A Proposal for Publishing Data Streams as Linked Data

http://streamreasoning.org
http://wiki.larkc.eu/c-sparql/

Davide F. Barbieri  
DEI – Politecnico di Milano  
dbarbieri@elet.polimi.it

Emanuele Della Valle  
DEI – Politecnico di Milano  
emanuele.dellavalle@polimi.it
Introduction
Real-Time Streams on the Web

• Streams are appearing more and more often on the Web in sites that distribute and present information in real-time streams.

• E.g. http://twitter.com/#search?q=just%20landed%20in

Realtime results for just landed in

- cnymegaphone | just landed in Washington D.C. http://myloc.me/86Ag3
  2 minutes ago from UberTwitter

- melodyxmadness just landed in dallas :) I in about an hour I will be headed HOME!
  2 minutes ago from txt

- nelsoncaban Just landed in Miami: quick breakfast at News Cafe, a call-a-cab at Wet Willie’s...then it’s on to the next trip (back to NYC)
  4 minutes ago from TweetDeck

• Checkout http://activitystrea.ms/ for a standard API
Introduction
Combining Streams and Static Information

• We anticipate a rapidly growing need of mashing up this streaming information with more static one.
• E.g., Twitter + MetaCarta

Background
Managing Streams

• Streams
  – unbounded sequences of time-varying data elements

• Stream Processing
  – Continuous queries registered over streams that are observed through windows
State-of-the-Art
Data Stream Management Systems (DSMS)

• Research prototypes:
  – STREAM http://infolab.stanford.edu/stream/
  – Aurora http://www.cs.brown.edu/research/aurora/
  – Borealis http://www.cs.brown.edu/research/borealis/public/

• Some Features are embedded in

• Start-ups
  – StreamBase http://www.streambase.com/

• Open Source
  – Esper http://esper.codehaus.org/
  – Data Turbine http://www.dataturbine.org/
Background

Continuous SPARQL (C-SPARQL)

• What is it?
  – an extension to SPARQL for continuous querying over (virtual) streams of RDF and static RDF graphs

• Architecture of our C-SPARQL Engine
  – Based on the Large Knowledge Collider (LarKC) conceptual framework
Background
RDF Stream

• RDF Stream Data Type
  – Ordered sequence of pairs, where each pair is made of an RDF triple and its timestamp $t$
    $(<\text{ triple }>, \ t)$
• E.g.,
  $(<\text{ :traveller1 :justLanded :placeA }>, \ T_1)$
  $(<\text{ :traveller2 :justLanded :placeB }>, \ T_1)$
  $(<\text{ :traveller3 :justLanded :placeA }>, \ T_2)$
  $(<\text{ :traveller1 :justLanded :placeC }>, \ T_3)$
Who has landed in USA in the last hour?

**REGISTER QUERY** WhoHasLandedInUSAInTheLastHour AS

PREFIX gno: <http://www.geonames.org/ontology#>
PREFIX c: <http://www.geonames.org/countries/#>
PREFIX : <http://example>

SELECT ?traveller ?place ?type
FROM <http://sws.geonames.org/nonExistingUSFeatureGraph>
FROM STREAM <http://someStreamGeneratedFromTwitter>
[ RANGE 60m STEP 5m ]

WHERE {
}
Background
An Example of C-SPARQL Query Explained

Who has landed in USA

REGISTER QUERY WhoHasLandedInUSAInTheLastHour AS
PREFIX gno: <http://www.geonames.org/ontology#>
PREFIX c: <http://www.geonames.org/countries/＃>
PREFIX : <http://example>
SELECT ?traveller ?place ?type
FROM <http://sws.geonames.org/nonExistingUSfeatureGraph>
FROM STREAM <http://someStreamGeneratedFromTwitter>
[ RANGE 60m STEP 5m ]
WHERE {
}
Proposal
Streaming Linked Data

• What
  – an extension of our C-SPARQL Engine that publishes data streams as Linked Data

• Architecture

![Diagram of C-SPARQL Engine and Clients]

HTML Clients
Linked Data Clients
Remote C-SPARQL Clients
Local C-SPARQL Clients

Streaming Linked Data Server

HTTP

C-SPARQL Engine

Data Streams
RDF Streams
RDF Graphs

HTML
RDF
Java
REST
• The problem
  – How to publish as linked an RDF Stream?
• Proposal
  – Use Named Graph
  – A *Stream Graph* (s-graph)
    • a metadata graph that describes the current content of the window over the stream
  – Several *Instantaneous Graphs* (i-graph)
    • one for each time stamp
  – `rdfs:seeAlso` is used (and reserved) to link the graphs
  – A *Streaming Linked Data Vocabulary* is used to describe the content of the graphs
A s-Graph (only metadata) and ...

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix sld: <http://www.streaminglinkeddata.org/schema#> .
@prefix : <http://example/> .
:sgraph sld:lastUpdate "T3"^^xsd:dateTime;
    sld.expires "T4"^^xsd:dateTime;
    sld:windowType sld:logicalTumbling;
    sld:windowSize "PT1H"^^xsd:duration .
:sgraph1 rdfs:seeAlso :igraph1 .
:igraph1 sld:receivedAt "T1"^^xsd:dateTime .
:sgraph1 rdfs:seeAlso :igraph2 .
:igraph2 sld:receivedAt "T2"^^xsd:dateTime .
:sgraph1 rdfs:seeAlso :igraph1 .
:igraph1 sld:receivedAt "T3"^^xsd:dateTime .
... and three i-Graphs (triples + few metadata)

:igraph1 sld:receivedAt "T_1"^^xsd:dataTime ;
    rdfs:seeAlso :sgraph .
:traveller1 :justLanded :placeA .
:traveller2 :justLanded :placeB .

:igraph2 sld:receivedAt "T_2"^^xsd:dataTime ;
    rdfs:seeAlso :sgraph .

:igraph3 sld:receivedAt "T_3"^^xsd:dataTime ;
    rdfs:seeAlso :sgraph .
:traveller1 :justLanded :placeC .

Emanuele Della Valle - visit http://streamreasoning.org
Streaming Linked Data
Raw Data Stream – naming the graphs

- Patterns
  - s-graphs http://ex.org/%stream-name%
  - i-graphs http://ex.org/%stream-name/%timestamp%

- Example
  - s-graph
    http://ex.org/just-landed-in
  - i-graphs
    http://ex.org/just-landed-in/2010-02-12T133412Z
    http://ex.org/just-landed-in/2010-02-12T133710Z
    http://ex.org/just-landed-in/2010-02-12T133933Z
Streaming Linked Data
Raw Data Stream – dereferencing rules

- Resource
  - http://ex.org/just-landed-in

- Linked Data Clients
  - http://ex.org/trdf/just-landed-in
  - http://ex.org/trdf/just-landed-in/2010-02-12T133441Z

- HTML Clients
  - http://ex.org/page/just-landed-in
Streaming Linked Data
Controlling the Window - Types of window

- **physical**: a given number of triples
- **logical**: a variable number of triples which occur during a given time interval (e.g., 1 hour)
  - **Sliding**: they are progressively advanced of a given STEP (e.g., 5 minutes)
  - **Tumbling**: they are advanced of exactly their time interval
Streaming Linked Data
Controlling the Window

• Physical Windows
  – Schema
    http://ex.org/%stream-name%/physical/%size%
  – Example, last 1000 triples
    http://ex.org/just-landed-in/physical/1000

• Logical Windows
  – Schema
    http://ex.org/%stream-name%/logical/%size%/%step%
  – Example, last hour sliding with a step of 1 minute
    http://ex.org/just-landed-in/logical/PT1H/PT10M

• NOTE 1: These URL patterns are translated in equivalent C-SPARQL queries that select a part of the stream
• NOTE 2: The lexical space of an interval is the same as xsd:duration, i.e., the format PnYnMnDTnHnMnS dened by ISO 8601
Streaming Linked Data
REST APIs to Control C-SPARQL Queries

• Operations:
  – **register** a C-SPARQL query
    • C-SPARQL queries have to be **registered** in the C-SPARQL Engine
  – **start** the computation
    • Be aware that the results are not “semantically” valid until the window is completely filled in
  – **pause** the computation
    • the windowing mechanism will keep working, but the window content is not processed
  – **stop** the computation
    • the window is emptied
  – **unregister** the C-SPARQL query
In our previous work we investigated C-SPARQL as an approach to treat non-RDF DSMSs as virtual RDF streams and graphs.

With this position paper, we propose an extension of our C-SPARQL Engine that publishes data streams as Linked Data.

We introduced the concept of

- A **Stream Graph** (s-graph)
  - a metadata graph that describes the current content of the window over the stream
- Several **Instantaneous Graphs** (i-graph)
  - one for each time stamp
  - `rdfs:seeAlso` is used (and reserved) to link the graphs

We define URL patterns

- to control the window
- to register, start, pause, stop, unregister a C-SPARQL query
Thank You!
Keep an eye on
http://www.larkc.eu/
http://streamreasoning.org/

Emanuele Della Valle
DEI – Politecnico di Milano
emanuele.dellavalle@polimi.it