Semantic Data Stream Mapping and Shape Constraint Validation Based on Collaboratively Created Annotations

Matthias T. Frank(✉) and Viliam Simko

FZI Research Center for Information Technology, Information Process Engineering, Haid-und-Neu-Str. 10-14, 76131 Karlsruhe, Germany
{frank,simko}@fzi.de

Abstract. Due to the Internet of Things, more and more data streams like environmental or traffic observations become publicly available over Web APIs. However, most of these APIs are still provided with ambiguous key-value pairs which do not contain any explicit semantics and are therefore hard to be processed without further data understanding and data transformation. We address this issue by mapping these data streams to a stream with explicit semantics based on collaboratively created annotations. Moreover, we enable shape constraint validation for observation messages which are evaluated on-the-fly. These shape constraints allow for generic pre-processing of data streams without further coding. To evaluate our approach, we process non-semantic data streams of public and private environmental observation stations, map them using explicit semantics from a semantic wiki platform and validate each message based on shape constraints from the same wiki platform. Any change on the metadata in the wiki immediately affects the resulting message stream without the need of adopting the code.

1 Introduction

More and more data streams like environmental or traffic observations become publicly available over Web application programming interfaces (APIs). Examples are public observation stations for traffic noise[1] or air pollution[2], but also private observation stations like senseBoxes[3] or other weather stations which post their observations in a machine readable format on the Internet. Preferential, these observations are also available over Web APIs. However, most of these APIs are still provided with ambiguous key-value pairs which do not contain any explicit semantics and are therefore hard to be processed without further data understanding and transformation. However, without explicit semantics, these values cannot be evaluated on-the-fly. A developer has to read and understand the human readable documentation of the data structure and build a tailored

[1] http://www4.lubw.baden-wuerttemberg.de/servlet/is/224275/.
[2] http://mnz.lubw.baden-wuerttemberg.de/messwerte/aktuell/statDEBW080.htm.
[3] https://sensebox.de/en/.

© Springer International Publishing AG, part of Springer Nature 2018
T. Mikkonen et al. (Eds.): ICWE 2018, LNCS 10845, pp. 321–329, 2018.
https://doi.org/10.1007/978-3-319-91662-0_26

solution that fits the implicit semantics in order to process the messages correctly. In some cases, where the documentation is unclear, incomplete, deprecated or even not present, a developer for a consuming application has to consult the data provider to fully understand and correctly implement the semantics of the data. Also, further transformation of the data could be required to interpret the values correctly. Whenever the data provider changes syntax or semantics of the data stream, the developer has to adopt the code of the consuming application manually. This challenge has also be identified by Wiener et al. [7] when integrating heterogeneous spatio-temporal data.

In this paper, we address the issue of lacking semantics in observation data streams by mapping data streams of public and private observation stations on-the-fly to explicit semantics for each observation. The semantic information is derived from collaboratively created semantic annotations of non-semantic data streams within a semantic wiki platform, based on the foundation we laid in [3]. In addition, we enable shape constraints for API messages which are validated during run time. These shape constraints allow for generic pre-processing of data streams without further coding.

In Sect. 2, we discuss related work in the field of Resource Description Framework (RDF) mappings of heterogeneous data, RDF stream processing and RDF shape constraints. Based on the related work, we introduce our approach for semantic annotations for data streams of observation data, semantic mapping of non-semantic data streams and constraint checking based on user created annotations in Sect. 3. The implementation of our approach is illustrated in Sect. 4. In Sect. 5, we show how our implementation performs on exemplary data streams of a public observation station and a private observation station. Finally, the results are discussed in Sect. 6.

2 Related Work

Mapping to explicit semantics: Dimou et al. [2] have outlined limitations of current mapping approaches like the need of manual alignment on a per-source, per-format or even per-case basis. The authors claim that incorporating data from multiple sources and different formats to Linked Open Data (LOD) remains complicated, although a significant number of tools exist for that purpose and introduce the RDF Mapping Language (RML) as a generic language for integrated RDF mappings of heterogeneous data. In contrast to their work, we aim on processing continuous streams of observation messages, rather than mapping gradually shaped data sets. Although our primary aim is not to incorporate the data of observation stations to LOD, but to enable consuming applications to correctly process observation data of heterogeneous data sources based on explicit semantics, we benefit from adopting the basic principle of uniform and interoperable mapping definitions.

Semantic stream processing: Several ways on how to process data streams are possible, especially when explicit semantic annotations have to be evaluated.

As an example, the stream processing could be implemented from scratch using any suitable programming language. This would enable the highest flexibility and minimize the complexity of the infrastructure needed. However, building the whole stream processing framework from scratch for every project would cause a lot of unnecessary workload which would make the development process inefficient. For this reason, distributed systems like Apache Spark[4] or Apache Flink[5] focus specifically on stream processing. Spark relies on micro batching which adds latency at the value of the batch interval, whereas Flink is designed as a real time stream processing engine. However, this difference is only relevant if observation messages have to be processed with high frequency in a latency critical setting, which is not the case in our scenario. For processing of RDF streams, Tommasini et al. [6] have introduced RSPLab, a cloud-ready and open-source framework for designing and executing tests that can be used to compare different implementations.

Semantic shape constraints: Rules, reasoning and constraint checking on RDF data are supported by Web Ontology Language (OWL)[6], Semantic Web Rule Language (SWRL)[7] or SPARQL Inferencing Notation (SPIN)[8]. Although OWL is primarily an ontology language that provides classes, properties, individuals, and data values for RDF documents, it also includes basic mechanisms for validation and inference that can be executed by OWL reasoners. As OWL allows only for basic reasoning like co-reference resolution respectively distinguishing and property restrictions for values and cardinality, SWRL has been introduced as an extension for the OWL terminology which makes use of the Rule Markup Language for more advanced rules, for example derived properties and assertions. SWRL does also provide a set of build-in functions for comparisons, math operations, and XML Schema Definition (XSD) data type specific manipulation functions. However, SWRL does not provide templates for shape constraints or a notion for user-defined functions. These limitations are addressed by SPIN, which aims to provide general business rules expressed in SPARQL Protocol and RDF Query Language (SPARQL). SPIN therefore allows for a more flexible implementation of user-defined models and constraints as they are required for our work. Knublauch et al. [4] have introduced the Shapes Constraint Language (SHACL) which can be regarded as the legitimate successor of SPIN[9]. It aims to describe and constraint especially the contents of RDF graphs by defining shapes that specify conditions that apply to a given RDF node using a high-level vocabulary. SHACL, which was firstly introduced as World Wide Web Consortium (W3C) public working[10] draft in October 2015 and is now available as W3C recommendation, lastly updated on 20 July 2017.

[4] https://spark.apache.org/.
[5] https://flink.apache.org/.
[6] https://www.w3.org/TR/owl2-overview/.
[7] https://www.w3.org/Submission/SWRL/.
[8] http://spinRDF.org/.
[9] http://spinRDF.org/spin-shacl.html.
[10] https://www.w3.org/TR/2015/WD-shacl-20151008/.

Discussion: From the related work we can learn that there are already tools that allow for RDF mapping, executing, processing and evaluation of RDF streams and also frameworks for modelling RDF constraints using explicit semantics and shared vocabularies. What is still missing is a framework that maps non-semantic data streams of heterogeneous observation stations to an RDF stream with meaningful explicit semantics based on collaboratively gathered annotations, including constraint validation and data provenance.

3 Approach

Initial situation: For our approach, we assume that series of observations from multiple observation stations are given as data streams S of continuous observation messages O. Each data stream is defined as $S_i = \{O_1, \ldots O_n\}$. Observation messages in a stream are represented as sets of key-value pairs which do not contain any explicit semantics, such as *"temperature=40"*. An observation message is defined as $O = \{(k, v) : k \in \mathbb{STR}, v \in \mathbb{VAL}\}$. Keys are interpreted as strings, whereas the values could be strings, numbers, booleans, objects, arrays, or null.

Requirement: In order to address the issue of lacking semantics of data streams, we require an adequate description of metadata. Let $M = \{(k, u) : k \in \mathbb{STR}, u \in \mathbb{URI}\}$ be a metadata description containing key-value pairs that we use for mapping keys to Uniform Resource Identifiers (URIs) which identify the linked concept, e.g. *(temperature, "quantity*[11]*:ThermodynamicTemperature").*

Metadata management: To manage the metadata, we introduce an annotation platform that allows for collaboratively created annotations of non-semantic data streams. With the annotation platform, we enable domain experts and other non-developers to intuitively annotate data sources with explicit semantics. Users of the annotation platform are provided with forms, where they can easily select applicable quantities, units and data types from a list of shared vocabularies. For context specific vocabularies, where no shared vocabularies can be used, it is also possible to define new classes and properties. The benefit of having one collaborative platform for all annotations in contrast to annotations per-source, per-format or even per-case is the logical abstraction of the underlying formats, data structures and serialization, and the possibility to reuse semantic annotations for similar data sources.

Semantic mapping: To combine messages of observation streams with the semantic annotations in a meaningful way, we introduce a semantic mapping process *SemMap*. For that process, we assume at least two diverse input streams S_1 and S_2 of observation messages. In addition, we assume a metadata repository which contains a machine processable representation of the semantic annotations and a mechanism that sends update notifications whenever there is a change in the metadata repository. The mapping of each observation message O based on

[11] http://qudt.org/schema/quantity#.

metadata M is defined as $SemMap(O, M) = \{(u, v) : (k, v) \in O, (k, u) \in M\}$. The output of the $SemMap$ process is data stream S_{LD} which consists of continuous messages from all input streams mapped to the explicit semantics of context, observed quantities, units and provenance. The output stream is defined as $S_{LD} = \{\{SemMap(O, M_1) : O \in S_1\} \cap \{SemMap(O, M_2) : O \in S_2\}\}$. The additional explicit semantics information enables a consuming application of data stream S_{LD} to interpret each message correctly without the need of any further data understanding or coding. Moreover, whenever there is a change in the metadata repository, for example due to a newly added or modified annotation, data stream S_{LD} immediately includes that new semantics and the consumer is able to interpret it correctly without any adoption of the code. An overview of this semantic mapping process is shown in Fig. 1.

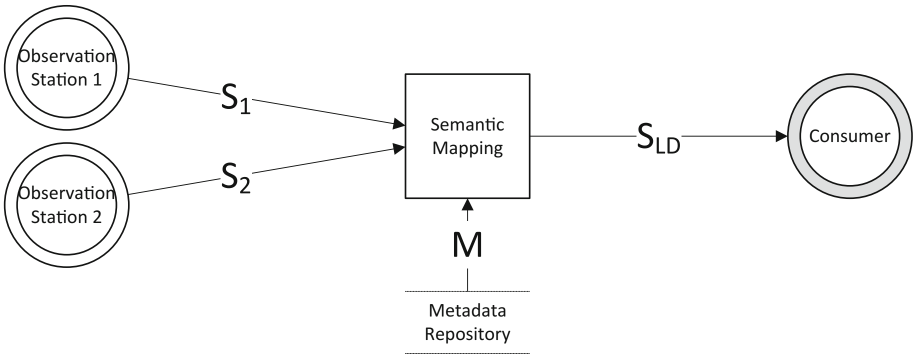

Fig. 1. Overview of semantic mapping process

Shape constraints: In addition to adding explicit semantics to observation messages, we do also allow for defining and evaluating shape constraints. These shape constraints can help to identify and filter invalid observations or add derived properties. For example, a message stream with values of air pollution can be evaluated on-the-fly whether the legal limit is exceeded or not, based on the metadata associated with that class of observations. Generic evaluation of observation data based on metadata which can easily be adopted by changing the related annotation spares developers to alter the programming code. The shape constraints can then be used for shape validation without adopting the code, just by creating annotations on the collaboration platform.

4 Implementation

For the first implementation of the annotation platform, we have decided to use the technically mature Semantic MediaWiki (SMW) [5] to ensure a reproducible result for the evaluation of our approach. SMW comes with hardly any predefined shared vocabularies, therefore the first step is to import shared vocabularies for sensor observations of observation stations, for describing quantities and units and to define shape constraints. For this purpose, we provide Semantic Sensor Network (SSN) [1] for describing sensors of observation stations and Quantities, Units, Dimensions and Data Types Ontologies (QUDT) for describing quantities, units and data types, but additional vocabularies can be imported. As discussed in Sect. 2, we employ SHACL as the basis for shape constraints. Therefore, we also import the SHACL shared vocabularies to our annotation platform and allow the platform user to annotate data sources with shape constraints and validate the shape of the produced messages based on the SHACL shape definitions provided by the metadata repository. For an easy editing of metadata and shape constraints of data sources, we implement templates and forms.

In the background, the terms are linked with the imported shared vocabularies which allows for interchange of these terms on other platforms with explicit semantics. However, it has to be ensured that the string value of the key for the new annotations is equal to the string value of the key produced by the annotated data source in order to ensure the correct assignment of metadata to observed values. The annotation platforms links these constraints to the according SHACL terms in the background for the evaluation later on.

For the stream mapping, we implement a consumer for configurable topics in Java, based on Apache Flink. Flink provides adapters for Apache Kafka[12] which we employ as our message broker. Whenever a new data source is registered to the metadata repository, the stream processing application receives an update notification and starts a new consumer for the according topic. We assume that messages in each input stream are serialized in JavaScript Object Notation (JSON). For each JSON object in the stream, the semantic mapping function performs a matching of keys for each member with keys received for the metadata from the metadata repository. For creating the RDF model, we employ Apache Jena[13]. For each message from an observation station, a Flink map function creates a new RDF instance of the class defined in the metadata repository for this station. For this instance, RDF properties are derived from the metadata repository for each match of a JSON member key and a metadata key. The value for the properties are extracted from the JSON object. Members of the message which are not defined in the metadata repository are interpreted as plain literal values and added to the RDF model as well. Finally, the RDF model is serialized to JSON-LD and published to the output stream. This output stream contains all the explicit semantics provided in the metadata repository and enables consumers to evaluate each message correctly, including provenance information. Whenever an annotation changes in the annotation platform, another update notification is send to the processing application and immediately affects the semantic mapping process without the need of any code adaption.

Rather than just adding explicit semantic to the observation messages, it is also possible to validate the semantic of each message based on shape

[12] https://kafka.apache.org/.
[13] https://jena.apache.org/.

constraints defined in the metadata repository. As we have imported and linked SHACL vocabularies to our annotation platform, any application that implements SHACL validation is suitable to perform the message validation.

5 Evaluation

To evaluate our approach, we process non-semantic data streams of public and private environmental observation stations, map them to a representation with explicit semantics retrieved from a semantic wiki platform and interpret each message based on shape constraints from the same wiki platform. The whole framework is executed on a system with Intel(R) Core(TM) i7-5600U CPU at 2.6 GHz and 12 GB memory.

For the evaluation, we implement a Kafka producer that creates a configurable amount of non-semantic messages to simulate third-party data streams. As templates for these messages we use the structure of a public environmental observation station and also a private environment observation station. For both types of messages we provide annotations on our annotation platform that fits to the keys of the members of each message type and also the shape information that we assume for a valid message from an observation station.

To evaluate the processing time, we create a timestamp when (1) a JSON message from the input stream is loaded into the application, (2) the RDF model is created including all metadata from the annotation platform, (3) the RDF model is serialized to JSON-LD and (4) the JSON-LD message is evaluated using the TopBraid SHACL API[14] in conjunction with the according shape constraint from the annotation platform. The test run covers a number of exactly 10,000 messages. As the test runs with limited hardware resources, a frequency of about 150 messages per second can be processed at the most. The results are shown in Table 1. The minimal and median values show that even with limited hardware resources a message from an observation stream can be mapped to an RDF model with explicit semantics, serialized to JSON-LD an validated with SHACL in less than one millisecond. Arithmetic mean, standard deviation and the maximum values show that there are also some outliers which require more processing time. Multiple executions of the test setup have produced similar results which

Table 1. Time in milliseconds for creating, serializing and validating one message

$f \approx 150 \frac{messages}{second}$	Arithmetic mean	Median value	Standard deviation	Min value	Max value
Creation	1.460	0.341	3.699	0.196	47.750
Serialization	0.342	0.270	1.372	0.135	130.886
Validation	3.020	0.195	5.684	0.062	129.788
Total	4.822	0.979	6.813	0.468	276.622

[14] https://github.com/TopQuadrant/shacl.

confirm this findings. As the architecture of our implementation is optimized for distributed systems, a higher message throughput can be realized by adding more nodes to the processing framework.

6 Conclusion

In this paper, we have shown how collaboratively created annotations from a semantic wiki platform can be exploited to map non-semantic data streams of public and private observation stations to a representation with explicit semantic information of observations, measured quantities, measuring units, and context information. Furthermore, we have shown how data shape constraints can be defined on a collaborative wiki platform and employed for constraint validation of observation data streams on-the-fly. To evaluate our approach, we have measured the time interval for creation, serialization and validation of messages from a continuous data stream using the message format of exemplary public and private environmental observation stations. The results show that our generic approach for mapping non-semantic streams of observation messages to a meaningful representation with explicit semantic information and validating the shape constraints of messages can be done efficiently on-the-fly without adopting the code to specific data sources.

As future work, we plan to assist domain experts in creating semantic annotations by deriving metadata of observation messages directly from data streams using collaboratively created pattern definitions of commonly observed quantities. Further, we plan to apply more cognitive consumers of semantically mapped data streams for exploiting the benefits of the explicit semantic information.

References

1. Compton, M., et al.: The SSN ontology of the W3C semantic sensor network incubator group. J. Web. Sem. **17**, 25–32 (2012)
2. Dimou, A., et al.: RML: a generic language for integrated RDF mappings of heterogeneous data. In: Proceedings of the 7th Workshop on Linked Data on the Web (2014)
3. Frank, M.: Integrating big spatio-temporal data using collaborative semantic data management. In: Bozzon, A., Cudre-Maroux, P., Pautasso, C. (eds.) ICWE 2016. LNCS, vol. 9671, pp. 507–512. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38791-8_38
4. Knublauch, H., Kontokostas, D.: Shapes constraint language (SHACL): W3C recommendation, 20 July 2017 (2017). https://www.w3.org/TR/shacl/
5. Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., Studer, R.: Semantic wikipedia. J. Web Sem. **5**(4), 251–261 (2007)

6. Tommasini, R., Della Valle, E., Mauri, A., Brambilla, M.: RSPLab: RDF stream processing benchmarking made easy. In: d'Amato, C., Fernandez, M., Tamma, V., Lecue, F., Cudré-Mauroux, P., Sequeda, J., Lange, C., Heflin, J. (eds.) ISWC 2017. LNCS, vol. 10588, pp. 202–209. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68204-4_21
7. Wiener, P., et al.: BigGIS: a continuous refinement approach to master heterogeneity and uncertainty in spatio-temporal big data. In: Proceedings of the 24th ACM SIGSPATIAL 2016, Burlingame, California, USA. ACM (2016)