An extensible architecture for an ecosystem of visualization web-components for Open Data

Gennaro Cordasco[2], Delfina Malandrino[1], Pina Palmieri[1], Andrea Petta[1], Donato Pirozzi[1], Vittorio Scarano[1], Luigi Serra[1], Carmine Spagnuolo[1], Luca Vicidomini[1]

[1]ISISLab, Dipartimento di Informatica, Università di Salerno (Italy)
[2]Dipartimento di Psicologia, Seconda Università di Napoli (Italy)

Abstract

We present here an architecture for open, extensible and modular web-components for visualization of Open Data datasets. The *datalet*s are web-components that can be easily built, and included into any standard HTML page, as well as used in other systems. The architecture is developed as part of the infrastructure needed to build a Social Platform for Open Data (SPOD) in the Horizon 2020 funded project ROUTE-TO-PA (www.routetopa.eu). We present the motivations, the examples and a sketch of the architecture. The software is currently under development, in a very early stage, but is already available, with MIT open source license, at deep.routetopa.eu.

I. Introduction

Open Data is available to anyone, who can freely use, modify, and share them for any purpose. With the advent of the Internet and the recent government initiatives (e.g., Data.gov.uk), open data are gaining even more interest since they can facilitate transparency, ensure accountability, and engage citizens and organisations, fostering their participation in public discussions and activities.

Open data is an inestimable initiative in favor of transparency [1]. Governments started to publish over the Internet making available a variety of data, such as balance data (e.g., OpenBilanci[1]), location-based data (e.g., Issy-les-Moulineaux data[2]). This is an initial step to reduce the gap between governments and citizens, but the road is still long because often datasets are in non-machine-readable format (e.g., scanned documents), aggregated, not cleaned and without an explanation of the fields meanings. Therefore, open data are there, ready to use at least for skilled and expert users, but effectively unusable for the most of citizens. Furthermore, the publication of Open data itself through the upload of data on a public institutional web-site does not means transparency; and Open Data and transparency are basically not synonymous. Transparency means that citizens know what their government is doing.

We believe that the visualisation of Open Data is a key tool to make understandable open data. This work contributes with an ***open source, extensible, modular and pluggable architecture for the discovery, distribution and usage of off-the-shelf visualisations***.

Our research is conducted within the project "*Raising Open and User-friendly Transparency-Enabling Technologies fOr Public Administrations*" (ROUTE-TO-PA, www.routetopa.eu), a Horizon 2020 European funded innovation project that combines expertise in the fields of e-government, computer science, learning science and economy, to improve the impact of ICT-based technology platforms for transparency. ROUTE-TO-PA focus is in contrasting the potential barriers to using ICT for transparency, that we identify in the complexity of the information provided, the lack of tools to facilitate the comprehension and the limited acceptance of ICTs for transparency among citizens. [2,3].

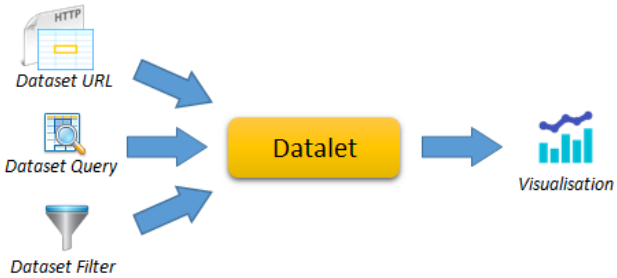
Concretely, the project is aiming to (a) develop a Social Platform for Open Data (SPOD) enabling social interactions among open data users and between open data users and government data; (b) build Transparency-Enhancing Toolset (TET) as extensions for existing major Open Data Platforms; and (c) develop a set of recommendations (GUIDE) as good practice guide for open data publishers for achieving higher quality

[1] OpenBilanci open-data web-site: http://www.openbilanci.it
[2] Issy-les-Moulineaux open data web-site: https://data.issy.com

transparency through open data. The architecture described here is ROUTE-TO-PA technological foundation to the development of SPOD and TET.

II. Datalets

In this paper, we introduce for the first time the main actor of our architecture, i.e., *Datalets,* A datalet is a off-the-shelf, reusable web-component able to load, query, filter, and visualise any dataset content (through HighCharts Javascript library). It is not just a visualisation library, but a whole building block that is to perform a sequence of steps to visualise dataset content (e.g., open datasets). This web-component is reusable since it can be placed in every web page, like institutional web-sites, blogs, forums and so on, without any knowledge on how it effectively is implemented. A datalet is interoperable and can query, filter, and visualise data from different sources. A datalet takes in input a dataset URL, a query to perform on the data, and a filter, and optionally additional configuration parameters. Currently, available datalets on deep.routetopa.eu are: *classical table-based visualisation, bar chart, column chart, line chart, column chart, map, treemap, area chart, 3D donut chart and column 3D chart*.



Datalets are an important visualization building block when one considers requirements of quality of the dataset. Data provenance, in fact, is crucial to determine whether information is trusted, who is the data originator and how to trust him, whether the data have been manipulated. Practically speaking, every time a dataset is manipulated, processed or visualised, the provenance ensure the trackability of the changes along the time. **Datalets guarantee the provenance of data** by (1) loading the dataset from the external source without changing the data, (2) providing the URL of the data source so that the user can check the visualised data on the original source, (3) introducing a certified repository of datalets. Datalets can be statically embedded in any web-page (e.g., blog, forum, institutional web-site, and so on) through a copy-and-paste of its source code. The following picture shows the map datalet embedded within our official Wordpress Installation web-site. It loads the data from a CKAN (Comprehensive Knowledge Archive Network[3]) example repository as shows the "source" text below the map.

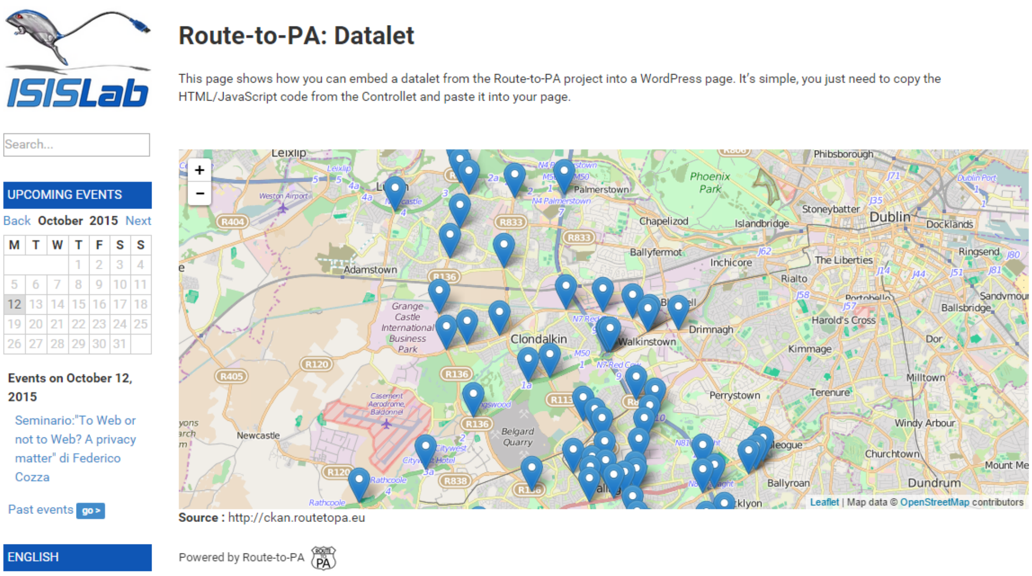


Fig. 2 - Datalet embedded within a Wordpress web-page.

Datalets have been designed to process any dataset as input. In the current implementation, datalets can visualise data from CKAN installations, and from commercial providers, such as OpenDataSoft, and Enigma. Work is under progress and further dataset providers will be covered.

[3] The open source data portal software http://ckan.org/

The datalet runs on the client-side within the user browser, thereby ensuring scalability on server-side: no additional heavy load is placed on the server, besides unloading few additional lines of Javascript. In order to visualise a dataset, the following steps need to be performed. The user types the url of a web-page which contains a datalet. The browser during the page content loading, takes the datalet URL and connects to the datalets repository. Then it loads the datalet source code to be executed (different visualization are rendered using different datalets). The datalet takes in input any external dataset URL and a query to recover the data. An optional filter can be applied to further refine data. Finally, the datalet renders an interactive visualisation by using the filtered data.

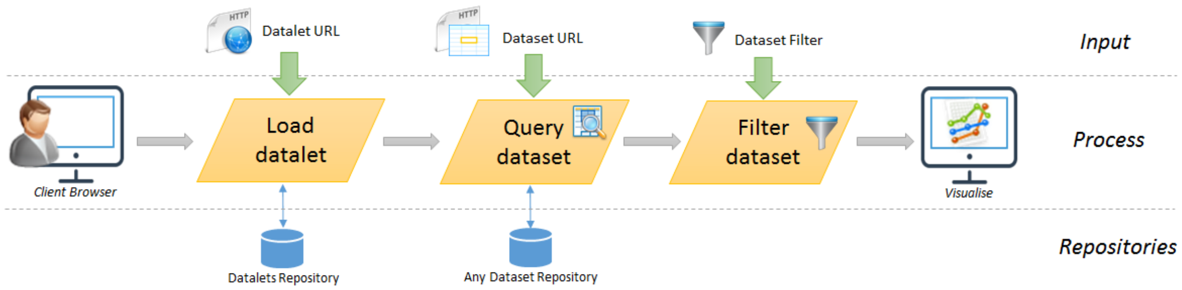


Fig. 3 - Process to load the datalet, query the dataset through its URL, filter the data and visualise it. All the processing is performed on the client-side within the user browser.

From a technical point of view, a datalet is a web-component developed with Polymer[4], so it is a mix of HTML, CSS, and Javascript content that can be included in any web-page. All the processing is performed client-side by the user browser. The datalet can query data providers provided that they return data in JSON or CSV format. All the datalets inherit from a base web-component called base-datalet.

III. How to generate a datalet

As we said, A datalet is a visualisation component that can be statically embedded within any web-page, but its creation can be complex and we developed several ways to generate a datalet.

The first method, used in the SPOD we are developing, is generating through a *controllet*, a wizard that guide the users through multiple steps in order to provide easily (for users) datalets inputs (e.g., dataset url, type of visualisation). In the first step, the user provides a reference to an external dataset. He/she can select the dataset from a popular list or can search it from a set of pre-configured data providers. In addition the controllet supports the loading of data from any external data provider by typing the dataset url. In the second step, the controllet loads the dataset content and shows the list of dataset fields. Then the user according to the desired datalet selects the fields to visualise. For instance the datalet map requires the longitude and latitude fields to place pinpoints. In the third step, the controllet shows a list of available datalets and the user assigns, using drag-and-drop, the fields selected during the second step to the datalet fields. For instance, Fig. 4 shows on the left the selection of a linechart and the mapping between data and datalets field; on the right a preview of the datalet. At the end, the controllet returns the HTML code that is (by the server side code) included and made permanent in the SPOD

The second way is to use a *Datalet Creator*, shown in Fig. 4, that is a static page that offers the same steps offered by the controllet, but at the end it simply allows to copy-and-paste the HTML code everywhere, thereby including the datalet statically into any HTML editor. Among the most popular CMSs, we verified the full compatibility with WordPress and Tumblr, More work is planned toward other platforms.

[4] Polymer project to develop web-components https://www.polymer-project.org

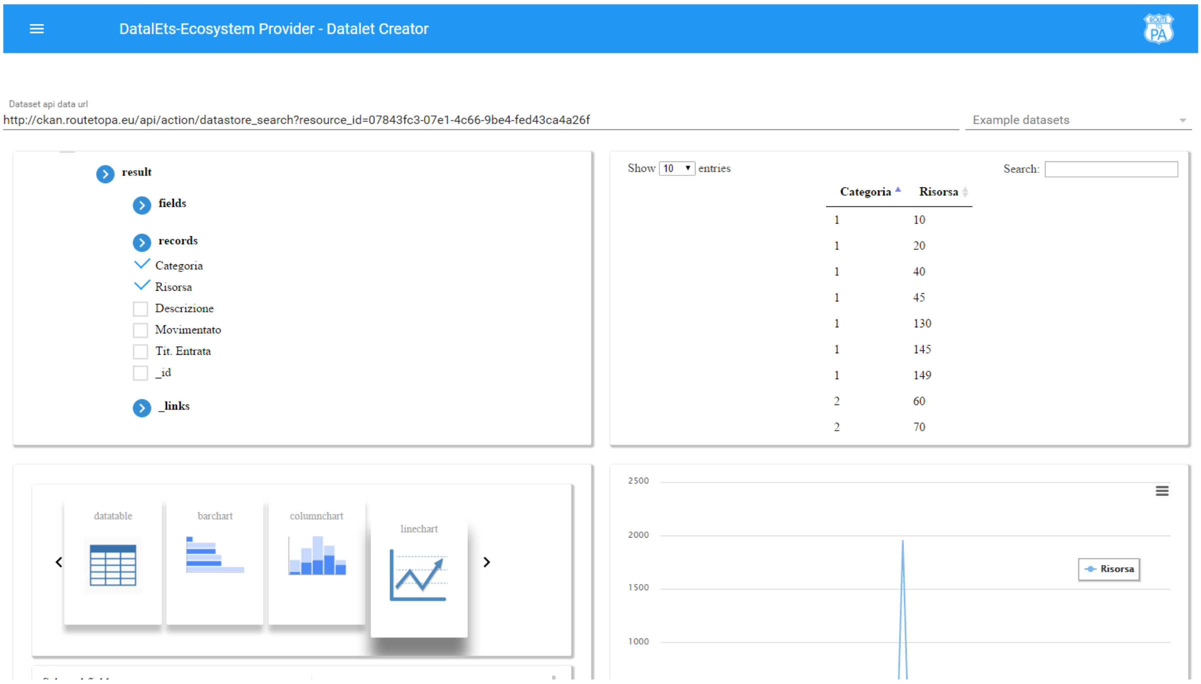


Fig. 4 – Datalet Creator (top part of the page). The interactive Controllet is simply a wizard with the same content, with three steps to choose dataset, fields and visualization.

IV. A sketch of the architecture

This section describes the architectural model that provides the following services: storing and lookup datalets. This section is essential also to describe how the provenance has been guaranteed.

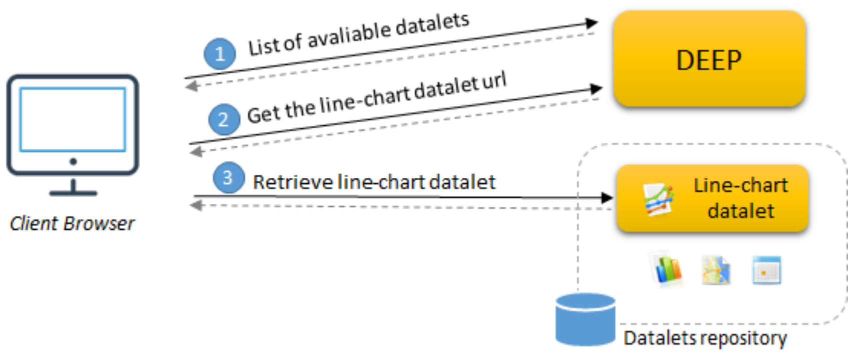


Fig. 5 - DEEP and datalet repository architecture. The DEEP provides the list of available datalets, and given the datalet name provides its implementation loaded from the repository.

The DatalEt-Ecosystem Provider (DEEP) component is a simple Restful service, providing the list of available datalets (i.e., listing service) and the mapping among the visualisation names and their relevant URL within the datalet repositories. The controllet described in the previous Section is based on this architecture. The controllet exploits the DEEP list service to get and show the list of available datalets visualisations. When the user selects a datalet during step three, the controllet using the corresponding URL, provided by the DEEP, download the datalet from a datalet repository.

Both the DEEP and the datalet repository have been designed to be extensible: they can collect all the visualisation requests so, as planned future work, they could also provide aggregated statistics on both users preferences and on data and their visualisations. For instance, the most popular datalet visualisations, most used datasets, most popular visualisations for a particular dataset, most visualised field for a particular dataset, and so on.

V. Conclusions and ongoing work

Our work is available already as open source software (MIT license) on deep.routetopa.eu and will be developed until January 2018 within the project ROUTE-TO-PA. We are at a very early stage, nevertheless, we decided to provide public access to our software as we develop it, with the full spirit of open-source availability. Although preliminary, and thereby subject to changes, we provide for all the datalets and the DEEP server full documentation for each component (datales and controllets), demo, example and fully commented code, and our developers will be mantaining and updating the site regularly, providing feedback on comments and bug reporting.

Beside being only partial, we are aware of several important issues that remain to be tackled. In fact, besides improving compatibility with dataset providers, and offering a wider choice of many diverse visualizations for datalets, we also plan to study and resolve several architectural issues.

First of all, we will deal with the scalability of DEEP workload, i.e. trying to achieve efficiency server-side. In fact, DEEP component and Datalet repository are centralised. This means that every single request to create a dynamic datalet will be processed by the DEEP, and each time a datalet is loaded and visualised, the datalet repository will serve the content. Both DEEP and Datalet repository process all the requests so they are a single point of load and failure (SPOF). In order to overcome these issues we provide a scalable, replicable with high availability architecture. Scalability means that the overall architecture is able to accommodate and process even more visualisation requests and manage burst of requests. Our plan is to design replicated DEEP component and datalets repositories, distributing them across multiple sites providing load balancing. This means that the architecture will evolve towards a federation of DEEPs, ensuring in any case the datalets certification. In this way, the architecture is distributed but have a central monitoring and management. Furthermore, distributed DEEPs and datalets repositories can be updated .

Then, efficiency from client-side will be tackled. by introducing caching mechanisms on the browser side. Datalets read data from external sources, that could be unavailable at some point for different reasons; in order to improve datalets loading performances and deal with unavailable data, caching mechanism will be designed and introduced in the architecture, carefully considering the impact on the provenance.

Finally, another interesting evolution is leverage on users statistics on the datalets usage (e.g., most used visualisations for a particular dataset) to suggest “good” or “popular” datalet for a given dataset or for a given type of dataset (see also [4]). Proving statistical usage in a distributed environment means that periodically a collecting and merging operations must be scheduled in order to update overall statistical data. The same federation can provide replicability by replicating the architectural components. This scalability improvements will be quantitatively measured through performance benchmarks.

Acknowledgments

The research leading to the results presented in this paper has been conducted in the project ROUTE-TO-PA (www.routetopa.eu) that received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 645860. The authors acknowledge several interesting discussions and comments with the ANCITEL S.p.a. team in ROUTE-TO-PA during the presentation of the architecture.

Bibliography

[1] Bertot J.C., Jaeger P.T. & Grimes Justin M. (2010). Using ICTs to create a culture of transparency: E-government and social media as openness and anti-corruption tools for societies,

in Government Information Quarterly, Volume 27, Issue 3, July, pp. 264-2.

[2] EU eGovernment Action Plan (2010). Europe eGovernment Action Plan 2011-2015, Harnessing ICT to promote smart, sustainable and innovative Government. COM(2010) 743 final.

[3] EU DG-CONNECT (2013). A vision for public services. European Commission, “Public Services” unit of Directorate-General for Communications Networks, Content and Technology (DG-CONNECT). June 13th, 2013.

[4] Thellmann, Klaudia, et al. "LinkDaViz–Automatic Binding of Linked Data to Visualizations.", The 14th International Semantic Web Conference (ISWC), Bethlehem, Pennsylvania, 2015