Sig.ma: live views on the Web of Data*

Giovanni Tummarello, Richard Cyganiak, Michele Catasta, Szymon Danielczyk, and Stefan Decker

Digital Enterprise Research Institute
National University of Ireland, Galway
Galway, Ireland
firstname.lastname@deri.org

Abstract. We demonstrate Sig.ma, both a service and an end user application to browse and perform tasks leveraging data coming from dozens of distributed and unrelated sources on the Web of Data.

1 Introduction

The amount of Resource Description Framework (RDF) documents, Microformats and RDFa available online has been growing tremendously in the past years. While the recent support for visual result enhancements by search engines (e.g. Yahoo! Searchmonkey) are certainly a notable incentive for marked up data productions, yet there is still a strong need to demonstrate convincing applications that can exploit the capabilities of leveraging multiple, distributed, unrelated data sources when solving a task of interest to the user.

We here demonstrate Sig.ma (`http://sig.ma/`), an application – with a matching developer API - to automatically integrate and make use of information coming from multiple web sources to fulfill the following tasks:

Browsing the Web of Data Starting from a textual search, the user is presented with a rich aggregate of information about the entity likely identified with the query. Queries can be about people as well as any entity type that is likely to have been described on the Web of Data (e.g. locations, name of documents, products, etc.) As the user visualizes the aggregate, she can follow links to other properties, switching the focus of visualization to that of neighboring entities.

Embedding, linking and Sig.ma alerts At any aggregation page, Sig.ma offers rich interaction tools to *expand* and *refine* the information sources that are currently in use as well as some *data oriented* clean-up functionalities to *hide* and *reorder* values and properties. As a result, it is possible to interactively create curated "views" on the Web of Data about a given entity which can be then addressed with permanent URLs, therefore passed in IMs or emails, or embedded using a specific markup in external HTML pages. These views are *live* and cannot be spammed: new data will appear on these views exclusively

* The work presented in this paper has been funded in part by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2) and in part by the FP7 EU Large-scale Integrating Project OKKAM - Enabling a Web of Entities (contract no. ICT-215032).

coming from the sources that the mashup creator has selected at creation time. As a new feature (to be announced by ISWC 2009) Sig.ma offers notifications of changes to profiles: a user can subscribe to a Sig.ma view and receive email alerts whenever new data is aggregated or changed.

Structured property search for multiple entities A user, but more interestingly an application, can make a request to Sig.ma for a list of properties and a list of entities. For example requesting "*affiliation, picture, email, telephone number, [...] @ Giovanni Tummarello, Michele Catasta [...]*" Sig.ma will do its best to find the specified properties and return an array (raw JSON or in a rendered page) with the requested values.

2 Test driving Sig.ma: example user interactions

In this section we will illustrate how Sig.ma presents itself to the end user. We also encourage the reader to see the other examples online on the Website as well as the screencast.

Sig.ma: Axel Polleres

In case of researcher "Axel Polleres", there exist plenty of data sources available: RDF sources such as DBLP, Ontoworld, Semanticweb.org but also Microformat sources provided by BOSS such as Polleres' public Facebook and LinkedIn profiles which, for instance, add more pictures to the mashup. Particularly rich sources such as the RDF coming from the DERI institute team page[1] add data such as his work phone number, some of the publications and related projects etc. As ambiguity on the name is low, pressing "Add More Info" button returns many more relevant results which provide social contacts alternative affiliations from previous employers and more.

The result of a 30 sources aggregation is shown partly in Figure 1.

Following the Web of Data browser paradigms, most of the results are clickable and lead to further information discovery. For example, clicking on the paper titled "Exposing Large Datasets with Semantic Sitemaps" reveals a Sig.ma coming from mostly 3 websites with complementary information such as the coauthors, the tags and alternative locations. Interestingly, a reference to a review of the paper from the Semantic review site (Revyu.com) is also given, see Figure 2.

Sig.ma's with ambiguous names: Galway and Trento

While returning plenty of relevant information, a Sig.ma for "*Galway*" shows the weakness of kickstarting the search with a simple text query but at the same time the importance of user feedback and interaction. Sig.ma will, at this time, not make a difference between Galway the Irish city and, for example, data found about "James Galway", a famous northern Ireland musician.

The power, in this case, lies in the structured data (which allows a neat user interface to juxtapose conflicting facts, e.g. 2 different types), but most of all in the user interaction paradigm: to remove these and many other false statements, the user must only be able to locate a single false value (e.g. a picture of the musician) and select the pop-up option "remove source" to remove the source itself and all the metadata associated.

Interestingly, ambiguity is not necessarily a bad thing in Sig.ma. For example querying for the Italian city "*Trento*" returns data from sources that describe

[1] `http://www.deri.ie/about/team/`

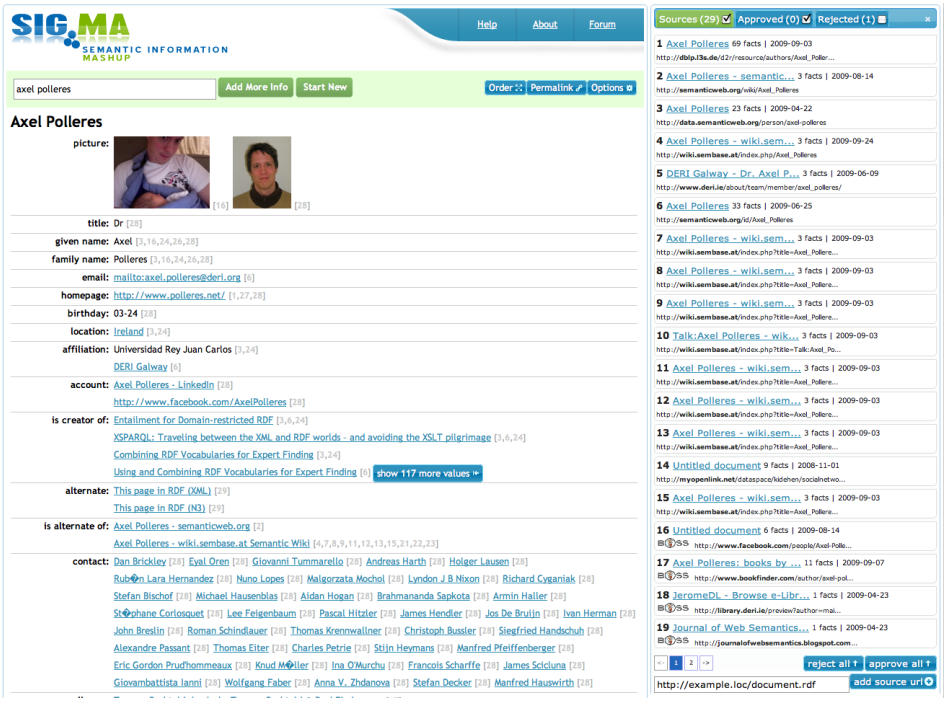

Fig. 1. Sig.ma screenshot for the query "Axel Polleres" when expanded to 30 sources – space usage is optimized using the capabilities of the web interface itself (e.g. property reordering and visualization options)

.

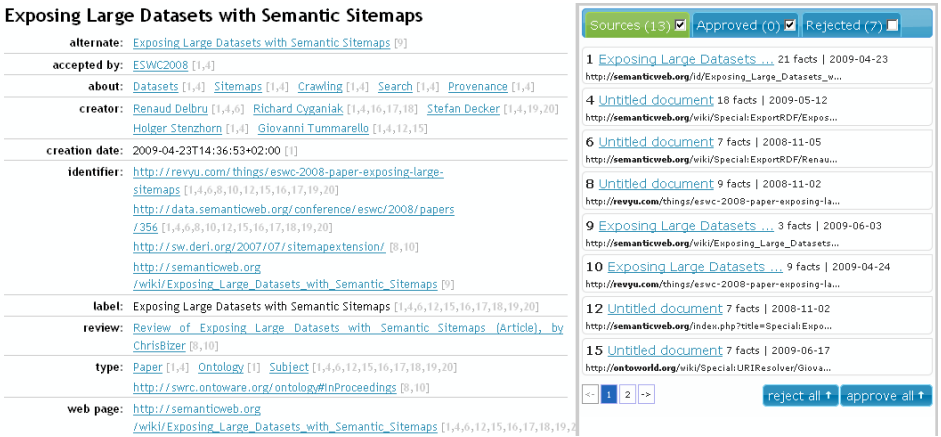

Fig. 2. Sig.ma screenshot for a paper. Information comes from multiple RDF sources also including a review from Revyu.com

.

formally different entities but that in practices can for an end user and for certain needs be usefully put together, resulting in a practically useful unified profile. As an example, merging Trento as a province and Trento as a city one can get a Sigma that would contain also an overall map of Italy with the Trento province highlighted and the name of many neighboring towns (due to the province source), which in certain contexts can be considered relevant pieces of information for the Trento entity.

3 Under the Hood

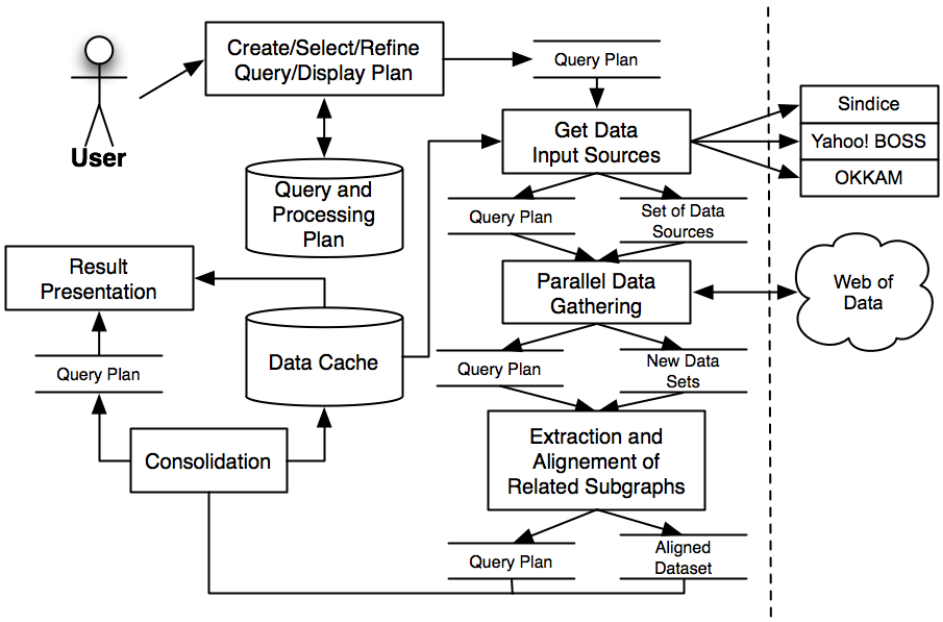

Fig. 3. Sig.ma dataflow

Sig.ma functionalities revolves around the creation of information aggregates called *Entity Profile*, or *the sig.ma* of an entity. Also, we define a *data source* as a Web document that contains structured data (e.g. a URL with RDF/XML or Microformats).

To get an entity profile, typically the user enters a textual query or inputs URLs or URIs. The keyword and structured queries provided by the user form the initial "Query Plan" in Sig.ma. The Query Plan, is then passed along and used as depicted in picture 1 and concisely described in the following steps:

Data Sources selection The query plan is expanded in a set of diverse Search Engine interrogations, which returns a list of related data sources. Yahoo Boss is used to retrieve normal pages (which we then scan for embedded data) while Sindice[3] is used in textual and structural query mode to leverage precise property searches.

If the source list is long, it is trimmed. The desired length is still subject to experimentation, but 25 sources seems to be a good compromise of response time, data variety, and it is still manageable in the user interface. The user interface has then a control for requesting more resources, which repeats the process with a higher source cutoff limit.

Parallel Data Gathering All the candidate data sources are retrieved from the Sindice HBase backed web cache, or fetched by a multithreaded service and then processed with the Sindice metadata extraction library *any23*[2]. , is used to extract RDF data from those different formats. In Sindice, the structured data extracted from web documents is also stored in the HBase–based *page repository*, which allows subsequent fast access to the documents' contents without incurring the cost of Web retrieval.

Extraction and Alignment of related subgraphs The structured data extracted from each source is broken down into chunks that each describe distinct entities (*resource descriptions*). Then, the set of sources is expanded based on `owl:sameAs` links and inverse functional properties found in the descriptions.

The structured RDF graph extracted from each source is broken down into chunks (called *resource descriptions*) that each describe distinct entities. This is made easy by the use of the RDF data model. A resource description contains the outgoing and incoming RDF triples of a specific resource.

In some cases it would be desirable to include more information into a resource description. An example are geographic locations, which are often attached to a resource via a property such as `foaf:based_near`, which points to another resource, often an RDF blank node, which in turn has properties `geo:lat` and `geo:long` that give the geographical coordinates. Obviously it would be good to have the coordinates included in the resource description, even though they are only indirectly attached to the resource in question. This could be solved either by manually identifying commonly occurring cases such as the one given here, or by using generic heuristics based on graph shape, e.g. include linked blank nodes that have less than a certain number of outgoing triples.

As an example of a decomposition into resource descriptions, consider the case of a typical FOAF[3] file that describes a person. It will be decomposed into one resource description for the file's owner, one small description for each of their friends listed in the profile, and possibly one description for the FOAF document itself, containing statements about its `foaf:maker` and `foaf:primaryTopic`.

Resource descriptions are now ranked. If the resource has one of the resource identifiers from the source acquisition step, then it will immediately receive a large boost, as there is almost total certainty that it described the entity in question.

Each description will be matched and scored against the keyword phrase, considering both RDF literals and (with a lower score) words in URIs. This helps to pick out the correct resource in cases such as FOAF files, which talk about multiple people, but it is easy to select the right one given a name.

Very small entities are slightly reduced in score, because experimental results show they are unlikely to contain interesting information, while cluttering up the source list in the user interface.

---

[2] `http://code.google.com/p/any23/`

[3] `http://www.foaf-project.org/`

Resource descriptions below a certain threshold are removed from consideration. We now have a ranked list of descriptions that are hoped to describe the same entity. Of course, since fuzzy keyword matching is used in several places in the process, there are chances of false positives.

A first cut of our algorithm used only the highest-ranking resource description from each source, discarding all others. This has proven to be problematic, as our ranking sometimes would score the document resource description higher than the description of the person or other entity described in the document, because both might have the same, highly salient, label. By including both, we leave the problem to the user, instead of risking the wrong pick.

If the number of highly-scoring resource descriptions is low at this point, then an attempt is made to discover additional sources, based on the RDF data we have already retrieved and established to likely describe the target entity. We obtain new resource identifiers for the target entity using searching for the URIs of the resources or `owl:sameAs` links from them or Inverse Functional properties or using more services such as, for example, the OKKAM service.

Any resource identifier discovered using these methods will be added into the *Query plan*, which will be then examined in the *refinement* step.

Consolidation All the descriptions are then merged into a single entity profile, by means of various heuristics. All selected resource descriptions are merged into a single entity profile. This simply means that all key-value pairs from all resource descriptions are combined into a single description. A reference to the original source is kept for each value.

Often different properties (keys in the key-value pairs that describe the entity) express the same thing. The next step is to consolidate the potentially large and chaotic list of properties into a simpler list that is more meaningful to the user. In RDF, properties are named with URIs; we consider only the last segment ("local part") of the URI. By convention, this local part is usually a good name for the property, written in CamelCase or with underscores or dashes, which are converted back into a more readable string consisting of space-separated words. In the future, we should also check the definition of the property (obtainable by dereferencing its URI, and often already in the page repository) for an `rdfs:label`.

The next step is to treat both incoming triples (of the shape "other-entity - relationship - our-entity") and outgoing triples (of the form "our-entity - relationship - value" or "our-entity - relationship - other-entity") as outgoing triples. This is done simply by flipping the incoming triples around, and adding an *inverse flag* to the relationship. For example, *A creator B* becomes *A is creator of B*.

Next, we apply some simple English-language heuristics on the property names. This is based on observing properties typically used in the wild (E.g. remove initial "has" in "has title").

Next, we apply a manually-compiled list of approximately 50 preferred terms and data transforms. For example, we replace all of the following property names with the preferred term "web page": work info homepage, workplace homepage, page, school homepage, weblog, website, public home page, url, web. Special attention has been given to terms that can be used in customized ways in the user interface: labels, depictions (images), short descriptions, web links. After consolidation, properties are ranked according to how many sources have values for it and well known properties are boosted. The number of distinct values for the property

is also factored in: properties where many sources agree on one or a few values (as observable e.g. with a person's name or homepage) receive a boost.

Value labeling For key-value pairs where the value is not a literal value (such as a name or a date), but a reference to another resource (usually by URI), a best-effort attempt is made to retrieve a good label for the resource which can also include more AJAX triggered web fetching and processing (done on behalf of the ajax code by backend services also using the Sindice cache).

Value consolidation If a property has several values with identical or very similar labels, then they are collapsed into one value to improve the visual presentation. For example, several sources that describe a scientist can state that they have authored a certain paper, using different identifiers for the paper. Without label-based consolidation, the paper would appear several times because the identifiers are not the same. After label consolidation, it appears only once. Both identifiers are retained internally. A click on the paper link will cause a new Sig.ma search that has the label and both of the URIs as input. Since labels are retrieved and displayed incrementally, the value consolidation has to be performed in the same fashion.

Source list refinement After the entity profile is presented to the user, it can be refined by modifying the sources list. This final step generates a closed loop in the Sig.ma data-flow, which actually represent an "expansion–refinement" loop which is driven by the user input.

After the entity profile is presented to the user, they can refine the result by adding or removing sources.

Almost any entity profile initially includes some poor sources that add noise to the results. Mixed into the desired entity profile are other entities that have the same or a similar name, or that for other reason ranked highly in the text search portions. The user interface allows quick removal of these. Widgets for source removal exist in the list of sources, and next to each value that is displayed in the profile. If the profile shows a poor label or unrelated depiction for the entity, a quick click will remove the offending source, and the next-best label or depiction will automatically take its place if present.

Since the profile is based on a fixed number of resources, it will often show only a subset of what is known about the entity. There is a button for including more sources in the source list. After having retrieved more sources and run the usual processing, it will simply mix the results into the profile.

We include also widgets that facilitate retrieval of more information of a specific kind, which show to be useful, for example, when a person's entity profile shows several academic publications that come from sources on a certain domain. Thus, it is likely that fetching more sources from that domain will yield more publications.

4 Related Works

So far there have been two notable approaches to provide a user experience with open data which make heavy use of Semantic Web technologies. In 2006, the SWSE Semantic Search engine demonstrated large scale aggregation of Semantic Web data [1]. To perform entity information consolidation, SWSE followed classic Semantic Web rules: consolidation via reuse of the same identifier across different data sources and several forms of lightweight reasoning such as explicit SameAs

statements, OWL Inverse Functional Properties, etc., and did not perform other data curing. As a result, the engine displays peculiar side effects. As an example, a query for "Giovanni Tummarello" returns 44 RDF nodes, each with different informations attached.

A completely different approach is that of the Tim Berners–Lee initiated Tabulator project[2] where multiple sources of data (RDF only) come into play only if directly interlinked and if the user performs the right browsing actions.

There are a number of further information services which can somehow be related with Sig.ma, given their ability to fulfill comparable user-driven information gathering tasks. For example, engines like "Spook" and "Pipl" allow retrieving people's entity profiles formed by NLP of web pages. While the coverage is certainly more extended of that of Sig.ma, the precision and the necessarily limited rules they employ limit the type and quality of the information provided.

5 Discussion and conclusions

Arguably the ultimate goal of the Semantic Web initiative is to enable automatic reuse and reconfiguration of information from multiple, not a priori known sources and for useful purposes no considered when the information was originally created[4].

We indeed believe that with Sig.ma, for the first time, *this is exactly the feeling* when querying for topics which have a reasonable coverage in terms of Web Data (and indexing provided by the underlying Sindice and top-K results from Yahoo BOSS).

In Sig.ma elements such as large scale semantic web indexing, logic reasoning, data aggregation heuristics, ad hoc ontology consolidation and, last but not least, user interaction and refinement, all play together to be able to take advantages of very many different sources and simplified markup practices.

We believe Sig.ma to be of great inspiration for a new breed of Semantic Web applications and API and, at the same time, a very valuable tool to show to other communities - outside the Semantic Web - what becomes possible once topics are covered by some machine readable annotations.

While Sig.ma does not currently attempt disambiguation, we believe this can be successfully achieved when rich enough descriptions exist. In turn, it is possible to conjure that such rich, easily disambiguable descriptions are bound to be more and more the standard on the Web.

References

1. A. Harth, A. Hogan, R. Delbru, J. Umbrich, S. O'Riain, and S. Decker. Swse: Answers before links! In *Semantic Web Challenge, ISWC*, 2007.
2. T. B. Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and analyzing linked data on the semantic web. In *In Procedings of the 3rd International Semantic Web User Interaction Workshop (SWUI06)*, page 06, 2006.
3. E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello. Sindice.com: a document-oriented lookup index for open linked data. *Int. J. of Metadata and Semantics and Ontologies*, 3:37–52, Nov. 10 2008.

[4] E.g. see the Semantic Web challenge goal