Payola: Collaborative Linked Data Analysis and Visualization Framework[*]

Jakub Klímek[1,2], Jiří Helmich[1], and Martin Nečaský[1]

[1] Charles University in Prague, Faculty of Mathematics and Physics
Malostranské nám. 25, 118 00 Praha 1, Czech Republic
{klimek, necasky}@ksi.mff.cuni.cz
[2] University of Economics, Prague
Nám. W. Churchilla 4, 130 67 Praha 3, Czech Republic

Abstract. Payola is a framework for Linked Data analysis and visualization. The goal of the project is to provide end users with a tool enabling them to analyze Linked Data in a user-friendly way and without knowledge of SPARQL query language. This goal can be achieved by populating the framework with variety of domain-specific analysis and visualization plugins. The plugins can be shared and reused among the users as well as the created analyses. The analyses can be executed using the tool and the results can be visualized using a variety of visualization plugins. The visualizations can be further customized according to ontologies used in the resulting data. The framework is highly extensible and uses modern technologies such as HTML5 and Scala. In this paper we show two use cases, one general and one from the domain of public procurement.

Keywords: Linked Data, RDF, visualization, analysis

1 Introduction

As the Linked Data begins to gain interest from general public in addition to semantic web developers, there is a growing need for user friendly linked data tools that do not require deep knowledge of semantic web technologies. The example of such a user can be a journalist who is interested in the data published as Linked Data, but does not want to learn SPARQL and RDF specifics. The problem arising here is that for each data domain, the data needs to be visualized differently to be comprehensible to non-technical users familiar with that domain. Therefore, it is hard to come up with a general analysis and visualization tool that would be usable by experts who know RDF and SPARQL and at the same time could shield the end-user from the RDF triples in a useful manner. Our vision is to build a community around a framework that provides all that is necessary to access and visualize Linked Data. The community would consist of both expert developers and end-users. Developers would create, share and reuse both general and domain-specific plugins for analysis and visualization of Linked Data. End-users could then easily use and combine the plugins, give feedback and suggest improvements, all without extensive technical knowledge.

2 Framework and plugins

In this section, we will briefly describe the framework and how the plugins fit together. Payola consists of a server part and a client part. The client part runs in an HTML5 enabled web browser using JavaScript and handles analyses editing and results visualization. The server part runs as a Scala program and handles users, datasource querying and data manipulation. Payola is a framework, so the concrete functionality and usability is the responsibility of plugins that populate the framework.
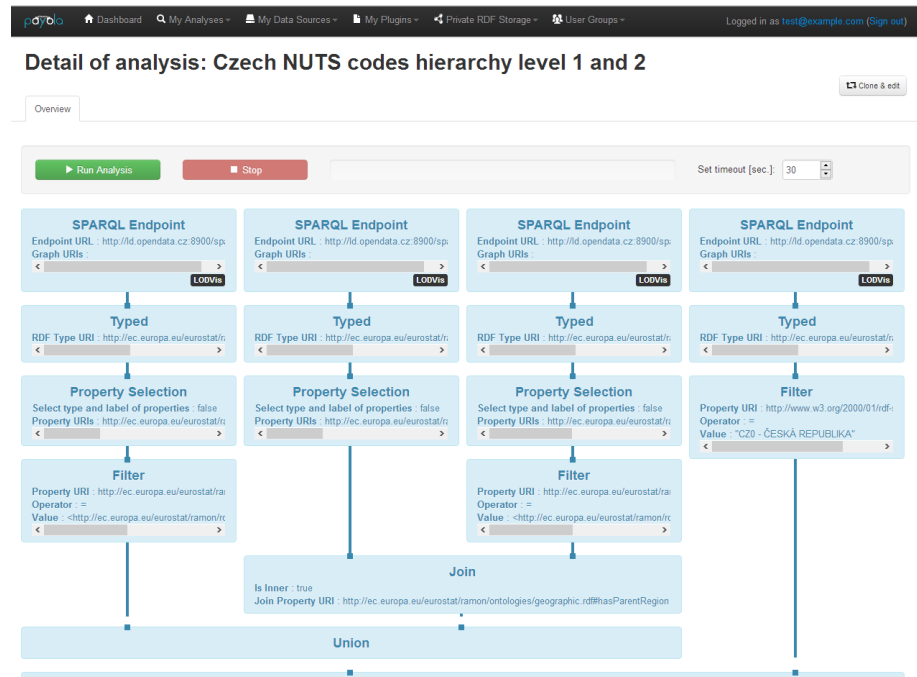

Fig. 1. Analysis editor - NUTS code hierarchy of the Czech Republic

There are two basic types of plugins. Firstly, the created analyses consists of interconnected analytical plugins - *analyzers*. When an analysis is executed and results are returned, they are visualized using visualization plugins - *visualizers*. As a proof-of-concept, we created a library of default analyzers that represent typical parts of SPARQL queries. These include selection of resources of a certain type, selection of properties that are interesting to the user and filters for resources which satisfy SPARQL filter conditions. However, more complex analyzers are possible. For example an ontological filter selects resources and properties which are instances of a given ontology. Currently under development is a set of domain-specific analyzers for the domain of public procurement. These analyzers are translated into SPARQL queries and executed on selected SPARQL endpoints. An issue to consider here is that when we connect ana-
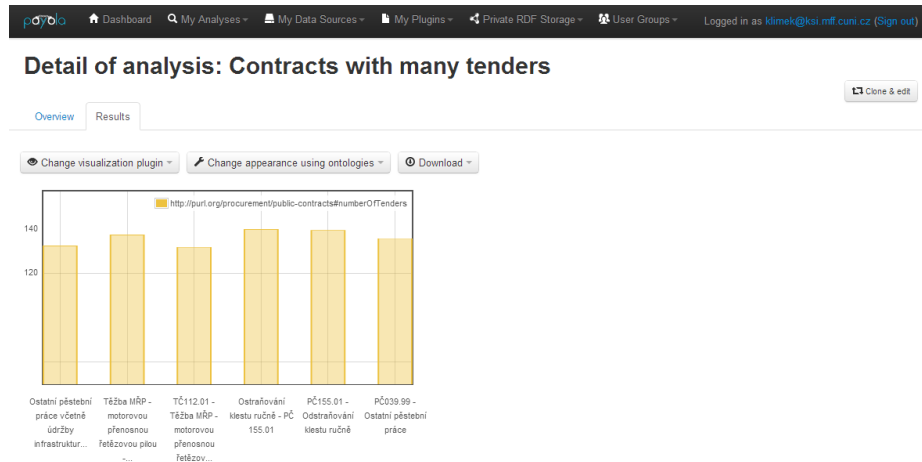
lyzers arbitrarily, the resulting SPARQL queries can be inefficient. The framework now includes a simple optimizer that combines many analyzers into one SPARQL query. However, it has its limits and it is a whole research area of query optimization which we

Fig. 2. Results browser - NUTS code hierarchy of the Czech Republic

need to focus on in order for the framework to be efficient even with more complicated analyses. An important feature for collaboration which is now under development is the ability to use a completed analysis as an analyzer (one box) in another analysis. So far a user can only clone an analysis and edit his copy. Once an analysis is correctly defined (see Figure 1 for an example of how an analysis can look like), it can be executed and the results visualized using different visualizers. We created a small proof-of-concept library of simple visualizers that include various graph and chart visualizations. An example can be seen in Figure 2 and Figure 3. While graph visualization is generically applicable for RDF data, for some data a chart visualization can be more suitable. The column chart visualizer has certain requirements on the input RDF graph. It has to contain entities of the same type, each equipped with a label and property with a value. When the analysis result graph data meets this criteria, it can be visualized as a column chart. Currently, the Payola framework is populated by proof-of-concept basic plugins which are not suited for the non-technical end-users yet. Currently under development is a feature for packaging an analysis an analyzer with user-friendly parameter selection. These packaged analyses are then directly usable by end-users.

Fig. 3. Column chart visualization - numbers of tenders for each public contract

3 Use cases

In this section, we will describe two use cases. The first goal is to reuse an analysis that shows a hierarchy of NUTS[3] codes for the Czech Republic, but for Germany. We will show how to adjust the existing analysis, which will show how easy it is to reuse analyses created by other users. The user finds the appropriate analysis in a list of available analyses. Then he creates his own copy (see Figure 1) and identifies the parameters that need to be changed. In future this step will be deprecated by creating analyzers out of existing analyses, which will be parametrized by selected values such as the country name. When the user changes the appropriate parameters, he executes the analysis and gets the results. Then the user can switch visualizers without the need to recompute the analysis.

In the second scenario, the user will see various visualizers applied to a result of a simple analysis from the domain of public procurement. It returns public contracts to which more than 120 tenders where submitted as a triple table. The analysis is labeled "Contracts with many tenders". The user can switch to, for example, the Gravity visualizer and see the results as a graph. Next, the general graph visualization can be customized for the Public Contracts Ontology[4] using the "Change appearance using ontologies". The user will see that the nodes representing public contracts are now green and look like a bobby pin. Because the result has a format required for the column chart visualizer (see Figure 3), the user can select it and see a column chart where the columns represent the number of tenders for each public contract in the result.

We asked our test users a couple of questions regarding usability and we got the responses we expected, which is that the general idea is great, but we need to work on the usability and easiness to use.

---

[3] Nomenclature of Territorial Units for Statistics - European region codes

[4] `http://purl.org/procurement/public-contracts`

4   Related Work

There are analytical tools such as *Semantic Pipes* [3]. It is a tool for transformation of data on the web which also supports RDF. However, it is focused solely on the analysis part and it is not possible to create an analysis without semantic web expertise, nor does the tool support creation of plugins by experts that would bridge this gap. There is a number of Linked Data visualization tools available, such as *Tabulator* [2] or *Explorator* [1], which usually visualize Linked Data as a graph and also support table view of the RDF triples. *Sgvizler*[5] allows web developers to render results of SPARQL queries as charts, maps, etc. These solely focus on the visualization part and offer a limited number of visualization techniques without support for extensibility. Also, there is *ViziQuer* [4] for exploring unknown SPARQL endpoints and getting and idea about what data is inside. None of the tools aim to support non expert users and none of them provide both analysis and visualization capabilities to a sufficient extent.

5   Conclusions

In this demo paper, we presented Payola, a framework for analysis and visualization of Linked Data which aims to combine the advantages found in some of the above mentioned tools to create an environment for analysis specification, execution an results visualization with a potential to be usable by non-expert users, for who the expert ones can create plugins, which are then easy to use. It supports analyses editing and collaboration via sharing and reuse. When an analysis is executed, results can be visualized using various plugins. The framework is currently populated by basic proof-of-concept plugins and more sophisticated plugins are currently under development. The tool is available at its project website: `http://live.payola.cz`.

References

1. S. Araujo, D. Shwabe, and S. Barbosa. Experimenting with Explorator: a Direct Manipulation Generic RDF Browser and Querying Tool. In *WS on Visual Interfaces to the Social and the Semantic Web (VISSW2009)*, 2009.
2. T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and analyzing linked data on the semantic web. In *3rd Int. Semantic Web User Interaction WS*, 2006.
3. D. Le-Phuoc, A. Polleres, M. Hauswirth, G. Tummarello, and C. Morbidoni. Rapid prototyping of semantic mash-ups through semantic web pipes. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 581–590, New York, NY, USA, 2009. ACM.
4. M. Zviedris and G. Barzdins. ViziQuer: A Tool to Explore and Query SPARQL Endpoints. In G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. Leenheer, and J. Pan, editors, *The Semantic Web: Research and Applications*, volume 6644 of *Lecture Notes in Computer Science*, pages 441–445. Springer Berlin Heidelberg, 2011.

---

[5] `http://code.google.com/p/sgvizler/`