ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

From Business Intelligence to semantic data stream management

Marie-Aude Aufaure [a], Raja Chiky [b,*], Olivier Curé [c], Houda Khrouf [d], Gabriel Kepeklian [d]

[a] MAS Lab Ecole Centrale Paris, France
[b] ISEP - LISITE, Paris, France
[c] LIP6 (UMR 7606/CNRS), Université Pierre et Marie Curie (UPMC), Paris, France
[d] Atos Integration, Paris, France

HIGHLIGHTS

- Evolution of Business Intelligence with emergence of Big Data technologies.
- New technologies and approaches the 3Vs (Volume, Velocity and Variety) of Big data.
- Stream reasoning over Big Data.
- Summarizing data streams (semantic and classic data).
- Semantic data matching in stream context.

ARTICLE INFO

Article history:
Received 21 May 2015
Received in revised form
4 November 2015
Accepted 10 November 2015
Available online xxxx

Keywords:
Data stream
Linked Data
Business Intelligence
Stream reasoning

ABSTRACT

The Semantic Web technologies are being increasingly used for exploiting relations between data. In addition, new tendencies of real-time systems, such as social networks, sensors, cameras or weather information, are continuously generating data. This implies that data and links between them are becoming extremely vast. Such huge quantity of data needs to be analyzed, processed, as well as stored if necessary. In this position paper, we will introduce recent work on Real-Time Business Intelligence combined with semantic data stream management. We will present underlying approaches such as continuous queries, data summarization and matching, and stream reasoning.

1. Introduction

The main objective of Business Intelligence is to transform data into knowledge for a better decision-making process. The constant growth of data and information, coming from heterogeneous data sources has led to new ways of interaction and the integration of new models and tools to cope with this heterogeneity. We manipulate more and more unstructured data documents, emails, social networks, contacts that need to be integrated with classical structured data like CRM, data stored in relational databases. We also need more and more interactivity, flexibility, dynamicity and expect the system to be proactive and reactive. Users expect immediate feedback, and want to find information rather than merely look for it. Moreover, the company tends to be organized in a collaborative way, called enterprise 2.0 [1]. All these evolutions induce challenging research topics for Business Intelligence, such as providing efficient mechanisms for a unified access and model to both structured and unstructured data. Semantic technologies are a perfect fit for integrating and matching data. Business Intelligence integrates collaborative and social software, by combining BI with elements from both Web 2.0 and the Semantic Web. Extracting value from all these data, a crucial advantage for companies, requires business analytics. In order to synthesize information and derive insights from massive, dynamic, ambiguous data, the use of data visualization techniques and visual analytics becomes critical. Business Intelligence is also impacted by big data, and need to account for the volume of data sources as well as the need of response in real-time for extracting value from trusted data.

This position paper addresses the integration of real-time analytics with semantic technologies. Many research work has been done separately in these two fields, but, to the best of our

* Corresponding author.
E-mail addresses: marie-aude.aufaure@ecp.fr (M.-A. Aufaure), raja.chiky@isep.fr (R. Chiky), olivier.cure@lip6.fr (O. Curé), houda.khrouf@atos.net (H. Khrouf), gabriel.kepeklian@atos.net (G. Kepeklian).

http://dx.doi.org/10.1016/j.future.2015.11.015
0167-739X/

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015

knowledge, only a few ones provide an integrated view. This is mainly due to scalability issues for semantic reasoning.

The rest of this paper is organized as follows. Section 2 describes the new needs in Business Intelligence and presents a generic architecture for semantic data stream management platform. Section 3 focuses on related work in the area of semantic data streaming. Section 4 describes data matching in an RDF stream context. Section 5 gives an overview of reasoning in the context of RDF stream processing. Finally, Section 6 concludes this paper and gives an outlook upon future research for managing large-scale semantic streaming data.

2. From BI to semantic data stream management

Business Intelligence (BI) refers to a set of tools and methods dedicated to collecting, representing and analyzing data to support decision-making in enterprises. BI is defined as the ability of an organization to take all input data and convert them into knowledge, ultimately, providing the right information to the right people at the right time via the right channel. During the last two decades, numerous tools have been designed to make available a huge amount of corporate data for non-expert users. Business Intelligence is a mature technology, widely adapted, but faces new challenges for incorporating new data such as unstructured data or data coming from sensors or social networks into analytics. A key issue is the ability to analyze in real-time these constantly growing amounts of data, taking their meaning into account. The complexity of BI tools and their interface is a barrier for their adoption. Thus, personalized systems and user modeling [2] have emerged to help provide more relevant information and services to the user. Information visualization and dynamic interaction techniques are key for enhancing the user experience in using such tools [3].

Traditional BI systems offer tools for structuring and storing data in a data ware-house, in which data are modeled with a multidimensional model representing the analysis axis. Key performance indicators can be computed from this model and restituted to the user in a static dashboard.

These systems can be extended with semantic technologies to capture the meaning of data and new ways of interacting with data, intuitive and dynamic. Semantic technologies [4,5] focus on the meaning of data and are capable of dealing with both unstructured and structured data. Having the meaning of data and a reasoning mechanism may assist a user during his analysis task. The vision of the FP7 CUBIST[1] project was to extend the ETL process to both structured and unstructured data, to semantically store data in a triple store and to provide user-friendly visual analytics capabilities leading to dynamic dashboards. Then, the information provided to the user is not composed of only quantitative values like key performance indicators, but can also integrate qualitative values represented by a graph or a lattice extracted from formal concepts (a formal concept is a set of objects sharing properties; the formal concepts are then organized in a lattice linked together by a relation of inclusion). The user can then navigate into these semantic data through a visual analytic tool [6].

More recently, business intelligence has been impacted by big data and, in particular, need to take into account the velocity i.e. the ability to provide information or alerts in real-time from streams. With the exponential growth of sensor networks, web logs, social networks and interconnected application components, large collections of data are continuously generated with high speed. These data are called "data streams": there is no limit on the total volume of data and there is no control over the order in which data arrive. The analysis methods (data mining, machine learning) should self-adapt to these data and process them on the fly in one pass and in the order of their arrival. These heterogeneous data streams [7] are produced in real time and consequently, should be processed on the fly. Then, they are maintained, interpreted and aggregated in the purpose of reusing their semantics and recommending relevant alerts to the targeted stakeholders in order to react to interesting phenomena occurring in the input streams. A precious decision-making value can be enhanced through the semantic analysis of data streams, especially while crossing them with other information sources.

Coming back to semantic technologies, numerous techniques can be used to extract some meaning or knowledge from data sources. Among them, we can cite natural language processing techniques, data mining, machine learning and ontology engineering. These techniques are used to extract patterns or models, to structure data and to transform any information in actionable knowledge. Semantic Web technologies can be used for linking, publishing or searching for data on the web, but also for large-scale structuring and enriching data with the RDF semantic model.

Semantic-based approaches are useful to simplify the integration of heterogeneous data sources by the mean of ontologies and for offering a unified metadata layer. Semantics can also be used for discovering and enriching information, and finally, to provide a unified data access mechanism. Semantics addresses the variety from the 3 V of Big Data (Volume, Variety and Velocity) to generate value from heterogeneous data. The value of data also increases when they can be linked to other data (Linked Data). Semantic technologies can then be seen as a great opportunity to reduce the cost and complexity of data integration.

Fig. 1 represents a generic architecture of what could be a Real-Time BI platform in which structured and unstructured data streams are processed on the fly. In a Real-Time BI platform, multiple heterogeneous data sources can be connected, and data can be static or dynamic. The static data comes from standard databases or from open data, and does not change or in a minor way. Dynamic data comes as a stream, in a semantic format (RDF for example) or not (raw data). To process raw data, they need first to be converted into semantic format using ontologies. The idea is to maintain into the system an homogeneous format and a meaningful model that can be processed by machines. This architecture is composed of different components for processing streams: semantic filtering and continuous queries, data summarization, matching and reasoning. Semantic filtering is used for processing a large volume of streams on the fly. Existing systems are mainly based on RDF and SPARQL. To manage infinite real-time data stream, the platform has to provide the ability to create persistent continuous queries, which allow users to receive new results when they become available. Moreover, in the context of Big Data with a huge volume of data coming in high velocity, the platform provides some summarizing and load shedding techniques [8] that randomly drop data from the streams when the load of the platform increases beyond what it can handle. Data matching is used to enrich data with additional knowledge and to combine data streams coming from distributed data sources. Discovering relations between data is a key factor to add contextual information which may enhance decision making. This task is particularly challenging in a stream context where time-efficient techniques are needed to ensure scalability over high stream rates. Finally, reasoning is a key component in an RDF stream context and still considered as an open problem, mainly for reasoning using parallelized computation and expressive ontology language. These components are not fully integrated in existing architectures for enabling semantic stream processing. We introduce in the following sections a survey of research work related to semantic data stream management, summarizing techniques, data matching and reasoning.

1 CUBIST EU FP7 project: http://www.cubist-project.eu/index.php?id=378.

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015

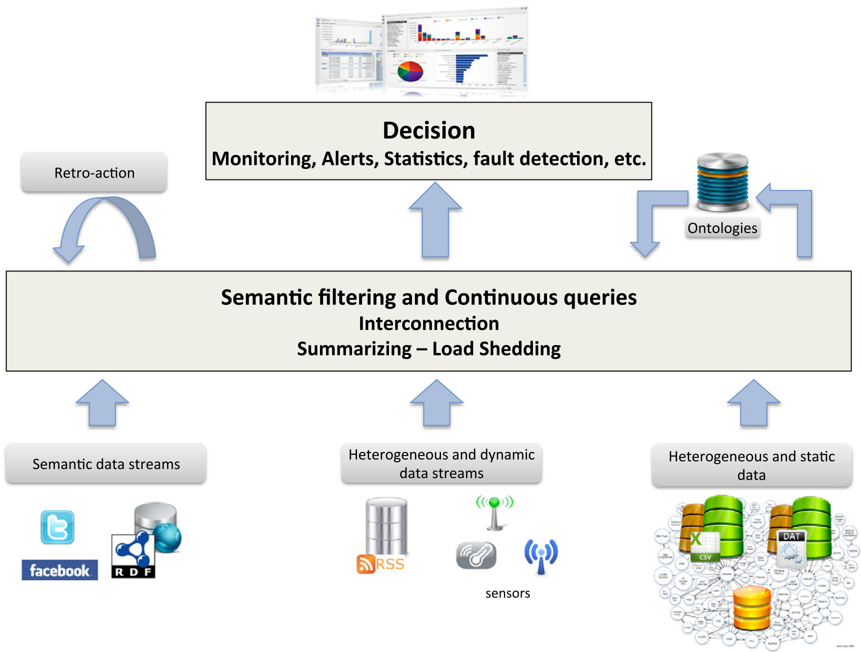

Fig. 1. A generic semantic real-time BI architecture.

3. Semantic data streaming

Massive data stream processing is a scientific challenge and an industrial concern. But with the current volumes of data streams, their number and variety, current techniques are not able to meet the requirements of applications. The Semantic Web tools, through the RDF for example, address the problem of heterogeneous data. Thus, the data stream are converted to semantic data stream by using RDF triples extended with a timestamp. To be able to query, filter, or reason on semantic data streams, the query language SPARQL must be extended to include concepts such as windowing, based on previous work in Data Stream Management Systems DSMS.

Data Stream Management Systems (DSMS) [9] are designed to perform continuous queries over data stream. Data elements arrive on-line and stay only for a limited time period in memory. In a DSMS, continuous queries evaluate continuously and incrementally arriving data elements. DSMS use windowing techniques to handle some operations like aggregation as only an excerpt of a stream (window) is of interest at any given time. A window may be physically defined in terms of a time interval (for instance the last week), or logically defined in terms of the number of tuples (for example the last 20 elements).

Several DSMS prototypes have been developed. Some of them are specialized in a particular domain (sensor monitoring, web application, etc.), some others are for general use (such as STREAM [10] and TelegraphCQ [11]).

The problem of "too much (streaming) data but not enough (tools to gain and derive) knowledge" was tackled by [12]. They envisioned a Semantic Sensor Web (SSW), in which sensor data are annotated with semantic metadata to increase interoperability and provide contextual information essential for situational knowledge. CQELS [13], SPARKWAVE [14], C-SPARQL [15] etc. are existing technologies to exploit these semantic and streaming (continuous and infinite) data, and are based on recommended standard RDF, as the format of representation. Their design and specification are based on DSMS's features.

CQELS [13] is a native approach in an RDF environment based on 'white-boxes'. It provides its own processing model and its own operators to deal with streams, for example, window operators or query semantic operators. C-SPARQL [15] on the other hand, uses a 'black-box' approach which delegates the processing to other engines such as stream/event processing engines and SPARQL query processors by translating to their provided languages.

Although almost all the engines are based on the SPARQL Language, there are only a few systems which are able to process big quantity of data on the fly. Moreover, these engines do not feature any tool that would allow them to reduce the processing efforts and improve the processing time. For many applications, we must obtain compact summaries of the stream. These summaries could allow accurate answering of queries with estimates, which approximate the true answers over the original stream [16].

3.1. Data summarization

In many fields, we are faced with the ever growing problem of how to manage and analyze large dynamic datasets. Database and data mining researchers often use synopsis (i.e. summaries) with great effect to scale up performance on these datasets with a small cost to accuracy. Perhaps the most basic synopsis of a data stream is a sample of elements from the stream. A key benefit of such a sample is its flexibility: other synopses can be built from a sample itself. The rest of this section summarizes the state of the art for data stream algorithms. We will focus primarily on the problems of creating sample structures for a single data stream, in addition, we will also present techniques used in a distributed environment. Most of these summary structures have been considered for traditional databases [17]. The challenge is to adapt some of these techniques to the data stream model.

Data stream sampling Sampling data streams is based on traditional sampling techniques, but also requires significant new innovations, especially to deal with the problem of infinite length streams. Windowing techniques are used to handle the unlimited nature of data: only an excerpt of a stream (window) is of interest at any given time. A window may be *physical*, defined in terms of a time interval (e.g., the last week), or *logical*, defined in terms of the number of tuples (e.g., the last 20 elements). These windows can be fixed with "fixed endpoints", or sliding with "moving endpoints" over time or tuples.

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015

The traditional online algorithm "Reservoir Sampling" was proposed by Vitter in 1985 [18] and is widely used to sample data streams. It produces a sample of fixed size and does not require prior knowledge of data stream length. Reservoir sampling is useful for insertions or updates but not for deletions in the case of a sliding window. The difficulty arises because elements must be removed from the sample as they expire, so that maintaining a sample of a specified size is nontrivial. Several algorithms for handling logical and temporal windows have been developed.

A simple approach was proposed in [19]. The algorithm maintains a reservoir sample for the first window of the data stream. When an element expires, it is replaced with the newly arrived element. This algorithm maintains a uniform random sample for the first window and requires little memory to store the sample, but has the disadvantage of being highly periodic. To handle this, another technique was proposed in [19]. Each new arrival is added to a "Backing sample" with a fixed probability and the sample is generated by down sampling the backing sample. As elements expire, they are removed from the backing sample.

Many other algorithms were developed to be applied to logical windows such as "chain sampling" [19], to temporal windows such as "priority sampling" or for particular use such as "concise sampling" [20]. To the best for our knowledge, all of these techniques sample the data stream individually. Moreover, these techniques exploit neither possibilities of computation in sensors, nor bidirectional communication between the sensors and the central server.

Distributed data stream sampling There are many applications where data is continuously produced by a large number of distributed sensors. Adaptive sampling has been developed for these applications to manage limited resources. They aim at conserving network bandwidth and storage memory by filtering out data that may not be relevant in the current context. The data collection rate becomes dynamic and adaptable to the environment.

Most existing adaptive sampling techniques sample data from each source (*temporal sampling*). An adaptive sampling scheme which adjusts data collection rates in response to the contents of the stream was proposed in [21]. A Kalman filter is used at each sensor to make predictions of future values based on those already seen. The sampling interval *SI* is adjusted based on the prediction error. If the needed sampling interval for a sensor exceeds that is allowed by a specified Sampling Interval range, a new *SI* is requested to the server. The central server delivers new *SI*s according to available bandwidth, network congestion and streaming source priority.

In [22], authors present a feedback control mechanism which makes the frequency of measurements in each sensor dynamic and adaptable. Sampled data are compared against a model representing the environment. An error value is calculated on the basis of the comparison. If the error value is more than a predefined threshold, then a sensor node collects data at a higher sampling rate; otherwise, the sampling rate is decreased. Sensor nodes are completely autonomous in adapting their sampling rate.

In [23], authors present a method to prevent sensor nodes to send redundant information; this is predicted by a sink node using an ARIMA prediction model. Energy efficiency is achieved by suppressing the transmission of some samples, whose ARIMA based prediction values are within a predefined tolerance value with respect to their actual values. A similar approach is proposed by Cormode and Garofalakis [24]. Their results show that reduced communication between sensors and the central server can be sufficient by using an appropriate prediction model. A wide range of queries (including heavy hitters, wavelets and multi-dimensional histograms) can be answered by the central server using approximate sketches.

On the other hand, [25] uses a *spatial sampling* technique called backcasting approach. Backcasting operates by first activating only a small subset of sensor nodes which communicate their information to a fusion center. This provides an estimate of the environment being sensed, indicating some sensors may not need to be activated to achieve a desired level of accuracy. The fusion center then backcasts information based on the estimate to the network and selectively activates additional sensors to obtain a target error level. In this approach, adaptive sampling can save energy by only activating a fraction of the available sensors.

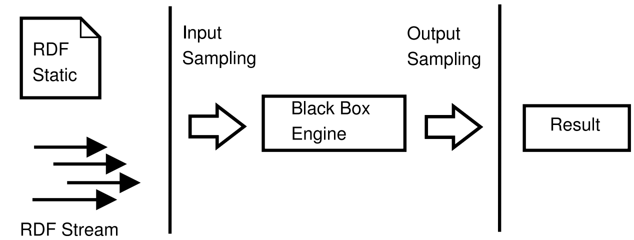

Fig. 2. Adding a sampling operator to semantic data stream management engine.

3.2. Summarizing semantic data streams

The growing generated data from web applications is becoming a problem for the processing systems, and the relation between data is causing troubles when attempting to exploit data repositories. Therefore, [26] proposed an extension of a real-time request system that allows to reduce processing tasks and memory space requirements. Authors propose the implementation of sampling operators that could be used in conjunction with existing semantic data streams engines considered as a Black Box engine. The sampling methods that have been implemented are: Uniform Random Sampling, Reservoir Sampling and Chain Sampling. Authors propose to extend existing semantic data stream querying engines by creating an external abstraction of the sampling operator as shown in Fig. 2.

However, this approach as implemented in [26] is only interesting when applied to semantic data streams including only independent RDF triples. The use of such approach is less successful when it comes to treat the data stream of a higher semantic level where RDF triples are linked to form RDF graphs. Indeed, this approach will lead to the destruction of semantic links constituting the structure of these graphs, thus reducing the level of semantics and affecting the data consistency.

Recently, two particular works take advantage of the fact that semantic data streams are constituted of a small set of RDF schema and have a very regular RDF graph structure according to a graph-based data model. In the first work, authors propose RDSZ [27] (RDF Differential Stream compressor based on Zlib [28]), an algorithm for lossless RDF stream compression. The approach combines a differential encoding mechanism with the general purpose stream compressor Zlib.

The second approach proposes an efficient interchange format for RDF streams ERI (Efficient RDF Interchange Format) [29]. This work is adapted from the encoding mechanism of the Efficient XML Interchange (EXI) format [30] (Efficient XML Interchange format). Its principle comes from the fact that the described entities in an RDF stream often follow a common schema. ERI multiplexes the information into structural (schema) and value (concrete data) channels. Then Zlib compressor is used in each channel, resulting in high compression ratios and effective processing time.

4. Data matching in an RDF stream context

RDF Streams are increasingly becoming available from various, distributed and autonomous sources. They cover different domains such as environment, energy and transportation. Combining

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015

and enriching these multiple streams is the vision of Semantic Sensor Web which aims to increase interoperability and to derive additional knowledge for reasoning enhancement. Indeed, semantic enrichment of sensor data is advantageous to provide contextual information which can increase reasoning capabilities to solve problems such as network management and event detection. Context has been particularly proven to be essential for accurate and robust anomaly detection in sensor data [31]. For example, what appears to be anomalous during a day or a year in an electrical sensor network may be found to be false positives when introducing context such as time of day (e.g. night) or specific events (e.g. holidays). Providing such context requires a system which is able to enrich streaming data by discovering semantic relations between RDF resources. This is known as data matching or link discovery task which has recently gained a crucial importance in Semantic Web. The main challenge is how to combine data residing at different sources, and resolve references at the instance level such as identity link using `owl:sameAs` or any other semantic relatedness between two real-world objects. This is a key factor to directly enhance two important dimensions of data quality, which are accuracy and completeness [32]. Accuracy is the extent to which data are correct, reliable and free of error. Using multiple representations from different sources has the advantage to detect conflicts and inaccurate data. Completeness is the extent to which data are of sufficient breadth, depth, and scope for the task at hand. Multiple object representations usually cover different properties, thus leading to more complete description.

Recently, several research studies have led to a plethora of matching tools based on manual configuration to define similarity function or on fully automated process. Most of these tools typically use offline matching given that the corpus of data is complete. However, there is very little research related to online and real-time matching that needs to link data as they appear and not as a batch. A variety of DSMS have been developed to manipulate, store and query multiple streams of data. However, none of them has tackled the problem of data matching in dynamic environment. In order to add context, a DSMS should be able to deal with high stream rates and perform semantic matching with high scalability. Such matching could be based on some time-efficient approaches proposed to deal with scalability for large datasets. This is ensured by pruning the search space using a candidate selection scheme in which the aim is to minimize the number of unnecessary similarity computations. One popular method, used in various tools [33,34], is filtering by utilizing inverted index structures. It reduces the search space by quickly excluding all pairs that do not share a common token. This optimization typically focuses on string similarity distances and on a specific discriminative property. Another advanced method proposes to form blocks of entities sharing an identical or approximate key. Many blocking-based techniques exist such as standard blocking (using a predefined blocking key), sorted neighborhood, and adaptive blocking [35]. Silk [36] and LIMES [37] are popular link discovery tools which make use of blocking techniques. Silk introduced a novel method called MultiBlock using a multidimensional index which subdivides the space into overlapping blocks. While standard blocking techniques block in one dimension, MultiBlock blocks by multiple properties using multiple dimensions. More precisely, it builds an index for each similarity measure in the matching function to preserve the distances of the entities. Then, it uses a compound multidimensional index to aggregate all indexes, and generate a comparison pair for each two entities which share an index. MultiBlock guarantees that similar entities share at least one block, thus reducing the runtime without sacrificing recall. It has the advantage to work on streaming data as it does not require to pre-process the whole dataset. LIMES proposed another lossless time-efficient algorithm called HYPPO (HYpersphere aPPrOximation algorithm) and dedicated for numeric values in metrics spaces. It utilizes the triangle inequality to compute pessimistic approximations of distances which are then used to filter out a large amount of obvious non-matches before executing comparisons. The triangle inequality has been also exploited in different tasks to improve the runtime, for example to optimize the performance of continuous queries on high dimensional streaming time series [38]. Overall, the reduction of the search space has been shown to enhance scalability by several orders of magnitude. Still, the sheer amount of links to be discovered can induce an impractical runtime. Thus, parallel processing have been recently investigated by some tools which involve advanced infrastructures such as MapReduce-based clusters or graphics processing units (GPU). For example, Silk, Zhishi.links [39] and LIMES-MR [40] support already the distributed computing with MapReduce thus optimizing the scalability. The performance of these frameworks is, however, limited by the input–output overhead. Thus, the use of parallel processing on massively parallel graphic processors was another solution that has been explored in [40].

While most of these methods are tailored to offline processing, many applications such as query answering systems or real-time credit monitoring still require an efficient and online approach to link data at runtime. In such a context, besides to scalability concern, the system should be able to take into account the dynamic nature of data and allow the matching model to evolve over the time. Currently, there is very little research related to link discovery that could be applied on highly dynamic streaming data. The work in [41] proposed to fetch the candidate solutions for each incoming resource using SPARQL filtering based on a supervised blocking technique. However, the supervised model in such environment would quickly become inaccurate as there is no guarantee that training data would still be representative of the space. Thus, an incremental approach is required to handle the data variations without processing the entire corpus of data. For example, the method described in [42] supports an online matching based on doubling hierarchical clustering with two stages: update stage that assigns entities to clusters, merge stage that combines clusters to prevent them from exceeding a fixed limit. Each new entity is then compared against other resources in the selected candidate clusters. This approach seems promising, but the author did not present any evaluation. Overall, there is a need for a more comprehensive support of real-time data matching in DSMS such as parallel processing, time-efficient algorithms, and incremental models to deal with scalability as well as the dynamics of data over the time.

5. Reasoning in an RDF stream context

This section emphasizes the importance of reasoning in the context of RDF stream processing. To fully grasp the potential of this feature, we first need to clarify the notion of reasoning in the context of the Semantic Web and Knowledge Bases in general. A Knowledge Base consists of a set of facts and some rules which are specifying the vocabulary used by the facts. For instance, consider the facts stating that `Camille teaches course1` and `Camille is pregnant` together with an ontology defining that `only a professor teaches a course` and `only a female can be pregnant`. Then a reasoner, i.e., a software dedicated to derive information and knowledge, can infer that `Camille is a female professor`.

In the Semantic Web, the graph-based RDF data format is used for the definition of both the facts and the vocabularies, also known as ontologies, corresponding to RDF Schema and OWL W3C languages. The ontologies are precisely the components that are supporting the derivation of implicit information and knowledge from explicit ones. This ability to derive inferences is an important differentiator between Knowledge Base systems and traditional Relational Database Management systems or NoSQL stores, graph-based ones included [43]. As a consequence, the reasoning feature is not one of the requirements presented in [44] for standard database stream processing. In this section, we argue that the full potential of RDF stream processing lies in this reasoning capacity, we point to some potential solutions for the design of such systems and we present first results toward this direction.

Reasoning in the Semantic Web has a long history that takes its roots in Artificial Intelligence and logic. Inferences drawn from ontologies support operations such as concept classification, qualification of the consistency of a knowledge base, retrieval of the instances of a given concept, finding the most specific concept an individual is an instance of and instance checking to name the main ones (read [45] for a detailed presentation of these reasoning services). They are generally implemented using automata theory, a translation to predicate logic, resolution-based methods or the semantic tableau approach. The latter two have been the most widely adopted among existing systems, e.g., Fact++, Pellet, HermiT, RacerPro for expressive ontologies, i.e., OWL2DL, and ELK, CEL, Quonto, Jena for lightweight ontologies, i.e., OWL2 profiles, namely EL, RL, QL, and RDFS.

These systems cannot be used out of the box in a stream context due to the near real-time nature that forces to obtain reasoning results within a given time period. In this situation, a particular attention is given to the ontology expressiveness/computational complexity trade-off which should ensure that the relevant inferences can be performed in a defined time-window constraint. This is the main reason for the adoption of RDFS reasoning, the least expressive ontology language of the W3C Semantic Web stack, in available systems. Nevertheless, some systems have considered an extension of RDFS, e.g., by introducing property transitivity, in RDF stream processing. Other ontology languages of the W3C stack, namely OWL2DL and its profiles, may be considered too computationally expensive in a streaming context and to the best of our knowledge have not implemented in existing systems.

Different types of RDF stream reasoning have been identified: data-driven, query-driven and hybrid solutions.

In the former, the information contained in RDF streams is extended at load-time, possibly using some external ontologies, by generating possible inferences. The main limitations of this approach, frequently denoted as materialization, is the latency implied at data loading time due to the computation of all deductions and the difficulty to maintain a valid dataset in the face of data updates, e.g., removing a single information can induce the deletion of an important number facts some of which can be derived by some other valid data. The main advantage of materialization is the performance of query processing due to the absence of any overhead. The Virtuoso RDF database management system is adopting this reasoning approach in a non-streaming context. We believe that this data-driven approach has some potential in a streaming context if the system is able to limit the number of materialization to a minimal and relevant set of triples. Hence ensuring that the materialization process can be executed within the defined time-window. In the case where an external, static knowledge base is used in the materialization process, a nice property would be to constrain the number of possible updates. The first requisite implies an analysis of both the continuous queries and the ontologies involved in stream processing. Since the continuous queries can be considered relatively static, i.e., once defined, they are being used for a certain time, this approach is realistic. The latter requisite may depend on the use case associated to the streaming application, i.e., update frequency of external knowledge bases. We both know of knowledge bases that are being updated several times per day, for instance in the bioinformatics domain, and of other domains where modifications are only occurring on a weekly or monthly basis.

This approach can be opposed to query-driven which rewrites, a.k.a., reformulates, the (continuous) queries in order to retrieve all valid answers. With this solution, the advantages consist of fast loading times and easier handling of data updates. The main drawback corresponds to slow query answering due to the overhead of reasoning over the data, knowledge pair which is required for all queries. Moreover, the rewriting can possibly generate a large number of queries, some of which can be semantically equivalent and syntactically different, resulting in a high rate of duplicate answers. GraphDB (formerly OWLIM) is an RDF store using this query-driven solution in a non-streaming context. If the continuous queries executed in a streaming context are relatively static, it may be worth considering this query-driven approach. In that situation, a set of queries would be reformulated prior to the stream processing and without any hard time constraint. A second step would analyze these rewritten queries to prevent query redundancy and unsatisfiable queries. This second step is clearly is more involved since ideally, these properties would be ensured at compile-time and not at run-time.

The latter solution is a combination of the previous two approaches, i.e., a method where only a portion of the data would be materialized and a part of the query would be reformulated. Blazegraph (formerly bigdata) is an example of a hybrid system in a non-streaming context. This approach may have a big impact on reasoning stream processing. It would require the definition of a cost-based approach for the definition of a boundary between materialization and rewriting. Such a solution would involve a detailed analysis of the query steaming workload and of the knowledge bases. Due to the complexity of the problem, a efficient approach would be based on the definition of heuristics.

In a non-streaming reasoning process, the order in which the information and knowledge is processed is not important. This is not the case in a streaming context where the order of stream tuples may imply a certain interpretation and implying the derivation of different facts and knowledge. We can distinguish between several order notions. Natural order is the most frequent one and generally corresponds to the order in which stream tuples arrive in the system. In the case of RDF data, a timestamp value can be added to the subject, predicate, object RDF triple, thus forming a quad. Some other forms of order, requiring a more or less involved computation, are also possible, e.g., popularity, frequency. To be the best of our knowledge, none of the current standard Semantic Web reasoners are adapted to handling ordered triples.

In [46], the authors present a taxonomy of implemented systems and open research problems in RDF stream reasoning. They argue that most implemented systems belong to the data-driven category with a natural order. IMaRS [47] and EP-SPARQL [48] are such systems which respectively belong to the DSMS and CEP categories. It is recognized that the query-driven approach is promising due to the static nature of queries in a streaming context. That is, the reformulation of the query would be performed once and its optimized version could potentially be executed an infinite number of times. The hybrid, mixing data and query driven approaches, presents the most important research challenges but also the highest potential in terms of task adequacy and efficiency.

A third dimension for RDF stream reasoning corresponds to the parallel computation of the inference services. In a non-streaming context, systems such as WebPie [49] and SAOR [50] have been implemented using the MapReduce [51]. The batch processing nature of MapReduce is not adapted to handle streams and hence, these systems cannot be used in a streaming context. Parallel systems dedicated to stream processing, e.g., Storm, Spark Streaming, Apache Flink, Samza or S4, have rarely been used by

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015

any RDF streaming processing system equipped with inference services. [52] is in fact, the only tentative that we know of in this direction. The system performs RDF streaming distributed on the S4 system. The paper describing the system describes an experimentation where RDFS reasoning is being performed. It does not seem that the software is being maintained or extended for more expressive ontologies.

Thus, RDF stream reasoning can be considered as an open problem in need for parallelized computation, expressive ontology language support and handling possibly complex order. The main challenge associated to this problem is to combine knowledge base and query analysis together with reasoning in parallel computation setting. This implies to define novel systems leveraging from Semantic Web and distributed streaming approaches.

6. Conclusion

The interconnection of massive data streams is a scientific challenge and a concrete industrial concern. But with the current volumes of data streams, their velocity and variety, current techniques are not able to meet the requirements of real applications. Yet we believe that this problem can be answered by taking advantage of recent advances in the techniques of querying, summarizing, matching and reasoning on semantic data streams. These techniques are part of the new generation of Real Time Business Intelligence.

The Semantic Web tools, through RDF for example, address the problem of heterogeneous data. Thus, data streams are converted to semantic data streams by using RDF triples extended with a timestamp. To be able to query, filter, or reason on semantic data streams, the SPARQL query language must be extended to include concepts such as continuous queries. Several research prototypes for semantic filtering have been presented recently. However, to the best of our knowledge, none of these works has been concerned about overloading when the semantic data stream management system is not able to handle an overwhelming incoming data. Load shedding techniques and summarization techniques exist in the field of DSMS as we had presented in this paper. The challenge is to adapt these techniques to semantic data streams by losing the least possible links between data. In addition, data streams are large, heterogeneous in nature and incrementally processed. Such complexities require a powerful online technique to handle link discovery and thus data enrichment in real time. However, most of research studies have mainly addressed the time-efficient linking to scale to very large datasets. Still, none of them has explored the incremental techniques to handle the dynamics of data over the time.

References

[1] Juan Trujillo, Alejandro Maté, Business intelligence 2.0: A general overview, in: Marie-Aude Aufaure, Esteban Zimányi (Eds.), Business Intelligence, in: Lecture Notes in Business Information Processing, vol. 96, Springer, Berlin, Heidelberg, 2012, pp. 98–116.
[2] Alfred Kobsa, Generic user modeling systems, in: Peter Brusilovsky, Alfred Kobsa, Wolfgang Nejdl (Eds.), The Adaptive Web, in: Lecture Notes in Computer Science, vol. 4321, Springer, 2007.
[3] Micheline Elias, Marie-Aude Aufaure, Anastasia Bezerianos, Storytelling in visual analytics tools for business intelligence, in: INTERACT 2013—14th IFIP TC13 Conference on Human–Computer Interaction, in: Lecture Notes in Computer Science, vol. 8119, Springer, Cape Town, South Africa, 2013, pp. 280–297.
[4] Tim Berners-Lee, James Hendler, Ora Lassila, The semantic web, Sci. Am. 284 (5) (2001) 34–43.
[5] Pascal Hitzler, Markus Krtzsch, Sebastian Rudolph, Foundations of Semantic Web Technologies, first ed., Chapman & Hall/CRC, 2009.
[6] Cássio A. Melo, Alexander Mikheev, Bénédicte Le Grand, Marie-Aude Aufaure, Cubix: A visual analytics tool for conceptual and semantic data, in: Jilles Vreeken, Charles Ling, Mohammed Javeed Zaki, Arno Siebes, Jeffrey Xu Yu, Bart Goethals, Geoffrey I. Webb, Xindong Wu (Eds.), ICDM Workshops, IEEE Computer Society, 2012, pp. 894–897.
[7] Charu Aggarwal (Ed.), Data Streams—Models and Algorithms, Springer, 2007.
[8] Nesime Tatbul, Uğur Çetintemel, Stan Zdonik, Mitch Cherniack, Michael Stonebraker, Load shedding in a data stream manager, in: Proceedings of the 29th International Conference on Very Large Data Bases—Volume 29, VLDB'03, VLDB Endowment, 2003, pp. 309–320.
[9] Lukasz Golab, M. Tamer Özsu, Issues in data stream management, SIGMOD Rec. 32 (2) (2003) 5–14.
[10] Arvind Arasu, Brian Babcock, Shivnath Babu, Mayur Datar, Keith Ito, Rajeev Motwani, Itaru Nishizawa, Utkarsh Srivastava, Dilys Thomas, Rohit Varma, Jennifer Widom, Stream: The stanford stream data manager, IEEE Data Eng. Bull. 26 (1) (2003) 19–26.
[11] Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Sailesh Krishnamurthy, Samuel R. Madden, Fred Reiss, Mehul A. Shah, Telegraphcq: Continuous dataflow processing, in: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, SIGMOD'03, ACM, New York, NY, USA, 2003, p. 668.
[12] Amit Sheth, Cory Henson, Satya S. Sahoo, Semantic sensor web, IEEE Internet Comput. 12 (4) (2008) 78–83.
[13] Danh Le-Phuoc, Minh Dao-Tran, Josiane Xavier Parreira, Manfred Hauswirth, A native and adaptive approach for unified processing of linked streams and linked data, in: Proceedings of the 10th International Conference on the Semantic Web—Volume Part I, PISWC'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 370–388.
[14] Srdjan Komazec, Davide Cerri, Dieter Fensel, Sparkwave: continuous schema-enhanced pattern matching over RDF data streams, in: Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems, DEBS'12, ACM, New York, NY, USA, 2012, pp. 58–68.
[15] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, Emanuele Della Valle, Michael Grossniklaus, C-sparql: Sparql for continuous querying, in: Proceedings of the 18th International Conference on World Wide Web, ACM, 2009, pp. 1061–1062.
[16] Edith Cohen, Graham Cormode, Nick Duffield, Structure-aware sampling on data streams, in: Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems, ACM, 2011, pp. 197–208.
[17] Paul G. Brown, Peter J. Haas, Techniques for warehousing of sample data, in: Ling Liu, Andreas Reuter, Kyu-Young Whang, Jianjun Zhang (Eds.), ICDE, IEEE Computer Society, 2006, p. 6.
[18] Jeffrey S. Vitter, Random sampling with a reservoir, ACM Trans. Math. Software 11 (1) (1985) 37–57.
[19] Brian Babcock, Mayur Datar, Rajeev Motwani, Sampling from a moving window over streaming data, in: Proceedings of the Thirteenth Annual ACM–SIAM Symposium on Discrete Algorithms, SODA'02, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002, pp. 633–634.
[20] Phillip B. Gibbons, Yossi Matias, New sampling-based summary statistics for improving approximate query answers, in: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, SIGMOD'98, ACM, New York, NY, USA, 1998, pp. 331–342.
[21] Ankur Jain, Edward Y. Chang, Adaptive sampling for sensor networks, in: Proceeedings of the 1st International Workshop on Data Management for Sensor Networks: In Conjunction with VLDB 2004, DMSN'04, ACM, New York, NY, USA, 2004, pp. 10–16.
[22] A.D. Marbini, L.E. Sacks, Adaptive sampling mechanisms in sensor networks. 2003.
[23] Chong Liu, Kui Wu, Min Tsao, Energy efficient information collection with the arima model in wireless sensor networks, in: GLOBECOM, IEEE, 2005, p. 5.
[24] Graham Cormode, Minos N. Garofalakis, Approximate continuous querying over distributed streams, ACM Trans. Database Syst. 33 (2) (2008).
[25] Rebecca Willett, Aline Martin, Robert Nowak, Backcasting: Adaptive sampling for sensor networks, in: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, IPSN'04, ACM, New York, NY, USA, 2004, pp. 124–133.
[26] Naman Jain, Manuel Pozo, Raja Chiky, Zakia Kazi-Aoul, Sampling semantic data stream: Resolving overload and limited storage issues, in: DaEng, 2013, pp. 41–48.
[27] Norberto Fernández, Jesús Arias, Luis Sánchez, Damaris Fuentes-Lorenzo, Óscar Corcho, RDSZ: An approach for lossless RDF stream compression, in: The Semantic Web: Trends and Challenges, Springer, 2014, pp. 52–67.
[28] Peter Deutsch, Jean-Loup Gailly, Zlib compressed data format specification version 3.3. Technical report, 1996.
[29] Javier D. Fernández, Alejandro Llaves, Oscar Corcho, Efficient RDF interchange (eri) format for RDF data streams, in: The Semantic Web–ISWC 2014, Springer, 2014, pp. 244–259.
[30] John Schneider, Takuki Kamiya, D. Peintner, R. Kyusakov, Efficient xml interchange (exi) format 1.0. W3C Proposed Recommendation, 20, 2011.
[31] Michael Hayes, Miriam A.M. Capretz, Contextual anomaly detection framework for big sensor data, J. Big Data 2 (1) (2015).
[32] Felix Naumann, Melanie Herschel, An Introduction to Duplicate Detection, Morgan and Claypool Publishers, 2010.
[33] Khai Nguyen, Ryutaro Ichise, Bac Le, SLINT: a schema-independent linked data interlinking system, in: 7th International Workshop on Ontology Matching, Boston, USA, 2012.
[34] Juanzi Li, Zhichun Wang, Xiao Zhang, Jie Tang, Large scale instance matching via multiple indexes and candidate selection, J. Knowl.-Based Syst. 50 (2013) 112–120.
[35] Rohan Baxter, Peter Christen, Tim Churches, A comparison of fast blocking methods for record linkage, in: ACM SIGKDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation, 2003.

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015

[36] Robert Isele, Anja Jentzsch, Christian Bizer, Efficient multidimensional blocking for link discovery without losing recall, in: 14th International Workshop on the Web and Databases, Athens, Greece, 2011.
[37] Axel-Cyrille Ngonga Ngomo, Lars Kolb, Norman Heino, Michael Hartung, Sören Auer, Erhard Rahm, When to reach for the cloud: Using parallel hardware for link discovery, in: 10th Extended Semantic Web Conference, ESWC, Montpellier, France, 2013.
[38] Zhengrong Yao, Like Gao, Xiaoyang Sean Wang, Using triangle inequality to efficiently process continuous queries on high-dimensional streaming time series, in: 15th International Conference on Scientific and Statistical Database Management, Cambridge, MA, USA, 2003.
[39] Xing Niu, Shu Rong, Yunlong Zhang, Haofen Wang, Zhishi.links results for OAEI 2011, in: 6th International Workshop on Ontology Matching, Bonn, Germany, 2011.
[40] Stanley Hillner, Axel-Cyrille Ngonga Ngomo, Parallelizing LIMES for large-scale link discovery, in: 7th International Conference on Semantic Systems, Graz, Austria, 2011.
[41] Houda Khrouf, Vuk Milicic, Raphaël Troncy, Mining events connections on the social web: Real-time instance matching and data analysis in eventmedia, J. Web Sem. 24 (2014) 3–10.
[42] Jennifer Sleeman, Online unsupervised coreference resolution for semi-structured heterogeneous data, in: Doctoral Consortium, 11th International Semantic Web Conference, Boston, USA, 2012.
[43] Olivier Curé, Blin Guillaume (Eds.), RDF Database Systems: Triples Storage and SPARQL Query Processing, first ed., Morgan Kaufmann, Boston, MA, USA, 2015.
[44] Michael Stonebraker, Ugur Çetintemel, Stanley B. Zdonik, The 8 requirements of real-time stream processing, SIGMOD Rec. 34 (4) (2005) 42–47.
[45] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, Peter F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, New York, NY, USA, 2003.
[46] Emanuele Della Valle, Stefan Schlobach, Markus Krötzsch, Alessandro Bozzon, Stefano Ceri, Ian Horrocks, Order matters! harnessing a world of orderings for reasoning over massive data, Semant. Web 4 (2) (2013) 219–231.
[47] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, Emanuele Della Valle, Michael Grossniklaus, Incremental reasoning on streams and rich background knowledge, in: The Semantic Web: Research and Applications, 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30–June 3, 2010, Proceedings, Part I, 2010, pp. 1–15.
[48] Darko Anicic, Paul Fodor, Sebastian Rudolph, Nenad Stojanovic, EP-SPARQL: a unified language for event processing and stream reasoning, in: Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28–April 1, 2011, 2011, pp. 635–644.
[49] Jacopo Urbani, Spyros Kotoulas, Jason Maassen, Frank van Harmelen, Henri E. Bal, Owl reasoning with webpie: Calculating the closure of 100 billion triples, in: ESWC (1), 2010, pp. 213–227.
[50] Aidan Hogan, Jeff Z. Pan, Axel Polleres, Stefan Decker, Saor: Template rule optimisations for distributed reasoning over 1 billion linked data triples, in: International Semantic Web Conference (1), 2010, pp. 337–353.
[51] Jeffrey Dean, Sanjay Ghemawat, Mapreduce: Simplified data processing on large clusters, in: OSDI, 2004, pp. 137–150.
[52] Jesper Hoeksema, Spyros Kotoulas, High-performance distributed stream reasoning using s4, in: Ordring Workshop at ISWC, 2011.

Marie-Aude Aufaure, who obtained a Ph.D. in Computer Science from the University of Paris 6, joined CentraleSupelec in 2003. She was head of the academic chair in Business Intelligence funded by SAP from 2008 to 2014, and member of the laboratory of Mathematics Applied to Systems. Before joining CentraleSupelec, she worked in industry and in academia, in particular at INRIA where she was associated partner from 2001 to 2014. Her domains of competencies cover digital technologies, Big Data, Business Intelligence, Semantic Technologies, Data Mining and Machine Learning. She is reviewer for many scientific journals and conferences, and is regularly invited for keynote talks. She was involved in many European and National collaborative projects as scientific manager, and has supervised fifteen Ph.D. students.

Marie-Aude Aufaure is in charge of the big data professional training at CentraleSupelec. She also acts as an independent expert for the European Commission and is scientific advisor for innovative companies involved in digital technologies.

Raja Chiky is currently Associate Professor at ISEP where she is head of the RDI team (Research and development in Information Technology) and responsible for database and data mining courses. She holds a Ph.D. in Computer Science from Telecom ParisTech obtained after a Master degree in data mining and an engineering degree in Computer Science. Before joining ISEP, she taught statistics, databases and language programming at the University of Paris Dauphine, University Paris 12 and Telecom ParisTech. She worked closely with EDF R&D on research projects related to data stream mining. Her research interests include statistics, data mining, data warehousing, data stream management, recommender systems, and cloud computing.

Olivier Curé is an assistant professor in Computer Science at the Université Paris-Est in France. He obtained his Ph.D. in artificial intelligence at the Université de Paris V, France. His research focuses on the relationships between databases and knowledge bases, reasoning in the context of the Semantic Web, data quality and ontology mediation and big data. He is part of the Models and Algorithms research team at LIGM CNRS lab (Université Paris-Est Marne la Vallée).

He has published 1 book, 4 book chapters, 8 journal papers and more than 50 papers in international, peer-reviewed conferences on databases, semantic web and ontologies.

Houda Khrouf is a post-doctoral researcher in Computer Science at ATOS SE. She received her M.Sc. from ESIEE Engineering in 2008 and her Ph.D. from Telecom ParisTeh in 2014. Her main research topics are knowledge representation, big data, recommender systems and data mining. She is currently involved in a French National project focusing on semantic stream processing and distributed systems. She was part of the team winning the Semantic Web Challenge at ISWC 2012 and the team winning the IESD Challenge at EKAW 2012.

Gabriel Kepeklian as research engineer at Thomson (1981–1988) in the field of computer languages and formal grammars – INSA 1981 (National Institute of Applied Sciences of Lyon) – has developed compilers and created supercomputer control languages. He is now R&D Director at Atos Integration France and develops projects focused on the technologies of web of data and linked data. Gabriel Kepeklian is currently Chairman of the DataLift association, which promote the development of web data, research, innovation and any activity to promote its uses as its technology aims. Its publications, courses and conferences have the same aims.

Please cite this article in press as: M.-A. Aufaure, et al., From Business Intelligence to semantic data stream management, Future Generation Computer Systems (2015), http://dx.doi.org/10.1016/j.future.2015.11.015