Ontology-based Access to Streaming Data

Jean-Paul Calbimonte[1], Oscar Corcho[1], and Alasdair J G Gray[2]

[1]Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
jp.calbimonte@upm.es,ocorcho@fi.upm.es
[2]School of Computer Science, The University of Manchester, United Kingdom
a.gray@cs.man.ac.uk

Abstract. The availability of streaming data sources is progressively increasing thanks to the development of ubiquitous data capturing technologies such as sensor networks. The heterogeneity of these sources introduces the requirement of providing data access in a unified and coherent manner. We present an ontology-based streaming data access service, based on extensions to the $\mathsf{R_2O}$ mapping language and its query processor ODEMapster, and to the C-SPARQL RDF stream query language. A preliminary implementation of the approach is also presented. With this proposal we expect to set the basis for future efforts in ontology-based streaming data integration.

1 Introduction

In the context of the Semantic Web vision, several initiatives that aim at providing semantic access to traditional (stored) data sources have been launched in the past years. Most of the existing approaches attempt to provide mappings between the elements in the relational and ontological models [4].However, similar solutions for streaming data mapping and querying using ontology-based approaches have not been explored yet in depth. This has become especially relevant due to the emergence of sources such as sensor networks, capable of ubiquitous data capture, processing and delivery.

2 Ontology-based Streaming Data Access

Our approach consists in creating an Ontology-based Streaming Data Access service, depicted in Fig 1. The service receives $\mathrm{SPARQL_{STR}}$queries specified in terms of of an ontology. Then the original query is transformed into queries in terms of the sources (*query canonisation*), using a set of $\mathsf{S_2O}$mappings. These are based on the $\mathsf{R_2O}$ mapping language, which has been extended to support streaming queries and data, most notably window and stream operators. The transformed queires are written in a continuous query language (e.g. SNEEql), that is expressive enough to deal with both streaming and stored sources, and to apply window, aggregates and window-to-stream operations.

Afterwards, the query processing phase starts, and the processor will deploy distributed query processing techniques [5] to extract the relevant data from the sources and perform the required operations. The result of the query processing will be a set of tuples that will be passed to a *data decanonisation* process, which will transform these tuples to ontology instances.

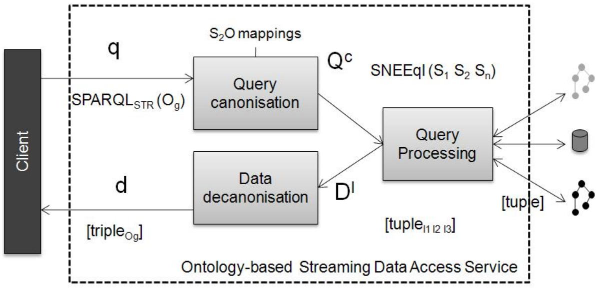

Fig. 1. Ontology-based Streaming Data Access service

3 Implementation and Walkthrough

The presented approach of providing ontology-based access to streaming data has been implemented as an extension to the ODEMapster processor [6]. This implementation generates queries that can be executed by the SNEE streaming query processor using the SNEEql query language [7].

In the example, consider a stream `windsamples` and a table `sensors`, and the $\mathsf{S_2O}$mapping to a *WindSpeedMeasurement* concept:

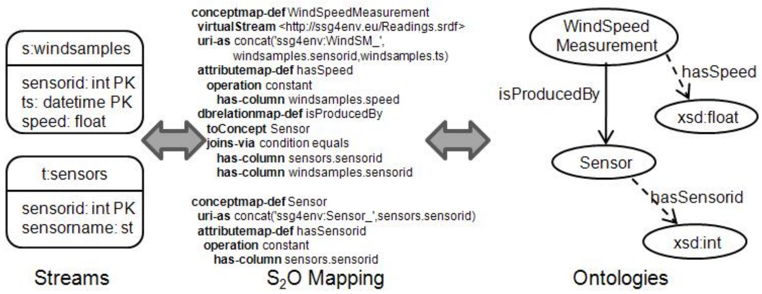

Fig. 2. $\mathsf{S_2O}$ mapping from stream to ontologies

Now we can pose a query over the ontology using $\mathrm{SPARQL_{STR}}$ for example to obtain the wind speed measurements taken in the last 10 minutes:

```
PREFIX fire: <http://www.ssg4env.eu#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?speed
FROM STREAM <www.ssg4env.eu/SensorReadings.srdf> [RANGE 10 MINUTE STEP 1 MINUTE]
WHERE { ?WindSpeed a fire:WindSpeedMeasurement; fire:hasSpeed ?speed;}
```

A class query atom $WindSpeedMeasurement(x)$ and a datatype property atom $hasSpeed(x, z)$ can be extracted from the $\mathrm{SPARQL_{STR}}$ query. The window specification $[t_i = now - 10, t_f = now, \delta = 1, unit = minutes]$ is also obtained. As it is defined in the $\mathsf{S_2O}$ mapping the *WindSpeedMeasurment* instances are generated based on the `sensorid` and `ts` attributes of the `windsamples` stream. Similarly the $\mathsf{S_2O}$ mapping defines that *hasSpeed* properties are generated from the values of the speed attribute of the `windsamples` stream. The query generated in the SNEEql language is the following:

```
SELECT RSTREAM concat('http://ssg4env.eu#WindSM',windsensor.id,windsensor.ts )
as  id ,( windsamples.speed ) as  speed
FROM windsamples[FROM NOW - 10 MINUTE]
```

The results will be transformed into tagged triples, instances of the class *WindSpeedMeasurement*.

References

1. Madden, S.R., Franklin, M.J., Hellerstein, J.M., Hong, W.: TinyDB: an acquisitional query processing system for sensor networks. ACM Trans. Database Syst. **30**(1) (2005) 122–173
2. Arasu, A., Babcock, B., Babu, S., Cieslewicz, J., Datar, M., Ito, K., Motwani, R., Srivastava, U., Widom, J.: Stream: The stanford data stream management system. In Garofalakis, M., Gehrke, J., Rastogi, R., eds.: Data Stream Management. (2006)
3. Galpin, I., Brenninkmeijer, C.Y., Jabeen, F., Fernandes, A.A., Paton, N.W.: Comprehensive optimization of declarative sensor network queries. In: SSDBM 2009. (2009) 339–360
4. Sahoo, S.S., Halb, W., Hellmann, S., Idehen, K., Jr, T.T., Auer, S., Sequeda, J., Ezzat, A.: A survey of current approaches for mapping of relational databases to RDF. W3C (January 2009)
5. Kossmann, D.: The state of the art in distributed query processing. ACM Comput. Surv. **32**(4) (2000) 422–469
6. Barrasa, J., scar Corcho, Gmez-Prez, A.: R2O, an extensible and semantically based database-to-ontology mapping language. In: SWDB2004. (2004) 1069–1070
7. Brenninkmeijer, C.Y., Galpin, I., Fernandes, A.A., Paton, N.W.: A semantics for a query language over sensors, streams and relations. In: BNCOD '08. (2008) 87–99