Public Data Integration with WebSmatch*

Emmanuel Castanier$^{(a)}$, Remi Coletta$^{(a)}$, Patrick Valduriez$^{(a)}$, Christian Frisch$^{(b)}$
(a) INRIA and LIRMM, Montpellier, France (b) Data Publica, Paris, France
(a) {FirstName.Lastname}@inria.fr (b) christian.frisch@data-publica.com

Abstract

Integrating open data sources can yield high value information but raises major problems in terms of metadata extraction, data source integration and visualization of integrated data. In this paper, we describe the demonstration of WebSmatch, a flexible environment for Web data integration, based on a real, end-to-end data integration scenario over public data from Data Publica[1]. WebSmatch supports the full process of importing, refining and integrating data sources and uses third party tools for high quality visualization. We use a typical scenario of public data integration which involves problems not solved by currents tools: poorly structured input data sources (XLS files) and rich visualization of integrated data.

1 Introduction

Recent open data government initiatives, such as `data.gov`, `data.gov.uk`, `data.gouv.fr` promote the idea that certain data produced by public organizations should be freely available to everyone to use and republish as they wish. As a result, a lot of open data sources are now available on public organization's web sites, in various formats.

Integrating open data sources from different organizations can yield high value information. For instance, matching gas emission data with climatic data for a given country or city can be valuable to better understand pollution. This rich local and targeted pool of information can also be leveraged to build new innovative services or, as a new source of business intelligence, to put in perspective business information with data such as weather, traffic, density of economic activities or touristic information in order to better understand current market dynamics and adapt product and services.

A large share of the available open data comes from large institutions (such as Eurostat, World bank, UN....) using structured data formats such as SDMX for statistical datasets or RDF for linked open data. However, the majority of the data that can be found on open data portals is available as unstructured data (such as spreadsheets). To integrate these sources or deliver data that web applications or mobile services can leverage, raw open data files must be structured through a processing workflow and delivered through APIs (Application Programming Interfaces). This workflow will ultimately transform "human usable information" such as spreadsheets into "computer usable data", drastically increasing the value of the open data published.

Based on this observation, Data Publica, a french company, provides added value over the public data sources they crawl, such as visualization of data sources or production of integrated data. Achieving this goal raises the followings problems:

Metadata extraction. Although structured formats exist to share and publish data, most of the public data available on the Web are Excel spreadsheets, with no difference between data and metadata. Detecting the metadata in such data sources is a mandatory step before performing data integration. To address this problem, we exploit computer vision techniques to deal with complex tabular representations of spreadsheets and machine learning techniques that take advantage of past human effort to automatically detect metadata in the next spreadsheets.

Data sources integration. In order to produce added value information over the public data sources, it is necessary to integrate data sources together. For this purpose, we need to perform schema matching, in order to match metadata structures. In the context of open data, schema matching is harder than in traditional data integration in distributed database systems [6], mainly because important metadata which are considered as implicit by document's authors, are simply

*The attached video of the whole demo is also available at `http://websmatch.gforge.inria.fr`
[1]`http://www.data-publica.com`

missing. In terms of matching capabilities, we rely on YAM++ [5], a powerful tool for schema matching and ontology alignment[2].

Visualization. To ease users's access to public data requires visualizing with high quality graphical representation. In Data Publica, the visualization task is delegated to Google Data Explorer, a powerful collection of visualization tools. However, Google Data Explorer imposes strict restrictions on input formats, such as separating data and metadata into different files and labeling metadata with some Google predefined concepts. Therefore, using Google Data Explorer requires metadata extraction and integration as preliminary steps.

To perform these tasks, Data Publica uses WebSmatch `http://websmatch.gforge.inria.fr/` , an environment for Web data integration with a service-oriented architecture with much flexibility for users and developers. Most tools for metadata integration are implemented as heavy clients and hard-coded with their own graphical interfaces. They need to be downloaded and installed, which make them hard to use with other independent tools (even if sources are provided) and reduce their dissemination. In contrast, WebSmatch is an open environment to be used as a Rich Internet Application (RIA).

In this paper, we describe the demonstration of WebSmatch based on a real-life, end-to-end data integration scenario over public data-sources.

The paper is organized as follows. Section 2 introduces the scenario of the demonstration in terms of inputs (poorly structured files) and outputs (rich visualization of integrated data). Section 3 presents WebSmatch metadata detection and integration services through the motiving example. Section 4 concludes.

2 Demonstration scenario

In this section, we describe our demonstration scenario by giving the inputs and outputs of the data integration process with WebSmatch.

Data Publica provides more than 12 000 files of public data. [1] However, even though data formats become richer and richer in terms of semantics and expressivity (e.g. RDF), most data producers do not use them much in practice, because they require too much upfront work, and keep using simpler tools like Excel. As an example, Data Publica has started to crawl public data available from the French administration, and found only 369 RDF files, compared with 148.509 .xls files. Unfortunately, no integration tool is able to deal in an effective way with spreadsheets. As far as we know, only two recent initiatives, OpenII [7] and Google Refine [3] deal with Excel files. However, their importers are very simple and make some strict restrictions over the input spreadsheets. For instance, they require to have exactly one table per sheet and all the attributes have to be in columns, at the first line of the sheet. Unfortunately, people do not use Excel in such proper way. And these importers proved to be useless on real spreadsheets from Data Publica. Thus, extracting metadata from such sources remains an open problem [3]. To illustrate this problem in the emonstration, we use the following spreadsheet files as input.

Input files

For simplicity purposes, the scenario of this demo involves only 2 data sources. To be representative of real-life public data, we choose two spreadsheet files:

`http://www.data-publica.com/publication/1341` is an Excel file. It contains data from the Climatic Research Unit (`http://www.cru.uea.ac.uk/`) about the temperature evolution in the world over the last decades. This file is quite well formed, it only contains some blank lines and comments.

`http://www.data-publica.com/publication/4736` is the Excel file depicted at the beginning of the video. It contains data from OECD (`http://www.oecd.org/`) about gas emissions in the world. The file contains the evolution on the last 20 years on several countries and 4 OECD geographic zones[4]. This spreadsheet is much more complex: it involves several sheets, with several

[2]YAM++ was recently ranked first at the Conference track of the OAEI competition over 15 participants. See the results at `http://oaei.ontologymatching.org/2011/` for more details.

[3]`http://code.google.com/p/google-refine/`

[4]See `http://stats.oecd.org/glossary/` for more details about these zones.

tables per sheet. It contains several blank lines and comments, making it hard to automatically detect the table. In addition, it involves bi-dimensional tabular structures and some column names are missing. For instance, the author of this file probably has in mind that the line containing $\{1995, 2000\}$ should be labelled by "year", which is not obvious in the context of automatic integration.

Expected results

Video presents the charts, maps and additional animations with timelines obtained after extraction of metadata and integration of the inputs described above.

To perform visualization, Websmatch exports the integrated data in Data Set Publishing Language (DSPL) format `https://developers.google.com/public-data/`) . DSPL is used by Google Public Data Explorer and Data Publica's own API and visualisation engine. Such format assumes the input data source to be precisely described. In particular, data and metadata need be distinguished. The metadata (source, title, author, header) are described in an XML file whereas the data are in Comma-Separated Values (CSV) files. In addition, metadata need to be tagged by some DSPL predefined concepts (hierarchy including times or geographical entities). Such format is too strict to be usable by a large public, and quite difficult to manipulate, even for computer scientists. Thus, although Google Data Explorer provides a powerful collection of visualization tools, it requires much upfront work from the user, in particular, with public spreadsheets like the ones described above.

3 Demonstration

WebSmatch is a Rich Internet Application (RIA), meaning that any third party is able to use it remotely, as a Web service, without any installation. To use all the WebSmatch components (integration, matching, clustering and export), one simply needs to put some redirection from his back office. The integration of WebSmatch and Data Publica is depicted in Figure 1. It involves the following flow:

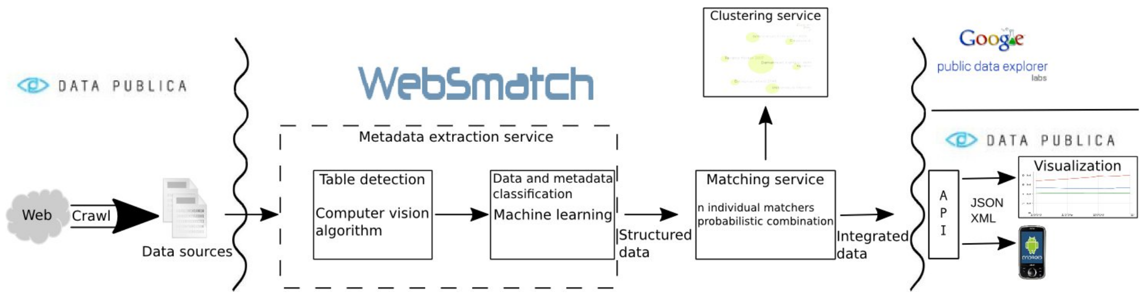


Figure 1: Data Integration process

Metadata Detection

After the Crawl, the user is redirected to the WebSmatch RIA. It is important to note that Excel files (such as .xls, for which there is no XML version) are not structured at all. As can be seen in the video, they can contain lots of artifacts such as blank lines, blank columns, titles, comments, and not only a simple table. To get all the metadata and data, the chosen file is parsed and then, two processes are applied to it. The first process relies on a combination of computer vision algorithms. Using the jexcelapi[5] library as a wrapper, the spreadsheet is first translated into a 0/1 bitmap (0 for void cell / 1 for non empty cell). In this bitmap, we run a connected component detection algorithm. Algorithm 1 takes as input a function indicating the color of a point in a

[5]`http://jexcelapi.sourceforge.net/`

bitmap (in our case, a datatype of a cell) and within a one step linear parse of the matrix, assigns a connected component to each cell.

```
Algorithm 1: TABLE DETECTION WITH CONNECTED COMPONENT
  input  : type(i,j): a function returning the datatype of each cell
  output: cc(i,j) : a function returning the connected component of each cell
  foreach $0 < i < n$ do
      foreach $0 < j < m$ do
          if $cc(i-1, j) \neq null$ then $cc(i, j) \leftarrow cc(i-1, j)$
          else $cc(i-1, j-1) \neq null$ $cc(i, j) \leftarrow cc(i-1, j-1)$
          else if $cc(i, j-1) \neq null$ then $cc(i, j) \leftarrow cc(i, j-1)$
          else if $cc(i-1, j+1) \neq null$ then $cc(i, j) \leftarrow cc(i-1, j+1)$
          else if $type(i, j) \neq void$ then
              $cc(i, j) \leftarrow new\ ConnectedComponent()$
```

Algorithm 1 allows us to partition the spreadsheet into regions. We then use more sophisticated computer vision approaches, such as morphologic transformation [2] and erode / dilate functions to refine the result of the connected component detection: remove too small connected components, merge connected components that have been splitt due to a single void line, etc... In the graphical interface (see video), the detected tables are drawn within a frame.

To decide whether data are line- or column-oriented, we exploit the following idea: if data are presented in lines, the datatypes of cells for each line are homogeneous, but the datatypes of cells for each column may be heterogeneous. We then compute the discrepancy in terms of cell datatypes for each line (1) and for each column (2). If (1) > (2), then the metadata are probably on the first lines, on the first columns otherwise.

$$\sum_{0<i<n} \left(\max_{t \in \{string, int, \ldots\}} \left(\sum_{0<j<m} (type_{[i,j]} = t) \right) \right) \quad (1)$$

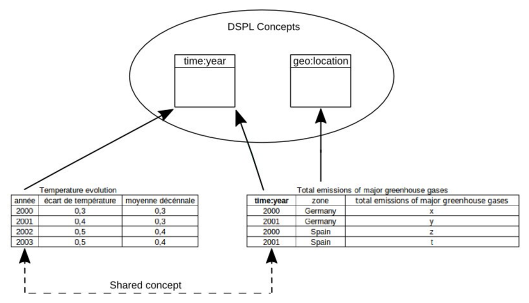
$$\sum_{0<j<m} \left(\max_{t \in \{string, int, \ldots\}} \left(\sum_{0<i<n} (type_{[i,j]} = t) \right) \right) \quad (2)$$

The end of the process relies on machine learning [4]. Using past experience and based on several criterions: the discrepancy measures, the datatype of a cell, the data type of the neighborhood of a cell, WebSmatch detects each important component in the spreadsheet file such as: titles, comments, table data, table header.

Machine learning is able to capture several spreadsheet users habits, such as: "cells on the very first line of a connected component, having the string datatype and bellow cells having a numeric datatype are often metadata" or "cells having the string datatype and void neighborhood and behind a table often are a title". The important feature is that such rules have not been designed by the user, but observed on several documents. They can be updated when new spreadsheets are performed by the user.

Matching

WebSmatch relies on YAM++ [5] to perform the matching task. YAM++ combines 14 different matching techniques, divided in 3 main groups: string matchers, dictionary and thesaurus matchers based on Wordnet[6] and instance-based matchers. Instance-based matcher is the generic name for matchers, which deals both with metadata and data. Such matchers are very useful when the column names are not informational enough, which is often the case in public data. The instance-based matcher implemented in YAM is very powerful and one of the main reasons for



Temperature evolution

année	écart de température	moyenne décénnale
2000	0,3	0,3
2001	0,4	0,3
2002	0,5	0,4
2003	0,5	0,4

Total emissions of major greenhouse gases

time:year	zone	total emissions of major greenhouse gases
2000	Germany	x
2001	Germany	y
2000	Spain	z
2001	Spain	t

Figure 2: Result of integration

[6]http://wordnet.princeton.edu/

YAM++ excellent results at the 2011 competition of the Ontology Alignment Evaluation Initiative (`http://oaei.ontologymatching.org`: first position at the Conference track and second position at the Benchmark track [5].

In the video, we observe that the cell “année” (i.e. year in french), which has been previously detected as metadata, is detected as ”time:year” concept by applying the instance-based matcher on its data collection $\{1990, 1991, \ldots\}$. Figure 2 depicts all the discovered matches over the two files of the scenario and the DSPL concepts we previously imported into the tool.

Notice that the line of the second spreadsheet contains a line within a collection of years but with a void cell as first column. Despite it is void, this cell is detected by WebSmatch to be a metadata. Indeed, it is at the first line and first column of the detected table and our machine learning algorithm detects the metadata to be placed in the first column. By applying the instance-based matcher, WebSmatch suggests this cell to be labelled with the ”time:year” concept.

Visualization

By detecting the blank cell, we are able to convert the bi-dimensionnal table from the initial spreadsheet into a classical (SQL-like) flat table (Figure 2). Thanks to the matching process, we are also able to identify concepts (from DSPL) over the data sources and to detect common attributes in order to produce integrated data.

At this step, we have distinguished data and metadata from the initial Excel files, and flatted bi-dimensionnal tables. We can easily generate an XML file describing the metadata (title, header, concepts) and the `.csv` files containing the data to fit the strict DSPL input format. As a result, we can take advantage of the powerful capabilities of Google Data Explorer in terms of visualization, or the cleaned sources can be exported to feed mash-up applications (such as `http://pipes.yahoo.com/pipes/`). The structured data can also be loaded into Data Publica's database as shown in Section 2.

4 Conclusion

In this paper, we described the demonstration of WebSmatch, a flexible environment for Web data integration, based on a real data integration scenario over public data from Data Publica. We chose a typical scenario that involves problems not solved by currents tools: poorly structured input data sources (XLS files) and rich visualization of integrated data. WebSmatch supports the full process of importing, refining and integrating data sources and uses third party tools for high quality visualization and data delivery. The demonstration will provide additional scenarios, involving more data sources and including data source recommendation using the clustering service. The attached video of the whole demo is also available at `http://www.youtube.com/watch?v=sqeU1lkXW_A`. Furthermore, it can be played with a test account at `http://websmatch.gforge.inria.fr`.

References

[1] F. Bancilhon and B. Gans. Size and structure of the french public sector information. In *gov.opendata.at*, 2011.

[2] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision.* Addison-Wesley, 1992.

[3] F. Hermans, M. Pinzger, and A. van Deursen. Automatically extracting class diagrams from spreadsheets. In *European Conference on Object-Oriented Programming*, pages 52–75, 2010.

[4] T. M. Mitchell. *Machine Learning.* McGraw-Hill Series in Computer Science, 1997.

[5] D. Ngo, Z. Bellahsene, and R. Coletta. YAM++ results for OAEI 2011. In *International Workshop on Ontology Matching*, 2011.

[6] T. M. Özsu and P. Valduriez. *Principles of Distributed Database Systems, third edition.* Springer, 2011.

[7] L. Seligman and al. OpenII: an open source information integration toolkit. In *Int. SIGMOD Conference*, pages 1057–1060, 2010.