



A First Step Towards a Streaming Linked Data Life-Cycle

Riccardo Tommasini²(✉) , Mohamed Ragab² , Alessandro Falcetta¹ ,
Emanuele Della Valle¹ , and Sherif Sakr²

¹ DEIB, Politecnico di Milano, Milan, Italy

{alessandro.falcetta,emanuele.valle}@polimi.it

² DataSystem Group, University of Tartu, Tartu, Estonia

riccardo.tommasini@polimi.it, {mohamed.ragab,sherif.sakr}@ut.ee

Abstract. Alongside with the ongoing initiative of FAIR data management, the problem of handling *Streaming Linked Data* (SLD) is relevant as never before. The Web is changing to tame Data Velocity and fulfill the needs of a new generation of Web applications. New protocols (e.g. WebSockets and Server-Sent Events) emerge to grant continuous and reactive data access. Under the Stream Reasoning initiative, the Semantic Web community has been actively working on query languages, engines, and vocabularies to address the scientific and technical challenges of taming Data *Velocity* without neglecting Data *Variety*. Nevertheless, a set of guidelines that showcase how to *reuse* existing resources to produce and consume streams on the Web is still missing. In this paper, we walk through the life-cycle of streaming linked data. We discuss the challenges of applying FAIR principles when publishing data streams. Moreover, we contextualise the usage of prominent Semantic Web resources, i.e., (i) **TripleWave**, **R2RML/RML**, **VoCaLS**, **RSP-QL**. We apply the guidelines to three representative examples of *real-world* Web streams: **DBpedia** Live changes, **Wikimedia** EventStreams, and the Global Database of Events, Language and Tone (**GDELT**). Last but not least, we open-sourced our code at <https://w3id.org/webstreams>.

Keywords: RDF Streams · Streaming linked data · RDF Stream Processing · Stream reasoning

1 Introduction

Alongside with the ongoing initiative of FAIR data management [30], the Semantic Web community, under the Stream Reasoning umbrella [12, 19], started investigating the problem of Streaming Linked Data (SLD) management.

The Web is changing to fulfill the needs of a new generation of Web applications that demands real-time data access [9]. On the one hand, new protocols like **HTTP Long Polling**¹ and **WebSockets** (see footnote 1), and **Server-Sent**

¹ <https://www.pubnub.com/learn/glossary/what-is-http-streaming/>.

Events² are becoming popular. On the other hand, the Stream Reasoning initiative is actively working to extend the Semantic Web Stack for RDF Stream Processing (RSP)³.

RSP aims at taming Data Velocity, i.e., the challenge of processing data as soon as they are produced and before they are no longer valuable, without neglecting Data Variety [10]. RSP extends the Semantic Web stack with a data model (i.e., RDF Stream), continuous extensions of SPARQL (e.g., RSP-QL [11]), and a number working prototypes that enable *on-the-fly* analysis of vast and heterogeneous data streams coming from complex domains [5, 24]. The literature also includes middleware systems like LSM and Ztreamy [3, 24], resources for publishing like TripleWave [20], and vocabularies to describe data streams [28].

Nevertheless, the quest for streams on the Web is still not FAIR. The Streaming Linked Data life-cycle is still unclear since a set of guidelines that explain how to produce and consume Streaming Linked Data is still missing. In this paper, we fill this gap presenting a Streaming Linked Data life-cycle. Additionally, we bootstrap a catalog of linked streams publishing three Real-World Web Streams [20]: DBpedia Live⁴ that shares the RDF changes on DBpedia, EventStreams⁵ that streams the changes across all the WikiMedia projects, and the Global Database of Events, Languages, and Tone (GDELT)⁶. We choose these streams to illustrate a comprehensive set of challenges that need to be tackled when publishing streams on the Web. In summary, the main contributions of this paper are:

1. We discuss the challenges to publish FAIR streaming data on the Web.
2. We present a publication lifecycle that is suitable for streaming linked data.
3. We include a set of representative examples for producing and consuming Streaming Linked Data that *reuse* existing resources (i.e., RML, TripleWave, VoCaLS), and RSP-QL.

This study is relevant for the Semantic Web community, in particular the RSP community, given the growing interest in Streaming Linked Data [23]. Moreover, the selected streams are relevant for those whose interests includes Knowledge Graph, news analysis, and Fact-Checking.

The remainder of the paper is organized as follows: Sect. 2 presents the Streaming Linked Data life-cycle, it discusses FAIR principles for streaming data management, and positions related resources. Sect. 3 presents the three real-world Web Streams, i.e., DBpedia Live, Wikimedia EventStreams, and GDELT. Sect. 4 applies the life-cycle to the selected streams. It highlights the assumptions, the design choices, and the open problems. Moreover, it shows examples of RDF Stream Processing using RSP-QL. Finally, Sect. 5 draws to conclusions, and presents future works.

² <https://developer.mozilla.org/en-US/docs/Web/API/EventSource>.

³ <https://www.w3.org/community/rsp/>.

⁴ <https://wiki.dbpedia.org/online-access/DBpediaLive>.

⁵ <https://stream.wikimedia.org>.

⁶ <https://gdeltproject.org>.

2 Streaming Linked Data Life-Cycle

In this section, we present the publication life-cycle for Streaming Linked Data. A premise to data publication is the identification of relevant sources. Figure 1 shows the three situations a practitioner might find when they publish Streaming Linked Data, i.e., our ultimate goal, which is identified by the *lower-right* quadrant.

The other quadrants present possible starting points, i.e., (upper-left) Web Data published in batches; (upper-right) Linked Data published in batches; and (lower-left) Web Data published as streams. Before introducing the life-cycle, we present a requirement analysis. In particular, we start from the FAIR principles [30] for data management, which aim at publishing data in a format that is Findable, Accessible, Interoperable, and Reusable, and we discuss their applicability to streaming data.

2.1 FAIR Principles for Streaming Data

Findable. (F1) data should be assigned unique and persistent identifiers, e.g., DOI or URIs. (F2) data should be assigned metadata that includes descriptive information, data quality, and context. (F3) metadata should explicitly name the persistent identifier since they often come in a separate file. (F4) Identifiers and metadata should be indexed or searchable.

Requirements related to findability immediately apply to Streaming Data. It is not hard to imagine a Web Stream as a resource that is uniquely identified, thoroughly described with rich metadata, indexed, searched.

For effective machine-readable stream representation, Tommasini et al. proposed The Vocabulary for Cataloging and Linking Streams (VoCaLS) [28]. VoCaLS builds on DCAT⁷ and is organized into three modules: Core, Service Description, and Provenance. In particular, *VoCaLS Core* follows the F-principles, i.e., a (i) *vocals:Stream* represents a Web stream, while a (ii) *vocals:StreamDescriptor* is an HTTP-accessible document that contains the stream's metadata.

Accessible. (A1) Data and metadata should be accessible via (a) free and (b) open-sourced, and (c) standard communication protocols, e.g., HTTP or FTP. Nonetheless, authorization and authentication are possible. (A2) metadata should be accessible even when data is no longer available.

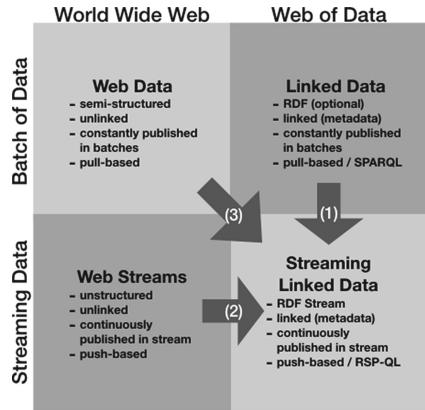


Fig. 1. Publication starting points.

⁷ <https://www.w3.org/TR/vocab-dcat/>.

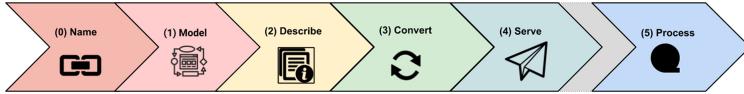


Fig. 2. Streaming linked data publication lifecycle.

Streaming data introduces a paradigm-shift for what concerns data access. Data are no longer put at rest and analysed. Instead, data are processed as soon as they arrive and before they are no longer useful. This change reached the Web architecture, which includes new protocols like *Server-Sent Events* for reactive processing and *WebSocket* for continuous consumption. These standardized solutions full-fill velocity-related requirements [26] as well as A1 and A2. Nevertheless, the Web is still founded on HTTP. Thus, Tommasini et al. proposed to decouple the stream description from the endpoint that enables data distribution. This approach ensures scalability and interoperability because a stream might have multiple endpoints. VoCaLS, in its current version, allows defining a *vocals:StreamEndpoint* that points to actual data sources.

Interoperable. (I1) Data and metadata must be written using formal languages and shared vocabularies that are accessible to a broad audience. (I2) Such vocabularies should also fulfill FAIR principles. (I3) Data and metadata should use qualified references to other (meta-)data.

In the literature, there is a general agreement on *RDF Stream* as the data-model of choice for making streaming data interoperable (cf Definition 1).

Definition 1. *An RDF Stream is a data stream where the data items o are RDF graph, i.e., a set of RDF triples of the form (subject, predicate, object) [11].*

Prominent vocabularies to use for the stream content include but are not limited to FrAPPE [4], SAO [14], SSN [7], and SIOC [22]. Recently, Schema.org includes the classes relevant for streams representation, i.e., DataFeed⁸ and DataFeedItem⁹, but their adoption has not been estimated yet.

Reusable. (R1) Data should adopt an explicit license for access and usage. (R2) Data provenance should be documented and accessible. (R3) Data and metadata should comply with community standards.

Requirements related to reusability apply straightforwardly to streaming data. In particular, VoCaLS provenance modules allow modeling streaming data generation and transformation (R2). Additionally, requirements and commandments recognized by the stream reasoning community are available (R3).

2.2 Five-Steps into Streaming Linked Data

In this section, we present our publication life-cycle for Streaming Linked Data. Figure 2 summarises our proposal that takes into account FAIR principles, W3C

⁸ <https://schema.org/DataFeed>.

⁹ <https://schema.org/DataFeedItem>.

best practices¹⁰, and the seminal of Hyland et al. [17], Hausenblas et al. [8] Villazon-Terrazas et al. [29]. Figure 2 shows the following steps: (0) Name (1) Model, (2) Describe, (3) Convert, (4) Publish, and (5) Processing. We elaborate on steps (0)–(4) in the following, while we postpone step (5), which is usually a corollary to publication, to Sect. 4.2.

Step 0 Name Things with (HTTP) URIs. The goal of Step 0 is to design (HTTP) URIs that identify the relevant resources [17]. To this extent, W3C presents a set of best practices for publishing¹¹, i.e., do not bind URIs to any implementation, keep them stable, and opaque¹². Moreover, Linked Data Principles prescribe to use (HTTP) URIs to identify resources.

In the RSP community, The general trend is decoupling the access to the stream resource, which is available via HTTP, and the stream content, which is not a resource and can use an arbitrary Web protocol (e.g. WebSocket) [6, 28]. Moreover, Sequeda et al. [25] proposed a URI-based mechanism to identify and access stream data items that take into account temporal and spatial aspects.

Step 1 Model the Streams. The goals of Step 1 are: (i) understanding and capturing the domain knowledge for enriching the raw [17] as well as (ii) identifying relevant resources.

In practice, this process requires collecting and reviewing applications, documentation, and data samples Then, with the help of knowledge engineers, domain knowledge, is captured into ontological models using a knowledge representation languages like RDFS or OWL2. The reuse of existing authoritative vocabularies is of paramount importance for FAIR data management.

During Step 1, the stakeholders ask questions that are translated into information needs [17]. For data streams, some information needs imply continuous semantics and their answer change over time, e.g., *What is the stream rate?* *What are the most relevant resources in the stream in the last 5 min?* Moreover, standard information needs are meaningful, e.g., *What is the stream main topic?* *For what statistics are useful?* *What problem you aim to solve?* The identified information needs should be formulated in natural language, and then *post-hoc* formalized using an appropriate formal language.

Step 2 Describe the Stream. Step 2 aims at providing representations for the streams [17] that can be consumed by both human and software agents.

To this extent, VoCaLS is the vocabulary of choice [28]. It allows writing machine-readable stream descriptions to share via HTTP. Moreover, it enables continuous data access via specified stream endpoints with streaming protocols.

In practice, describing a stream requires identifying relevant metadata such as publishers, the domain of provenance, documentation, and related resources. If the streaming data re-uses ontologies or schemas, they should be linked alongside the license for accessing the streaming data. Additionally, time-varying statics such as the stream rate and the number of active consumers should be available.

¹⁰ <https://www.w3.org/TR/ld-bp/>.

¹¹ <https://www.w3.org/TR/cooluris/#cooluris>.

¹² I.e., do not let the user understand the underlying infrastructure.

Step 3 Convert Data to RDF Streams. The goal of Step 3 is fostering interoperability by enabling RDF (Stream) provisioning. Indeed, data are frequently shared using document-based format (e.g., JSON, XML), APIs, or using (semi)-structured data formats (e.g., TSV, CSV).

In practice, the conversion mechanism occurs automatically using approaches based on mapping languages that decouple conversion and modeling [17]. For instance, R2RML¹³, and its extension RML [13], allow converting almost any non-RDF data source. Nevertheless, existing mapping engines were successfully used only for *batch* data conversion and, thus, they must be adapted or extended to work with Streaming Linked Data. The conversion mechanism involving data stream, due to their infinite nature, calls for *continuous* semantics, i.e., producing an infinite output from an infinite input [12]. Although this research problem is still open, a suitable solution is converting one stream-element a time [20].

Step 4 Serve Streams on the Web.

The goal of this step is to provide the data to the audience of interest. Static Datasets – opportunely described with contextual vocabularies [1] – are either added to the Linked Open Data (LOD) Cloud, shared using REST APIs, or exposed via SPARQL endpoints. Critical aspects of Step 6 are licensing, audit, and access control [8, 17].

Currently, one cannot add Streaming Linked Data as they are to the LOD Cloud [20, 28], nor one can query them via SPARQL endpoints [12]. However, *Triple-Wave* [20] is a resource for publishing Streaming Linked Data that exposes an *S-Graph* via HTTP, i.e., a “static” named graph called that acts as stream descriptor. The S-Graph can be added to the LOD Cloud or indexed by search engine, making the stream findable and accessible via available endpoints.

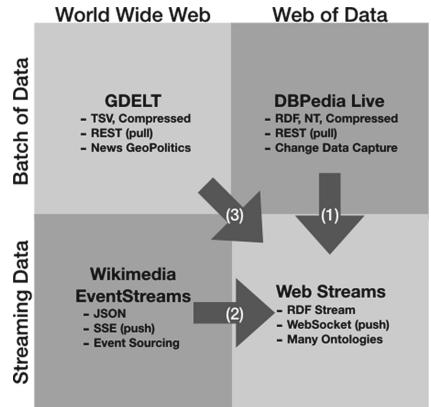


Fig. 3. Stream examples overview.

3 Real-World Web Streams

In this section, we present three representative examples of real-world Web Streams published as Streaming Linked Data, i.e., DBpedia Live, Wikimedia EventStream, and the Global Dataset of Event, Language and Tone (GDELT). Our goal is to provide guidelines that (i) motivate the samples of RDF streams we selected to bootstrap our catalog, and (ii) facilitate future extensions of the catalog. Therefore, we selected three examples that cover all the publication scenarios. Figure 3 positions each of them w.r.t. the scenarios from Fig. 1.

¹³ <https://www.w3.org/2001/sw/rdb2rdf/r2rml/>.

3.1 DBpedia Live

DBpedia is a community project to extract structured data from Wikipedia in the form of an Open Knowledge Graph. DBpedia is served as linked data using Entity-Pages, REST APIs, or SPARQL endpoints. The latest version of DBpedia counts more than 470 millions of RDF triples¹⁴.

DBpedia Live (DBL) is a changelog stream continuously published to keep DBpedia replicas in-synch. A Synchronization tool designed to consume the batches and update the local DBpedia copy is available [21]. DBL uses a pull-based mechanism for data provisioning. DBL shares RDF data using DBPedia ontology (DBO), which is a cross-domain ontology, manually created from the most used Wikipedia info-boxes. The latest version of DBO covers 685 classes and 2795 different properties. It is a direct-acyclic graph, i.e., classes may have multiple super-classes, as required to map it to `schema.org`.

A DBL update consists of four compressed N-Triples (NT) files. Two main different files, i.e., *removed*, and *added*, determine the insertion and deletion stream. Two further streams share files for clean updates that are optional to execute: *reinserted*, which corresponds to unchanged triples that can be reinserted, and *clear*, which prescribe the delete queries that clear all triples for a resource. Although edits often happen in bulk, this information is not present in DBL, i.e., changes come with no timestamps.

3.2 Wikimedia EventStreams

```

1 meta: [...], timestamp: 1554284688, id: 937929642, bot: false,
2 type: 'edit', title: 'Q31218558',
3 user: 'Tagishsimon', wiki: 'wikidatawiki'
4 comment: '...', parsedcomment: [...],
5 minor: false, namespace: 0, patrolled: true,
6 length: { new: 5530, old: 5445 },
7 revision: { new: 901332361, old: 756468340 },
8 server_name: 'www.wikidata.org',
9 server_script_path: '/w', server_url: 'https://www.wikidata.org',
```

Listing 1.1. Wikimedia EventStream recentchanges example data.

Wikimedia EventStream (WES) is a web service created at Wikimedia Foundation, i.e., an American non-profit organization that hosts, among the other, open-knowledge projects like Wikipedia. In particular, Wikimedia invests financial and technical resources for the maintenance of projects that foster free and open knowledge. WES was originally used for internal data analysis and was open-sourced in 2018. It exposes streams of structured data using SSE.

WES data is gathered from the internal *Kafka* cluster. it includes logs and change-data captures. In practice, WES refers to eight distinct *JSON* streams: (i) recentchanges (ii) revision-create (iii) page-create (iv) page-property-change

¹⁴ <http://dbpedia-live.openlinksw.com/live/>.

(v) page-links-change (vi) page-move (vii) page-delete (viii) page-undelete. In this section, we focus only on the *recentchanges* stream, which is the one with the most complex content.

Listing 1.1 shows an example of a *recentchange* data item. In WES’s recent changes, the event title links to the Wikidata entity entity, e.g., in Listing 1.1 title points <https://www.wikidata.org/entity/Q31218558.ttl>. Each item is timestamped (line 1) and typed (line 2), helping us to introduce the concept of event [18]. Four kinds of events are possible: “edit” (cf. Listing 1.1) for existing page modification; “new”, for new page creation, “log” for log action, “external” for external changes, and “categorize” for category membership change.

3.3 Global Database of Events, Language and Tone (GDELT)

Table 1. Example of GDELT event data.

GLOBALEVENTID	Actor1Code	EventCode	AvgTone	DATEADDED
35209457	GOV	020	-3.4188	20190401203000
835209458	LEG	120	1.39860	20190401203000
835209459	USA	040	1.39860	20190401203000

The GDELT is the largest open-access *spatio-temporal* archive for human society. Its Global Knowledge Graph spans more than 215 years and connects people, organizations, locations worldwide. GDELT captures themes, images, and emotions into a single holistic global network. GDELT data come from a multitude of news sources using Natural Language Processing techniques.

GDELT consists of three different streams, i.e., Events, Mentions¹⁵, and Global Knowledge Graphs (GKG)¹⁶, delivered as compressed TSV every 15 min: The *Event stream* shares Geo-Political events appearing in the news. Each event refers to two Actors, e.g., nations or public figures, participating in each event and the action they perform, e.g., a *diplomatic visit*. The *Mention stream* records every mention of a particular event in the news over time, along with the article’s timestamp. The mention stream is linked to the Event stream by the Global Event ID field. The *GKG stream* connects people, organizations, locations, themes, news sources, and events across the planet into a massive network. GKG stream is rich and multidimensional. Fields like the GCAM, Themes, or Persons include more than one entry, using a CSV or similar formats.

GDELT streams use the dyadic format for Conflict and Mediation Event Observations (CAMEO) [15], and they contain Global Content Analysis Measures (GCAM). Table 1 shows few relevant fields for the Event stream, i.e., GLOBALEVENTID and DATEADDED that identify the event uniquely and over time; Actors and Event code that link the event to CAMEO, and as an example of GCAM dimension, AvgTone.

¹⁵ [Link to GDELT-Event_Codebook-V2.0.pdf](#).

¹⁶ [Link to GDELT-Global_Knowledge_Graph_Codebook-V2.1.pdf](#).

4 Putting the Life-Cycle into Practice

In this section, we walk through the life-cycle steps for DBL, WES, and GDELT.

4.1 Publication

The Streaming Linked Data life-cycle starts with data publication, i.e., Steps (0)-(4) described in Sect. 2.

Step 0 Name. Regarding *Step 0*, we opted for re-naming the published linked streams. Our base-uri is <http://linkeddata.stream>. Moreover, we adopt the DBpedia best practice for URI design, i.e.,

- `/resource/stream` indicates Web stream URI and its RDF representation
- `/page/stream` provides an HTML representation of the stream resource;
- `/ontology/onto` provides the PURL to vocabulary of the stream resource.

Step 1 Model the Streams. Regarding *Step 1*, we opted for using OWL 2 and RDFS as knowledge representation languages.

DBL provisions data in RDF make use of DBPedia ontology (DBO), therefore we did not need to take care of any further modelling effort.

For *WES recentchange*, we collected the information about the streams schemas into an OWL 2 ontology(see footnote 19). WES are designed around the notion of event. Thus, we ported related classes from contextual vocabularies like the Event Ontology¹⁷. Regarding the *recentchanges* stream, we emphasizes the modeling of the events types in our ontology, i.e., “edit”, “new”, “log”, “categorize”, or “external”. Similarly, we take into account what could be represented as external resources like Wikidata. Since data items are timestamped individually, and we used this timestamp to name the graph containing all the event data.

For GDELT, we collected all the information regarding the streams schemas into an OWL 2 Ontology that involves both CAMEO¹⁸ and GCAM¹⁹. The CAMEO ontology is a coding scheme designed for the study of third-party mediation in international disputes. It contains a hierarchical coding scheme for dealing with sub-state actors, event types, and an extensive taxonomy for religious groups and ethnic groups. For CAMEO, we focus on event types and actors, creating a comprehensive hierarchy for the stream. GCAM is a pipeline of 18 content analysis tools. Each news article monitored by GDELT goes through GCAM pipeline that captures over 2230 dimensions, reporting density, and value scores for each. Using GCAM, you can assess the density of “*Anxiety*” speech via Linguistic Inquiry and Word Count (LIWC), or “*Smugness*” via WordNet Affect. The GCAM Master Codebook lists of all of the dimensions available²⁰. We converted the Codebook in RDF, and we refined it manually to link it to DBPedia and WordNET.

¹⁷ <http://motools.sourceforge.net/event/event.122.html>.

¹⁸ <http://linkeddata.stream/ontologies/cameo.owl>.

¹⁹ <http://linkeddata.stream/ontologies/gcam.owl>.

²⁰ <http://data.gdeltproject.org/documentation/GCAM-MASTER-CODEBOOK.TXT>.

Step 2 Describe the Stream. For *Step 2*, we opted for using VoCaLS and DCAT as vocabularies for writing stream descriptions. Relevant metadata about the streams include license, data formats, documentations, and human-readable descriptions. We collected all this information for each of the aforementioned Web streams into a *vocals:StreamDescriptor*.

```

1 :dbl a vocals:StreamDescriptor ;
2 dcat:title "DPedia Live" ; dcat:publisher <http://www.dbpedia.org> ;
3 rdfs:seeAlso <https://wiki.dbpedia.org/services-resources/ontology>
4 dcat:license <https://creativecommons.org/licenses/by-nc/4.0/> ;
5 dcat:publisher <http://www.linkeddata.stream/about> ;
6 dcat:dataset :dblstream .
7 :dblstream a vocals:RDFStream .
8 vocals:hasEndpoint :dblendpoint, :dblendpointold .
```

Listing 1.2. A DBL Stream VoCaLS description. Prefixes omitted.

Listing 1.2 presents the *vocals:StreamDescriptor* that contains necessary information about the DBL²¹, which contains basic information about the publisher and the license (line 6) as well as links to other relevant datasets (lines 5-6).

```

1 wes:recentchange a vocals:StreamDescriptor ;
2 dcat:dataset :recentchange ;
3 dcat:title "Wikimedia Recentchanges Event Stream"^^xsd:string ;
4 dcat:publisher <http://www.linkeddata.stream/about> ;
5 dcat:license <https://foundation.wikimedia.org/wiki/Terms_of_Use/en> .
6 :recentchange a vocals:Stream ;
7     vocals:hasEndpoint :wesendpoint, :wesendpointold .
```

Listing 1.3. Wikimedia EventStream recentchanges sGraph. Prefixes Omitted.

Listing 1.3 shows the *vocals:StreamDescriptor* for the *recentchanges* stream²², which includes the Wikimedia terms of use as license, and the OWL 2 ontology we designed to describe WES' domain.

```

1 :events a vocals:StreamDescriptor ;
2 dcat:title "GDELT Event Stream"^^xsd:string ;
3 dcat:publisher <http://www.linkeddata.stream/about> ;
4 dcat:description "GDELT Events Stream"^^xsd:string ;
5 dcat:license <https://creativecommons.org/licenses/by-nc/4.0/> .
6 dcat:dataset :eventstream .
7 :eventstream a vocals:Stream ;
8     vocals:windowType vocals:logicalTumbling ;
9     vocals:windowSize "PT15M"^^xsd:duration ;
10    vocals:hasEndpoint :eventEndpoint, :oldEndpoint .
```

Listing 1.4. A GDELT Stream description using VoCaLS. Prefixes omitted.

²¹ <http://linkeddata.stream/resource/dbl>.

²² <http://linkeddata.stream/resource/recentchanges>.

Listing 1.4 shows a VoCaLS description for the GDELT Event Stream. GDELT does not use a license format, thus we include a license that is compliant with the terms of use. We also linked ontologies and mappings using `rdfs:seeAlso`. Since the stream is published regularly as 15 min batch, we include metadata about the rate, i.e., `vocals:windowType` and `vocals:windowSize` at lines 6, 7.

Step 3 Convert Data to RDF Streams. For *Step 3*, we faced difference cases, RDF triples, JSON, and TSV. Thus, we discuss the best choice for each scenarios to the extent of providing useful guidelines.

DBpedia Live already provides RDF Data. Thus, we do not have to apply any conversion mechanism. Nevertheless, DBL updates are shared as compressed NT triple files, which allow discussing the alternative RDF Streams serializations, i.e., triple-based (TB) and graph-based (GB). Although the two serializations have been proven to be semantically equivalent, some trade-offs are worth unveil. Table 2 lists the most important differences:

(i) Triple-based RDF Stream serialization will require to extend the RDF format to enable Event-Time processing. Indeed, no additional information like the timestamp can be carried when streaming a single RDF triple. On the other hand, graphs named after their time annotation. For instance, using the Time-Ontology, can be used as a form of punctuation. (ii) While TP is optimized for triple-pattern evaluations [23], GB simplifies BGPs'. (iii) Finally, GB simplifies time-based provenance tracking as graphs are annotated with timestamps. On the other hand, TP simplifies the window maintenance as new items arrive as triple to add/remove already.

On the other hand, in WES and GDELT stream, the elements provisioned rich document formats, i.e., JSON and TSV. Therefore, we must set up a conversion pipeline to foster interoperability (I).

```

1  "@context": {
2    "@vocab": "http://linkeddata.stream/ontology/wes/",
3    "xsd": "http://www.w3.org/2001/XMLSchema#",
4    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
5    "timestamp": { "@type": "xsd:dateTime" },
6    "title": { "@id": "rdfs:about" },
7    "type": "@type", "id": { "@id": "@id", "@type": "Event" }}}
```

Listing 1.5. JSON-LD Context for Event in Listing 1.1

Table 2. Triple-based vs Graph-based RDF Stream Serialization

Triple	Graph
Custom RDF format for Event Time	Use named graph for punctuation
Optimised for triple patterns evaluation	Optimised for (named) basic graph patterns
Simplifies window maintenance	Naturally supports provenance

In WES's recent changes, the event title links to the Wikidata entity, e.g., in Listing 1.1 title points <https://www.wikidata.org/entity/Q31218558.ttl>. The first approach for *recentchange* conversion will require de-reference this information at conversion time and stream out the stream content with contextual domain data. However, such an approach is discouraged as we have the size of the referred entity might be inappropriate for stream provisioning (E.g., the entity above contains 6166 triples). Alternatively, being WES's data JSON, it is sufficient to attach a valid JSON-LD context to each document of Listing 1.5.

In GDELT, data arrive in TSV format. Thus, we cannot count on the *JSON-JSON-LD* compatibility, we must set up a mapping-based conversion. We opt for the RML mapping language [13] and for a modified version of CARML mapping engine that handles the annotation process incrementally to minimize the translation latency.

```

1 <GEM> a rr:TriplesMap ; rml:logicalSource <source> ;
2   rr:subjectMap [
3     rr:template "http://linkeddata.stream/resource/gdelt/{GLOBALEVENTID
4       }";
5     rr:class gdelt:Event;
6     rr:graphMap
7       [ rr:template "http://linkeddata.stream/resource/time/{DATEADDED}"
8         ];
9     rr:predicateObjectMap [ rr:predicate cameo:type;
10      rr:objectMap
11        [rr:template "http://linkeddata.stream/ontologies/cameo/{EventCode}";
12          ];
13     rr:predicateObjectMap [
14       rr:predicate cameo:actor;
15       rr:objectMap [ rr:parentTriplesMap <Attr1TM> ]]; [...].
```

Listing 1.6. RML Mapping for GDELT Event Stream Conversion (Subset).

Listing 1.6 shows a portion of the GDELT event RML mapping. We created a named graph after the event timestamp using `rr:graphMap` at line 5. We used `rr:class` to assign the `sem:Event` type to the data at line 4. Moreover, we also assign the CAMEO type using `cameo:type` at line 7. To model the actors and their hierarchy of types, we include a separate triple-map at line 12.

Step 4 Serve Streams on the Web. Regarding *Step 4*, we enable continuous data access to the content of the streams using a customized version of `TripleWave` that connects to the streams of choice and provision the RDF Streams via WebSocket. Nevertheless, to make this access sustainable over time, we require the interested user to run, locally to their machine, a docker image that executes the conversion pipeline accessing data at the source.

```

1 :dbendpoint a vocals:StreamEndpoint ;
2   dcat:format frmt:JSON-LD ;
3   dcat:accessURL "ws://localhost:8081/dbl" .
4 :dbendpointold a vocals:StreamEndpoint ;
5   dcat:format frmt:NT ;
6   dcat:accessURL "http://downloads.dbpedia.org/live/changesets/
    lastPublishedFile.txt".

```

Listing 1.7. DBL Stream Endpoints. Prefixes omitted.

To enable data access, we added *vocals:StreamEndpoint* to each stream description that links to the exposed WebSocket. Moreover, for provenance reasons we included a *vocals:StreamEndpoint* that points to the original data source. Listing 1.7, Listing 1.8, and Listing 1.9 show such endpoints for DBL, WES recentchange, and GDELT Events, respectively.

```

1 :wesendpoint a vocals:StreamEndpoint ;
2   dcat:format frmt:JSON-LD ;
3   dcat:accessURL "ws://localhost:8081/wes/recentchanges" .
4 :wesendpointold a vocals:StreamEndpoint ;
5   dcat:format frmt:JSON ;
6   dcat:accessURL "https://stream.wikimedia.org/v2/stream/recentchange".

```

Listing 1.8. WES recentchange Stream Endpoints. Prefixes omitted.

```

1 :eventEndpoint . a vocals:StreamEndpoint ;
2   dcat:format frmt:JSON-LD;
3   dcat:accessURL "ws://localhost:8080/gdelt/events" .
4 :oldEndpoint . a vocals:StreamEndpoint ;
5   dcat:format frmt:TSV;
6   dcat:accessURL "http://data.gdeltproject.org/gdeltv2/lastupdate.txt"

```

Listing 1.9. GDELT Event Stream Endpoints. Prefixes omitted.

4.2 Processing

This last step, which is a corollary to the publication cycle, aims at making sense of the published streams using standard languages and protocols.

Although many of SPARQL dialects for continuous query exist [6, 23, 27], RSP-QL is the reference model of choice for RDF Stream Processing (RSP). Moreover, it is also query language that includes all the existing SPARQL extensions [11] RSP-QL allows defining Stream-to-Stream transformations using a simple query model that is backward-compatible with SPARQL 1.1. and includes the operators' families: *Stream-to-Relation* (S2R) operators, which bridge the world of unbounded streams with the world of finite stream portions. Listing 1.10 shows in line 4 a typical operator of this kind, i.e., a Time Window. *Relation-to-Relation* (R2R) operators, which can be executed over the finite outputs of the S2R. In

the RSP context, R2R coincides with SPARQL algebraic operators. *Relation-to-Stream* (R2S), which returns to infinite data streams. Listing 1.10 shows in line 2 the RStream, which is an example of R2S that append a timestamp to the output of the R2R operator.

```

1 PREFIX dbl <http://linkeddata.stream/resource/dbl/>
2 SELECT (COUNT (?entity) AS ?count) ?entity
3 FROM NAMED WINDOW <wa> ON dbl:added [RANGE PT1M STEP PT10S]
4 FROM NAMED WINDOW <wr> ON dbl:removed [RANGE PT1M STEP PT10S]
5 WHERE { WINDOW ?w { ?entity ?p ?o . } }
6 GROUP BY ?entity ORDER BY DESC(?count)

```

Listing 1.10. RSP-QL Query counting the most edited entities in a minute.

DBL was successfully used to analyze DBpedia evolution. In particular, DBL was used to satisfy information needs like *How many entities are updated in last 5 min?*. DBpedia live statistics is a daily updated Web page that shows analyses like top-k entity changes. We decided to compute the aforementioned statistics using RSP-QL. Listings 1.10 shows how the query that counts the top-20 most edited entities in the last minute looks like.

A recent challenge triggered a real-time data analysis of WES data, most of which focus on the data visualization aspect²³. The projects empower simple statistical analyses such as comparison of *bot vs. human* editor, *minor vs. major* changes detection or categorizing the type of events. Listing 1.11 shows an example of RSP-QL query calculating the stream rate every minute.

```

1 PREFIX wes : <http://linkeddata.stream/resource/wes>
2 SELECT (COUNT{*}/60) ?ratesec
3 FROM NAMED WINDOW <w> ON wes:recentchange [RANGE PT60S STEP PT60S]
4 WHERE { WINDOW <w> { ?s a wes:Event } }

```

Listing 1.11. WES Recentchange rate

Several studies have been running using GDELT. In particular, data visualization techniques that take into account the spatio-temporal metadata of GDELT extracted events. GDELT offers different APIs to run analysis and a Google Big Query²⁴ access to the database. GDELT exposes examples of pre-configures analyses via the analysis service. For instance, the Event TimeMapper visualizes events matching a given search over time.

²³ [Wikimedia EventStream Terms Of Service](#).

²⁴ <https://cloud.google.com/bigquery/>.

```

1 PREFIX gdelt : <http://linkeddata.stream/resource/gdelt>
2 SELECT (COUNT(?newsa) AS ?tot)
3 FROM NAMED WINDOW <e> ON gdelt:events [RANGE 30MPT STEP 15MPT]
4 FROM NAMED WINDOW <m> ON gdelt:mentions [RANGE 30MPT STEP 15MPT]
5 FROM NAMED WINDOW <g> ON gdelt:gkg [RANGE 30MPT STEP 15MPT]
6 WHERE {
7   WINDOW <e> { ?event :quadClass cameo:4 ; :actionGeo_cc "IZ". }
8   WINDOW <m> { ?event :mentions ?newsa. }
9   WINDOW <g> { ?newsa :theme gcam:kill ; :location geo:iraq. } }
10 GROUP BY ?newsa

```

Listing 1.12. RSP-QL Crossing GDELT Streams. Prefixes omitted.

5 Discussion and Conclusion

In this paper, we walked through the life-cycle of Streaming Linked data discussing how FAIR principles for data management applies and showcasing three examples of the Web Streams, i.e., DBpedia Live Stream, Wikimedia EventStream, and GDELT, that we work as open-source at <https://w3id.org/webstreams>.

While Naming, Describing, and Serving can count to sufficient examples that ease the user experience, *Modelling* and *Data Conversion* still demand a huge amount of manual work. For the former most of the complexity lies in the data domain understanding. Making sense of domain data like GDELT requires very specific knowledge. Moreover, knowledge representation still hides some challenges when it concern streaming data. For latter, best practices are emerging from the community around R2RML/RML.

We consider an interesting research direction investigating how knowledge engineering changes when data analysis is bound to specific temporal constraints. To the best of our knowledge, current works focus on temporal data modeling for historical data management [16]. Moreover, we consider the following exciting future works: (i) extending and maintaining the catalog of streams, the released ontologies, and used resources; (ii) extending the Streaming Linked Data processing step of the life-cycle with dedicated guidelines, i.e., identifying canonical problems for streaming linked data processing. (iii) the introduction of performance assessment and benchmarking as an explicit part of the life-cycle, taking into account the ongoing work of the RSP community and LDBC [2].

Acknowledgments. Dr. Tommasini acknowledges support from the European Social Fund via IT Academy program.

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, 20 April 2009 (2009)

2. Angles, R., et al.: The LDBC social network benchmark. CoRR abs/2001.02299 (2020)
3. Arias-Fisteus, J., García, N.F., Fernández, L.S., Fuentes-Lorenzo, D.: Zstreamy: a middleware for publishing semantic streams on the web. *J. Web Semant.* **25**, (2014)
4. Balduini, M., Della Valle, E.: FrapPE: a vocabulary to represent heterogeneous spatio-temporal data to support visual analytics. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9367, pp. 321–328. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25010-6_21
5. Barbieri, D.F., Braga, D., Ceri, S., Della Valle, E., Grossniklaus, M.: Querying RDF streams with C-SPARQL. *SIGMOD Rec.* **39**(1), 20–26 (2010)
6. Barbieri, D.F., Della Valle, E.: A proposal for publishing data streams as linked data - a position paper. In: Proceedings of the WWW 2010 Workshop on Linked Data on the Web, LDOW 2010, Raleigh, USA, 27 April 2010 (2010)
7. Compton, M., et al.: The SSN ontology of the W3C semantic sensor network incubator group. *J. Web Sem.* **17**, 25–32 (2012)
8. Consortium, W.W.W., et al.: Best practices for publishing linked data (2014)
9. Della Valle, E., Balduini, M.: Listening to and visualising the pulse of our cities using social media and call data records. In: Abramowicz, W. (ed.) BIS 2015. LNBP, vol. 228, pp. 3–14. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26762-3_1
10. Della Valle, E., Dell'Aglio, D., Margara, A.: Taming velocity and variety simultaneously in big data with stream reasoning: tutorial. In: DEBS (2016)
11. Dell'Aglio, D., Della Valle, E., Calbimonte, J., Corcho, Ó.: RSP-QL semantics: a unifying query model to explain heterogeneity of RDF stream processing systems. *Int. J. Seman. Web Inf. Syst.* **10**(4), 17–44 (2014)
12. Dell'Aglio, D., Della Valle, E., van Harmelen, F., Bernstein, A.: Stream reasoning: a survey and outlook. *Data Sci.* **1**(1–2), 59–83 (2017)
13. Dimou, A., et al.: Mapping hierarchical sources into RDF using the RML mapping language. In: 2014 IEEE International Conference on Semantic Computing, Newport Beach, CA, USA, 16–18 June 2014, pp. 151–158 (2014)
14. Gao, F., Ali, M.I., Mileo, A.: Semantic discovery and integration of urban data streams. In: Proceedings of the Fifth Workshop on Semantics for Smarter Cities a Workshop at the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, 19 October 2014, pp. 15–30 (2014)
15. Gerner, D.J., Schrodt, P.A., Yilmaz, O., Abu-Jabr, R.: Conflict and mediation event observations (cameo): a new event data framework for the analysis of foreign policy interactions. International Studies Association, New Orleans (2002)
16. Gottschalk, S., Demidova, E.: Eventkg - the hub of event knowledge on the web - and biographical timeline generation. *Semantic Web* **10**(6), 1039–1070 (2019)
17. Hyland, B., Wood, D.: The joy of data-a cookbook for publishing linked government data on the web. In: Wood, D. (ed.) Linking Government Data, pp. 3–26. Springer, Heidelberg (2011). https://doi.org/10.1007/978-1-4614-1767-5_1
18. Luckham, D.: The power of events: an introduction to complex event processing in distributed enterprise systems. In: Bassiliades, N., Governatori, G., Paschke, A. (eds.) RuleML 2008. LNCS, vol. 5321, pp. 3–3. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88808-6_2
19. Margara, A., Urbani, J., van Harmelen, F., Bal, H.E.: Streaming the web: reasoning over dynamic data. *J. Web Sem.* **25**, 24–44 (2014)
20. Mauri, A., et al.: TripleWave: spreading RDF streams on the Web. In: Groth, P., et al. (eds.) ISWC 2016. LNCS, vol. 9982, pp. 140–149. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46547-0_15

21. Morsey, M., Lehmann, J., Auer, S., Stadler, C., Hellmann, S.: DBpedia and the live extraction of structured data from wikipedia. *Program* **46**(2), 157–181 (2012)
22. Passant, A., Bojärs, U., Breslin, J.G., Decker, S.: The SIOC Project: semantically-interlinked online communities, from humans to machines. In: Padget, J., et al. (eds.) COIN -2009. LNCS (LNAI), vol. 6069, pp. 179–194. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14962-7_12
23. Phuoc, D.L., Dao-Tran, M., Tuán, A.L., Duc, M.N., Hauswirth, M.: RDF stream processing with CQELS framework for real-time analysis. In: Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, DEBS 2015, Oslo, Norway, 29 June-3 July 2015, pp. 285–292 (2015)
24. Phuoc, D.L., Nguyen-Mau, H.Q., Parreira, J.X., Hauswirth, M.: A middleware framework for scalable management of linked streams. *J. Web Semant.* **16**, 42–51 (2012)
25. 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, Washington DC, USA 2009
26. Stonebraker, M., Çetintemel, U., Zdonik, S.B.: The 8 requirements of real-time stream processing. *SIGMOD Rec.* **34**(4), 42–47 (2005)
27. Tommasini, R., Della Valle, E.: Yasper 1.0: towards an RSP-QL engine. In: Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC) (2017)
28. Tommasini, R., et al.: VoCaLS: vocabulary and catalog of linked streams. In: Vrandečić, D., et al. (eds.) ISWC 2018. LNCS, vol. 11137, pp. 256–272. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00668-6_16
29. Villazón-Terrazas, B., Vilches-Blázquez, L.M., Corcho, O., Gómez-Pérez, A.: Methodological guidelines for publishing government linked data. In: Wood, D. (ed.) Linking Government Data, pp. 27–49. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-1767-5_2
30. Wilkinson, et al.: The fair guiding principles for scientific data management and stewardship. *Sci. Data* **3**(1), 160018 (2016). <https://doi.org/10.1038/sdata.2016.18>