Adding Meaning to Facebook Microposts via a Mash-up API and Tracking Its Data Provenance

Thomas Steiner*, Ruben Verborgh†, Joaquim Gabarró Vallés*, Rik Van de Walle†
*Universitat Politècnica de Catalunya, Department LSI, 08034 Barcelona, Spain
Email: {tsteiner, gabarro}@lsi.upc.edu
†Ghent University – IBBT, ELIS – Multimedia Lab, B-9050 Ledeberg-Ghent, Belgium
Email: {ruben.verborgh, rik.vandewalle}@ugent.be

***Abstract*—The social networking website Facebook offers to its users a feature called "status updates" (or just "status"), which allows users to create microposts directed to all their contacts, or a subset thereof. Readers can respond to microposts, or in addition to that also click a "Like" button to show their appreciation for a certain micropost. Adding semantic meaning in the sense of *unambiguous intended ideas* to such microposts can, for example, be achieved via Natural Language Processing (NLP). Therefore, we have implemented a RESTful mash-up NLP API, which is based on a combination of several third party NLP APIs in order to retrieve more accurate results in the sense of emergence. In consequence, our API uses third party APIs opaquely in the background in order to deliver its output. In this paper, we describe how one can keep track of provenance, and credit back the contributions of each single API to the combined result of all APIs. In addition to that, we show how the existence of provenance metadata can help understand the way a combined result is formed, and optimize the result combination process. Therefore, we use the HTTP Vocabulary in RDF and the Provenance Vocabulary. The main contribution of our work is a description of how provenance metadata can be automatically added to the output of mash-up APIs like the one presented here.**

I. Introduction

According to official Facebook statistics [7], the social networking website has more than 500 million active users out of which half log on to Facebook in any given day. The average Facebook user has 130 friends, and creates 90 pieces of content each month. This sums up to the impressive number of overall twenty-two billion five hundred million pieces of content per month. Similar to the microblogging website Twitter with its full text Twitter search[1], Facebook as well offers both a search feature on the website, and a JSON-based search API over status updates from all global Facebook members[2]. In order to perform data mining, a statistically significant amount of microposts is necessary (this is also known as access to the "firehose"). However, while Twitter grants selected parties access to its Streaming API [17] for research purposes, for Facebook there is no such documented option. To address this shortage, we have developed an extension called Facebook Swarm NLP for the Google Chrome Web browser. This extension[3] first injects JavaScript code into the Facebook.com homepage to perform data analysis on the encountered set of microposts, and then sends the results to a central data processing point. Given a broad enough installation base, this extension allows for a random sample of microposts to be analyzed as they become available on Facebook – in effect a very modest firehose. The extension first checks if the user is logged in to Facebook.com, and if so, retrieves all status updates from the contacts that are displayed on the current user's Facebook homepage. Second, the extension performs named entity extraction via Natural Language Processing (NLP) using a remote NLP API on each of these status updates in order to add semantic meaning to them. The extracted named entities are then displayed below each post, as illustrated in Figure 1. Finally the extracted named entities are sent to a central Google Analytics [11] profile to compute basic or advanced trends, for example by ranking the most discussed-about named entities per day, or by pivoting named entities by Analytics data, like users' geographic locations.

Fig. 1. Facebook Swarm NLP Chrome extension. Extracted named entities have a pale yellow background.

As mentioned before, in order to perform named entity extraction, we rely on a mash-up API that calls existing third party NLP APIs in the background and that delivers the combined results of these APIs in a consolidated way. Obviously it is very desirable (i) to credit back the contribution of each single third party API to the joint results, and (ii) to track the provenance of the joint results in order to understand how they were formed. We will show at the concrete example of the mash-up NLP API used for our Facebook Swarm NLP extension how these two constraints can be fulfilled in a generalizable way.

[1] https://search.twitter.com/
[2] Facebook search for "salamanca": http://bit.ly/ogpsearch
[3] http://bit.ly/facebookswarmnlp

The remainder of this paper is structured as follows: we discuss related work in Section II. In Section III, we introduce APIs that allow for unstructured data to be converted into Linked Data. In Section IV, we describe how we automatically maintain provenance metadata in our API. Section V presents future work. Finally Section VI ends the paper with a conclusion.

II. Related Work

In [8], Groth et al. describe how through tools and technologies such as Yahoo Pipes, Really Simple Syndication (RSS) and APIs, so-called mash-ups can be created in a dynamic, just-in-time way, combining data from different data sources. The authors are driven by the motivation to allow for trust and confidence in mash-ups, and therefore consider it critical to be able to analyze the origin of combined results. They suggest an approach based on OWL and XML, with a focus on process documentation. However, different from us, where the goal is to transparently add provenance data at API invocation level, their focus is more on overall process documentation in the context of a mash-up application.

The focus of Carroll et al. in [4] is on the provenance of triples in the Semantic Web world, namely, for making statements about triples in graphs. Therefore, the paper introduces the concept of named graphs, an extension to RDF. In contrast to our work, Carroll et al. focus purely on using triples to make statements about triples (that is, they stay in the RDF world), whereas our approach uses RDF to make statements about potentially any API result. That is, our approach is not limited to RDF results, albeit in the concrete case we use RDF in addition to JSON as API result.

In the WS-* world, BPEL4WS, described by Curbera et al. in [6] provides a formal language for the specification of business processes and business interaction protocols. This allows for the combination of several APIs, however, it does not credit back concrete outputs of a combined API to the underlying APIs.

III. Structuring Unstructured Data

Sir Tim Berners-Lee has introduced Linked Data in a W3C Design Issue [2], where he defines the four rules for Linked Data:

1) Use URIs as names for things.
2) Use HTTP URIs so that people can look up those names.
3) When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL).
4) Include links to other URIs, so that they can discover more things.

In order to represent extracted named entities from Facebook microposts in an unambiguous way, we apply the first and the second Linked Data principle by representing named entities with HTTP URIs. This is taken care of by the third party NLP APIs that we use for our Chrome extension, namely OpenCalais [16], Zemanta [18], DBpedia Spotlight [15], and AlchemyAPI [1]. These APIs take a text fragment as input, perform named entity extraction on it, and then link the extracted entities back into the Linking Open Data (LOD) cloud[4]. We use these APIs in parallel, and by combining their results aiming at the emergence effect in the sense of Aristotle: "*[. . .] the totality is not, as it were, a mere heap, but the whole is something besides the parts [. . .]*"[5].

A. Combining Results from Different NLP APIs

We have implemented a mash-up API for the four NLP APIs. While the original calls to each particular NLP API are all HTTP `POST`-based, we have implemented a `GET`- and `POST`-based mash-up API. All NLP APIs return entities with their types and/or subtypes, names, relevance, and URIs that link into the LOD cloud. Our mash-up API supports two output formats, namely application/json and text/turtle (an RDF [12] serialization format). The combined output in JSON form for the micropost *"Tom has the LaTeX, BibTeX, LaTeX, LaTeX blues..."* (see Figure 1) is shown below:

```
[
  {
    "name": "LaTeX",
    "relevance": 0.7128319999999999,
    "uris": [
      {
        "uri": "http://dbpedia.org/resource/LaTeX",
        "source": "zemanta"
      }
    ],
    "source": "zemanta"
  },
  {
    "name": "BibTeX",
    "relevance": 0.8143277777777777,
    "uris": [
      {
        "uri": "http://dbpedia.org/resource/BibTeX",
        "source": "zemanta"
      }
    ],
    "source": "zemanta"
  }
]
```

These joint results come from a request to our mash-up API via `GET /entity-extraction/combined/Tom%20has%20the%20LaTeX%2C%20BibTeX%2C%20LaTeX%2C%20LaTeX%20blues...`. Obviously our API abstracts away the different output formats of the underlying APIs and returns a JSON object structure instead.

B. The Need for Providing Provenance Metadata

Hartig et al. mention in [9] some reasons that justify the need for provenance metadata. Among those is linked dataset replication and distribution on the Web with not necessarily identical namespaces: based on the same source data, different copies of a linked dataset can be created with different degrees of interconnectedness by different publishers.

We add to this list the automatic conversion of legacy unstructured data to Linked Data with heuristics where extracted entities – while being consolidated and backed up by different data sources – might still be wrong. Especially with our "mash-up"-like approach, it is very desirable to be able

[4] `http://lod-cloud.net/`

[5] Aristotle, Metaphysics, Book H 1045a 8-10.

to track back to the concrete source where a certain piece of information comes from. This enables (i) to correct the error at the root of our API (fighting the cause), (ii) to correct the concrete error in an RDF annotation (fighting the symptom), and (iii) to judge the trustworthiness and quality of a dataset, which is probably the most important reason.

IV. Tracking Provenance With Multiple Sources

As outlined before, we use several data sources (APIs) in the background in order to add meaning to Facebook microposts. Extracted named entities from a Facebook micropost might in consequence be the result of up to four agreeing (or disagreeing) API calls (see Section III). In order to track the contributions of the various sources, we have decided to use the Provenance Vocabulary [10] by Hartig and Zhao with the prefix `prv`, the HTTP Vocabulary in RDF [13] by Koch et al. with prefix `http`, and a vocabulary for Representing Content in RDF [14] by the same authors with prefix `cnt`. We have chosen the HTTP Vocabulary in RDF for the fact that it is a W3C Working Draft developed by the Evaluation and Repair Tools Working Group (ERT WG), which is part of the World Wide Web Consortium (W3C) Web Accessibility Initiative (WAI). The Provenance Vocabulary was chosen because of its relatively broad implementation in several projects, such as Pubby[6], Triplify[7], and D2R Server[8].

While our mash-up API supports two output formats (application/json and text/turtle), we have added provenance information exclusively to the text/turtle variant. In order to represent the extracted named entities in a Facebook micropost, we use the Common Tag vocabulary [5]. A micropost is `ctag:tagged` with a `ctag:Tag`, which consists of a textual `ctag:label` and a pointer to a resource that specifies what the label `ctag:means`. The Common Tag vocabulary is well-established and developed by both industry and academic partners. In order to make statements about a bundle of triples, we group them in a named graph. We use the TriG [3] syntax:

```
:G = {
  <https://www.facebook.com/Tomayac/posts
       /10150175940867286> ctag:tagged [
      a ctag:Tag ;
      ctag:label "BibTeX" ;
      ctag:means <http://dbpedia.org/resource/BibTeX>
    ] .
} .
```

A. The Provenance Vocabulary

In this section, we outline the required steps in order to make statements about the provenance of a group of triples contained in a named graph `:G` that was generated using several HTTP `GET` requests to third party APIs. We use the Provenance Vocabulary [10] with prefix `prv`, the HTTP Vocabulary in RDF [13] with prefix `http`, and the Representing Content in RDF [14] vocabulary with prefix `cnt`.

First, we state that *:G* is both a *prv:DataItem* and obviously an *rdfg:Graph*. *:G* is *prv:createdBy* the process of a *prv:DataCreation*. This *prv:DataCreation* is *prv:performedBy* a *prv:NonHumanActor*, a *prvTypes:DataProvidingService* to be precise (simplified as *http://tomayac.no.de/entity-extraction/combined* in the listing). This service is *prv:operatedBy* a human (`http://tomayac.com/thomas_steiner.rdf#me`). Time is often important for provenance, so the *prv:performedAt* date of the *prv:DataCreation* needs to be saved. During the process of the *prv:DataCreation* there are *prv:usedData*, which are *prv:retrievedBy* a *prv:DataAcess* that is *prv:performedAt* a certain time, and *prv:performedBy* a non-human actor (our API) that is *prv:operatedBy* a human (`http://tomayac.com/thomas_steiner.rdf#me`. For the *prv:DataAccess* (there is one for each third party API involved), we *prv:accessedService* from a *prv:DataProvidingService* of which we *prv:accessedResource* at a certain *irw:WebResource*. Therefore, we *prvTypes:exchangedHTTPMessage* which is an *http:Request* using *http:httpVersion* "1.1" and the *http:methodName* "GET".

B. Provenance RDF Overview

This section provides a shortened overview of the provenance RDF in Turtle syntax for a Facebook micropost tagged with the label "BibTeX" and the assigned meaning `http://dbpedia.org/resource/BibTeX`. The named graph `:G` in the first part of the listing contains the absolute data (the fact that the Facebook micropost with the URI `https://www.facebook.com/Tomayac/posts/-10150177486072286` is tagged with the label "BibTeX", which is represented by the HTTP URI `http://dbpedia.org/resource/BibTeX`). The second part with metadata about `:G` says that these facts were generated via two calls, one using the HTTP method `GET`, and the other `POST`:

```
:G = {
  <https://www.facebook.com/Tomayac/posts
       /10150177486072286> ctag:tagged [
     a ctag:Tag ;
     ctag:label "BibTeX" ;
     ctag:means <http://dbpedia.org/resource/BibTeX> ;
  ] .
} .


:G
  a prv:DataItem ;
  a rdfg:Graph ;
  prv:createdBy [
    a prv:DataCreation ;
    prv:performedAt "2011-05-20T15:06:30Z"^^xsd:dateTime ;
    prv:performedBy <http://tomayac.no.de/entity-
         extraction/combined> ;
    prv:usedData [
      prv:retrievedBy [
        a prv:DataAcess ;
        prv:performedAt "2011-05-20T15:06:30Z"^^xsd:
             dateTime ;
        prv:performedBy <http://tomayac.no.de/entity-
             extraction/combined> ;
        prv:accessedService <http://spotlight.dbpedia.org/
             rest/annotate> ;
        prv:accessedResource <http://spotlight.dbpedia.org
             /rest/annotate?text=Tom%20has%20the%20LaTeX%2
             C%20BibTeX%2C%20LaTeX%2C%20LaTeX%20blues...&
             confidence=0.4&support=20> ;
```

[6] http://www4.wiwiss.fu-berlin.de/pubby/
[7] http://triplify.org/Overview
[8] http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/

```
      prvTypes:exchangedHTTPMessage [
        a http:Request ;
        http:httpVersion "1.1" ;
        http:methodName "GET" ;
        http:mthd <http://www.w3.org/2008/http-methods#
            GET> ;
      ] ;
    ] ;
  ] ;
  prv:usedData [
    prv:retrievedBy [
      a prv:DataAcess ;
      prv:performedAt "2011-05-20T15:06:41Z"^^xsd:
          dateTime ;
      prv:performedBy <http://tomayac.no.de/entity-
          extraction/combined> ;
      prv:accessedService <http://api.zemanta.com/
          services/rest/0.0/> ;
      prv:accessedResource <http://api.zemanta.com/
          services/rest/0.0/> ;
      prvTypes:exchangedHTTPMessage [
        a http:Request ;
        http:httpVersion "1.1" ;
        http:methodName "POST" ;
        http:mthd <http://www.w3.org/2008/http-methods#
            POST> ;
        http:headers (
          [
            http:fieldName "Content-Type" ;
            http:fieldValue "application/x-www-form-
                urlencoded" ;
          ]
        )
        http:body [
          a cnt:ContentAsText ;
          cnt:characterEncoding "UTF-8" ;
          cnt:chars """method=zemanta.suggest_markup
          &api_key=Your_API_Key
          &text=Tom%20has%20the%20LaTeX%2C%20BibTeX%2C
              %20LaTeX%2C%20LaTeX%20blues...
          &format=json
          &return_rdf_links=1""" ;
        ] ;
      ] ;
    ] ;
  ] ;
] .
```

It is to be noted that statements such as in the listing above refer to the triple objects as an identifier for a Web resource (where the Web resource is a representation of the result of the API call at the time where it was `prv:performedAt`). As provenance metadata always refers to the time context in which a certain statement was made, it is essentially unimportant what representation the resource returns in future.

V. Future Work

Already commenced future work is to explore ways to drastically simplify the descriptions by reducing their verbosity, but still try to be compatible with existing standards such as the HTTP in RDF and Provenance vocabularies. While it is always easier to come up with a specialized vocabulary that does one task well (for example, we could imagine a simple vocabulary with the sole purpose to log the API call of an API invocation), broader reuse and acceptance can be gained by reusing existing vocabularies. We will investigate how to find the right balance here.

VI. Conclusion

We have introduced an API for adding semantic meaning to Facebook microposts. As mash-up data sources for our API we have presented NLP APIs, and then focused on the necessary RDF vocabularies to annotate Facebook microposts with the thereof extracted named entities in the form of common tags. Due to their different "mash-up"-like history of origins, we needed to track provenance metadata in order to assure the trustworthiness of the generated data. We showed how the Provenance Vocabulary can be used to keep track of the original third party API calls that led to the consolidated results. We have shown how a concrete multi-source RESTful API can automatically maintain provenance metadata. We believe that being able to track back the origin of a piece of data is of crucial importance, however, the generated provenance-related triples are very verbose, and in consequence stating even simple facts like that a combined result is based on two separate sub-results takes up a lot of space. The verbosity is mainly due to the used vocabularies, the Provenance Vocabulary and the HTTP Vocabulary in RDF. We are conscious that our current approach is a first step in the right direction, however, that some more steps are ahead to take. Our vision is to establish a common method for specifying provenance data for mash-up APIs.

Acknowledgment

T. Steiner is partially supported by the European Commission under Grant No. 248296 FP7 I-SEARCH project. J. Gabarró is partially supported by TIN-2007-66523 (FORMALISM) and SGR 2009-2015 (ALBCOM).

References

[1] AlchemyAPI. http://www.alchemyapi.com/api/entity/.
[2] T. Berners-Lee. Linked Data, Juli 27, 2006. http://www.w3.org/DesignIssues/LinkedData.html.
[3] C. Bizer and R. Cyganiak. The TriG Syntax, July 30, 2007. http://www4.wiwiss.fu-berlin.de/bizer/TriG/.
[4] J. Carroll, C. Bizer, P. Hayes, and P. Stickler. Named graphs, provenance and trust. In *WWW '05: Proceedings of the 14th Int. Conference on World Wide Web*, pages 613–622, New York, USA, 2005. ACM.
[5] Common Tag. Common Tag Specification, January 11, 2009. http://commontag.org/Specification.
[6] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana. The next step in web services. *Commun. ACM*, 46:29–34, October 2003.
[7] Facebook. Press Room – Statistics, May 16, 2011. https://www.facebook.com/press/info.php?statistics.
[8] P. Groth, S. Miles, and L. Moreau. A model of process documentation to determine provenance in mash-ups. *ACM Trans. Internet Technol.*, 9:3:1–3:31, February 2009.
[9] O. Hartig and J. Zhao. Publishing and Consuming Provenance Metadata on the Web of Linked Data. In *3^{rd} Int. Provenance and Annotation Workshop (IPAW'10)*, Troy, NY, USA, 2010.
[10] O. Hartig and J. Zhao. Provenance Vocabulary Core Ontology Specification, January 25, 2011. http://purl.org/net/provenance/ns.
[11] A. Kaushik. *Web Analytics 2.0: The Art of Online Accountability and Science of Customer Centricity*. SYBEX Inc., Alameda, CA, USA, 2009.
[12] G. Klyne and J. J. Carrol. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, Feb. 2004.
[13] J. Koch, C. A. Velasco, and P. Ackermann. HTTP Vocabulary in RDF 1.0, May 10, 2011. http://www.w3.org/TR/HTTP-in-RDF10/.
[14] J. Koch, C. A. Velasco, and P. Ackermann. Representing Content in RDF 1.0, May 10, 2011. http://www.w3.org/TR/Content-in-RDF10/.
[15] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia Spotlight: Shedding Light on the Web of Documents. In *Proceedings of the 7th Int. Conference on Semantic Systems (I-Semantics)*, 2011.
[16] OpenCalais. http://www.opencalais.com/documentation/.
[17] Twitter. Streaming API, May 16, 2011. http://dev.twitter.com/pages/streaming_api_methods.
[18] Zemanta. http://developer.zemanta.com/docs/.