RDF Spreadsheet Editor: Get (G)rid of Your RDF Data Entry Problems

Markus Schröder[1], Christian Jilek[1], Jörn Hees[1], Sven Hertling[2], and Andreas Dengel[1,3]

[1] German Research Center for Artificial Intelligence (DFKI) GmbH
Trippstadter Straße 122, 67663 Kaiserslautern, Germany,
{markus.schroeder, christian.jilek, joern.hees, andreas.dengel}@dfki.de,
[2] Data and Web Science Group, University of Mannheim, Germany,
sven@informatik.uni-mannheim.de,
[3] Knowledge-Based Systems Group, Department of Computer Science,
University of Kaiserslautern, P.O. Box 3049, 67653 Kaiserslautern, Germany

Abstract. Spreadsheets are widely used by knowledge workers, especially in the industrial sector. Their methodology enables a well understood, easy and fast possibility to enter data. As filling out a spreadsheet is more accessible to common knowledge workers than defining RDF statements, in this paper, we present a tool which uses the spreadsheet metaphor to enable various kinds of users to easily create RDF data whether they are RDF experts or novices. Our approach aims at supporting users in manually filling a knowledge base with their expertise. This is usually a collaborative process involving a team of domain experts and knowledge engineers. To immediately see modelling consequences and updates by others, each entry is simultaneously transferred into triple statements.

Keywords: spreadsheet, RDF data entry, filling knowledge base

1 Introduction

Our experience from industry projects shows that knowledge workers are usually more skilled in dealing with spreadsheets than working with triple statements. In order to enable our partners to easily communicate their expertise in the form of semantic data, we created a tool which supports this knowledge transfer. The widely known spreadsheet metaphor is well suited for enabling all kinds of users to manually enter data. Such a modelling process typically involves a lot of collaboration and communication (expert-to-expert, expert-to-knowledge engineer, knowledge engineer-to-knowledge engineer). We therefore developed a *web-based* spreadsheet editor (i.e. hosted on the intranet – or the internet if confidentiality requirements permit this). To immediately see modelling consequences and updates by others, each entry into a cell of the spreadsheet is simultaneously transferred to RDF statements. In contrast to the fixed structure of a spreadsheet, the RDF graph additionally allows for running complex queries or dynamically restructuring the data for other issues and applications (for example by using SPARQL).

2 Related Work

Converting spreadsheets to RDF is not a novel idea, several approaches have been presented in the past. For most of them the user has to provide a mapping of how to transfer the spreadsheet data to RDF (e.g. RDF123 [2]), but the mappings needed for transformation are rather hard to define for inexperienced users. Some other approaches like Any23[4] also support semi- or fully automated conversions without the necessity of defining a mapping. However, the transformation steps are designed as a batch process and are not done simultaneously. Because of these facts, it is not easy to create RDF data with them, especially when working collaboratively or if non-experts are involved.

In contrast to taking filled spreadsheets as an input and converting them, the following approaches focus on supporting the user in entering and working with RDF data. Pohl [3] published *rdfedit*, a web-based tool to create RDF data. It could also be used by Semantic Web laymen using a subject-predicate-object table and the discontinued search engine Sindice. RDForms[5] enables users to enter data in forms, which are simultaneously transferred to RDF triples. Nevertheless, these forms have be defined in advance for each use case. Tripcel [1] also uses the spreadsheet metaphor to read and process RDF data. The author created an expression language which allows for defining RDF terms in a sheet to call functions on them.

Unlike our tool they still require considerable RDF knowledge in order to be used or configured. None of them uses the spreadsheet metaphor in a way that enables a fast data entry as we do. Note that our application primarily focuses on providing a convenient way – especially for non-experts – to create instance data. It thus allows for an easy manual RDF data ramp-up, a task none of the existing editors seems to be appropriate for. Although it allows for ontology modelling to some extend, it is not intended to compete with full-fledged ontology editors like the widely known Protégé[6]. Additionally, the emerging knowledge graph should be maintained using dedicated applications.

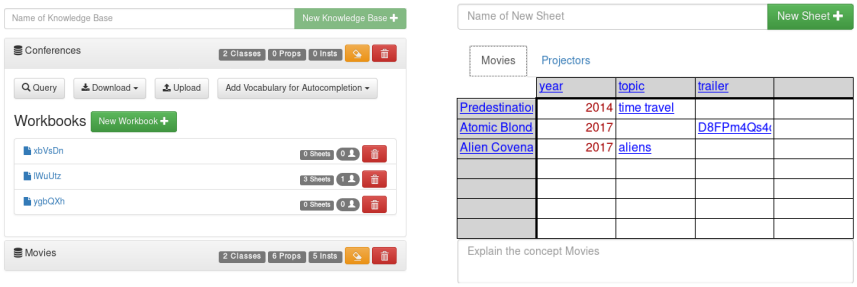
3 RDF Spreadsheet Editor

Our approach is a web-based spreadsheet editor, that simultaneously transfers spreadsheet inputs to RDF statements and also adds them to a knowledge base. In this process we use a simple and fixed *class per sheet* and *entity per row* mapping similar to csv2rdf[7] (which will be explained in the following). The main features are creating and manipulating RDF classes, properties, instances and assertions. Instead of providing a full-featured ontology editor, we primarily focus on creating instance data (ABox). Our tool supports the user by automatically inferring and creating domain and range statements as well as auto completion of resource labels, for example.

[4] `https://any23.apache.org`
[5] `http://rdforms.org`
[6] `http://protege.stanford.edu`
[7] `https://www.w3.org/TR/csv2rdf/`

For a first impression we kindly refer the reader to this project's website[8], which presents an online demo as well as a video. In the following we describe the main features of the system in more detail.

Using the **administration page** (Figure 1a) an expert can create and delete knowledge bases (KBs). The query button opens a new page to write SPARQL queries against the KB. Common vocabularies (like FOAF) can simply be added in order to reuse their classes and properties. Knowledge bases may contain one or more workbooks, which again contain the actual spreadsheets. A workbook's page link can be shared among members of a team in order to work collaboratively. The KBs can be imported and exported using typical triple data formats, whereas workbooks can be downloaded as (annotated) Excel files or JSON.



(a) Administration page to organize knowledge bases and workbooks

(b) Workbook page containing sheets where entries are transferred to RDF statements simultaneously

Fig. 1: Main pages of the RDF Spreadsheet Editor

By using the **spreadsheet metaphor** we enable users to work with semantic data in a familiar way. Figure 1b shows the workbook page where users create sheets and enter data. In general, for each new resource a random UUID is generated and serves as its URI while the user input serves as its label. Using a text area below, a comment for each focused resource can be stated, which is then presented when mouse hovering it. Adding a sheet creates a new RDFS class. The row header is used to create resources which are instances of the class. Entering a column header generates a new property having the sheet's class as its domain. A cell instantiates a new resource and links it to the one of the row header using the property given in the column header. This results in a triple statement. Forcing the application to create a literal instead of a resource is done by prepending a single quotation mark as known from common spreadsheet tools. There is also an auto completion feature which operates on the resources' literals to encourage their reuse. Once defined, resources can be used several times within one or among different sheets. Users can thereby link instances (ABox) resulting in a more interconnected RDF graph. To explicitly refer to an existing resource, users have two possibilities: using the previously introduced auto completion feature, which shows a list of suggestions while typing or copy & pasting a resource from one cell to another.

[8] `http://www.dfki.uni-kl.de/~mschroeder/demo/rdf-spreadsheet-editor`

For each knowledge base a **SPARQL endpoint** is available using Fuseki[9]. A graphical front-end mainly realized with YASGUI[10] allows an expert to write SPARQL queries. In contrast to the fixed structure of a spreadsheet, this allows for running complex queries or dynamically restructuring the data for other issues and applications. Predefined queries help to get a quick and easy overview of classes, properties and instances. User defined queries can be saved and shared with others. This feature, for example, allows knowledge engineers to create specialized views for non-experts, which is a first means of providing direct feedback showing modelling consequences. We can also think of another feature to directly feed existing knowledge services.

Providing an online **collaborative work environment** helps domain experts from different fields to work together on a single knowledge base simultaneously. That is why changes in sheets need to be propagated immediately, so every team member is directly aware of recent changes. This fosters the active contribution of all members resulting in a vivid collaboration. Auto completion features promote the reuse of classes, properties and instances.

In an **early evaluation**, 17 rather inexperienced users were able to model a demo scenario in shorter time compared to writing triples in turtle syntax as well as using Protégé.

4 Conclusion and Outlook

In this paper, we presented a tool that uses the spreadsheet metaphor to enable all kinds of users, especially those inexperienced in Semantic Web concepts, to create RDF data. We focused on supporting users in manually filling a knowledge base with their expertise as well as easily collaborating with their team while doing so. Transferring cells into triple statements allows for running complex queries or dynamically restructuring the data for other issues and applications.

In the future, we plan to enhance our app to support the import of existing structured data, linking to other knowledge bases, capabilities to directly feed existing knowledge services and multilingualism.

Acknowledgements Parts of this work have been funded by the German Federal Ministry of Economic Affairs and Energy in the project PRO-OPT (01MD15004D) and by the DFG in the project Managed Forgetting (DE 420/19-1).

References

1. Tripcel: Exploring rdf graphs using the spreadsheet metaphor. In: 8th International Semantic Web Conference (ISWC2009) (October 2009)
2. Han, L., Finin, T., Parr, C., Sachs, J., Joshi, A.: RDF123: from spreadsheets to RDF. The Semantic Web-ISWC 2008 pp. 451–466 (2008)
3. Pohl, O.: rdfedit: User Supporting Web Application for Creating and Manipulating RDF Instance Data, pp. 54–59. Springer International Publishing, Cham (2014)

[9] `https://jena.apache.org/documentation/fuseki2`
[10] `https://github.com/OpenTriply/YASGUI`