On Integrating Data Services Using Data Mashups*

Muhammad Intizar Ali[1], Reinhard Pichler[1],
Hong-Linh Truong[2], and Schahram Dustdar[2]

[1] Database and Artificial Intelligence Group, Vienna University of Technology
{intizar,pichler}@dbai.tuwien.ac.at
[2] Distributed Systems Group, Vienna University of Technology
{truong,dustdar}@infosys.tuwien.ac.at

Abstract. Mashups are applications that aggregate functionality, presentation, and/or contents from existing sources to create a new application. Contents are usually generated either using web feeds or an application programming interface (API). Both approaches have limitations as web feeds do not provide powerful data models for complex data structures and lack powerful features of database systems. On the other hand, API's are usually limited to a specific application thus requiring different implementations for each of the sources used in the mashups. We propose a query based aggregation of multiple heterogeneous data sources by combining powerful querying features of XQuery and SPARQL with an easy interface of a mashup tool for data sources in XML and RDF. Our mashup editor allows for automatic generation of mashups with an easy to use visual interface.

1 Introduction

The amount of structured and semi-structured data available on the internet has been steadily increasing and many companies are now providing their data publicly accessible through API's, querying interfaces, RESTful web services, or data services [1]. The rapid growth of Web 2.0 technologies has motivated many big companies to make their contents reusable for the creation of new applications using existing data. Many publicly accessible API's such as Google Maps[1], Amazon[2] and DBPedia[3] are available for the users to generate their own new applications using their existing contents. A typical example of such a scenario is the combination of the list of hotels in a particular city with Google Maps to generate an interactive map of hotels or data collected from several news sites and merged together to provide a single access point to the user.

Mashups are web applications that consume the available data from third parties and combine/reuse them to build a new application. Mostly the contents are in the form of web feeds or API's. All the contents are combined either on client side using client-side scripts or on server-side using some available server-side technology such as ASP,

* This work was supported by the Vienna Science and Technology Fund (WWTF), project ICT08-032.

[1] http://maps.google.com

[2] http://www.amazon.com

[3] http://www.dbpedia.org

JSP, etc. Mashups are different from traditional web applications because they are usually dynamically created to serve a very specific and short lived task. Several mashup editors have been launched to encourage people to build new applications using the massive amount of publicly available contents. Yahoo Pipes[4], Google Mashups[5] and IBM Mashup Center[6] are a few examples of the popular mashup editors. However, the limitation of existing mashup editors is that they focus only on web feeds or API's. These web feeds can represent simple information but lack the capability to represent or query data items provided by querying interfaces or data services [2]. On the other hand, API's are usually limited to a specific application thus requiring different implementations for each of the sources used in the mashups. Currently, the development of data mashups to deal with complex data structures requires strong programming skills, making mashups hard to create for novice users.

We utilize the concept of data mashups and use it to dynamically integrate heterogeneous web data sources by using the extension of XQuery proposed in [3]. All the available data sources over the internet are considered as a huge database and each data source is considered as a table. Data mashups can generate queries in extended XQuery syntax and can execute the sub-queries on any available data source contributing to the mashup. XML and RDF are the prevailing data formats for web data sources. To query these data sources, one can use XQuery and SPARQL – their respective query languages. The novelty of our tool is that it integrates the powerful features of database querying into a data mashup tool. It provides an easy to use interface of a mashup editor to generate complex queries visually for the integration of a multitude of distributed, autonomous, and heterogeneous data sources.

2 Database Oriented Mashups

A mashup application comprises three major components, which are (1) data level, (2) process level, and (3) presentation level [4]. The data level is mainly concerned with accessing and integrating heterogeneous web data sources. These sources can provide structured, semi-structured or unstructured data. Existing data mashup tools cannot deal with structural and semantic diversities of heterogeneous data sources. Recently, the importance of using data mashups for data integration using database oriented mashups has been realized [2]. Inspired by Yahoo pipes, there are a few attempts such as MashQL [5] and Deri Pipes [6] to generate semantic queries from data mashups. However, to the best of our knowledge, there exists no data mashup tool which allows the user to formulate queries over web data sources using their respective query languages and at the same time deals with the heterogeneity of the data sources. Our tool is similar to MashQL and Deri Pipes, but we focus on the XQuery extension of [3] with additional support of the SPARQL query language. Using our approach, existing data integration support for mashups is further enhanced to formulate a single query containing inside sub-queries of different query languages to deal with heterogeneous data integration.

[4] http://pipes.yahoo.com/pipes

[5] http://code.google/com/gme

[6] www.ibm.com/software/info/mashup-center/

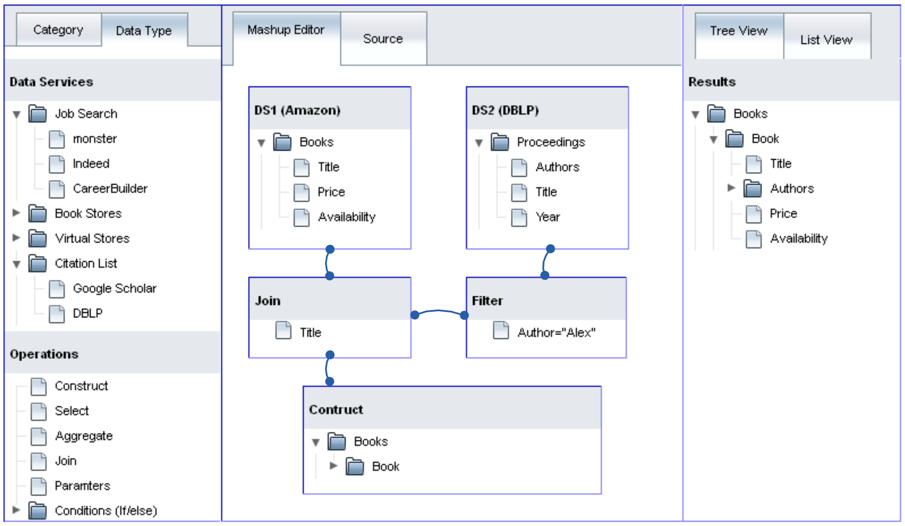

Fig. 1. Mashup Editor

3 Data Mashups Using XQuery

Figure 1 shows the interface of our system. The main window is divided into three panels, namely data source selection, mashup editor, and query results.

Data Source Selection. All available/registered data services are shown in the left panel of Figure 1. Each data service describes its available data source, its functionality and schema (if provided) which help the user to select the most suitable data service. Data services can be arranged in different categories based on metadata provided while registering a data source. Alternatively, data services can be grouped according to their data format (i.e., XML or RDF) by choosing the "data type" option in the left panel.

Mashup Editor. The central panel in Figure 1 is the mashup editor. The user can select any data service from the left panel and can easily drag and drop it into the mashup editor. These data services can be combined via several data operations, which are also selected by drag and drop from the left panel. The mashup is implemented by generating an extended XQuery expression with sub-queries in SPARQL or XQuery following the syntax in [3]. Figure 2 shows a sample extended XQuery expression generated by the mashup editor after integrating XML and RDF data sources. This query contains a SPARQL query as a sub-query inside XQuery. The mashup editor has both a design view (by choosing the "mashup editor" option in the central panel) and a command line interface (via the "source" option in the central panel). The design view provides an easy graphical interface while the command line interface is used by an expert user to write queries in the extended XQuery syntax described in [3]. For the creation of the queries from the graphical interface we use a similar approach as described in [7] for XQuery generation and [8] for SPARQL query generation.

Query Results. In the right panel of Figure 1, the result of executing the query from the mashup editor is displayed. The data format of the result is always XML. The user can

```
for $a in
doc("http://WISIRISFuzzySearch/License.xml")/agreement,
$b in SPARQLQuery(" SELECT ?Availability ?ExecutionTime
WHERE {
?x <http://www.w3.org/2001/sub#avail> ?Availability .
?x <http://www.w3.org/2001/sub#QoS> ?ExecutionTime "
      },http://WISIRISFuzzySearch/QoS.rdf)/result
RETURN
   <Result>
     <ServiceTitle>{$a/title}</ServiceTitle>
     <Requirement>{$a/requirement}</Requirement>
     <Availability>{$b/availability}</Availability>
    <ExecutionTime>{$b/ExecutionTime}</ExecutionTime>
   </Result>
```

Fig. 2. A Sample query in extended XQuery – generated from the Mashup visual interface [3]

choose between two different views of the XML result: either in tree form (as shown in Figure 1) or in table form (by choosing the "list view" option in the right panel).

4 Conclusion

We provide a database oriented mashup tool for integrating heterogeneous data sources with a visual interface which allows for an easy definition of complex data mashups. This tool can be used as plug-in for web applications to generate powerful and efficient data integration mashups.

References

1. Dan, A., Johnson, R., Arsanjani, A.: Information as a service: Modeling and realization. In: Proc. SDSOA 2007. IEEE Computer Society, Los Alamitos (2007)
2. Vancea, A., Grossniklaus, M., Norrie, M.C.: Database-driven web mashups. In: Proc. ICWE, pp. 162–174. IEEE, Los Alamitos (2008)
3. Ali, M.I., Pichler, R., Truong, H.L., Dustdar, S.: DeXIN: An extensible framework for distributed XQuery over heterogeneous data sources. In: Filipe, J., Cordeiro, J. (eds.) ICEIS 2009. LNBIP, vol. 24, pp. 172–183. Springer, Heidelberg (2009)
4. Lorenzo, G.D., Hacid, H., Paik, H.-Y., Benatallah, B.: Data integration in mashups. SIGMOD Record 38(1), 59–66 (2009)
5. Jarrar, M., Dikaiakos, M.D.: Querying the data web: The MashQL approach. IEEE Internet Computing 14(3), 58–67 (2010)
6. Morbidoni, C., Tummarello, G., Polleres, A.: Who the FOAF knows Alice? a needed step toward semantic web pipes. In: Proc. SWAP. CEUR Workshop Proceedings, vol. 314. CEUR-WS.org (2008)
7. Li Xiang, J.F.B., Gennari, J.H.: XGI: A graphical interface for XQuery creation. In: Proc. AMIA Symposium 2007, pp. 453–457. American Medical Informatics Association (2007)
8. Russell, A., Smart, P.R.: NITELIGHT: A graphical editor for SPARQL queries. In: International Semantic Web Conference (Posters & Demos). CEUR Workshop Proceedings, vol. 401. CEUR-WS.org (2008)