Scalable Semantic Access to Siemens Static and Streaming Distributed Data*

E. Kharlamov[1] S. Brandt[2] M. Giese[3] E. Jiménez-Ruiz[1] Y. Kotidis[4] S. Lamparter[2]
T. Mailis[5] C. Neuenstadt[6] Ö. Özçep[6] C. Pinkel[7] A. Soylu[8] C. Svingos[4]
D. Zheleznyakov[1] I. Horrocks[1] Y. Ioannidis[4] R. Möller[6] A. Waaler[3]

[1]Uni. Oxford [2]Siemens [3]Uni. Oslo [4]AUEB [5]Uni. Athens [6]Uni. Lübeck [7]fluidOps [8]NTNU

Abstract. Numerous analytical tasks in industry rely on data integration solutions since they require data from multiple static and streaming data sources. In the context of the Optique project we have investigated how Semantic Technologies can enhance data integration and thus facilitate further data analysis. We introduced the notion Ontology-Based Stream-Static Data Integration and developed the system Optique to put our ideas in practice. In this demo we will show how Optique can help in diagnostics of power generating turbines in Siemens Energy. For this purpose we prepared anonymised streaming and static data from 950 Siemens power generating turbines with more than 100,000 sensors and deployed Optique on distributed environments with 128 nodes. The demo attendees will be able to see do diagnostics of turbines by registering and monitoring continuous queries that combine streaming and static data; to test scalability of our devoted stream management system that is able to process up to 1024 concurrent complex diagnostic queries with a 10 TB/day throughput; and to deploy Optique over Siemens demo data using our devoted interactive system to create abstraction semantic layers over data sources.

1 Introduction

Motivation. Siemens runs service centres dedicated to diagnostics of thousands of power-generation appliances across the globe. One typical task for these centres is to detect in real-time potential failure events caused by, e.g., an abnormal temperature and pressure increase. Such tasks require simultaneous processing of *(i)* sequences of digitally encoded coherent signals produced and transmitted from thousands of gas and steam turbines, generators, and compressors installed in power plants, and *(ii)* static data that include the structure of relevant equipment, history of its exploitation and repairs, and even weather conditions. These data are scattered across a large number of heterogeneous data streams in addition to static DBs with hundreds of TBs of data.

Even for a single diagnostic task, such as checking if a given turbine might develop a fault, Siemens engineers have to analyse streams with temperature and other measurements from up to $2,000$ sensors installed in different parts of the turbine, analyse historical temperature data, compute temperature patterns, compare them to patterns in other turbines, compare weather conditions, etc. This requires to pose a collection of hundreds of queries, the majority of which are semantically the same (they ask about temperature), but syntactically different (they are over different schemata). Formulating and executing so many queries, and then assembling the computed answers, takes up to 80% of the overall diagnostic time [10].

* This demo accompanies our ISWC'16 paper [14] and extends [8, 11]. This research was funded by the EU project Optique (FP7-IP-318338) and the EPSRC grants DBonto, MaSI^3, and ED^3.

Our Proposal. In order to streamline the diagnostic process at Siemens, we propose a data integration approach based on Semantic Technologies that extends a well-known Ontology Based Data Access and that we call *Ontology-Based Stream-Static Data Integration* (*OBSSDI*). It follows the classical data integration paradigm that requires the creation of a common 'global' schema that consolidates 'local' schemata of the integrated data sources, and mappings that define how the local and global schemata are related. In *OBSSDI* the global schema is an *ontology*: a formal conceptualisation of the domain of interest that consists of a *vocabulary*, i.e., names of classes, attributes and binary relations, and *axioms* over the terms from the vocabulary that, e.g., assign attributes of classes, define relationships between classes, compose classes, class hierarchies, etc. *OBSSDI* mappings relate each ontological term to a set of queries over the underlying data. For example, the generic ontology attribute *temperature-of-sensor* is mapped to all specific data and procedures that return temperature readings from sensors in dozens of different turbines and DBs storing historical data, thus, all particularities and varieties of how the temperature of a sensor can be measured, represented and stored are captured in these mappings. In *OBSSDI* the integrated data can be accessed by posing queries over the ontology, i.e., *ontological queries*. These queries are *hybrid*: they refer to both streaming and static data. Evaluation of an ontological query in *OBSSDI* has three stages: *(i)* in the *enrichment* stage ontology axioms are used to expand the ontological query in order to access as much of relevant data as possible; *(ii)* in the *unfolding* stage the mappings are used to translate the enriched ontological query into (possibly many) queries over the data; and *(iii)* in the *execution* stage the unfolded data queries are executed over the data. *OBSSDI* differs from traditional OBDA since the latter assumes that data is in (static) relational DBs, e.g [3, 18], or streaming, e.g., [2, 4], but not of both kinds. Moreover, we are different from existing solutions for unified processing of streaming and static semantic data e.g. [17], since they assume that data is natively in RDF while we assume that the data is relational and mapped to RDF.

Contributions. We developed an *OBSSDI* system OPTIQUE with several novel parts:

- BOOTOX: semi-automatic support to construct high quality ontologies and mappings over relational and streaming data.
- STARQL: query language over ontologies that combines streaming and static data, and allows for efficient enrichment and unfolding that preserves the semantics of ontological queries.
- STREAMVQS: end-user oriented query formulation support to construct continuos ontological queries.
- EXASTREAM: backend for optimising large numbers of queries automatically generated via enrichment and unfolding, and efficiently execute them over distributed streaming and static data.

BOOTOX [7] is practically important since it can dramatically speed up deployment and maintenance of *OBSSDI* systems. STARQL [16] is crucial since, to the best of our knowledge, no dedicated query language for hybrid semantic queries has the required properties. STREAMVQS (that extends OptiqueVQS [19]) is essential since it allows for fast and easy data access for non-experts to state-of-the-art technologies. EXASTREAM [15] is vital since even in the context where the data is only static and not distributed, query execution without dedicated optimisation techniques performs poorly since the queries that are automatically computed after enrichment and unfolding can be very inefficient, e.g., they may contain many redundant joins and unions.

2 System Overview

OPTIQUE is an integrated system that consist of multiple components to support *OBSSDI* end-to-end [6, 9, 12, 13]. For IT specialists OPTIQUE offers support for the whole lifecycle

of ontologies and mappings: semi-automatic bootstrapping from relational data sources, importing of existing ontologies, semi-automatic quality verification and optimisation, cataloging, manual definition and editing of mappings. For end-users OPTIQUE offers tools for query formulation support, query cataloging, answer monitoring, as well as integration with GIS systems. Query evaluation is done via OPTIQUE's query enrichment, unfolding, and execution backends EXASTREAM that allow to execute up to thousands complex ontological queries in highly distributed environments.

We now give an overview of EXASTREAM, our component for scalable streaming and static relational data processing that is in the focus of this demonstration. Relational queries produced by an unfolding component of Optique are handled by EXASTREAM, our high-throughput distributed *Data Stream Management System* (*DSMS*). The EXASTREAM *DSMS* is embedded in EXAREME, a system for elastic large-scale dataflow processing in the cloud [15, 20]. In the following, we present some key aspects of EXASTREAM.

EXASTREAM is built as a streaming extension of the SQLite *DBMS*, taking advantage of existing Database Management technologies and optimisations such as *query planners*. It provides a declarative language, namely SQL$^{\oplus}$, for querying data streams and relations that conform to the CQL semantics [1]. EXASTREAM natively supports *User Defined Functions* (*UDFs*) with arbitrary user code. The engine blends the execution of *UDFs* together with relational operators using JIT tracing compilation techniques speeding up the execution time. *UDFs* allow to express very complex dataflows using simple primitives. For OPTIQUE we used *UDFs* to implement communication with external sources, window partitioning on data streams, data mining algorithms such as the *Locality-Sensitive Hashing* technique [5] for computing the correlation between values of multiple streams. More importantly, the main operators that incorporate the algorithmic logic for transforming SQLite into a *DSMS* are implemented as *UDFs*.

In order to enable efficient processing of data streams of very high velocity we have implemented a number of optimisations in the stream processing engine, such as *adaptive indexing*. With this technique EXASTREAM collects statistics during query execution and, adaptively, decides to build main-memory indexes on batches of cached stream tuples in order to expedite query processing.

3 Demonstration Scenarios

For the demonstration purpose we selected 20 diagnostic tasks typical for Siemens service centres and expressed these tasks in STARQL and STREAMVQS. Then, we prepared a demo data set of streaming and static data from 950 gas and steam turbines in the time from 2002 to 2011. This data is anonymised in a way that preserves the patterns needed for demo diagnostic tasks. During the demo we will 'play' the streaming data and thus emulate real time streams. Then, we distributed the demo-data in several installations with different number of nodes (VMs) ranging from 1 to 128, where each node has 2 processors and 4GB of main memory. To demonstrate diagnostics results we prepared a dedicated monitoring dashboard for each diagnostic task in the catalog. Dashboards show diagnostics results in real time, as well as statistics on streaming answers, relevant turbines, and other information that is typically required by the service engineers at Siemens. Finally, we deployed OPTIQUE over the Siemens data by bootstrapping ontologies and mappings with BOOTOX and then manually post-processing and extending them so that they reach the required quality and contain necessary terms and mappings to cover 20 Siemens diagnostic tasks.

During the demo OPTIQUE will be available in three scenarios:

[S1] *Diagnostics with user's deployment*: the attendees will be able to deploy OPTIQUE over the Siemens data by bootstrapping ontologies and mappings, saving them, and

observing and possibly improving them in dedicated editors. Then, they will query their deployed instance with diagnostic tasks either from the Siemens catalog or their own, i.e., they will be able to formulate such tasks in STREAMVQS as parametrised continuous queries and register concrete instances of these tasks over data streams.

[S2] *Diagnostics with our deployment*: The attendees will be able to query our preconfigured (high quality) Siemens deployment using diagnostic tasks either from the Siemens catalog and their own.

[S3] *Performance showcase of our deployment*: the attendees will be able to run various tests over our deployment using one of 128 preconfigured Siemens distributed environments and one of 10 test sets of queries. While running the tests they will monitor the throughput and progress of parallel query execution processes.

4 References

[1] A. Arasu, S. Babu, and J. Widom. The CQL Continuous Query Language: Semantic Foundations and Query Execution. In: *VLDBJ* (2006).

[2] J. Calbimonte, Ó. Corcho, and A. J. G. Gray. Enabling Ontology-Based Access to Streaming Data Sources. In: *ISWC*. 2010.

[3] C. Civili, M. Console, G. De Giacomo, D. Lembo, et al. MASTRO STUDIO: Managing Ontology-Based Data Access applications. In: *PVLDB* 6.12 (2013).

[4] L. Fischer, T. Scharrenbach, and A. Bernstein. Scalable Linked Data Stream Processing via Network-Aware Workload Scheduling. In: *SSWKBS@ISWC*. 2013.

[5] N. Giatrakos, Y. Kotidis, A. Deligiannakis, V. Vassalos, and Y. Theodoridis. In-network approximate computation of outliers with quality guarantees. In: *Inf. Systems* 38.8 (2013).

[6] I. Horrocks, T. Hubauer, E. Jiménez-Ruiz, E. Kharlamov, et al. Addressing Streaming and Historical Data in OBDA Systems: Optique's Approach. In: *KNOW@LOD*. 2013.

[7] E. Jiménez-Ruiz, E. Kharlamov, D. Zheleznyakov, I. Horrocks, C. Pinkel, M. G. Skjaeveland, E. Thorstensen, and J. Mora. BootOX: Practical Mapping of RDBs to OWL 2. In: *ISWC*. 2015.

[8] E. Kharlamov, S. Brandt, E. Jimenez-Ruiz, Y. Kotidis, et al. Ontology-Based Integration of Streaming and Static Relational Data with Optique. In: *SIGMOD* (2016).

[9] E. Kharlamov, D. Hovland, E. Jiménez-Ruiz, D. L. C. Pinkel, et al. Enabling Ontology Based Access at an Oil and Gas Company Statoil. In: *ISWC*. 2015.

[10] E. Kharlamov, N. Solomakhina, Ö. L. Özçep, D. Zheleznyakov, et al. How Semantic Technologies Can Enhance Data Access at Siemens Energy. In: *ISWC*. 2014.

[11] E. Kharlamov, S. Brandt, M. Giese, E. Jiménez-Ruiz, et al. Enabling semantic access to static and streaming distributed data with optique: demo. In: *DEBS*. 2016.

[12] E. Kharlamov, E. Jiménez-Ruiz, C. Pinkel, M. Rezk, et al. Optique: Ontology-Based Data Access Platform. In: *ISWC Posters & Demos*. 2015.

[13] E. Kharlamov, E. Jiménez-Ruiz, D. Zheleznyakov, D. Bilidas, et al. Optique: Towards OBDA Systems for Industry. In: *ESWC (Selected Papers)*. 2013.

[14] E. Kharlamov, Y. Kotidis, M. Theofilos, C. Neuenstadt, et al. Towards Analytics Aware Ontology Based Access to Static and Streaming Data. In: *ISWC*. 2016.

[15] H. Kllapi, P. Sakkos, A. Delis, D. Gunopulos, and Y. E. Ioannidis. Elastic Processing of Analytical Query Workloads on IaaS Clouds. In: *CoRR* abs/1501.01070 (2015).

[16] Ö. L. Özçep, R. Möller, and C. Neuenstadt. Stream-Query Compilation with Ontologies. In: *AI 2015*. 2015.

[17] D. L. Phuoc, M. Dao-Tran, J. X. Parreira, and M. Hauswirth. A Native and Adaptive Approach for Unified Processing of Linked Streams and Linked Data. In: *ISWC*. 2011.

[18] M. Rodriguez-Muro, R. Kontchakov, and M. Zakharyaschev. Ontology-Based Data Access: Ontop of Databases. In: *ISWC*. 2013.

[19] A. Soylu, E. Kharlamov, D. Zheleznyakov, E. Jiménez-Ruiz, M. Giese, and I. Horrocks. Ontology-Based Visual Query Formulation: An Industry Experience. In: *ISVC*. 2015.

[20] M. M. Tsangaris, G. Kakaletris, H. Kllapi, G. Papanikos, et al. Dataflow Processing and Optimization on Grid and Cloud Infrastructures. In: *IEEE Data Eng. Bull.* 32.1 (2009).