Where are the RDF Streams?

Deploying RDF Streams on the Web of Data with TripleWave

Andrea Mauri[1], Jean-Paul Calbimonte[2], Daniele Dell'Aglio[1], Marco Balduini[1], Emanuele Della Valle[1], and Karl Aberer[2]

[1] DEIB, Politecnico di Milano, Italy
{name.surname}@polimi.it
[2] EPFL, Switzerland
{name.surname}@epfl.ch

Abstract. RDF Stream Processing (RSP) bridges the gap between semantic technologies and data stream systems. Although a number of RSP systems have been recently proposed, no RDF streams are actually made publicly available on the Web. To cope with this, RSP engines require ad-hoc wrappers in order to be fed from non-RDF streams available on the Internet. In this paper we present TripleWave: an approach for publishing existing streams on the Web as RDF streams, using mappings to perform live transformation of data, and following the Linked Data principles. We implemented and deployed TripleWave for a concrete use-case: a live feed of updates of Wikipedia.

1 Introduction

Data streams are one of the main sources of data on the Web, and are actively being produced in several domains, ranging from social networks to IoT. A data stream [1] can be defined as an ordered sequence of items (e, t), where e is a data item and t is a time annotation. Recently, the notion of RDF stream, where the data item e is modeled as RDF, has gained prominence. Several RDF stream Processing (RSP) approaches and systems [8] have been proposed to deal with this type of data, powering applications in several domains [6] and launching standardization efforts[3]. Surprisingly, public RDF Streams are generally missing in the landscape of RDF stream processing. In fact, to the best of our knowledge, there are currently no RDF streams published and publicly available over the Web. Existing RSP engines have circumvented this issue in different ways: e.g. the SLD framework has internal components to lift the data in RDF [2], while SPARQL$_{stream}$ [5] uses the notion of virtual RDF streams. Other engines expose programmatic APIs and delegate to the users the task to manage the streams and to feed the system through the APIs [7].

Inspired by the experience of the community in publishing static data sets as RDF data sets, in this work we present TripleWave, a framework to transform existing streams in RDF streams, and publish them on the Web. Moving

[3] W3C RSP community Group: http://www.w3.org/community/rsp/

from static to dynamic data poses new requirements not addressed by existing solutions. In particular, the high velocity on which the data is generated makes it difficult to to permanently store the whole data stream and to publish it as Linked Data. To overcome this limit, we extend the initial proposal about publishing RDF streams in [3], defining two types of triple graphs: stream graphs and instantaneous graphs, as detailed in Section 3. As output, TripleWave produces a JSON stream in the JSON-LD format: each stream element is described by an RDF graph and the time annotation is modeled as an annotation over the graph. Using this format compliant with existing standards, TripleWave enables processing RDF streams not only with RSP engines, but also with existing frameworks and techniques for RDF processing (e.g. SPARQL engines).

As a case study, we consider the change stream of Wikipedia[4]. This stream features all the changes that occur on the Wikipedia website. This stream is characterized by heterogeneity: it comprehends not only elements related to the creation or modification of pages (e.g., articles and books), but also events related to users (new registrations and blocked users), and discussions among them.

```
{ "page": "Naruto: Ultimate Ninja",
  "pageUrl": "http://en.wikipedia.org/wiki/Naruto:_Ultimate_Ninja",
  "url": "https://en.wikipedia.org/w/index.php?diff=669355471&oldid=669215360",
  "delta": -7, "comment": "/ Characters /",
  "wikipediaUrl": "http://en.wikipedia.org",
  "channel": "#en.wikipedia", "wikipediaShort": "en",
  "user": "Jmorrison230582", "userUrl": "http://en.wikipedia.org/wiki/User/Jmorrison230582",
  "unpatrolled": false, "newPage": false, "robot": false,
  "namespace": "article" }
```

Listing 1.1. A fragment of the change stream of Wikipedia

Listing 1.1[5] shows a fragment of the stream of changes of Wikipedia. In particular, it shows that the user `Jmorrison230582` modified an article of the `English` Wikipedia about `Naruto: Ultimate Ninja`. Furthermore, the delta attribute tell us that the user deleted some words, and the `url` attribute refers the to the Wikipedia page that describes the event.

2 R2RML to create RDF streams

Streams on the Web are available in a myriad of formats, so to adapt and transform them to RDF streams we use a generic transformation process that is specified as R2RML[6] mappings. Although these mappings were originally conceived for relational database inputs, we can use light extensions that support other formats such as CSV or JSON (e.g. as in RML[7]). The example below specifies how a Wikipedia stream update can be mapped to a graph of an RDF stream[8]. This mapping defines first a triple that indicates that the generated subject is of type `schema:UpdateAction`. The `predicateObjectMap` clauses add two more

[4] Cf. `https://en.wikipedia.org/wiki/Special:RecentChanges`
[5] Data collected with the API provided by `https://github.com/edsu/wikistream`
[6] R2RML W3C Recommendation: `http://www.w3.org/TR/r2rml/`
[7] RML extensions: `http://rml.io`
[8] We use schema.org as the vocabulary in the example.

triples, one specifying the object of the update (e.g. the modified wiki page) and the author of the update. The graph is specified using the `graphMap` property.

```
:wikiUpdateMap a rr:TriplesMap; rr:logicalTable :wikistream;
  rr:subjectMap [ rr:template "http://131.175.141.249/TripleWave/{time}";
                  rr:class schema:UpdateAction; rr:graphMap :streamGraph ];
  rr:predicateObjectMap [rr:predicate schema:object; rr:objectMap [ rr:column "pageUrl" ]];
  rr:predicateObjectMap [rr:predicate schema:agent;  rr:objectMap [ rr:column "userUrl"] ];.
```

Additional mappings can be specified, as in the example below, for providing more information about the user (e.g. user name):

```
:wikiUserMap a rr:TriplesMap; rr:logicalTable :wikistream;
  rr:subjectMap   [ rr:column "userUrl";
                    rr:class schema:Person; rr:graphMap :streamGraph ];
  rr:predicateObjectMap [ rr:predicate schema:name; rr:objectMap  [ rr:column "user" ]];.
```

A snippet of the resulting RDF Stream graph, serialized in JSON-LD, is shown in in Listing 1.2.

```
{"http://www.w3.org/ns/prov#generatedAtTime": "2015-06-30T16:44:59.587Z",
 "@id": "http://131.175.141.249/TripleWave/1435682699587",
 "@graph": [
   { "@id": "http://en.wikipedia.org/wiki/User:Jmorrison230582",
     "@type": "https://schema.org/Person",
     "name": "Jmorrison230582" },
   { "@id": "http://131.175.141.249/TripleWave/1435682699587",
     "@type": "https://schema.org/UpdateAction",
     "object": {"@id": "http://en.wikipedia.org/wiki/Naruto_Ultimate_Ninja"},
     "agent":  {"@id": "http://en.wikipedia.org/wiki/User:Jmorrison230582"}
   }
 ],
 "@context": "https://schema.org/"  }
```

Listing 1.2. Portion of the timestamped element in the RDF stream.

3 Publishing stream elements as Linked Data

TripleWave is implemented in Node.js and streams out the RDF stream using HTTP with chunked transfer encoding. Consumers can register at the endpoint `http://131.175.141.249/TripleWave/wiki.json` and receive the data following a push paradigm. In cases where consumers may want to pull the data, TripleWave allows publishing the data accordingly to the Linked Data principles [4]. Given that the stream supplies data that changes very frequently, data is only temporarily available for consumption, assuming that recent stream elements are more relevant. In order to allow the consumer to discover which are the currently available stream elements, we use and extend the framework proposed in [3]. According to this scheme, TripleWave distinguishes between two kinds of Named Graphs: the Stream Graph (*sGraph*) and the Instantaneous Graphs (*iGraphs*). Intuitively, an iGraph represents one stream element, while the sGraph contains the descriptions of the iGraphs.

```
tr:sGraph  sld:contains (tr:1435682699954 tr:1435682699587) ;
           sld:lastUpdate "2015-06-29T15:46:05"^^xsd:dateTime .
tr:1435682699587 sld:receivedAt "2015-06-30T16:44:59.587Z"^^xsd:dateTime .
tr:1435682699954 sld:receivedAt "2015-06-30T16:44:59.954Z"^^xsd:dateTime .
```

Listing 1.3. The sGraph pointing to the iGraph described in Listing 1.2.

As an example, for the Wikipedia RDF stream, the sGraph is published at the address `http://131.175.141.249/TripleWave/sGraph`. By accessing the sGraph, consumers discover which are the stream elements (identified by iGraphs) available at the current time instants. The sGraph in Listing 1.3 describes the sGraph and the current content that can be retrieved. The ordered list of iGraphs is modeled as an `rdf:list` with the most recent iGraph as the first element, and with each iGraph having its relative timestamp annotation. Next, the consumer can access the iGraphs dereferencing the iGraph URL address. As example, when the consumer accesses the graph[9] at `http://131.175.141.249/TripleWave/1435682699587`, it retrieves the content of the graph reported in Listing 1.2.

4 Conclusion

We have presented TripleWave, a system that allows deploying RDF Streams on the Web, taking existing streams of non-RDF data and converting them to stream graphs using declarative mappings. TripleWave allows publishing these streams as Linked Data, as well as a live stream of RDF graphs. We have implemented and deployed a use-case that feeds from a live stream of Wikipedia updates. This constitutes a first step towards ubiquitous deployment of RDF streams, based on the ever-increasing amount of data streams available on the Web and the upcoming Internet of Things. Moreover, the provision of RDF stream data will prompt new challenges to existing Linked Data solutions, and will contribute to the maturity of RSP technologies.

References

1. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *PODS*, pages 1–16. ACM, 2002.
2. M. Balduini, E. Della Valle, D. DellAglio, M. Tsytsarau, T. Palpanas, and C. Confalonieri. Social listening of city scale events using the streaming linked data framework. In *ISWC*, pages 1–16. Springer, 2013.
3. D. F. Barbieri and E. Della Valle. A proposal for publishing data streams as linked data - A position paper. In *LDOW*, 2010.
4. T. Berners-Lee, C. Bizer, and T. Heath. Linked data-the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
5. J.-P. Calbimonte, H. Jeung, O. Corcho, and K. Aberer. Enabling query technologies for the semantic sensor web. *Int. J. Semantic Web Inf. Syst.*, 8:43–63, 2012.
6. E. Della Valle, S. Ceri, F. Van Harmelen, and D. Fensel. It's a streaming world! reasoning upon rapidly changing information. *IEEE Intelligent Syst.*, (6):83–89, 2009.
7. D. Gerber, S. Hellmann, L. Bühmann, T. Soru, R. Usbeck, and A.-C. N. Ngomo. Real-time rdf extraction from unstructured data streams. In *ISWC 2013*, pages 135–150. 2013.
8. A. Margara, J. Urbani, F. van Harmelen, and H. Bal. Streaming the web: Reasoning over dynamic data. *J. Web Semantics*, 25:24–44, 2014.

[9] This iGraph is expired at the time of submission. It is possible to consult the sGraph to get the list of the non-expired graphs.