the task was in the use of parent triples maps. The creation of this construct within the tool was considered difficult for participants with no previous knowledge on R2RML. We have also used a standard usability test to validate the visual representation. The group familiar with R2RML had the highest scores in the usability test, followed by Web developers and Knowledge engineers. As mentioned before in our analysis, this may be explained by the expectations of these users. The usability test also indicated that information quality within the tool was deficient for some users. We believe that improvements in this characteristic will have a positive effect in other usability aspects.

Future work includes incorporating transformation functions, as proposed in [5], to the visual representation, and applying the block metaphor to other mapping languages. We also intend to assess and compare the cognitive load of our approach to others. Though our approach generates R2RML mappings that are compliant, we have chosen not to support the reuse of resources across different parts of a mapping; e.g., the reuse of an object map in different predicate object maps. Because of this, we cannot yet load arbitrary R2RML mappings in our tool. Future work will thus also look into rewriting R2RML mappings for inclusion in our tool or support for the reuse of resources in different places of our representation.

Acknowledgements. This paper was supported by CNPQ, National Counsel of Technological and Scientific Development – Brazil and by the Science Foundation Ireland (Grant 13/RC/2106) as part of the ADAPT Centre for Digital Content Technology (http://www.adaptcentre.ie/) at Trinity College Dublin.

References

1. Blinkiewicz, M., Bak, J.: SQuaRe: A visual support for OBDA approach. In: Proceedings of the Second International Workshop on Visualization and Interaction for Ontologies and Linked Data (VOILA@ISWC 2016).
2. Bottoni, P., Ceriani, M.: SPARQL playground: A block programming tool to experiment with SPARQL. In: Proceedings of the International Workshop on Visualizations and User Interfaces for Ontologies and Linked Data (VOILA@ISWC 2015).
3. Brooke, J.: SUS - A quick and dirty usability scale. Usability evaluation in industry, 189-194 (1996).
4. Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF Mapping Language. (2012) https://www.w3.org/TR/r2rml/.
5. Debruyne, C., O'Sullivan, D.: R2RML-F: Towards Sharing and Executing Domain Logic in R2RML Mappings. In: Workshop on Linked Data on the Web (LDOW 2016).
6. 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: Workshop on Linked Data on the Web (LDOW 2014).
7. Fraser, N.: Google Blockly - a visual programming editor. (2014). URL: https://developers.google.com/blockly. Accessed Aug 2017.
8. Heyvaert, P., Dimou, A., Herregodts, A.L., Verborgh, R., Schuurman, D., Mannens, E., Van de Walle, R.: RMLEditor: A Graph-based Mapping Editor for Linked Data Mappings. In: The Semantic Web - Latest Advances and New Domains (ESWC 2016).
9. Junior, A. C., Debruyne, C., O'Sullivan, D.: Juma: an Editor that Uses a Block Metaphor to Facilitate the Creation and Editing of R2RML Mappings. In: The Semantic Web - Latest Advances and New Domains (ESWC 2017).
10. Knoblock, C., Szekely, P., Ambite, J.L., Goel, A., Gupta, S., Lerman, K., Muslea, M., Taheriyan, M., Mallick, P.: Semi-Automatically Mapping Structured Sources into the Semantic Web. In: 9th Extended Semantic Web Conference (ESWC 2012).
11. Lembo, D., Rosati, R., Ruzzi, M., Savo, D.F., Tocci, E.: Visualization and management of mappings in ontology-based data access (progress report). In: Informal Proceedings of the 27th International Workshop on Description Logics (2014).
12. Lewis, J. R.: Psychometric evaluation of the post-study system usability questionnaire: The PSSUQ. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting (Vol. 36, No. 16, pp. 1259-1260). SAGE Publications (1992).
13. Neto, L.E.T., Vidal, V.M.P., Casanova, M.A., Monteiro, J.M. R2RML by assertion: A semiautomatic tool for generating customised R2RML mappings. In: 10th The Semantic Web: Satellite Events (ESWC 2013).
14. Noy, N.F., Sintek, M., Decker, S., Crubézy, M., Fergerson, R.W., Musen, M.A.: Creating semantic web contents with Protégé-2000. IEEE Intell. Syst. 2, 60–71 (2001).
15. Pinkel, C., Binnig, C., Haase, P., Martin, C., Sengupta, K., Trame, J.: How to best find a partner? An evaluation of editing approaches to construct R2RML mappings. In: 11th European Semantic Web Conference (ESWC 2014).
16. Rodrıguez-Muro, M., Hardi, J., Calvanese, D.: Quest: efficient SPARQL-to-SQL for RDF and OWL. In: 11th International Semantic Web Conference on Posters and Demonstrations (ISWC 2012).
17. Sengupta, K., Haase, P., Schmidt, M., Hitzler, P.: Editing R2RML mappings made easy. In: 12th International Semantic Web Conference (ISWC 2013).
18. Siciliaa, Á., Nemirovskib, G., & Nolleb, A.: Map-on: A web-based editor for visual ontology mapping. Semantic Web Journal, (Preprint):1–12.