DataConf: A Full Client-Side Web Mashup for Scientific Conferences

Lionel Médini[1,2], Florian Bacle[2], Benoît Durant de la Pastellière[2], Fiona Le Peutrec[2], and Nicolas Armando[2]

[1] LIRIS lab, University of Lyon, Lyon, France
[2] Universite Claude Bernard Lyon, University of Lyon, Lyon, France
lionel.medini@liris.cnrs.fr,
{florian.bacle,benoit.durant-de-la-pastelliere,fiona.le-peutrec,
nicolas.armando}@etu.univ-lyon.fr

Abstract. This paper describes DataConf, a mobile Web mashup application that mixes Linked Data and Web APIs to provide access to different kinds of data. It relies on a widely used JavaScript framework and on a component-based approach to manage different datasources. It only requires static server-side contents and performs all processing on the client side.

DataConf aggregates conference metadata. It allows browsing conference publications, publication authors, authors' organizations, but also authors' other publications, publications related to the same keywords, conference schedule or resources related to the conference publications. For this, it queries the SPARQL endpoint that serves the conference dataset, as well as other open or custom endpoints and Web APIs that enrich these data.

Keywords: mobile Web, client-side mashup, Linked Data, Web API, SPARQL, SWC, SWDF, Backbone.js, publication browsing, conference events.

1 Introduction

The WWW'2012 conference that was held in Lyon was an occasion for the local Web community to initiate several innovating projects around conference material and Web technologies. We designed a mobile Web application to augment the poster track. It targeted conference attendees and allowed them to navigate among poster metadata (title, authors, abstract, keywords...), as well as authors and other publications metadata, using their smartphones and tablets. For this, it takes advantage of Linked Data [1] for enriching conference metadata from several sources. It dynamically constructs complex SPARQL queries, sends them to cross-domain endpoints and allows users to browse these metadata using either a textual or a graphical interface. It relies on a main conference publication dataset. Since 2006, such datasets are available on the SWDF[1] SPARQL

[1] http://data.semanticweb.org/

P. Cimiano et al. (Eds.): ESWC 2013, LNCS 7955, pp. 337–344, 2013.

endpoint. These datasets mainly rely on the SWC vocabulary [2], [3], thus integrating FOAF, SWRC or iCal [4] elements.

After the conference, we refactored our application to be reusable and easily deployed for any conference that has its publication metadata available on the Web. This new version is named DataConf and is designed to be configurable and extendable. Several instances are already deployed for different conferences, as shown in Fig. 1.

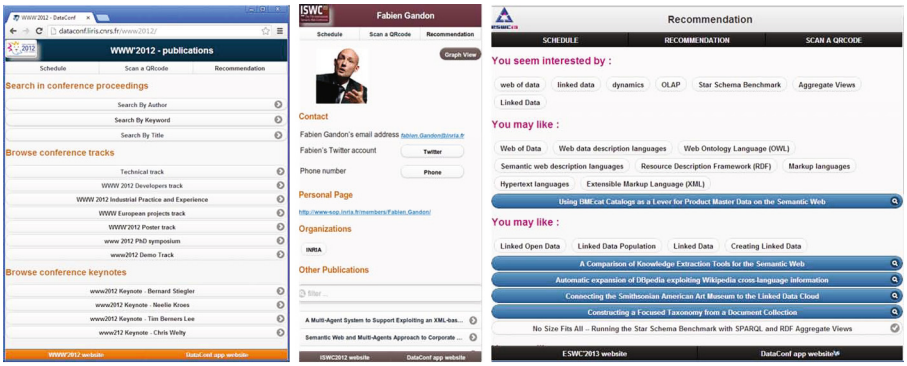

Fig. 1. Three different types of views in three different DataConf instances, respectively related to the WWW'2012, ISWC'2012 and ESWC'2013 conferences

We oriented the refactoring process around the two following tasks: conference chairs writing a configuration file for a new conference event and Web developers creating a new component that handles a particular datasource. We herein detail the architectural choices and structure, so that our application can be adopted by the community. The application homepage at http://dataconf.liris.cnrs.fr/ provides access to different DataConf instances. More information on DataConf and its custom datasources are available on our wiki at http://liris.cnrs.fr/dataconf/. DataConf sources are downloadable at https://github.com/ucbl/DataConf.

This paper is organized as follows: we firstly present the architecture of our DataConf application. Then, we detail the different datasources that can be integrated with DataConf. We conclude and present evolution perspectives of this application.

2 DataConf Architecture

DataConf is an application that aggregates heterogeneous data about scientific conference materials, such as publications, authors and keywords. For this, it queries several datasources that can either be SPARQL endpoints (e.g. SWDF, DBLP[2]), or other Web services that provide information about these materials (e.g. conference schedule, external resources available on the Web, etc.).

[2] `http://dblp.l3s.de/d2r/`

In this section, after a short overview of mashup creation frameworks, we present the DataConf architecture. In order to facilitate its reuse for other conference, we then explain how to configure a DataConf instance and how to develop a component to connect it to a new datasource.

2.1 Mashup Framework Choice

We refactored our application to make it configurable, reusable and extendable. In order to facilitate its appropriation by the community, we chose to build it on top of an existing framework. For that matter, we chose an open source and widely used framework. This section describes the rationale of this choice.

Several frameworks exist in the field of enterprise mashup technologies, such as JackBe[3] or Convertigo[4]. However, enterprise mashups [5] can be powerful applications that require users to log in and involve complex business workflows. On the contrary, we wanted our app to remain lightweight, since it only relies on components that query datasources, process their results and integrate them into the interface.

As our application originally targeted a WWW series conference, technical choices were oriented toward a full Web application applying recent/emerging Web technologies and standards. In particular, HTML5 APIs encouraged us designing the application as much client-sided as possible. We therefore chose to use a Web client-side (i.e. JavaScript) framework. Numerous JS frameworks now exist and several comparisons such as [6] are available, based on various criteria. As we aimed at favoring the adoption of our app by Web developers, we focused on community size, documentation and syntax simplicity. We thus chose to use Backbone.js[5]. On top of this framework, we developed the necessary objects to build our mashup application, among which a templating system and a workflow engine that can process queries to different datasources for each route (hash path) defined in a configuration file.

The Backbone.js framework provides a powerful routing system that permits the construction of new routes on the fly. The URL is the key of the routing system: by listenning on the URL change event, it is able to match it to a route and to trigger the associated callback function. The router comes with a ready to go history handler which permits an automatic support of page navigation.

2.2 Architecture Overview

The core of the DataConf engine is an on-purpose built Backbone.js router. At runtime (when a new hash is selected), it plays the role of a controller and performs all the machinery needed to request the datasources, according to the commands specified in the configuration file (see next section). Fig. 2. shows the main objects involved in our architecture.

[3] `http://www.jackbe.com/`
[4] `http://www.convertigo.com/`
[5] `http://backbonejs.org/`

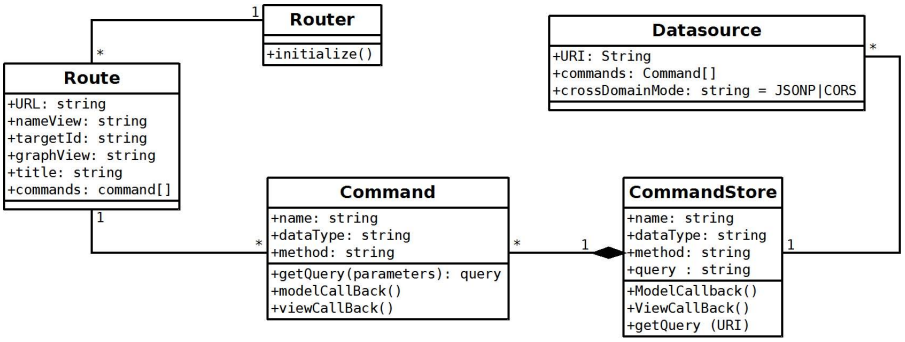

Fig. 2. Global view of the DataConf layer (on top of Backbone.js)

A command is a particular query type that can be sent to a datasource. For instance, in SWC, there will be a particular command to retrieve the publication corresponding to an URI. Commands contain the following fields: **name**, HTTP request **method** and returned **datatype**, as well as three methods:

- **getQuery**: returns the query object to send to the datasources
- **modelCallback**: handles the response from the datasource, performs calculations over the response data and transforms it in an internal common representation formatted
- **viewCallback**: is triggered by the router to integrate these formatted data into the views.

In order to cope with datasource heterogeneity problems, DataConf relies on an internal data representation that can differ from the data format contained in the response. For this, the model callback function that handles a response is limited to transforming the data into the appropriate common representation format (depending on its nature: title, author name, homepage...) and storing them using the LocalStorage API. Moreover, this architecture allows caching data in the LocalStorage object and more complex data composition into the views, since the view callback functions can wait for several types of data to be ready before starting the rendering process.

The datasource querying and integration process in DataConf is structured as follows. For each command associated to the current route, the router: i) gets the corresponding datasource and command objects, ii) verifies the data availability in the LocalStorage, iii) if not, constructs an asynchronous cross-domain request using the datasource and command configuration values, iv) sends the request, v) triggers the command model callback when the response is available, vi) triggers the command ViewCallback to render the view when the model callback has returned. This process is illustrated in Fig. 3.

2.3 Configuring a DataConf Instance

To be deployed for a particular conference, DataConf relies on a static configuration file. No sensitive information is present in the configuration file and this

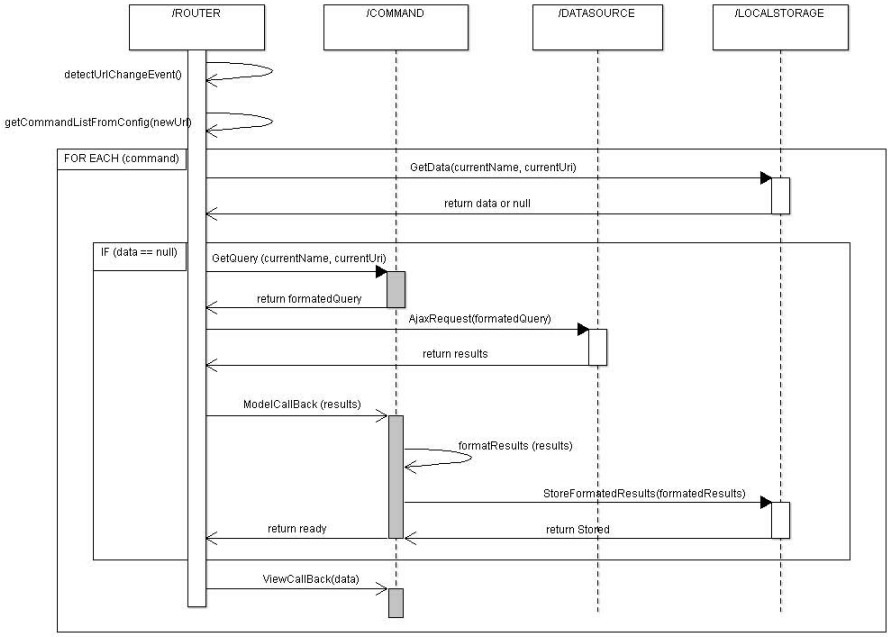

Fig. 3. Detailed view of the data retrieving process

file can be processed on the client side. This configuration file is written in JSON and contains the following elements:

- **Conference general information**, such as its name, base URI and logo URI.
- **View templates** that allow presenting different entities (publications, authors...) using different layouts. These templates are described by their locations and names. During initialization, all view templates are loaded and precompiled to create a single-page application that will be able to handle its presentation layer without calling the server again.
- The set of **datasources** that can be queried by the application. A datasource description contains a server URI, a cross-domain mode (JSONP / CORS)[6] and references to the component that handles this datasource. This component is a JS file defining a set of commands and is called a *CommandStore*.
- **Routes** that link the URI hash paths to datasource commands.

[6] JSONP (JSON with Padding, `http://bob.ippoli.to/archives/2005/12/05/remote-json-jsonp/`) and CORS (Cross-Origin Resource Sharing, `http://www.w3.org/TR/cors/`) are two techniques for sending cross-domain requests from a Web browser. The former exploits the same-origin policy exception for the HTML 'script' tag, whereas the latter uses a set of on-purpose HTTP headers issued in a recent W3C specification (currently: candidate recommendation).

2.4 Developing a Datasource Component

Developers who wish to add a new datasource to their DataConf instance are welcome to do so. For this, they need to gather in the same JS file (CommandStore), all the command objects associated to requests that can be sent to this datasource. Once this done, they need to declare this datasource and link it to their CommandStore in the configuration file and associate the commands with the appropriate routes.

3 DataConf Ecosystem

At the ESWC'2013 conference, we demonstrated the integration of our DataConf application with the following datasources:

- **Main datasource**: SPARQL endpoint serving the conference metadata in the Semantic Web Conference (SWC) ontology format (see section 1). It provides all the data about publications and authors, and refers to them using their URIs in the dataset.
- **DBLP**: SPARQL endpoint used to access authors' other publications in the DBLP database. By clicking on a DBLP publication, users can browse the DBLP datasource using DataConf as they browse the conference ones. In the ESWC'2013 instance, DataConf uses the RKBExplorer[7] SPARQL endpoint to access the DBLP datasource. This endpoint is queried using the author's full name, which can cause homonymy issues. An example of DBLP integration in DataConf is shown in the "Other Publications" section of Fig. 1.b.
- **DuckDuckGo!**[8]: Search engine used to enrich organizations' information, such as homepage URL and logo. This engine is queried using the organization full name. We rely on its "I'm feeling ducky!" feature to retrieve the most probable query result. We noticed that it provides quite good results for organizations, but less good results for people. Indeed, we originally used DuckDuckGo! for retrieving authors' homepages, but switched to Google while finding too many inaccurate results on this particular query type.
- **Google Web search API**[9]: used to enrich data by finding authors' homepages, as explained previously. Homepages are displayed in the "Personal Page" section of the authors' views, see Fig. 1.b.
- **SimpleSchedule**: Custom datasource that allows retrieving the actual schedule of the conference events. Experience shows that the schedule is a dynamic artifact, whereas a conference dataset is not. In order to provide an accurate version of the time schedule, we developed a web application that gathers all the events from the conference dataset and proposes a back office graphical user interface for easily creating, rescheduling and deleting these events. DataConf users can thus view the time and location of a paper

[7] `http://dblp.rkbexplorer.com/` -- `http://www.informatik.uni-trier.de/~ley/db/`

[8] `http://duckduckgo.com/1/c/RDF_data_access`

[9] `https://developers.google.com/web-search/`

presentation, even if it has been rescheduled by the conference organizers. SimpleSchedule can be part of the conference management process and exposes the conference schedule as a RESTful Web service. SimpleSchedule exports event data in ICS[10]. An example of SimpleSchedule integration in DataConf can be seen in the DataConf homepage (Fig. 1.a), where all the different types of events (tracks, keynotes...) are displayed.

- **DataPaper**: Custom datasource that allows conference actors (authors, chairs) to provide external resources about their own descriptions and publications. For this, DataPaper stores information about the URIs, description texts and format of these external resources in a NoSQL database (CouchDB) and can be queried by DataConf using the database Web API. These data are stored in key-value pairs, using for instance a publication URI in the conference dataset as a key and the URI of a web page that gives more details about this publication as a value. We also developed a backoffice interface for enriching the DataPaper database, in which conference actors (identified by their foaf:profile and role in the dataset) can add external resources linked to their profile or publications. DataConf then queries this service using the URI of an entity present in the conference dataset, in order to retrieve all the external resources that DataPaper stores about this entity. Even if the backoffice interface is freely available as an unofficial WordPress plugin, we suggest to keep using our DataPaper instance for later conferences, in order to keep the benefits of the contents added by former conference actors about their profiles. Fig. 1.b. shows how DataPaper contents, such as photo and contact information, can be integrated to enrich an author's view, using the data he/she already entered in DataPaper.
- **Reasoner**: Custom "local datasource" that performs client-side reasoning on the conference publication keywords. It is based on a keyword ontology that is loaded at initialization time. During user navigation, DataConf stores the keywords of the publications viewed by the user. The reasoner datasource allows retrieving super and sub-keywords (and thus the publications they refer to), and recommending to users publications they did not yet see and referring to keywords close to those they have already explored. This is performed using an adapted version of the OWLReasoner JS engine, that is embedded in a Web worker and queried in SPARQL (with limitations[11]), la JSONP. Thus, its data can be processed and integrated in the views in the same manner as if it was a distant endpoint. Fig. 1.c shows an example of recommendation interface in the DataConf ESWC'2013 instance.

4 Conclusions

DataConf is a mobile Web mashup designed w.r.t. recent Web standards and technologies and so that all computation is done on client side. It proposes

[10] `http://www.ietf.org/rfc/rfc2445.txt`

[11] Currently, our version of the OWLReasoner engine can process subClassOf queries in SPARQL, but with the constraint of having only one triple per query. In addition, OWLReasoner does not support literals and therefore, data properties.

various features, like querying and browsing conference publications and allowing users to access and enrich the metadata from two different SPARQL endpoints, four Web services and a local inference engine. Conference chairs can easily configure a DataConf instance as soon as the conference dataset is available on a SPARQL endpoint. We also designed this application to be reusable and extendable by Web developers: it uses the Backbone.js framework and relies on view templates and simple components that encapsulate datasource processing logics. Several DataConf instances are currently available and query datasets of different conferences, including ESWC'2013.

In the future, we hope that this application will gain audience in the Web and Semantic Web communities, so that it can provide even more other useful services for scientific conference attendees. The next envisaged version should allow users to authenticate, in order to gain service personalization and social features. Moreover, as templates and datasources can be changed and specified in a configuration file, we plan to explore the reusability of our application infrastructure for another application field than a conference, in order to evaluate the interest of this architecture as a lightweight framework for mobile Web mashups.

References

1. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. International Journal on Semantic Web and Information Systems (IJSWIS) 5(3), 1–22 (2009)
2. Müller, K., Heath, T., Handschuh, S., Domingue, J.: Recipes for SemanticWeb dog food - The ESWC and ISWC metadata projects. In: 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference (ISWC 2007+ASWC 2007), Busan, South Korea, November 11-15 (2007)
3. Müller, K., Bechhofer, S., Heath, T.: Semantic Web Conference Ontology (2009), `http://data.semanticweb.org/ns/swc/ontology`
4. Internet Calendaring and Scheduling Core Object Specification (iCalendar) - RFC 2445 - IETF (November 1998), `http://www.faqs.org/rfcs/rfc2445.html`
5. Hinchcliffe, D., Benson, J.: EMML changes everything: Profitability, Predictability & Performance through Enterprise Mashups. Hinchcliffe & Company 1940 Duke Street, Alexandria, Virginia 22314 1-877-HIN-CHCL (877.446.2425) (2009), `http://openmashup.org/whitepaper/docs/oma_whitepaper_120309.pdf`
6. Hempton, G.L.: The Top 10 Javascript MVC Frameworks Reviewed (2012), `http://codebrief.com/2012/01/the-top-10-javascript-mvc-frameworks-reviewed/`