# A Domain-Specific Language for Do-It-Yourself Analytical Mashups

Julian Eberius, Maik Thiele, and Wolfgang Lehner

Technische Universität Dresden
Faculty of Computer Science, Database Technology Group
01062 Dresden, Germany
{julian.eberius,maik.thiele,wolfgang.lehner}@tu-dresden.de

**Abstract.** The increasing amount and variety of data available in the web leads to new possibilities in end-user focused data analysis. While the classic data base technologies for data integration and analysis (ETL and BI) are too complex for the needs of end users, newer technologies like web mashups are not optimal for data analysis. To make productive use of the data available on the web, end users need easy ways to find, join and visualize it.

We propose a domain specific language (DSL) for querying a repository of heterogeneous web data. In contrast to query languages such as SQL, this DSL describes the visualization of the queried data in addition to the selection, filtering and aggregation of the data. The resulting data mashup can be made interactive by leaving parts of the query variable. We also describe an abstraction layer above this DSL that uses a recommendation-driven natural language interface to reduce the difficulty of creating queries in this DSL.

**Keywords:** data analytics, data mashups, natural language queries.

## 1 Introduction

The increasing amount and variety of data available in the web leads to new possibilities in end-user focused data analysis. In the course of the *Open Data* trend, public agencies have started to make governmental data available using web services. In addition, there is a large amount of "crowdsourced" data from services such as Yelp (venue ratings) or Twitter (trending topics, sentiments).

To make productive use of this data, two elements are needed: first, a way to integrate the heterogenous data into a common representation, second, a way to analyze the integrated data to make it usable. cities The well-known solutions to these two problems are data integration through ETL processes into data warehouses, and the usage of BI (business intelligence) tools for analytics. These tools could basically be applied to these new forms of data as well, but for end-user data analysis they have two disadvantages: First, they are designed for skilled users. Second, ETL processes are constructed for static sets of input sources and are not suitable for on-demand joining of web data sources.

A. Harth and N. Koch (Eds.): ICWE 2011 Workshops, LNCS 7059, pp. 337–341, 2011.

To make web data accessible to end users, new integration and analysis methods are necessary. They should accommodate to the skill levels of end-users, but also to their needs: compared with the business intelligence in enterprises, the skill level as well as the query complexity are much lower.

With regard to the vast amount tools for end-user driven mashup development that have been developed in recent years, we argue that there is room for improvement. Specifically, we argue that their scope, general mashup application development, and their user interface styles, for example data- and work flow graphs, are not optimal for the problem of end user data analytics.

We will discuss our view of the requirements of end-user data analytics in the next section (Section 2). We will then propose an approach to tackle the presented problems in Section 3 and finally discuss related work in Section 4.

## 2 Research Questions

Consider an exemplary use case for end-user business intelligence: a user plans to open a cafe, and needs to decide on its location. He requires to join data from multiple heterogenous sources. He needs statistical data about the districts of the city, such as average income, rent or age structure, data which is available from public agencies. In addition he needs data about the popularity of existing cafes in the various districts, available from services such as Yelp. When he has found the data, the user needs to to join, filter and aggregate it. For example, he needs to merge the statistical information about the cities districts with the average rating of existing venues in the district.

In a next step, a visualization the be preferable to a tabular display of the result data. The type and the properties of the visualization should be configurable by the user. In a last step, it would be beneficial if the user could easily vary the parameters of the mashup, e.g., aggregation or filter parameters, to enable an exploratory style of data analysis.

From this scenario, a number of requirements and research questions can be derived.

- How to deal with heterogenous data sources with varying degrees of structure when creating data mashups?
- How well-suited are the interfaces currently used (e.g. drag and drop data flow languages) for end-user mashup construction and are there alternatives?
- How to enable users to find the data sets that contain the information they are interested in?
- How can techniques such as (automatic) tagging and matching be used to recommend data sets that could fit into the user's data mashup?
- How to facilitate the selection of visualizations and interaction patterns that are appropriate for the data?

## 3 Approach

To support the outlined use cases we propose a declarative domain specific language (*DSL*), as well as a higher level natural language interface that supports

the user in creating data mashups using this DSL. The language allows the user to query a repository of possibly heterogenous information that can have various degrees of structure, ranging from free text over CSV files and graph-structured data to relational databases.

It supports a set of operations such as joining of different data sets, filtering according to a given predicate or grouping. In contrast to the result of a query in a relational database system, executing a query in the proposed DSL results in a *data mashup*, which is a visualization of the selected data. The form of the visualization, e.g. a chart or map, can be specified in the query, or be automatically inferred from the data used in the mashup. In addition, it includes interaction features that can be specified in the query. Specifically, for each value in the query that is given as a variable, an interaction feature (slider, drop-down menu etc.) that allows to set this value will be present in the mashup.

To accommodate the language to the needs of end users while keeping it expressive enough for developers, the language can be used on two abstraction levels.

1. End-User Level: This level is suitable for end-user mashup creation. In contrast to previous mashup systems that mostly either use WYSIWYG application editing or a pipeline-style graphical connection of operators, we propose a iterative, recommendation-supported natural language interface, which will be described below.
2. DSL-Level: On this level, the actual domain specific language, i.e. the query language described above, resides. On this level, the language is similar to typical data flow languages, with the addition of the visualization operators, and variables which result in interaction features in the mashup. Input from the high-level interface are mapped to executable operators on this level to create executable mashups.

With the higher level interface, users can enter a query in natural language, which is then incrementally refined until a fitting data mashup can be created from the query. The first step would be very similar to systems like WolframAlpha[1] in which users enter entities and attributes which they want to compare, e.g., "*unemployment usa germany*". In the proposed system, more complex cases including visualization directions or filter conditions are also possible, in the style of "*plot the unemployment rates in the usa and germany between 1990 and 2010.*"

However, instead of presenting an answer on a best-effort basis, the platform would go through an incremental process of assisting the user in refining the query, as shown in Figure 1. In this process, the system will interpret the query using techniques from natural language processing to find the elements needed for the construction of the mashup: data sets, joins/filters/aggregates, visualization and interaction forms. These elements will be mapped to concrete operators on the DSL-level. For every missing element the user will be prompted to refine the query, giving recommendations based on the elements that have been recognized.

[1] http://www.wolframalpha.com/

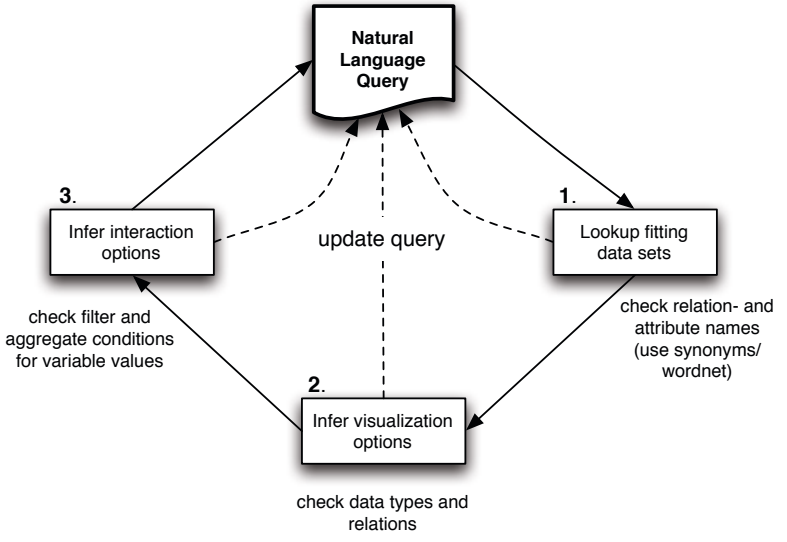


**Fig. 1.** Incremental Refinement of the Original Query Input

## 4 Related Work

An overview of available general end-user mashup development systems is given by Grammel et al [2]. Beyond these general systems, a number of data mash-up, analysis and visualization platforms have been proposed. Google Fusion Tables [1] provides tools for users to upload tabular data files, join, filter and aggregate the data and visualize the results. The interface is a standard, menu-based point and click interface, no steps of the process are assisted or automated. Similar tools are, for example, GeoCommons[2] and to some extend ManyEyes[3], which focus on the visualization and do not offer analytical functions.

One of the more successful platforms focusing on end-user data mashups is Yahoo Pipes[4]. It uses a visual data flow language to merge and filter feeds and to model user input. Executing a pipe (a data flow) results in a new feed, which can include parameters that the user specifies on execution. Resulting feed data can be displayed as a list, or on a map if the items contain spatial data. The system offers many operators and thus a high degree of flexibility, but lacks visualization or data other data exploration features, instead focusing on merging and processing of data. Furthermore, to use the system, the user has to understand the concept of data flow graphs, as well as many specific mashup problems, for example how web services are called with URL parameters, or that a geo-coding component has to be inserted into the pipe to display addresses on a map.

[2] geocommons.com
[3] manyeyes.alphaworks.ibm.com
[4] pipes.yahoo.com

There are many current systems that explore the application of natural language querying to semantic data bases. Kaufmann et al. propose a classification for these system that ranges from completely free form query entry to more structured or guided approaches with almost formal query languages [4]. They evaluate several systems and conclude that neither end of this spectrum is optimal for end-users. They argue that a guided free entry approach is preferable.

Recommending components to mashup creators has been explored for example by Greenshpan et al [3]. They developed a system that offers autocompletions based on previous mashups created by other users. Picozzi et al. on the other hand propose a system that recommends components to be added to a mashup using quality metrics that take both the new components as well as the already chosen components into account [5].

## 5 Conclusion and Planned Contributions

The increasing amount of publicly available data on the web raises the question how this data can be made usable for end-users. There is a need for simple tools for data joining, analyzing and visualizing web data from different sources.

The following specific contributions are planned:

- A DSL for data mashup construction, with query, visualization and interaction operators for working on heterogenous web data.
- A high-level natural language query interface and an iterative process for refining queries and mapping them to the concrete DSL.
- A recommendation engine for finding data sets, visualization and interaction forms fitting the given query.

The project is in the initial research phase. The next steps in the research plan include concretizing the operators of the DSL, exploring the capabilities of current NL-querying systems and experimenting with different ways of mapping natural language input to operators of the DSL and data sets in the repository.

## References

1. Gonzalez, H., Halevy, A.Y., Jensen, C.S., Langen, A., Madhavan, J., Shapley, R., Shen, W., Goldberg-Kidon, J.: Google fusion tables: web-centered data management and collaboration. In: SIGMOD 2010 (2010)
2. Grammel, L., Storey, M.-A.: A Survey of Mashup Development Environments. In: Chignell, M., Cordy, J., Ng, J., Yesha, Y. (eds.) The Smart Internet. LNCS, vol. 6400, pp. 137–151. Springer, Heidelberg (2010)
3. Greenshpan, O., Milo, T., Polyzotis, N.: Autocompletion for mashups. In: VLDB 2009 (2009)
4. Kaufmann, E., Bernstein, A.: Evaluating the usability of natural language query languages and interfaces to semantic web knowledge bases. In: Web Semantics: Science, Services and Agents on the World Wide Web (2010)
5. Picozzi, M., Rodolfi, M., Cappiello, C., Matera, M.: Quality-Based Recommendations for Mashup Composition. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 360–371. Springer, Heidelberg (2010)