

DERI Pipes: visual tool for wiring Web data sources

Danh Le Phuoc¹, Axel Polleres¹, Giovanni Tummarello¹, Christian Morbidoni²

¹ Digital Enterprise Research Institute, NUI Galway, Ireland
{firstname.lastname}@deri.org

² SeMedia Group, Universita' Politecnica delle Marche, Ancona, Italy
christian@deit.univpm.it

Abstract. Making effective use of RDF data published online (such as RDF DBLP, DBpedia, FOAF profiles) is, in practice, all but straightforward. Data might be fragmented or incomplete so that multiple sources need to be joined, different identifiers (URIs) are usually employed for the same entities, ontologies need alignment, certain information might need to be "patched", etc. The only approach available to these problems so far has been custom programming such transformations for the specific task to be performed in a Semantic Web application. In this demo, we illustrate a paradigm for creating and reusing such transformation in an easy, visual web-based and collaborative way: DERI Pipes.

Keywords: Semantic Web, data mashup.

1 Introduction

Web is a vast collection of data sources published in several formats such as XML, RSS, XHTML, RDF, JSON, etc. These data sources can be reused to build new more powerful, useful and relevant data sources in a way that their creator did not quite intend. However, the published data sources are rarely in exactly the form that they are going to be used; they need to be transformed to compose an expected data source. The common approach for this transformation is custom programming which is not trivial for everyone.

On top of that, there is an increasing amount of RDF data exposed by current Web applications such as DBLP¹, DBpedia [Auer et al., 2007], blogs, wikis, forums, etc that expose their content as e.g. SIOC [Breslin et al., 2005], FOAF [Brickley et al., 2005] data through SIOC exporters, Triplify plugins². Furthermore, these RDF data are offered in a variety of formats, such as interlinked RDF/XML files, RDF statements embedded in HTML/XML pages, etc³.

Unfortunately, there is no clear and established model on how to use this amount of information coming from many diverse sources. In general, such data is unlikely to be directly injected into an end application for multiple reasons. The common

¹ DBLP Bibliography - Home Page. <http://www.informatik.uni-trier.de/~ley/db/>

² Triplify <http://triplify.org/>

³ Custom Rdf Dialects <http://esw.w3.org/topic/CustomRdfDialects>

approach to these problems so far has been the custom programming of such transformations for the specific task to be performed in a Semantic Web application. Hence, we empower the vision of turning the web into programmable environment for everyone. Our DERI Pipes⁴ is a step towards making web data sources rewiring easier for people to make more control over information services they consume without requiring full programming skills.

This metaphor has been inspired by Yahoo Pipes⁵, Microsoft Popfly, etc⁶, which allows to implement customized services and information streams by processing and combining Web sources (usually RSS feeds) using a cascade of simple operators. Since Web pipes are themselves HTTP retrievable data sources, they can be reused and combined to form other pipes. Also, Web pipes are "live": they are computed on demand at each HTTP invocation, thus reflect the current status of the original data sources.

Unfortunately, Yahoo Pipes are engineered to operate using fundamentally the RSS paradigm (item lists) which does not map well to the graph based data model of RDF. For this purpose, Similar to Yahoo Pipes, Microsoft Popfly⁷, etc in targeting to non-programmer users, SWP provides a user-friendly visual editor to drag and drop web data sources then wire them with processing blocks to redirect them into expected output. Although, SWP Pipes supports several formats (XML, RSS, XHTML, RDF, JSON, etc), as an extended version of Semantic Web Pipes [Morbidoni et al., 2007], it strongly focuses on utilizing features of RDF data processing. On top that, DERI Pipes is an open source project which enables developers to extend processing blocks and integrate it to enterprise application.

SWP offers specialized operators that can be arranged in a graphical web editor to perform the most important data aggregation and transformation tasks without requiring programming skills or sophisticated knowledge about Semantic Web formats. This enables developers as well as end users to create, share and re-use semantic mashups that are based on the most current data available on the (semantic) web.

When a pipe is invoked (by a HTTP GET request with the pipe URL), the external sources of the pipe are fetched dynamically. Therefore, the output of a pipe always reflects the data currently available on the web. Since SWP are themselves HTTP retrievable data sources, they can be reused in other pipes and combined to form pipes of increasing complexity....

2 Semantic Web Pipes engine and features

While it would be possible to implement pipe descriptions themselves in RDF, our current ad hoc XML language is more terse and legible. If an RDF representation will be later needed, it will be possible to obtain it via GRDDL [Conolly et al., 2007]. The executable pipe XML syntaxes are stored in a database. When they are invoked, the

⁴ DERI Pipes. <http://pipes.deri.org/>,

⁵ Yahoo Pipes(<http://pipes.yahoo.com>),

⁶ Microsoft Popfly (<http://www.popfly.com/>).

execution engine fetches data from remote sources into an in-memory triple store, and then executes the tree of operators. Each operator has its own triple buffer where it can load data from input operators, execute the SPARQL[Prud'hommeaux et al., 2005] query or materialize implicit triples of RDF closures.

DERI Pipes has executing engine which is responsible for processing data defined in an ad hoc XML language. The executable pipe XML syntaxes are stored in a database. When they are invoked, the execution engine fetches data from remote sources into processing buffers (triple store, XML pipeline processing stream XProc[Walsh et la 2008], string buffer, etc) and then executes the tree of operators. These processing buffers can be filtered, transformed and queried by using Regrex, SPARQL, XQuery, XPath, XSLT, etc. Since each operator is implemented as a processing unit, it can be scheduled in parallel and distributed processing structure to improve scalability.

In this version, DERI Pipes is supporting 4 types of operators. The first type includes the operators for fetching data from RDF, SPARQL-XML result, XML, HTML (with embedded RDFa and microformats). The second type is for issuing SPARQL query like SELECT, CONSTRUCT. Processing operators like mixing, patching and reasoning represent another type. The last type of operators are input operators like parameter, URL builders. All the specifications of these operators can be found at <http://pipes.deri.org/documentation.html>.

A graphical editor (figure 1) with a visual and user friendly user interface has been implemented based on the framework ZK. We are using ZK as integrating framework to control several Javascript based GUI libraries by Java which is used for server-side manipulation of RDF data. The drag and drop editor lets users view and construct their pipeline, inspecting the data at each step in the process. Furthermore, any existing pipe can be copied and integrated into the pipe that is edited by the user.

Thanks to HTTP content negotiation, humans can use each Semantic Web Pipe directly through a convenient web user interface. The format of a pipe output depends on the HTTP header sent in the request. For example, RDF-enabled software can retrieve machine-readable RDF data, while users are presented a rich graphical user interface. Therefore, along with supporting various common serialized RDF formats (RDFXML[Hayes, 2004],N3[Berners-Lee, 1998],Turtle[Beckett, 2006],etc), we also support JSON formats for pipe output which is suitable for light-weight data-level web mashup. For example, our pipe's preview interface has been created from Javascript-based Exhibit facet browser consuming Exhibit [Huynh et al., 2007] JSON data directly from pipes.

3 Use cases at work

This section will give some typical use cases which were created on pipes.deri.org as pipes by using graphical editor. Firstly, we present a simple pipe that shows how to remixing data from various sources. Data about Tim Berners-Lee is available on various sources on the Semantic Web, e.g. his FOAF file, his RDF record of the DBLP scientific publication listing service and from DBpedia. This data can not simply be merged directly as all three sources use different identifiers for Tim. Since

we prefer using his self-chosen identifier from Tim's FOAF file, we will create a pipe as an aggregation of components that will convert the identifiers used in DBLP and DBpedia. This is performed by using the Construct-operator with a SPARQL query (see TBLonTheSW pipes at [Morbidoni, 2007]). The whole showcase is then easily addressed by the pipe shown in Fig. 1: URIs are normalized via the CONSTRUCT-operators and then joined with Tim's FOAF file.

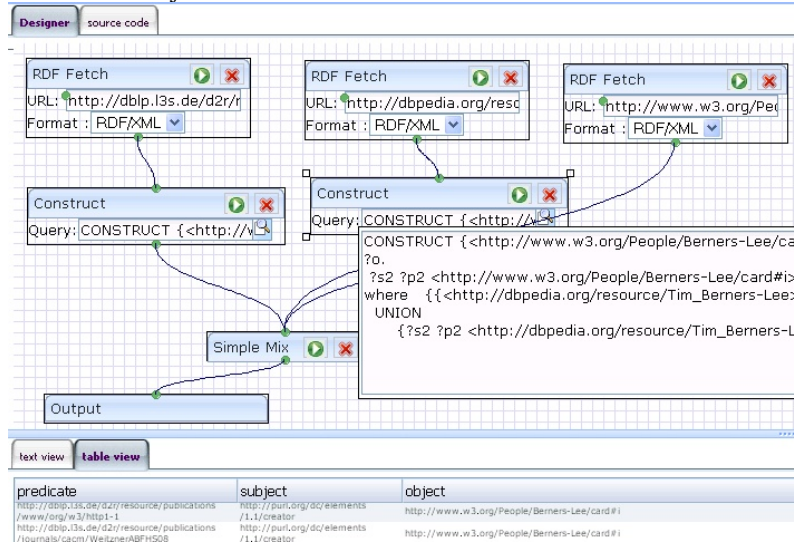


Fig1. remixing RDF data from different sources

Inspired by the Yahoo Pipes use cases of cross-search engine feed aggregation, DERI Pipes should enable us to fetch data from parametric sources as well as extracting RDF statements from non-native RDF formats. For example, we want to collect facts about London that are published as RDF statements on heterogeneous sources like news, web blogs, geo-based web services, etc. These RDF statements stated in HTML-based documents as RDFa, microformats or are provided by web services calls which support RDF format outputs. On top of that, we do not know beforehand which are URLs containing such RDF data, so we have to employ semantic web search engines like Sindice to search for such URLs. Firstly, we ask for RDF URIs indexed in Sindice⁸, then use the SELECT operator to filter the interested URIs. After that, we use the FOR operator to repeatedly fetch RDF data from those URIs. Furthermore, we can get geo information from Geonames⁹. Finally, we use the Simplemix operator to mix these RDF data to create a parametric pipe (c.f. the cityfacts pipe at [Morbidoni, 2007]) which allows users to enter a city name for searching RDF data about that city.

Examples for all operators and other showcases can be found at <http://pipes.deri.org>.

⁸ Sindice – The Semantic Web Index. <http://www.sindice.com/>.

⁹ GeoNames. <http://www.geonames.org/>

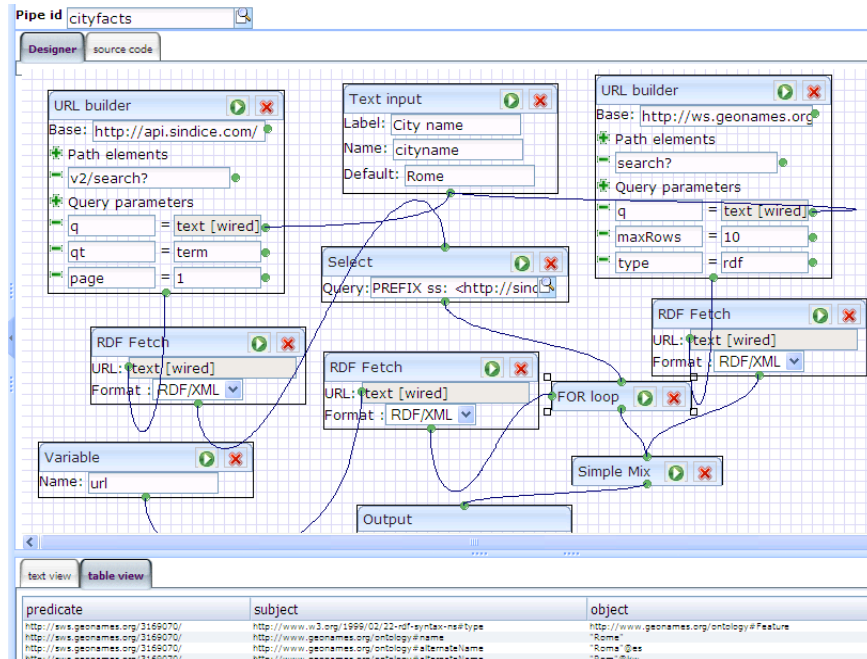


Fig. 2. Cross engine feeds

References

- [Auer et al., 2007]. Auer, S., Bizer, C., Kobilarov, G., Lehmann, L., Cyganiak, R., and Ives Z. : Dbpedia: A nucleus for a web of open data. In 6th Int.l Semantic Web Conf., Busan, Korea, November 2007.
- [Conolly et al., 2007]. Conolly, D. (ed.): Gleaning Resource Descriptions from Dialects of Languages (GRDDL), July 2007. W3C Prop. Rec., <http://www.w3.org/TR/grddl/>.
- [Breslin et al., 2005]. Breslin, J.G., Harth, A., Bojars, U., Decker, S., "Towards Semantically-Interlinked Online Communities", Proceedings of the 2nd European Semantic Web Conference (ESWC '05), LNCS vol. 3532, pp. 500-514, Heraklion, Greece, 2005. <http://rdfs.org/sioc/spec/>
- [Brickley et al., 2005]. Brickley, D. and Miller, L.: FOAF Vocabulary Spec., July 2005. <http://xmlns.com/foaf/0.1/>.
- [Morbidoni et al., 2007]. Morbidoni, C., Polleres, A., Tummarello, G., and Le-Phuoc, D.: SemanticWeb Pipes. Technical Report, see <http://pipes.deri.org/>, Nov. 2007.
- [Prud'hommeaux et al., 2005] Prud'hommeaux, E., Seaborne, A (eds.) : SPARQL Query Language for RDF, 2005.
- [Hayes, 2004] Hayes, P. : RDF semantics, Feb.2004. W3C Rec.
- [Berners-Lee, 1998] Berners-Lee, T.: Notation 3, since 1998. <http://www.w3.org/DesignIssues/Notation3.html>.

- [Beckett, 2006]. Beckett, D. : Turtle - Terse RDF Triple Language, Apr. 2006.
<http://www.dajobe.org/2004/01/turtle/>.
- [Huynh et al., 2007] Huynh, D. F., Karger, D. R., and Miller, R. C. 2007. Exhibit: lightweight structured data publishing. In Proceedings of the 16th international Conference on World Wide Web (Banff, Alberta, Canada, May 08 - 12, 2007). WWW '07. ACM, New York, NY, 737-746.
- [Walsh et al., 2008] Walsh, N., Millowski, A., Thompson, H.S.; XProc: An XML Pipeline Language. <http://www.w3.org/TR/2008/WD-xproc-20080814/>. (Aug. 2008)