Towards VoIS: a Vocabulary of Interlinked Streams

Yehia Abo Sedira, Riccardo Tommasini and Emanuele Della Valle

Politecnico di Milano, DEIB, Milan, Italy
yehiamohamed.abosedera@mail.polimi.it
riccardo.tommasini, emanuele.dellavalle@polimi.it

Abstract. Since the mid 2000s', the Linked Data principles, and the tools they inspired, became a solid foundation for a decentralized yet connected Web of Data where static (or slowly evolving) data can be shared and traversed. However, the growing popularity of (fast) streaming data is now calling for an extension of Linked Data towards a Web of Data Streams. In this paper, we identify several challenges to solve towards this vision. Moreover, we present VoIS, a Vocabulary of Interlinked Streams, and we show how it contributes in addressing such challenges.

1 Introduction

The vision of a Web of Data Streams– a decentralized ecosystem of streaming and static data published and consumed by intelligent systems following Linked Data principles – is appealing as never before. This is thanks to the availability on the Web of data sources like social media, news-feeds and application logs, which are both natively data stream and Web-based.

Since the late 90s, the popularity of streaming data grew every year [2]. The Semantic Web did – and it is still doing – its part [7]. Processing data stream, using extensions of Web technologies, captured most of the attention. However, decentralization was advocated [13,5] and recently Triplewave [11] made a first step in prototyping a solution for decentralized publishing of data stream.

The Web of Data Streams scenario promises interesting use-cases, but can we handle streams in Linked Data applications? We analyzed the status of streams in relation of Linked Data principles in Table 1.

Linked Data Principles	Static Data	Data Streams
Decentralised URI	Yes	Yes
Cross Links	Yes	Maybe
Metadata	VoID, DCat, DC terms	No
Served with HTTP	RDF/HTML	No
Standards	RDF & SPARQL	Ongoing

Table 1. How linked data best practice apply to Streaming Data

Like static data, streams are referenced using URIs. Cross Links, i.e. the navigation from one data source to another, is possible from data items flowing in the streams to Linked Data, but it is unclear how the opposite can work in the general case, due to the volatile nature of those flowing data items.

Stream representation is an open problem. While many vocabulary are available to describe datasets, e.g. VoID, Dcat, DC terms, none of them is suitable to represent a data stream. Listing 1.1 clearly shows that some information is missing when we deal with streaming data. See for instance Line 9, how can we expect to dump a potentially infinite data stream? Moreover, the dump will be continuously updated, but how can we describe this phenomena? One may argue that Partition (see Lines 10-16) can solve the issue, but still metadata to describe the Window (e.g., length, starting time, or periodicity) are missing. Last but not, streams are not generally served via HTTP.

```
1  :MilanTrafficDS a void:Dataset ; # No datastream concept
2      dcterms:subject dbc:Traffic ;
3      dcterms:subject dbc:Milan ;
4      dcterms:title "Milan traffic stream";
5      dcterms:description "Streaming traffic aspect of milan
           areas";
6      terms:license <https://creativecommons.org/licenses/by-nc
           /4.0/> ;
7    foaf:homepage <http://www.example.com/>;
8    void:vocabulary <http://dbpedia.org/> ;
9    void:dataDump <http://www.example.com/dump.ttl> ;
10   void:classPartition [ void:class foaf:Organization; ];
11     void:propertyPartition [ void:property foaf:name; ] ;
12     void:subset :Window_1 .
13 :Window_1 a void:Dataset;
14     dcterms:title "Window of Milan traffic stream";
15     dcterms:description "Window of Milan traffic stream";
16     void:dataDump <http://www.example.com/dump.ttl> .
```

Listing 1.1. VoID limits for stream representation

Linked Data Notification (LDN) [6] is a recent approach that enables asynchronous communications between decentralized actors. Although LDN captures the idea of a dynamic environment and focuses on the transmission of linked data (not necessarily bounded to the HTTP protocol), it does offer a solution for decentralized data streams.

Velocity in Linked Data [8] was always seen as changes/updates and handled by means of versioning systems capable of answering comparative SPARQL queries across historical instances. Unfortunately, this mechanisms does not scale to scenarios where changes happens extremely frequently like in data streams. Indeed, the Stream Reasoning community defined SPARQL extensions that can specifically target streaming data by means of the continuous semantics, however, much as LDN, they focus on the description and the processing of the flowing data items without paying much attention to describe the data stream.

In this paper, we investigate the following research question: *Can we represent and publish data stream on the Web as Linked Data?* In order to answer such a research question, we designed VoIS[1], i.e. a Vocabulary of Interlinked Datastreams (VoIS), which is the main contribute of this paper.

The remainder of the paper is organized as follows. Section 2 reports on the requirement analysis we conducted starting from the challenge we perceive in Web of Data Streams. Section 3 illustrates the overall solution we propose with our Vocabulary of Interlinked Datastreams (VoIS). Section 4 explains how VoIS can be used to address the challenges introduced in Section 2. In Section 5, we position VoIS w.r.t. the state of the art. Finally, in Section 6, we draw some conclusions.

2 From Challenges to a Requirement Analysis

In this section, we present the challenges that we perceive in the Web of Data Streams. By discussing them, we elicit the requirements for a vocabulary able to positively answer our research question. Table 2 summarizes our requirement analysis.

Challenges	Requirememnts
Stream Discovery	Metadata are required to enable lookup, selection, linking, and licensing of streaming data sources in a decentralized way.
Stream Access	Applications must be able to negotiate access to data via streaming protocols and standards.
Stream Recall	At least the most recent part of a data stream must be accessed via Linked Data standards and protocols.
Provenance	The provenance of operations that can generate, manipulate or delete a stream must be tracked.

Table 2. Challenges and requirements in the Web of Data Streams

Stream Discovery. For a Linked Data application, discovery is the process of finding available datasets that are relevant to a certain task [12]. The process is based on the dataset metadata (i.e. void:DatasetDescription) and available endpoints (e.g. void:sparqlEndpoint).

A vocabulary must enable the discovery process by specifying which annotations are relevant for a data stream, e.g. stream rate, source and content information. Moreover, it is worth to mention that licensing plays a crucial role in a decentralized Web environment where actors can exchange data for goods.
Stream Access. In Linked Data, access usually refers to the problems of fetching a data dump or delegating the SPARQL query execution to some endpoint.

Generally, streaming data are not served using HTTP nor queried using Linked Data standards (RDF/SPARQL). It is rather provisioned via streaming APIs or queried by means of systems that support continuous queries.

[1] https://github.com/streamreasoning/vois

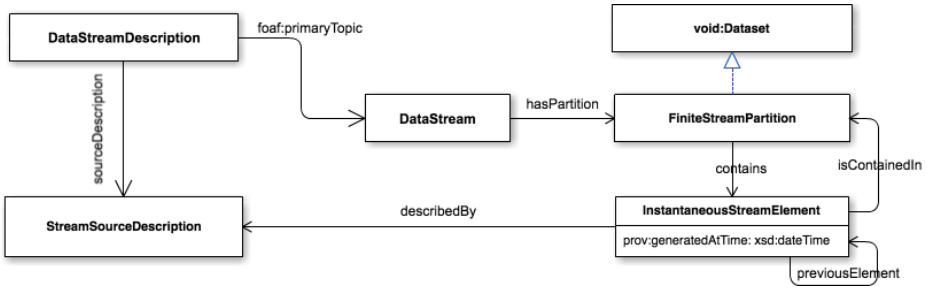

Fig. 1. Modeling stream elements using VoIS

A vocabulary is needed to define how to access streams, what are the available options and who can access them. Moreover, it must describe, if the access requires any extensions beyond HTTP-based content negotiation mechanisms.
Stream Recall. Changes in Linked Data are usually handled by versioning system and data dumps; this allows to query the previous version of a dataset by means of specialized endpoints and SPARQL extensions [14].

Unfortunately, a general assumption for stream processing is that streams are theoretically unbounded and it is not possible (or meaningful) to store them entirely. This practically means that any attempt to access streaming data with one-time queries hardly ends up with the same result twice.

A vocabulary must explain how to access past data when possible and how we can possibly do it using one-time access to data issued via traditional protocols (i.e. HTTP) and standards (RDF/SPARQL).
Provenance. A final challenge regards tracking the provenance of those processes that generate or manipulate a data stream. Indeed, discovery, processing and recall need to trust the data provisioner and they have to be able to evaluate the semantics of the data stream. A vocabulary must capture the transformations that might influence both the provenance of a stream as a whole and the one of the various time-varying data items that flow in the data stream.

3 Vocabulary of Interlinked Datastreams (VoIS)

In this section, we introduce the first version of the Vocabulary of Interlinked Datastreams (VoIS) which extends VoID to represent streams as Linked Data.

As depicted in Figure 1, the following abstractions regulate the definition of a stream and, thus, the access to its content: a (i) `vois:InstantaneousStremElement` is unit of content at a given time instant. (ii) a `vois:DataStream` – which is usually served though a streaming API [4] – is an unbounded sequences of `vois:InstantaneousStremElement`. Streams metadata are available within a (iii) `vois:DataStreamDescription`, which is a Web resource that can be accessed through HTTP. Finally a recent portion of the time-varying data items that flowed on the stream, i.e.

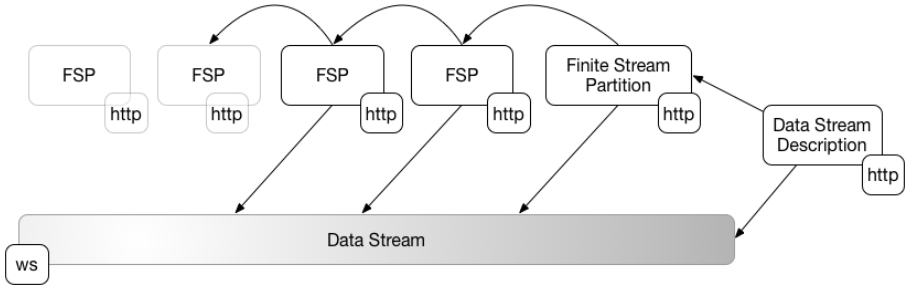

Fig. 2. How VoIS abstractions allows representing time-varying data items in the streaming–side of a λ architecture.

(iv) `vois:FiniteStreamPartition`, are made available via HTTP to enable consuming part of the stream with standard SPARQL (one-time) queries.

VoIS design takes inspiration from the λ-architecture [10] – an architectural pattern typical of Big Data applications. In order to deal with data velocity without turning down accuracy, a pipeline that computes the results on-the-fly is places side-by-side to the one that performs batch computations with a coarser time granularity. Figure 2 shows how VoIS abstractions allows representing time-varying data items in the streaming side of this architecture. Note that FSP stays for Finite Stream Portion and the leftmost FSP is not linked because old enough FSP can be forgotten from the streaming–side of the λ-architecture, since they are now available from the the batch–side. The batch side, which is not illustrated, can be implemented with Linked Data.

4 Addressing Challenges with VoIS

In this section, we show how VoIS solves the challenges we presented in Section 2. Table 3 summarizes how each requirement is satisfied in VoIS.

Challenges	Vois
Stream Discovery	DataStream, SourceDescription and StreamDescription.
Stream Access	StreamEndpoint(ws), RSPEndpoint, SPARQLendpoint.
Stream Recall	FiniteStreamPartition, Windowing
Provenance	(S) StreamToRelation (R), R2R, R2S, S2S operators with PROV-O.

Table 3. Challenges and VoIS solutions for a Web of Data Streams.

Stream Discovery. As mentioned in Section 2, traditional data discovery is based on the dataset and endpoint metadata. Similarly, we identified different set of metadata to enable discovery of data streams:

– stream metadata, i.e. time-varying metadata that refers to the stream content (e.g. rate),

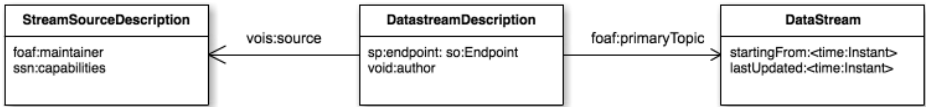

Fig. 3. Using VoIS for discovery

- source metadata, i.e. metadata that refer to the source of the stream (e.g. sensor capabilities),
- publication metadata, that connects sources, streams and possible samples and makes the stream actually findable by means of registry or repositories.

VoIS allows to capture all these levels of metadata by means of three classes, i.e., `vois:DataStream`, `vois:StreamSourceDescription`, and `vois:StreamDescription`. Figure 3 depicts the structure of the representation, while Listing 1.2 shows an example that uses Dublin Core[2] and VoID vocabularies for stream metadata about subject and licensing. SSN is used to describe the source, which is a sensor. The `vois:StreamDescription` is the file itself.

```
<> a vois:StreamDescription ;
   foaf:primaryTopic [:MilanTrafficDS a  vois:DataStream ] .

:MilanTrafficDS dcterms:title "Milan traffic stream";
   dcterms:subject dbc:Traffic ;
   dcterms:subject dbc:Milan ;
   dcterms:description "Traffic Data about Milan";
   dcterms:license <https://cc.org/licenses/by-nc/4.0/> ;
    vois:hasStreamEndpoint :MilanTrafficDS_StreamEndpoint ;
    vois:hasRSPEndpoint :MilanTrafficDS_RSPEndpoint .
    vois:hasFinitePartion :Window_1 ;
    vois:hasFinitePartion :Window_2 .

:SourceDescription_1 a vois:SourceDescription .

:MilanTrafficDS_StreamEndpoint a vois:Endpoint ;
    vois:uri "ws://example.org/milan/traffic/" .
:MilanTrafficDS_RSPEndpoint a vois:Endpoint ;
    vois:uri "http://example.org/milan/traffic/rspql" .
```

Listing 1.2. Source Description attached to data stream with VoIS

Stream Access. In a Web of Data Streams, data are processed as soon as they arrive by systems capable to deal with information flows. Streaming data access relies either on protocols specifically designed to reduce the overhead of the request or registering a continuous query to a Stream Processor that is exposed as an endpoint. Either ways require to extend the Linked Data content negotiation protocol to enable direct access and continuous querying, since streaming data are neither generally served using HTTP nor querable by with SPARQL engines.

[2] http://dublincore.org/documents/dcmi-terms/

VoIS captures this extension by means of `vois:Endpoint`, that describes the available endpoints to access a data stream. Currently, we defined two types of endpoints widely used in stream processing: `vois:StreamEndpoint` that represents streaming API (e.g. Web Socket) [11] and `vois:RSPEndpoint` that refers to endpoints that expose methods to register continuous queries [3].

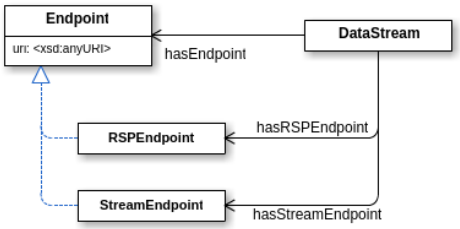

Fig. 4. Using VoIS for discovery

Stream Recall. Versioning is not a traditional stream processing use-case, however, in a Web of Data Streams providing Liked Data compliant access to, at least, a recent part of past information serves the purposes of many applications.

In VoIS, we introduced two classes `vois:InstantaneousStreamElement` and `vois:FiniteStreamPartition` for this reason. As the name suggests, the former represents a single data item in the stream with temporal metadata, while the latter is a sample that contains stream elements. More precisely, we defined `vois:Window` as a partition that is generated according to certain criteria and correspond to a time interval that contains some stream elements.

An crucial point is maintaining the order between stream elements. Indeed, any`vois:InstantaneousStreamElement` are is linked with its predecessor by means of the property `vois:previousElement`. Similarly, any `vois:Window` is part of a list linked through the property `vois:previousWindow`.

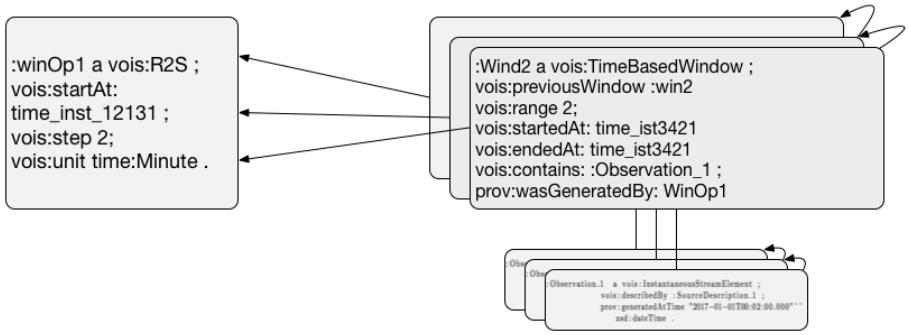

Fig. 5. Using VoIS for Recall.

Figure 5 illustrates the relation between window and stream elements. Thanks to these abstractions it is possible to locate and access stream data items in the past exploiting the temporal metadata as shown in Listing 1.3.

```
1 SELECT ?w
```

```
WHERE {  ?w a vois:Window ; vois:startedAt ?instant .
         ?instant time:inXSDDateTime ?s .
FILTER(?s > "2017-06-05T00:38:00.000"^^xsd:dateTime .)}
```

Listing 1.3. SPARQL query retrieving window overlapping with specific time instance

Provenance. As mentioned in Section 2, two are the types of transformation that can influence the provenance of a stream.

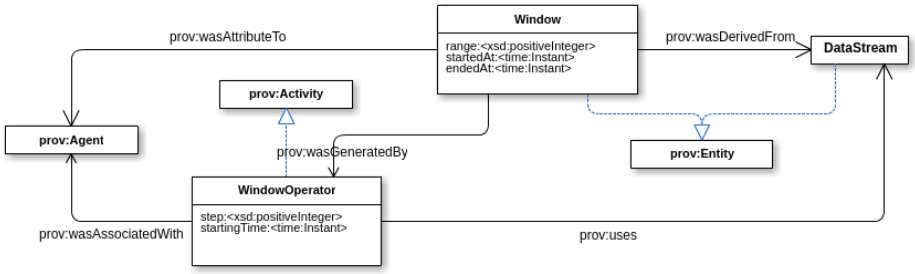

Fig. 6. Modeling window using VoIS

Figure 6 shows how the windowing mechanism is modeled in VoIS by means of PROV-O ontology. We used the Agent-Activity-Entity design pattern to represent the windowing action on a stream.

Figure 7 illustrates how to model *Stream-level transformations* in VoIS. A new relation `vois:hasAffluent` between streams was introduced to model the fact that a stream – namely a swollen one – is derived from others – namely affluent ones. Accordingly, two subtypes of streams are modeled: `vois:SwollenStream` to refer to a stream derived from other streams, and `vois:AffluentStream` to refer to stream contributing to a swollen stream.

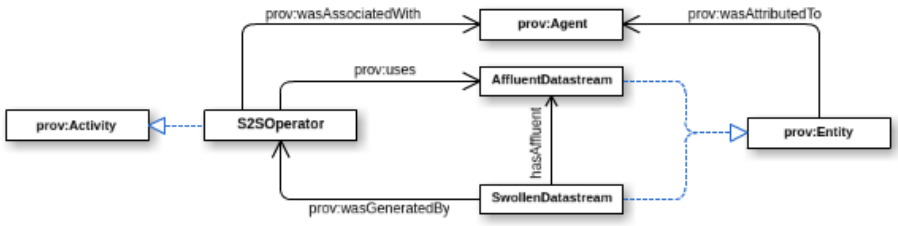

Fig. 7. Using VoIS for provenance: Composition

5 Related Work

In this section, we discuss the related work with regard to the challenges we presented in Section 2. Table 4 summarizes the discussion.

DCterms, DCAT and VoID. Several vocabularies exist to describe dataset by means of metadata. *Dublin Core Terms*[3] is the first one made to describe both physical and Web resources. It provides fifteen generic terms (eg. Title, Creator, Subject and Description). *Data Catalog Vocabulary (DCAT)* [9] is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web. It focuses on describing how static data catalogs and datasets are accessible and distributed. *Vocabulary of Interlinked Datasets (VoID)* [1] aims at describing RDF datasets and cross dataset links. VoID's use-cases comprise dataset discovery, selection and query optimization.

None of these vocabularies fully solves the challenges we identified for a Web of Data Streams. They neither capture the dynamic nature of data stream nor to represent time-varying data elements. Although discovery is still possible relying on source metadata, streaming data access is never taken into account. Recall can be partially achieved by periodically creating dumps of the stream data. However, this approach does not scale well with streaming data. Provenance tracking is possible, but limited to stream data sources only.

Linked Stream Data [13]. Sequeda and Corcho proposed a URI based mechanism to identify and access stream data coming from sensor networks. This proposal takes into account temporal and spatial aspects that are relevant from a query perspective. LSD enables discovery by referencing streams. However, authors neither proposed any protocol extensions that would support streaming access, nor discussed the problem of recalling old portions of the stream. Provenance tracking is possible by means of data source descriptions (Sensors), but not in terms of stream transformations.

Streaming Linked Data [5]. Barbieri et al. proposed to publish data streams as Linked Data by means of an RSP engine. To this extent they introduced the concepts of Stream Graph (or s-graph) and Instantaneous Graph (or igraph). This makes possible fully solves the problem of discovery and we can partially track the provenance of the RSP engine activity. However, they did not consider any protocol extension nor described the problem of recall. VoIS build on SLD introducing a better description of data stream access and provenance.

[3] http://dublincore.org/documents/dcmi-terms/

Challenges	DCterms, DCAT & VoID	SLD [5]	LSD [13]	LDN [6]	VoIS
Stream Discovery		✓	✓		✓
Stream Access		≃		≃	✓
Stream Recall	≃				✓
Provenance	≃	≃	≃	≃	✓

Table 4. Challenges and VoIS solutions for a Web of Data Streams. Symbol Legend: Empty cell, i.e. not covered; ≃, i.e. partially covered; ✓, i.e. covered.

Linked Data Notification (LDN) [6]. Capadisli et al. proposed a protocol – now officially a W3C recommendation[4] – that aims at making Web Notifications de-referenceable, persistent and reusable, i.e. compliant to Linked Data principles. Such a protocol orchestrates the communication between *senders*, *receives* and *consumers*. Stream Access might be possible using LDN since they are not bounded to any specific transmission protocols, although the communication methods between consumer and receives are RESTful. We can track the provenance of the involved actors, but not specifically stream transformations. Finally, neither Stream Discovery nor Recall challenges are in the scope of the work, which targets communication/sharing between actors rather than exploration and querying.

6 Conclusion and Future Work

In this paper, we advocated Data Streams as part of the Linked Data ecosystem. To this extent, we presented four challenges that we perceived in the Web of Data Streams. We elicit the requirements for a vocabulary able to support machines in automatically addressing those challenges in a decentralized environment and we contribute Vocabulary of Interlinked Datastreams (VoIS) as a solution.

VoIS has the potential to enable:

- Stream Discovery – it allows identifying streams that are relevant to one task according to the metadata the publishers used to describe the nature of their streams, the endpoints where they can be accessed and any other relevant information (encoded with a specialized vocabulary).
- Stream Access – it describes the information necessary to extend the Linked Data content negotiation approach in order to allow accessing data streams by the means of streaming APIs and consuming on the fly the time-varying data that flows on them.
- Stream Recall – it offers a way for a stream publisher to exposed as Linked Data a recent portion of the (naturally unbounded) data streams and, thus, permits to issue one-time SPARQL query to recall recent time-varying data.
- Provenance – it enables provenance tracking in terms of stream source and transformations both at the level of the entire data stream and at the level of the time-varying data elements that flow on it.

As for future works, evaluating the vocabulary both functionally and statistically is our current priority. We plan to do the former by asking for reviews to Semantic Web / Stream Reasoning experts, Linked Data practitioners and students. We plan to do the latter following ontology evaluation best practices.

Moreover, we are working on the implementations of a machinery (based on [3]) that practically demonstrates the advantages of using VoIS.

[4] https://linkedresearch.org/ldn/

References

1. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing linked datasets. In: Proceedings of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009, Madrid, Spain, April 20, 2009. (2009)
2. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA. pp. 1–16 (2002), http://doi.acm.org/10.1145/543613.543615
3. Balduini, M., Della Valle, E.: A restful interface for RDF stream processors. In: International Semantic Web Conference (Posters & Demos). CEUR Workshop Proceedings, vol. 1035, pp. 209–212. CEUR-WS.org (2013)
4. Balduini, M., Della Valle, E., Dell'Aglio, D., Tsytsarau, M., Palpanas, T., Confalonieri, C.: Social listening of city scale events using the streaming linked data framework. In: International Semantic Web Conference (2). Lecture Notes in Computer Science, vol. 8219, pp. 1–16. Springer (2013)
5. Barbieri, D.F., Della Valle, E.: A proposal for publishing data streams as linked data - A position paper. In: Proceedings of the WWW2010 Workshop on Linked Data on the Web, LDOW 2010, Raleigh, USA, April 27, 2010 (2010)
6. Capadisli, S., Guy, A., Lange, C., Auer, S., Sambra, A.V., Berners-Lee, T.: Linked data notifications: A resource-centric communication protocol. In: ESWC (1). Lecture Notes in Computer Science, vol. 10249, pp. 537–553 (2017)
7. DellAglio, D., Della Valle, E., van Harmelen, F., Bernstein, A.: Stream reasoning: A survey and outlook. Data Science (Preprint), 1–24
8. Käfer, T., Abdelrahman, A., Umbrich, J., O'Byrne, P., Hogan, A.: Observing linked data dynamics. In: ESWC. Lecture Notes in Computer Science, vol. 7882, pp. 213–227. Springer (2013)
9. Maali, F., Cyganiak, R., Peristeras, V.: Enabling interoperability of government data catalogues. In: Electronic Government, 9th IFIP WG 8.5 International Conference, EGOV 2010, Lausanne, Switzerland, August 29 - September 2, 2010. Proceedings. pp. 339–350 (2010)
10. Marz, N., Warren, J.: Big Data: Principles and Best Practices of Scalable Real-time Data Systems. Manning (2015), https://books.google.it/books?id=HWkMQEACAAJ
11. Mauri, A., Calbimonte, J., Dell'Aglio, D., Balduini, M., Brambilla, M., Della Valle, E., Aberer, K.: Triplewave: Spreading RDF streams on the web. In: The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II. pp. 140–149 (2016)
12. Nikolov, A., d'Aquin, M.: Identifying relevant sources for data linking using a semantic web index. In: WWW2011 Workshop on Linked Data on the Web, Hyderabad, India, March 29, 2011 (2011)
13. Sequeda, J.F., Corcho, Ó.: Linked stream data: A position paper. In: Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN09), collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington DC, USA, October 26, 2009. pp. 148–157 (2009)
14. Tappolet, J., Bernstein, A.: Applied temporal RDF: efficient temporal querying of RDF data with SPARQL. In: The Semantic Web: Research and Applications, 6th European Semantic Web Conference, ESWC 2009, Heraklion, Crete, Greece, May 31-June 4, 2009, Proceedings. pp. 308–322 (2009)