EFESTO: A platform for the End-User Development of Interactive Workspaces for Data Exploration

Giuseppe Desolda[1], Carmelo Ardito[1], Maristella Matera[2]

[1]Dipartimento di Informatica, Università degli Studi di Bari Aldo Moro
Via Orabona, 4 – 70125 – Bari, Italy
{name.surname}@uniba.it
[2]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano
Piazza Leonardo da Vinci, 32 – 20134 – Milano, Italy
maristella.matera@polimi.it

Abstract. This demo will show the EFESTO platform at work for the creation of interactive workspaces supporting end users in the exploration and seamless composition of heterogeneous data sources. By means of a visual paradigm implemented within a Web composition environment, the end-users dynamically create "live" mashups where relevant information, extracted from different types of data sources including the Linked Open Data, and functions that can be performed on it can be flexibly shaped-up at runtime.

Keywords: Mashups, Web Composition Environments, Data Integration.

1 Introduction

Mashups are data-centric applications that can assist users (even when they are not experts in programming) in easily composing heterogeneous resources. They are considered a solution for supporting data exploration processes that exceed one-time interactions and allow users to progressively seek for information. As studied in [1], typically users invoke general-purpose search engines and/or specialized verticals, and then use "their brain" (or suitable cognitive aids, e.g., annotations or clipboards) for remembering results to be used next. Mashups solve (at least partially) these limitations, as they try to accommodate users' needs for data integration within personal, ad-hoc created workspaces. However, some factors still prevent a wider use of mashups in real contexts. Besides the complexity of the composition paradigm [2], the active interaction with the retrieved data, by means of advanced exploration and manipulation actions, is hardly supported.

This demo paper presents EFESTO, a platform for the **End-User Development of mashups**. EFESTO is characterized by a paradigm for the exploration and composition of heterogeneous data sources whose design has resulted from a series of elicitation studies [2], [3], in the attempt to accommodate the end-user mental model for a light-weight data integration within personal workspaces. To overcome limitations highlighted both in literature and during our field studies [2], EFESTO offers: *i)* visual mechanisms to integrate data retrieved from different data sources; *ii)* a new "polymorphic" data source model that, by exploiting the Linked Open Data (LOD) cloud, enables the access to "mutable" information depending on the situational needs of the mashup under construction; *iii)* a set of tools to organize, visualize and manipulate extracted data according to specific functions.

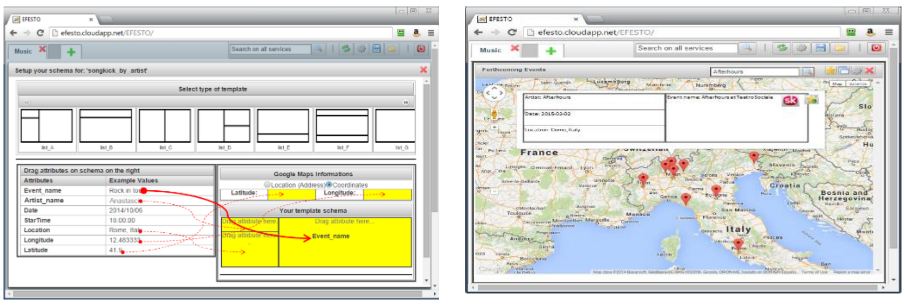
2 The EFESTO Composition Paradigm

To illustrate the main EFESTO features (also highlighted in bold in the following), let us consider a usage scenario. We suppose that Maria interacts with the EFESTO Web composition environment to explore information about musical events. Maria starts looking for pertinent services among those registered into the platform. A **wizard procedure** guides her to make a selection from a popup window where services are classified by category (e.g., videos, photos, music, social). Maria selects SongKick, a service that provides information on music events, and a map *UI template* for displaying the retrieved information. As shown in Fig. 1a, the data attributes returned by SongKick are visualized in a panel on the left. To make the attributes understandable by the user, the system shows some example values. Maria then chooses a table UI template for visualizing, when required, some further details. She selects and drops the desired attributes in the fields of the table template (colored in yellow in Fig. 1a). These actions represent queries that will be executed to create the mashup data set. Fig. 1b reports an example of the created mashup which is **immediately executed** in the Web browser. By typing "Afterhours" in the search box, the forthcoming events of this band are visualized as pin on the map. If a pin is selected, a table appears and shows the details of the corresponding event.

Maria can also **integrate data** through *join* and *merge* operations [3] that she expresses visually through drag&drop actions operated on the **running mashup**. For example, since SongKick does not provide location images, Maria joins SongKick and Flickr, so that the location name becomes the keyword for extracting from Flickr a sequence of related images. When clicking on the location name in the table shown in Fig. 1b, another box visualizes the Flickr images.

Maria also wants to know further details about the artists, such as genre, starting year of activity and artist photo. Searching among the services registered in the platform, she does not find what can satisfy her needs. Thus, she decides to bind the SongKick artist attribute with a Dbpedia-based *polymorphic* data source. The platform shows a list of properties related to the musical artist class[1]. Maria creates a new data source based on the properties *genre*, *starting year of activity* and *artist photo*. Henceforward, Maria can find a list of upcoming events on SongKick and visualize the additional artist's information, retrieved through the new data source, when clicking on a specific artist on SongKick. We call this data source *polymorphic* because it can provide

[1] When a service is registered in the platform, each attribute is automatically annotated with a Dbpedia class that is semantically close to the attribute meaning [4]; for example the SongKick *Artist* attribute is annotated with the Dbpedia *Musical Artist* class.



Drag attributes on schema on the right

Attributes	Example Values
Event_name	Rock in lou
Artist_name	Anastasci
Date	2014/10/06
StarTime	18.00.00
Location	Rome, Italy
Longitude	12.483333
Latitude	41.9

(a) (b)

Fig. 1: (a) Visual mapping between data attributes and UI template fields; (b) Example of a created mashup lively executed in the composition environment.

different information (properties) according to the data source attribute it is bound to. For example, if Maria had started from the SongKick *place* attribute, properties like borough, census, year, demographics would be proposed. Pre-registered data sources (e.g., SongKick) only provide a pre-defined, invariable set of properties.

In the previous scenario, Maria has aggregated and composed information into her workspace, according to a paradigm which is similar in some aspects to the ones provided by other mashup platforms [5]. Our field studies [1] however revealed that mashups generally lack in offering *data manipulation functions* that can be useful to support daily tasks. For this reason, EFESTO provides a set of tools (exploiting functions local to the platform or exposed by remote APIs) that provide visualizations of or actions on the extracted contents, for example to collect&save favourites, to compare items, to plot data items on a map, to inspect full content details, or to arrange items in a mind map to highlight relationships. For example, Maria can organize data on concerts so that selected concerts become node in a mind map which are connected to other node representing close restaurants. Transitions across different tools then determine different organizations of data or progressively enable different functions.

3 Architecture and Feature Checklist

Fig. 2 illustrates the EFESTO architecture. The platform is able to manage **heterogeneous components** (data, UI, logic) on which an **orchestration logic** and a **UI synchronization** at the presentation layer is applied to build **hybrid mashups**. The interaction layer supports the **dynamic composition of running mashups** through an **iconic visual language** enabling an **implicit control flow**. *UI templates* are used to build and visualize the mashup data set which consists of *UI items*, i.e., data elements rendered according to a selected UI template. UI templates also play the role of schemas specifying how the *Mashup Engine* has to query the involved APIs and integrate the resulting data. *Tools* instead manage the organization of UI items by invoking (functional) APIs. *Event Listeners* catch the events (e.g., the drag of a UI item from a UI Template to a Tool) and send them to an *Event Manager* where they are translated into the invocation

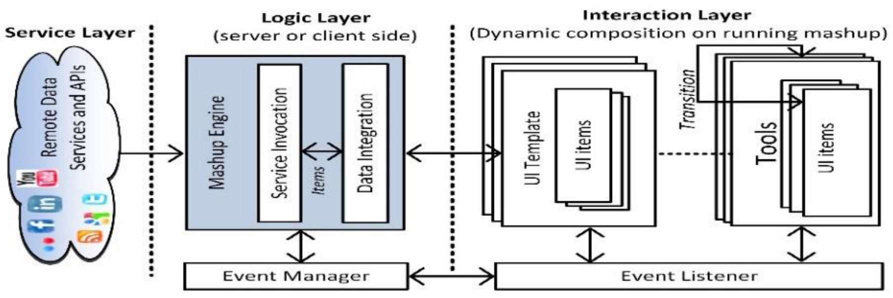


Fig. 2. Main components of the EFESTO platform.

of (functional) APIs through the Mashup Engine. **The runtime engine can be distributed along the client and the server**, or located only at the **client-side**. Server-side module offers however the advantage of implementing a **long-lasting instantiation logic** with the additional possibility of supporting the **collaborative composition** of interactive workspaces, as already discussed in some previous papers [6].

4 Conclusions

In several application domains there is an increasing demand by end users to access, integrate, and use flexibly multiple resources available online. The EFESTO platform tries to respond to this need by letting users easily integrate, by means of an EUD paradigm, heterogeneous information that otherwise would be totally unrelated. With respect to other platforms, EFESTO also enables a seamless transition of the retrieved data across different organizations, visualizations and functionality. We believe this is a characterizing feature that can pave the way to a new conception of mashups as effective tools for supporting users' tasks.

References

1. White, R.W., Roth, R.A.: Exploratory Search: Beyond the Query-Response Paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 1(1)**,** 1-98 (2009)
2. Ardito, C., Costabile, M.F., Desolda, G., Lanzilotti, R., Matera, M., Piccinno, A., Picozzi, M.: User-Driven Visual Composition of Service-Based Interactive Spaces. *Journal of Visual Languages & Computing* 25(4)**,** 278-296 (2014)
3. Ardito, C., Costabile, M.F., Desolda, G., Lanzilotti, R., Matera, M., Picozzi, M.: Visual Composition of Data Sources by End-Users. In *Proc. of AVI '14*, pp. 257-260. (2014)
4. Desolda, G.: Enhancing Workspace Composition by Exploiting Linked Open Data as a Polymorphic Data Source. In *Proc. of IIMSS '15*, in print. (2015)
5. Daniel, F., Matera, M.: Mashups: Concepts, Models and Architectures. Springer (2014)
6. Ardito, C., Bottoni, P., Costabile, M.F., Desolda, G., Matera, M., Picozzi, M.: Creation and Use of Service-Based Distributed Interactive Workspaces. *Journal of Visual Languages & Computing* 25(6)**,** 717-726 (2014)