Open Mashup Platform – A Smart Data Exploration Environment

Tuan-Dat Trinh, Ba-Lam Do, Peter Wetz, Amin Anjomshoaa, Elmar Kiesling, and A Min Tjoa

Vienna University of Technology, Vienna, Austria
{tuan.trinh,peter.wetz,ba.do,amin.anjomshoaa,
elmar.kiesling,a.tjoa}@tuwien.ac.at

Abstract. The number of applications designed around Linked Open Data (LOD) has expanded rapidly in recent years. However, these applications typically do not make use of the vast amounts of LOD datasets, but only provide access to predefined, domain-specific subsets. Exceptions that do allow for more flexible exploration of LOD are not targeted at end users, which excludes users who have limited experience with Semantic Web technologies from realizing the potential of the so-called LOD cloud. This paper introduces a Mashup Platform that models, manages, reuses, and interconnects LOD web applications, thereby encouraging initiative and creativity of potential users. Figuratively, our approach allows developers to implement building blocks whereas the platform provides the cement so that end users can build houses by themselves.

1 Introduction

More than ten years after the concept of LOD has been introduced, can end users really benefit from it? In spite of considerable effort made by researchers, the answer, unfortunately, is "just a little". This issue is becoming more pressing as the number of LOD dataset increases; as of 2014[1], there are already more than 60 billion triples from 928 datasets.

The limited adoption of LOD by end users may be explained by the following observations: (i) There are currently no platforms that allow end users to manage and reuse LOD applications. An "*LOD App Store*" – by analogy with digital distribution platforms for mobile apps such as *Google Play Store* or *App Store* – would be an interesting concept to foster diffusion among end users. (ii) Most LOD providers publish their datasets without paying regard to how their data may be used effectively or how it may be combined with data provided by others. After all, specific use cases are typically the most effective way to illustrate that the data is useful for end users. To save their own times, LOD providers need tools that support them in implementing such applications as efficiently as possible. (iii) Most importantly, end users currently play a passive role, waiting for developers to deliver LOD applications rather than leveraging LOD according

[1] http://stats.lod2.eu/

to their individual needs. To make good use of LOD, users currently need to equip themselves with knowledge about Semantic Web technologies as well as the SPARQL query language. Since this cannot generally be expected, the key question we address here is: "Is there any way to utilize the users' capabilities and creativity and allow them to explore LOD themselves without the need to acquire specialized knowledge?"

To address these issues, we propose an *Open Mashup Platform*. The key idea of this platform is to compose LOD applications from linkable *Web widgets* provided by data publishers and developers. These widgets are divided into three categories, i.e., *data*, *process*, and *visualization* widgets which perform the data retrieval, data processing/data integration, and data presentation tasks, respectively. The internal mechanics of these complicated tasks are not visible to end users because they are encapsulated inside the widgets. Widgets have inputs and outputs and can be easily linked to each other by end users. Being web applications, they can run on various platforms and be shared and reused. When a LOD provider publishes a new dataset, they can develop new widgets and add them to the platform so that users can dynamically, actively and creatively combine these widgets to compose LOD applications across multiple LOD datasets. This paper presents the most basic functionalities of the Platform – a smart data exploration environment for end users available at `http://linkedwidgets.org`.

2 Prototype System

2.1 Graph-based model and the Annotator tool

To communicate and transmit data between widgets, each of them implements its own well-defined model as well as interfaces to provide the features required by the Platform. A widget is similar to a service in that it has multiple inputs and a single output. To model them, however, instead of capturing the functional semantics and focusing on input and output parameters like SAWSDL [6], OWL-S [2], WSMO [1] etc., we use a graph-based model similar to [5]. This approach has a number of advantages and is a prerequisite for the semantic search and auto-composition algorithms described in this paper. For example, from the model for a *Film Merger* widget that requires an *Actor* and a *Director* as its two inputs and returns a list of *Films* starring the actor and being directed by the director (cf. Fig. 1a), the relation between input and output instances is immediately apparent.

To allow developers to create and annotate widgets correctly and efficiently, we provide a *Widget Annotator* tool. Developers simply drag, drop and then configure three components, i.e., *WidgetModel*, *Object*, and *Relation* to visually define their widget models. After that, the OWL description file for the model as well as the corresponding HTML widget file are generated automatically. The latter includes the injected Java Script code snippet served for the widget communication protocol and a sample *JSON-LD* input/output of the widget according to the defined model. Based on that, developers can implement the widget's processing function which receives input from widgets and returns output to others. De-

velopers can also rewrite/improve widgets using server-side scripting languages. Finally, as soon as they have deployed their widgets, developers can submit their work to the platform where it is listed and can be reused with other available widgets; in particular, the widgets annotations are published into the LOD of widgets which can be accessed from the graph <*http://linkedwidgets.org*> of the `http://ogd.ifs.tuwien.ac.at/sparql` SPARQL endpoint.

2.2 Semantic Widget Search

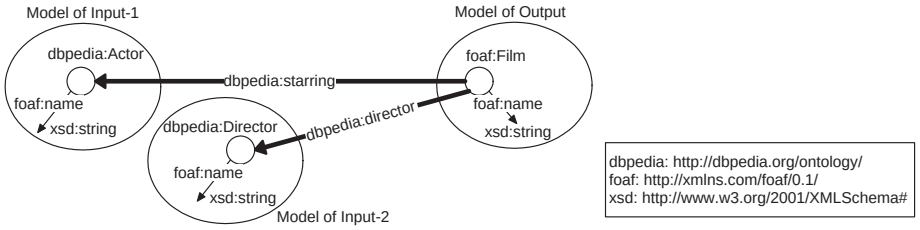
In line with the growth of the LOD cloud, the number of available widgets can be expected to grow rapidly. In this case, to ensure that users can find widgets on the platform, a *semantic search* will be provided in addition to conventional search methods by keywords, category, tags, etc. Because the widgets' RDF metadata is openly available via the SPARQL endpoint, other third parties, if necessary, can also develop their own widget-search tool. Our search tool is similar to the annotator tool, but it is much simpler and directed at end users. By defining the constraints for input/output, they, for example, can find widgets which return *Films* with particular properties for each *Film*, or even find widgets which consist of relationship between *Films* and *Actors*.

2.3 Mashup Panel

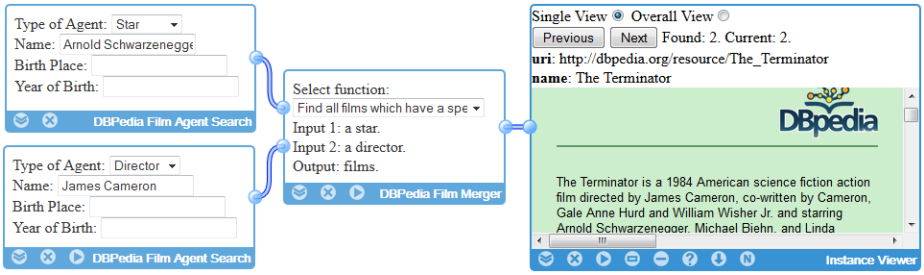
The *Mashup Panel* is the most crucial part of the platform; it allows users to compose, publish and share their applications, thereby enabling them to dynamically and actively explore LOD datasets without special skills or knowledge.

Widgets are grouped into *Widget Collections* to offer a group of scenarios after being combined to each other. Users can create their own collections or choose to work with existing collections from other users. The list of widgets that belong to the selected collection is placed at the left-hand side of the *Mashup Panel*. Users simply drag and drop widgets into the mashup area at the right-hand side. For each chosen widget, available operations are *resize*, *run*, *view/cache output data*, *get detailed information* about the widget based on its URI. In the next step, users can wire the input of a widget to the output of another one and thus build up a data-processing flow. The connected widgets, under the coordination of the platform, will communicate and transmit data to each other, from the very first data widgets to the visualization widgets. Finally, if the whole mashup is saved, parameters set in HTML form inputs from each widget will be automatically detected and stored so that users can publish the final result displayed inside the visualization widget onto their websites. Furthermore, the combined application are semantically annotated and can be shared between users via their URLs or URIs.

We implemented two algorithms to help users acquaint themselves with their new widgets: *auto-matching* and *auto-composition*. The *auto-matching* algorithm enables users to find – given input/output terminal A – all terminal B from all of other widgets such that connection between A and B is valid. The *auto-composition* algorithm is a more advanced approach in that it can automatically



(a) Model of the Film Merger Widget



(b) A complete sample LOD Application

Fig. 1: Sample widget model and widget combination

compose a complete application from a widget, or a complete branch that consumes/provides data for a specific output/input terminal. "*Complete*" in this context means that all terminals must be wired. This as well as the semantic search feature distinguish our platform from similar contributions, e.g., [4] or [3]. A sample application that collects all movies played by *Arnold Schwarzenegger* and directed by *James Cameron* is shown in Fig. 1b. Many other use cases can be found on the platform at `http://linkedwidgets.org`.

References

1. de Bruijn, J., et al.: Web Service Modeling Ontology (WSMO) (2005), `http://www.w3.org/Submission/WSMO/`
2. David, M., et al.: OWL-S : Semantic Markup for Web Services (2004), `http://www.w3.org/Submission/OWL-S/`
3. Imran, M., et al.: ResEval mash: a mashup tool for advanced research evaluation. In: 21st international conference companion on World Wide Web. pp. 361–364 (2012)
4. Le-Phuoc, D., et al.: Rapid prototyping of semantic mash-ups through semantic web pipes. In: 8th international conference on World Wide Web. p. 581. ACM Press, New York, New York, USA (2009)
5. Taheriyan, M., Knoblock, C.: Rapidly integrating services into the linked data cloud. In: 11th International Semantic Web Conference. pp. 559–574 (2012)
6. Wc, C.B., Ibm, J.F.: SAWSDL : Semantic Annotations for WSDL and XML Schema. IEEE Internet Computing 11(6), 60–67 (2007)