Chapter 8
Velocity on the Web

Riccardo Tommasini

8.1 Introduction

The World Wide Web (Web) is a distributed system designed around a global naming system called URI [25]. Resources, the central abstraction, are not limited in their scope[1] and, although it is not mandatory, they are typically published along with a representation of their state. Using Hyper-Text Transfer Protocol (HTTP), Web applications, or more generically agents can access, exchange, and interact with resources' representations.

The decentralized nature of the Web makes it scalable but causes the spread of Data Variety [36]. Indeed, resources are heterogeneous and noisy. The Web of Data (WoD) is an extension of the Web that enables interoperability among Web applications encouraging data sharing. Semantic technologies like RDF, SPARQL, and OWL are the pillars of the WoD's technological stack.

From Smart Cities [27] to environmental monitoring [1], from Social Media analysis [31] to fake-news detection [32], a growing number of Web applications need to access and process data as soon as they arrive and before they are no longer valuable [16]. To this extent, the Web infrastructure is evolving,and new protocols are emerging, e.g., WebSockets and Server-Sent Events, and Application Public Interfaces (API) are becoming reactive, e.g., WebHooks.

In the big data context, the challenge mentioned above is known as Data Velocity [36], and Stream Processing (SP) is the research area that investigate how to handle it. SP solutions are designed to analyze *streams* and detect *events* [14] in real-time. However, SP technologies are inadequate to work on the Web. Data Velocity appears together with Data Variety and they lack flexible data models and expressive

[1] A *resource* is anything to which we can assign a Uniform Resource Identifier (URI).

R. Tommasini (✉)
University of Tartu, Narva Mnt 18, Tartu, Estonia
e-mail: riccardo.tommasini@ut.ee

A. Geraci (ed.), *Special Topics in Information Technology*,
PoliMI SpringerBriefs, https://doi.org/10.1007/978-3-030-62476-7_8

manipulation languages that are necessary to handle heterogeneous data. On the other hand, semantic technologies are not designed for continuous and reactive processing of streams and events. Therefore, *Web applications cannot tame Data Velocity and Variety at the same time* [16].

Data Velocity and Variety affect the entire data infrastructure. On the Web, it means technologies for the *identification*, *representation*, the *interaction* with resources [25]. Therefore, our investigation focus on the following research question.

Can we identify, represent, and interact with heterogeneous streams and events coming from a variety of Web sources?

The research question above implies that streams and events, the key abstractions of Stream Processing, become are valid Web resources. Nevertheless, the nature of streams and events, which as they are respectively *unbounded* and *ephemeral*, contrasts with the nature of Web resources, which are stateful. How this impact identification, representation, and processing is the focus of our investigation. Notably, the seminal work on Stream Reasoning and RDF Stream Processing has paved the road that goes in this direction [21].

To guide the study, we follow the Design Science (DS) research methodology, which studies how *artifacts*, i.e., software components, algorithms, or techniques, interact with a *problem context* that needs improvement. Such interaction, called *treatment*, is designed by researchers to solve a problem. The ultimate goal of DS is to design theories that allow exploring, describing, explaining, and predicting phenomena [51]. The validation of treatments is based on principle compliance, requirements satisfaction, and performance analysis.

Outline. Section 8.2 presents the state-of-the-art on Stream Reasoning and RDF Stream Processing. Section 8.3 formulates the research problems. Section 8.4 presents the major contributions of this research work. Finally, Sect. 8.5 concludes the chapter.

8.2 Background

According to Cugola et al. a data stream is an unbounded sequence of data items ordered by timestamp [14]. On the other hand, an event is an atomic (happened entirely or not at all) domain entity that describes something that happened at a certain time [29]. While streams are infinite and shall be analyzed continuously, events are sudden and ephemeral and shall be detected to react appropriately.

Previous attempts to handle Data Velocity on the Web belong to the Stream Reasoning (SR) [22] and RDF Stream Processing (RSP) [30] literature. Existing works discuss how to identify [7, 39] and represent stream and events [24, 35, 40], but most of the literature focuses on processing RDF streams [11, 19, 20, 30] and detect RDF events [4, 18]. Due to lack of space, we cannot list all the relevant works and we invite the interested readers to consult the recent surveys [21, 30].

Works on *identification* of streams and events directly refer to the notion of Web resource [25]. Barbieri and Della Valle [7] propose to identify streams and each element in the stream. In particular, they propose to use RDF named graphs for both the stream (sGraph) and its elements (iGraphs). The sGraph describes a finite sub-portion the stream Stream made of relevant iGraphs. Moreover, Sequeda and Corcho [39]'s proposal includes a set of URI schemas that incorporate spatio-temporal metadata. This mechanism is suitable to identify sensors and their observations but makes the URI not opaque.

Works on *representation* focus on RDF Streams, i.e., data streams whose data items are timestamped RDF graphs or triples. The Semantic Web community proposed ontological models for modelling such items from a historical point of view. However, neither of these focuses on the infinite nature of streams nor the ephemeral nature of events. Only Schema.org recently included a class that can identify streams as resources.[2] In the context of events, relevant vocabularies are the Linking Open Descriptions of Events (LODE) [40] and Simple Event Model [24]. Efforts on representing RDF streaming data include, but are not limited to FrAPPe [5], which uses pixels as an element for modelling spatio-temporal data, SSN [13] which models sensors observations, and SIOC [33] which include social media micro-posts.

Works on *interaction* divide into protocols for *access* and solutions for *processing* streams and events. The former successfully relies on HTTP extensions for continuous and reactive consumption of Web data, e.g., WebSockets and Server-Sent Events. The latter includes, but it is not limited to works on (i) *Analysis* of streams, i.e., filtering, joining, and aggregating; (ii) *Detection* of events, i.e., matching trigger conditions on the input flow and take action, and (iii) *Reasoning* over and about time, i.e., deducing implicit information from the input streams using inference algorithms that abstract and compose stream elements.

RSEP-QL is the most prominent work in the context of Web stream and event processing [18, 20]. Dell'Aglio et al. build on the state-of-the-art SPARQL extensions for stream analysis event detection. RDF Streams are the data model of choice. Moreover, a continuous query model allows expressing window operations and event patterns over RDF Streams.

Works on reasoning include ontology-based streaming data access (OBSDA) [11], and incremental reasoning [19]. These approaches aim at boosting reasoning performance to meet velocity requirements. Works on OBSDA use query-rewriting to process Web streaming data directly at the source while works on incremental reasoning focus on making reasoning scalable in the presence of frequent updates.

8.3 Problem Statement

According to Design Science, research problems divide into *Design Problems* and *Knowledge Questions*. The former are the problems to (re-)design an artifact so that it

[2] https://schema.org/DataFeed.

better contributes to achieving some goal. A solution to a design problem is a design, i.e., a decision about what to do that is generally not unique. The latter include explanatory or descriptive questions about the world. Knowledge questions do not call for a change in the world, and the answer is a unique falsifiable proposition.

From the related literature, it emerges that the *identification* of streams and events using URIs is possible [6, 34, 39]. Moreover, new protocols like WebSockets enable continuous data *access* on the Web. Therefore, our investigation focuses on how to *represent* and *process* streams and events on the Web.

The *representation problem* calls for *improving the Web of Data by enabling conceptual modeling and description of new kinds of resources, i.e., infinite ones like streams and ephemeral ones like events*.

A possible solution to solving the *representation problem* is an ontology, i.e., the specialization of a conceptualizations [23]. However, existing ontologies do not satisfy the requirements of Web users, when they are interested in representing infinite and ephemeral knowledge (cf Sect. 8.2). Existing ontologies were designed to model time-series or historical events that one can query by looking at the past. Moreover, these works neglect the peculiar natures of stream/event transformations, which are continuous, i.e., they last forever [26]. In practice, a shared vocabulary to describe streams and events on the Web is still missing.

The *processing problem* calls for *improving the Web of Data by enabling expressive yet efficient analysis of Web streams and detection of Web events*.

A possible solution to the *processing problem* is to combine semantic technologies and stream processing ones. However, existing solutions show high-performance but limited expressiveness or vice versa, they are very expressive but not efficient [3]. In practice, an expressive yet efficient Stream Reasoning approach that combines existing ones was never realized [42].

Query languages, algorithms, and architectures designed to address the processing problem are typically evaluated using benchmarks like CityBench [2]. However, recent works indicate the lack of a systematic and comparative approach to artifact validation [17, 37]. These limitations focus on two formal properties of the experimental results, i.e., repeatability and reproducibility. The former refers to variations on repeated measurements on the object of study under identical conditions. The latter refers to measurement variations on the object of study under changing experimental conditions.

Therefore, we formulate an additional *validation problem* that calls for *improving validation research by enabling a systematic comparative exploration of the solution space*. A possible solution to the *validation problem* is a methodology that guides researchers to design reproducible and repeatable experiments.

8.4 Major Results

In this section, we present the significant results of our research work. The various contributions are organized according to the identified problems.

Our primary contribution to solving the *representation problem* is the Vocabulary for Cataloging Linked Streams (VoCaLS) [50]. VoCaLS is an OWL 2 QL ontology that includes three modules: a *core* module that enables identifying streams as resources; *service description* that allows describing stream producers and consumers as well as catalogs, and *provenance* that allows auditing continuous transformations. Following the design science methodology, we inquired the community to collect our requirements [38]. Then, to verify VoCaLS compliance, we used the vocabulary in real-world scenarios. Moreover, we validated VoCaLS by expert opinion (peer review) and using Tom Gruber's ontology-design principles [23]. Listing 1 shows an example of a stream description and publication. It shows that VoCaLS allows (i) identifying the stream as a resource, e.g., an RDF Stream; (ii) it provides providing a static stream description including metadata like the license; it enables (iii) accessing the stream content via endpoints that decouple identification (which happens via HTTP) from consumption that uses more appropriate protocols, e.g., WebSockets.

```
PREFIX : <http://linkeddata.stream/resource/> .
<http://linkeddata.stream>  a vsd:PublishingService .
:descriptor  a vocals:streamDescriptor ;
             dcat:dataset :stream1 ;
             dcat:publisher <http://linkeddata.stream>;
             dcat:license <https://creativecommons.org/licenses
                 /by-nc/4.0/> ;
             dcat:description "stream of sensors observations".
:stream1 a vocals:RDFStream ;
    vocals:hasEndpoint [   a vocals:streamEndpoint ;
                           dcat:format frmt:JSON-LD ;
                           dcat:accessURL "ws://example.org/iot
                               /sensor". ] .
```

Listing 1 Example of RDF stream using VoCaLS.

Our primary contribution to solve the *processing problem* is the Expressive Layered Fire-hose (ELF) (formerly streaming MASSIF) [9]. ELF is a stream reasoning platform designed after a renovated Cascading Stream Reasoning (cf Fig. 8.1). ELF can identify the best trade-off between efficiency and expressiveness. To this extent, it organizes the stream reasoners in a layered network and orchestrates the processing to be inversely proportional to the input streams rate.

In the *Continuous Processing* layer (L1), data are in the form of structured streams. L1's operations are elementary and can sustain very high-rates (millions of items per minute). Examples of operations include filters, left-joins, and simple aggregations. In this level, data streams can be converted to foster data integration in the lay-

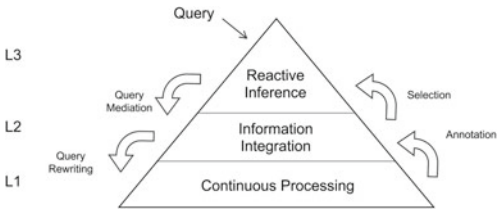

Fig. 8.1 A renovated Cascading Reasoning vision w.r.t. [41]

ers above. Possible implementations leverage on window-based stream processing languages for structured data like Streaming SQL [49].

The *Information Integration* layer (L2) aims at building a uniform view over the input streams using a conceptual model. In L2, data streams are typically semi-structured, e.g., RDF Stream. L2's operations are slightly more complex than L1's, e.g., pattern matching over graph streams, and the input rate is reduced to hundreds of thousands items per minute. Moreover, at this level, data streams can be *interpreted* using background domain knowledge. In these regards, our further contribution is C-SPRITE, i.e., an algorithm for efficient hierarchical reasoning over semantic streams [10]. C-SPRITE applies a hybrid reasoning technique that outperforms existing reasoners for Instance Retrieval, even when the number of sub-classes to check is more than a hundred. Possible implementations of this layer include RSP engines, which allows enriching and joining multiple streams, or approaches for Ontology-Based streaming data access [12].

The *Reactive Inference* layer (L3) calls for reactive operations that combines and compare high-level abstractions from various domain. In L3, streams are usually symbolic, e.g., event types, and operations can be very expressive because they deal with a reduced input rated (thousands of items per minute). Possible reasoning framework that are suitable for L3 are Description Logic (DL), temporal logical, and Answer Set Programming (ASP). Our further contribution concerning L3 is Ontology-Based Event RecognitiON (OBERON) (formerly OBEP) [9, 10, 43], i.e., (i) is an A Domain-Specific Language that treats events as first-class objects [43]. OBERON uses two forms of reasoning to detect and compose events over Web streams, i.e., Description Logics reasoning and Complex Event Recognition. Notably, machine learning techniques such as Bayesian networks or hidden Markov models are also suitable approaches for this layer.

The investigation related the *validation problem* develops in [46–48]. Validation research is comparative, and thus, it relies on the notion of experiment [28]. To guarantee repeatability and reproducibility, researchers must have full control over both the experimental environment and the object of study. Thus, our main research contribution to solve the validation problem are (i) a methodology for experiment design for RSP [42] and the architecture for an experimental environment based on the notion of Test-Stand [46], and (ii) a Web environment for experimentation called RSPLab, which guarantees reproducibility and repeatability of experimental results using containerization techniques in the context of RDF Stream Processing.

Moreover, in [44], we highlight the issues related to designing a query language based on RSP-QL formalization that treats streams as first-class objects and keeps the constructs minimal, homogeneous, symmetric and orthogonal [15]. The work evolved into a reference implementation for RSP-QL called YASPER [45] and a framework for rapid-prototyping.[3]

8.5 Conclusion

In this chapter, we summarize the work presented by the research work on representing and processing streams and events on the Web.

With the growing popularity of data catalogs like Google's Dataset Search [8], the research around VoCaLS is potentially impactful. Streams and events are novel kinds of Web resources that are relevant for a number of applications. VoCaLS is a first step towards modelling unbounded and/or ephemeral knowledge. Nevertheless, more work is left to be done in terms of knowledge representation and reasoning. To this extent, an updated version of VoCaLS, which includes better conceptualizations for Web streams and events, is in progress.

Moreover, with the spread of Knowledge Graphs (KG), efficient yet expressive reasoning techniques are relevant as never before. Indeed, KGs are vast and constantly evolving. Therefore, scalable and event-driven reasoning techniques look promising. In particular, the work on C-SPRITE hits a significant trade-off between expressiveness and efficiency. In these regards, pushing efficient reasoning further to incorporate more sophisticated language features, e.g., transitive property, is extremely appealing.

References

1. Alahakoon D, Yu X (2016) Smart electricity meter data intelligence for future energy systems: a survey. IEEE Trans Ind Inform 12(1):425–436
2. Ali MI, Gao F, Mileo A (2015) Citybench: a configurable benchmark to evaluate RSP engines using smart city datasets. In: The Semantic web - ISWC 2015 - 14th international semantic web conference, Bethlehem, PA, USA, 11–15 Oct 2015, proceedings, part II, pp 374–389. https://doi.org/10.1007/978-3-319-25010-6_25
3. Anicic D (2012) Event processing and stream reasoning with ETALIS. Ph.D. thesis, Karlsruhe Institute of Technology
4. Anicic D, Fodor P, Rudolph S, Stojanovic N (2011) EP-SPARQL: a unified language for event processing and stream reasoning. In: WWW. ACM, pp 635–644
5. Balduini M, Della Valle E (2015) Frappe: a vocabulary to represent heterogeneous spatio-temporal data to support visual analytics. In: International semantic web conference (2). Lecture notes in computer science, vol 9367. Springer, Berlin, pp 321–328
6. Barbieri DF, Braga D, Ceri S, Della Valle E, Grossniklaus M (2010) Incremental reasoning on streams and rich background knowledge. In: ESWC (1). Lecture notes in computer science, vol 6088. Springer, Berlin, pp 1–15

[3] https://github.com/riccardotommasini/yasper.

7. Barbieri DF, Della Valle E (2010) A proposal for publishing data streams as linked data - A position paper. In: LDOW. CEUR workshop proceedings, vol 628. CEUR-WS.org
8. Benjelloun O, Chen S, Noy NF (2020) Google dataset search by the numbers. https://arxiv.org/abs/2006.06894
9. Bonte P, Tommasini R, Della Valle E, Turck FD, Ongenae F (2018) Streaming MASSIF: cascading reasoning for efficient processing of iot data streams. Sensors 18(11)
10. Bonte P, Tommasini R, Turck FD, Ongenae F, Della Valle E (2019) C-sprite: efficient hierarchical reasoning for rapid RDF stream processing. In: DEBS. ACM, pp 103–114
11. Calbimonte J, Corcho Ó (2014) Evaluating SPARQL queries over linked data streams. Linked data management. Chapman and Hall/CRC, Boca Raton
12. Calbimonte J, Mora J, Corcho Ó (2016) Query rewriting in RDF stream processing. In: ESWC. Lecture notes in computer science, vol 9678. Springer, Berlin, pp 486–502
13. Compton M, Barnaghi PM, Bermudez L, Garcia-Castro R, Corcho Ó, Cox SJD, Graybeal J, Hauswirth M, Henson CA, Herzog A, Huang VA, Janowicz K, Kelsey WD, Phuoc DL, Lefort L, Leggieri M, Neuhaus H, Nikolov A, Page KR, Passant A, Sheth AP, Taylor K (2012) The SSN ontology of the W3C semantic sensor network incubator group. J Web Sem 17:25–32
14. Cugola G, Margara A (2010) TESLA: a formally defined event specification language. In: DEBS. ACM, pp 50–61
15. Date CJ (1984) Some principles of good language design (with especial reference to the design of database languages). SIGMOD Rec 14(3):1–7
16. Della Valle E, Ceri S, van Harmelen F, Fensel D (2009) It's a streaming world! reasoning upon rapidly changing information. IEEE Intell Syst 24(6):83–89
17. Dell'Aglio D, Calbimonte J, Balduini M, Corcho Ó, Della Valle, E (2013) On correctness in RDF stream processor benchmarking. In: International semantic web conference (2). Lecture notes in computer science, vol 8219. Springer, Berlin, pp 326–342
18. Dell'Aglio D, Dao-Tran M, Calbimonte J, Phuoc DL, Della Valle E (2016) A query model to capture event pattern matching in RDF stream processing query languages. In: EKAW. Lecture notes in computer science, vol 10024, pp 145–162
19. Dell'Aglio D, Della Valle E (2014) Incremental reasoning on RDF streams. Linked data management. Chapman and Hall/CRC, Boca Raton, pp 413–435
20. Dell'Aglio D, Della Valle E, Calbimonte J, Corcho Ó (2014) RSP-QL semantics: a unifying query model to explain heterogeneity of RDF stream processing systems. Int J Semantic Web Inf Syst 10(4)
21. Dell'Aglio D, Della Valle E, van Harmelen F, Bernstein A (2017) Stream reasoning: a survey and outlook. Data Sci 1(1–2):59–83
22. Germano S, Pham T, Mileo A (2015) Web stream reasoning in practice: on the expressivity vs. scalability tradeoff. In: RR. Lecture notes in computer science, vol 9209. Springer, Berlin, pp 105–112
23. Gruber TR (1995) Toward principles for the design of ontologies used for knowledge sharing? Int J Hum-Comput Stud 43(5–6):907–928
24. van Hage WR, Malaisé V, Segers R, Hollink L, Schreiber G (2011) Design and use of the simple event model (SEM). J Web Sem 9(2):128–136
25. Jacobs I, Walsh N (2004) Architecture of the world wide web, volume one. W3C recommendation, W3C. https://www.w3.org/TR/webarch/
26. Keskisärkkä R (2016) Representing RDF stream processing queries in RSP-SPIN. In: International semantic web conference (posters & demos). CEUR workshop proceedings, vol 1690. CEUR-WS.org
27. Kolozali S, Bermúdez-Edo M, Puschmann D, Ganz F, Barnaghi PM (2014) A knowledge-based approach for real-time iot data stream annotation and processing. In: 2014 IEEE international conference on internet of things, Taipei, Taiwan, 1–3 Sep 2014, pp 215–222
28. Kuehl RO (2000) Design of experiments stastistical principles of research design and analysis. No. Q182. K84 2000
29. Luckham D (2008) The power of events: an introduction to complex event processing in distributed enterprise systems. In: RuleML. Lecture notes in computer science, vol 5321. Springer, Berlin, p 3

30. Margara A, Urbani J, van Harmelen F, Bal HE (2014) Streaming the web: reasoning over dynamic data. J Web Sem 25:24–44
31. Mathioudakis M, Koudas N (2010) Twittermonitor: trend detection over the twitter stream. In: SIGMOD conference. ACM, pp 1155–1158
32. Nixon LJB, Fischl D, Scharl A (2019) Real-time story detection and video retrieval from social media streams. In: Mezaris V, Nixon LJB, Papadopoulos S, Teyssou D (eds.) Video verification in the fake news era. Springer, Berlin, pp 17–52. https://doi.org/10.1007/978-3-030-26752-0_2
33. Passant A, Bojars U, Breslin JG, Decker S (2009) The SIOC project: semantically-interlinked online communities, from humans to machines. In: COIN@AAMAS&IJCAI&MALLOW. Lecture notes in computer science, vol 6069. Springer, Berlin, pp 179–194
34. Pimentel V, Nickerson BG (2012) Communicating and displaying real-time data with websocket. IEEE Internet Comput 16(4):45–53
35. Rinne M, Blomqvist E, Keskisärkkä R, Nuutila E (2013) Event processing in RDF. In: WOP. CEUR workshop proceedings, vol 1188
36. Russom P, et al (2011) Big data analytics. TDWI best practices report, fourth quarter
37. Scharrenbach T, Urbani J, Margara A, Della Valle E, Bernstein A (2013) Seven commandments for benchmarking semantic flow processing systems. In: The semantic web: semantics and big data, 10th international conference, ESWC 2013, Montpellier, France, 26–30 May 2013. Proceedings, pp 305–319. https://doi.org/10.1007/978-3-642-38288-8_21
38. Sedira YA, Tommasini R, Della Valle E (2017) Towards vois: a vocabulary of interlinked streams. In: DeSemWeb. CEUR workshop proceedings, vol 1934. CEUR-WS.org
39. Sequeda JF, Corcho Ó (2009) Linked stream data: a position paper. In: SSN. CEUR workshop proceedings, vol 522, pp 148–157. CEUR-WS.org
40. Shaw R, Troncy R, Hardman L (2009) LODE: linking open descriptions of events. In: ASWC. Lecture notes in computer science, vol 5926. Springer, Berlin, pp 153–167
41. Stuckenschmidt H, Ceri S, Della Valle E, van Harmelen F (2010) Towards expressive stream reasoning. In: Semantic challenges in sensor networks. Dagstuhl seminar proceedings, vol 10042. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany
42. Tommasini R (2015) Efficient and expressive stream reasoning with object-oriented complex event processing. In: DC@ISWC. CEUR workshop proceedings, vol 1491. CEUR-WS.org
43. Tommasini R, Bonte P, Della Valle E, Ongenae F, Turck FD (2018) A query model for ontology-based event processing over RDF streams. In: EKAW. Lecture notes in computer science, vol 11313. Springer, Berlin
44. Tommasini R, Della Valle E (2017) Challenges & opportunities of RSP-QL implementations. In: WSP/WOMoCoE. CEUR workshop proceedings, vol 1936, pp 48–57. CEUR-WS.org
45. Tommasini R, Della Valle E (2017) 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)
46. Tommasini R, Della Valle E, Balduini M, Dell'Aglio D (2016) Heaven: a framework for systematic comparative research approach for RSP engines. In: ESWC. Lecture notes in computer science, vol 9678. Springer, Berlin, pp 250–265
47. Tommasini R, Della Valle E, Balduini M, Sakr S (2020) On teaching web stream processing - lessons learned. In: Bieliková M, Mikkonen T, Pautasso C (eds.) Web engineering - 20th international conference, ICWE 2020, Helsinki, Finland, 9–12 June 2020, proceedings. Lecture notes in computer science, vol 12128. Springer, Berlin, pp 485–493. https://doi.org/10.1007/978-3-030-50578-3_33
48. Tommasini R, Della Valle E, Mauri A, Brambilla M (2017) Rsplab: RDF stream processing benchmarking made easy. In: ISWC, pp 202–209
49. Tommasini R, Sakr S, Valle ED, Jafarpour H (2020) Declarative languages for big streaming data. In: Bonifati A, Zhou Y, Salles MAV, Böhm A, Olteanu D, Fletcher GHL, Khan A, Yang B (eds.) Proceedings of the 23nd international conference on extending database technology, EDBT 2020, Copenhagen, Denmark, March 30 - April 02, 2020. pp. 643–646. OpenProceedings.org. https://doi.org/10.5441/002/edbt.2020.84

50. Tommasini R, Sedira YA, Dell'Aglio D, Balduini M, Ali MI, Phuoc DL, Della Valle E, Calbimonte J (2018) Vocals: Vocabulary and catalog of linked streams. In: International semantic web conference (2). Lecture notes in computer science, vol 11137. Springer, Berlin, pp 256–272
51. Wieringa R (2014) Design science methodology for information systems and software engineering. Springer, Berlin