WWW'2012 posters app: augmenting the WWW'2012 posters session

Lionel Médini
lionel.medini@liris.cnrs.fr

Nguyen Hoang Tan
duy-tan.nguyen-hoang@etu.univ-lyon1.fr

Barthélémy Deluy
barthelemy.deluy@etu.univ-lyon1.fr

Introduction

Our mobile Web application targets WWW'2012 conference attendees, owning a recent, internet-enabled mobile device and being physically in front of one of the 100 posters displayed at the conference. Each poster is associated with a QR-Code; flashing a code leads to the poster homepage in the application. It provides extra information about this poster and allows users navigate among posters, authors and other publications metadata, communicate with posters authors who could not attend the conference and access other functions.

As it targets a WWW series conference, technical choices were oriented towards a full Web application applying recent/emerging Web technologies and standards. In particular, HTML5 APIs encouraged us designing the application as client-sided as possible. We thus explored the feasibility, using currently available browsers on mobile devices, of: (i) dynamically constructing and sending SPARQL queries to several endpoints using the CORS recommendation, (ii) representing and allowing browsing among these metadata using a textual and a graphical interface, (iii) locally building, classifying and querying an ontology and (iv) capturing and processing images using the device built-in camera. Before the conference, a beta version of the application can be tested on the WWW'2011 edition metadata. This version mostly focuses on metadata access and browsing, as miscellaneous functions are still under development.

This paper is organized as follows: we first present the two main functions related to metadata querying and processing, followed by miscellaneous functions and general architecture of the application. We then conclude and discuss the advantages and limitations of this app.

SPARQL Endpoints interrogation and navigation among metadata

From a poster homepage, conference attendees can view its metadata (title, authors, abstract) and access its 2-page PDF description that was accepted by the posters session PC. Posters metadata are enriched so that authors and keywords are navigable. Clicking on an author provides information such as name, most probable homepage, affiliation organization, co-authors, publications in the posters session and other publications (extracted from the DBLP database). Among these data, the organization, co-authors and publications items are navigable. The "organization" page displays the name, homepage and other authors referred in DBLP for this organization, and the "Other publications" page displays DBLP-specific metadata about the publications (title, DOI URI, year, publication type and name of the conference / journal). The co-authors list of an author is a typical Linked Data usage example of the available data: it is generated after parsing the other publications list and gathering each all authors of each publication. Another feature of our application is that from an author or publication view, users can browse the DBLP database. We first wanted to limit this possibility to prevent users from browsing too far away from the WWW'2012 posters session. But we chose to allow it, as it seemed interesting for evaluating the interest conference attendees would find in our interface for browsing large publications databases. To extract end enrich metadata, we use 3 SPARQL endpoints:

- SWDF (http://data.semanticweb.org/) contains (among other conferences) the metadata about the WWW'2012 conference. We rely on this endpoint for accessing posters metadata and PDF files. SWDF is accessed using the CORS (http://www.w3.org/TR/cors/) recommendation working draft.
- DBLP / L3S (http://dblp.l3s.de/d2r/) provides a wider dataset, used to enrich SWDF posters metadata, such as other publications of an author; but it lacks publications keywords and (obviously!) does not provide access to the PDF files. As L3S is not CORS enabled, this server is accessed using the JSONP technique (http://www.json-p.org/).
- DuckDuckGo! RDF data access (http://duckduckgo.com/1/c/RDF_data_access) is a search engine that can be queried in SPARQL. We also use it for metadata enrichment. As we do not want to retrieve all data satisfying a query but get the most probable value (e.g. get an author's or organization homepage), we use its "I'm feeling ducky" feature. DuckDuckGo! is accessed using JSONP.

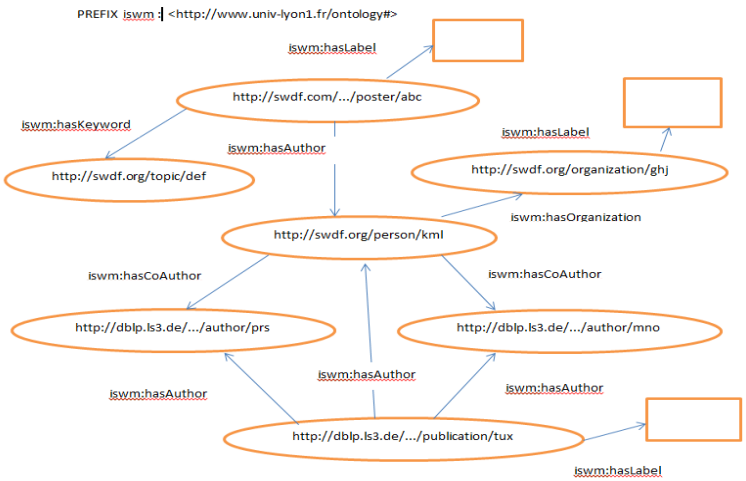

Figure 1. Example of the ontology generated from a poster URI.

Local ontology enrichment and reasoning

Our application also allows browsing by keywords. As DBLP does not provide information about publications keywords, navigable keywords list is only accessible for posters and other publications that have an entry in SWDF. For each of those publications, our application dynamically constructs an ontology composed of the publication, its keywords, its authors, their other publications and the keywords of these publications. The ontology structure is shown in Figure 1.

Once the data gathered, we launch a classifying process using the OWLReasoner (http://code.google.com/p/owlreasoner/) JS inference engine in order to be able to launch SPARQL queries on keywords (e.g. suggesting other publications that match the same keywords, more generic ones, or publications of the same authors that match the user's keywords). For this, we rely on the taxonomy of the conference keywords (extracted from the call for papers pages) and the locally stored user's keywords in this taxonomy.

Other functions

Other features are also provided by our application (or will be, as some are still under development at the time of submitting this paper):

- Flashing a QR-code inside the Web application: in browsers that support the getUserMedia API or the capture input type, it is possible to take a photograph of the QR-code associated to a poster and send it to the server. It is then decoded using the ZXing library (http://code.google.com/p/zxing/) and the user is redirected to the poster homepage.
- View metadata as graphs: users can choose to view and navigate in tree-based graphical representations of the publications, authors or keywords. The graphical representation is built using the JavaScript Infovis toolkit (http://thejit.org/).
- Publication and author search: a form allowing to access directly a poster or author's description page.
- Extra information access: URLs of authors-provided extra material (detailed PDF, video presentation…) are stored in our database and displayed in the homepage of the corresponding poster.
- Chatting: users dispose of a Web chat client connected to our XMPP server and can participate in conversations about posters or conference keywords. Specific chatrooms are created for posters and keywords. Messages updates in the current version is based on HTTP refresh but we plan to use the WebSocket API in future versions (as soon as Tomcat will support this API).
- Ask question about one part of the poster: using the same process as for flashing QR-codes, users can take a photo of one specific part of a poster and send a question about it by email. The email addresses of the authors are automatically extracted from the PDF files. Sending the email is done out of the application, using the device built-in email client.
- Vote for a poster: during the conference, users will be able to vote for the Best Poster and Best Student Poster awards.

Application architecture

As previously said, most functions provided by our application are performed on the client. The client side of our app is based on HTML5 pages and various scripts. It uses scripts from jQuery (http://jquery.com), jQuery validation plugin (http://bassistance.de/jquery-plugins/jquery-plugin-validation/) and CSS3 templates from jQuery UI (http://jqueryui.com/).

The server is only used to store extra data about posters (PDF, video presentations…), to decode the QR-codes and as a gateway to allow users to chat over XMPP. We use nginx (http://nginx.org/) as front server, Tomcat (http://tomcat.apache.org/) as servlet container and Openfire (http://www.igniterealtime.org/projects/openfire/) as XMPP server. The library used for decoding QR-Codes is ZXing (http://code.google.com/p/zxing/).

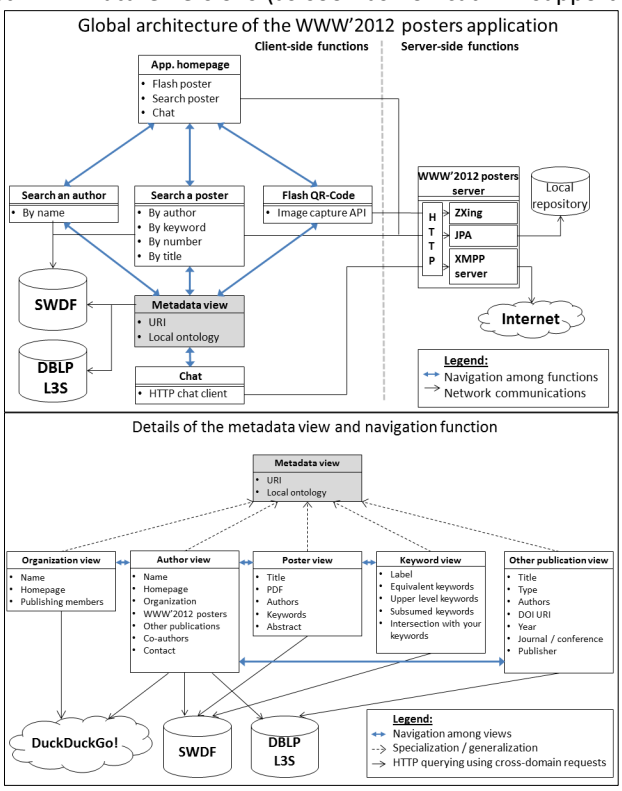

Figure 2. Schemas of the application architecture.

Conclusion

Our WWW'2012 posters application proposes various features about the WWW'2012 posters session. It has been designed w.r.t. recent Web standards and technologies and so that as much computation as possible is done on client side: we use cross domain requests to access and enrich the metadata from three different SPARQL endpoints; during the navigation, the application constructs a local ontology and uses it for suggesting publications to the user. Server-side dynamic content is only used to propagate authentication requests, store extra data about the posters and as a gateway to the XMPP server that provides chat service. In this paper, we highlight access and navigation in the posters and other publications metadata. A (supposedly) stable version of the app can be tested at http://ter.barthdeluy.com:8080/www2012-stable/paper/view/1.html.

First experiments show that recent smartphones perform well for relatively heavy computations, such as multiple cross-domain requests bringing large amounts of data, graph construction and view, reasoning over a (small) ontology. Therefore, many technologies are much more device/browser-dependent than expected: few browsers currently support the getUserMedia API or the capture input type that allows flashing QR-Codes from inside our app (of course they can also be flashed using a non-web application to retrieve a poster homepage). Thus features provided by our app on different browsers may vary depending on the technologies they embed, but the core functions (i.e. metadata navigation) will be available on all recent browsers.

Moreover, the CORS recommendation needed for sending cross domain requests to the metadata servers is neither supported by all browsers, nor by all servers. As webapps perform more and more client-side computations, a suggestion that could be made for SPARQL endpoints administrators to help Linked Data development is to conform to this recommendation, as it is now stable and as these data are not supposed to be concerned by security issues.