Enabling Semantic Complex Event Processing in the Domain of Logistics*

Tobias Metzke[1], Andreas Rogge-Solti[1], Anne Baumgrass[1], Jan Mendling[2], and Mathias Weske[1]

[1] Hasso Plattner Institute at the University of Potsdam, Germany
tobias.metzke@student.hpi.uni-potsdam.de
{firstname.lastname}@hpi.uni-potsdam.de
[2] Institute for Information Business at Vienna University of Economics and Business, Austria
jan.mendling@wu.ac.at

Abstract. During the execution of business processes, companies generate vast amounts of events, which makes it hard to detect meaningful process information that could be used for process analysis and improvement. Complex event processing (CEP) can help in this matter by providing techniques for continuous analysis of events. The consideration of domain knowledge can increase the performance of reasoning tasks but it is different for each domain and depends on the requirements of these domains. In this paper, an existing approach of combining CEP and ontological knowledge is applied to the domain of logistics. We show the benefits of semantic complex event processing (SCEP) for logistics processes along the specific use case of tracking and tracing goods and processing related events. In particular, we provide a novel domain-specific function that allows to detect meaningful events for a transportation route. For the demonstration, a prototypical implementation of a system enabling SCEP queries is introduced and analyzed in an experiment.

1 Introduction

The enterprise system landscape has significantly changed in the last decade. Sensors are increasingly used to track objects via Global Positioning System (GPS), measure temperature, energy consumption and other types of data. Sensors provide this data in event streams. An event, in general, is something that happens or occurs and might change the current state of a system [1].

For the detection of complex and meaningful patterns in event streams, users can rely on different event operators and temporal relationships that are provided by *Complex Event Processing (CEP)* technology [1]. The enrichment of event streams with high-level knowledge is required for handling the context in which the stream data is interpreted and analyzed [2]. At present, research in this area

* The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement 318275 (GET Service).

A.R. Lomuscio et al. (Eds.): ICSOC 2013 Workshops, LNCS 8377, pp. 419–431, 2014.

has formed the term of *Semantic Complex Event Processing (SCEP)* [2,3,4]. There are prototypes combining ontological knowledge with CEP techniques, dealing with specific domains like smart grids, advanced facility management and stock markets. The domain of logistics, however, has not been covered by these considerations yet. In logistics, considerable amounts of event-based data are produced, e.g. traffic and weather information [5]. These data are relevant for logistics processes to ensure timely transport of goods [5,6]. Besides the events that occur directly on the planned routes, also events in geographical proximity of planned routes can affect logistics processes. For example, you might want to circumnavigate congestions, flooded areas, or other dangerous regions in safe distance.

In this paper, we address the need for domain-specific event operators to serve logistics scenarios. Thereby, this paper makes the following contributions. First, the usage of domain-specific knowledge in form of an ontology for CEP is shown for the logistics domain. To use the DBpedia as a top-level ontology[1] only minor extensions where necessary. Second, we use SCEP to implement a novel logistics-specific built-in function. This function is able to consider geo-spatial distance and, thus, support the special requirements of logistics processes for CEP query creation. In this way, our contribution can serve as a basis for a convenient routing, re-routing and transportation of goods in the domain of logistics. To demonstrate the applicability of the approach, we implemented a lightweight SCEP prototype. This prototype is used in an evaluative experiment to illustrate the effectiveness of the proposed concepts.

The remainder of this paper is structured as follows: Section 2 provides an overview of the characteristics of CEP in the domain of logistics and introduces several use cases. Section 3 outlines the general approach taken for the inclusion of ontological knowledge in CEP and explains the mechanisms that allow for semantic querying in logistics. Based on the SCEP concepts, Section 4 presents the logistic-specific adaption for identifying transportation-related events. Following, Section 5 depicts the details of the prototypical implementation of the approach before Section 6 evaluates the general benefits and shortcomings of it. Section 7 then summarizes research related to the presented approach. This work concludes with Section 8, summarizing the work done and providing an outlook on future steps.

2 Usage of SCEP in Logistics

An important use case in the logistics domain is *track and trace* [5,6]. Logistics service providers (LSPs) monitor their means of transportation like trucks, ships, and containers on their respective routes worldwide. LSPs depend on the detection of relevant complex events from numerous data sources. In particular, events from external sources need to be correlated to the respective routes and transportation means.

[1] See `http://dbpedia.org/About`

Weather events like floods and storms as well as road blockages can have an impact on transporation routes. Therefore, events located near these routes can affect the schedule of a transportation plan. Thus, LSPs need to detect those events to be able to react appropriately and timely. The more information an LSP gains on such events, the more precise its reaction can be. To this end, complex event queries can be constructed to listen to event streams. Without external knowledge, however, it is inconvenient to construct queries that capture the necessary information for such scenarios. More specifically, query designers would need to know the characteristics of all the routes that the LSP's trucks, ships, or containers are on. These specifics could include knowledge about the waypoints on the routes, the overall lengths of the routes as well as the time needed to complete them. This hampers convenient and efficient query creation for this and more complex use cases. Therefore, CEP engines that are used in logistics should make use of background knowledge.

SCEP engines [3,7,8] provide knowledge to query designers in a convenient way, by abstracting from domain-specific complexities. Furthermore, semantic technologies enable automatic normalization of event stream sources by semantic annotation. With ontological background knowledge, a query designer can conveniently create high-level queries that capture relevant information, instead of having to cope with stream source specifics. Section 3 provides further information on how semantically rich CEP queries can be written and how they are evaluated.

3 Querying Semantic Events

Semantic CEP querying on event streams allows for the combination of CEP and semantic web technologies. The user defines queries, in this case *semantic queries*, which are registered in a *semantic CEP engine*. Incoming event streams are monitored by this engine and evaluated against the defined user queries with the help of *ontological background knowledge* and CEP capabilities. The results of these evaluations are gathered and the user is informed about the matching queries. The model of semantic events is introduced in Section 3.1. In Section 3.2, the structure of SCEP queries and their difference to CEP queries is given. Afterwards, the process of *semantic querying* is explained in Section 3.3.

3.1 Semantic Event Model

In order to integrate semantics in CEP queries, Zhou et al. [9] propose a *semantic event model* that captures event attributes, their semantics, domain entities and the relations between them. This model describes how event attributes are matched to semantic entities and linked to external knowledge bases in order to enrich the information of the events. In the end, incoming *raw events* (e.g. lightweight data tuples) are transformed into *semantic events* (graph-based data, see Fig. 1) that point into existing ontologies. Such ontologies comprise entities, literals and links between them. The resulting graph structure

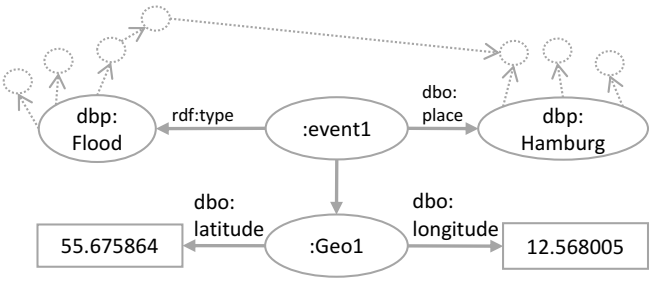


Fig. 1. A semantic event dynamically created from the raw event in Listing 1.1

allows for graph-specific queries and algorithms to be applied on the information stored. There have been several approaches to introduce a common logistics domain-ontology [10,11]. However, common knowledge ontologies (or top-level ontologies) like the DBpedia include most of the transportation-related entities already. Minor extensions to this knowledge base are provided in the paper where necessary.

With the help of semantic events, semantic queries can be defined that make use of knowledge in ontologies events are linked to. For example, the *raw event* shown in Listing 1.1 is a flood warning for Hamburg, a city in Germany. Such an event can be dynamically annotated either with the help of semantic annotation techniques [12] or static mapping files that are maintained for specific event streams [3]. The resulting *semantic event* is shown in Fig. 1. It references entities from the underlying DBpedia ontology like *dbp:Flood* and *dbp:Hamburg*. The *dbo* namespace describes all DBpedia ontology predicates and some basic classes while the *dbp* namespace primarily describes entities that have a Wikipedia website. Thus, the event is linked to all the knowledge available for these two entities in that ontology, indicated by the dotted circles and arrows. SCEP queries can then use these connections and work with knowledge the event itself does not provide but it is linked to.

Listing 1.1. An incoming raw event with a list of string-based key-value pairs

```
[ ('type', 'FloodWarning'), ('city', 'Hamburg'), ('location','52.181701 11.600692')]
```

3.2 Semantic Event Queries

Based on the semantic event model, a basic structure for semantic queries can be introduced. These SCEP queries start with a traditional CEP pattern and can further be specified by a semantic part as shown in Listing 1.2.

Listing 1.2. Structure of a semantic query

```
SCEP Query ::=
  [PREFIX <namespace>]
  [CEP Subpattern ::=
   SELECT <event*, attribute*, aggregation*>
    FROM <input stream AS event>*
    (WHERE <relational constraints>)?
    (SEQ <event, event, ...>)?
    (WINDOW <window specifications>)?]
  [Semantic Subpattern ::=
    {(<subject URI> <predicate URI> <object URI|Literal>.)*}]
```

First, the *CEP subpattern* specifies the temporal and relational constraints of events based on their attributes. The CEP subpattern consists of the traditional *SELECT* and *FROM* statements specifying the data stream and the projection of attributes and aggregations the query returns. The *event* specified in this pattern will be passed on to the semantic subpattern upon evaluation process of an incoming event. The *where*, *sequence* and *window* selectors are optional and help to further specify the query. The CEP subpattern does not directly correspond to one existing CEP language definition but rather generalizes common features of several ones [8,13].

Second, the *semantic subpattern* places semantic constraints over events and their associated domain entities [3]. Semantic patterns are written in SPARQL[2] triple notation. The pattern is a list of statements that comprise a subject, a predicate, and an object. While the subject and predicate have to be given by an uniform resource identifier (URI), the object can be either defined by an URI or a literal (e.g. a *string*, an *integer*, or any other data type) (see Listing 1.2).

Following the example use case of Section 2 and the event described in Listing 1.1 and Fig. 1, a query following the SCEP pattern is shown in Listing 1.3. This specific query asks for the given event type and its geographical location, if it contains information about the city of *Hamburg* in Germany and warns about a flood.

Listing 1.3. An example for a SCEP query

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/resource/>
SELECT type, location
FROM WorldWeatherStream as $event
{$event dbo:place ?city.
$event rdf:type dbp:Flood.
filter(?city = dbp:Hamburg)}
```

[2] See `http://www.w3.org/TR/sparql11-query/`

The query in Listing 1.3 basically works with information that is present in the event itself and has simply been transferred into an ontology. Furthermore, a SCEP query can work with data that is not provided by the event directly. For example, the population of a city the event happens at could be taken into account, although it is not provided by the event itself (see Listing 1.4).

3.3 Semantic CEP Querying

In order to semantically evaluate incoming events, they have to be transformed into semantic events (see Section 3.1) and evaluated with the help of registered semantic queries (see Section 3.2). More specifically, the semantic subpatterns of the semantic queries are used for the semantic evaluation. Therefore, the semantic part is transformed into a SPARQL ASK query[3], as shown in Listing 1.4.

Listing 1.4. An example ASK query

```
PREFIX dbo: <http://dbpedia.org/ontology/>
ASK {$event dbo:place ?city.
?city dbo:populationTotal ?population.
filter(?population > 1500000)}
```

In general, these queries return either *True* or *False* upon evaluation. The evaluation workflow of the given query displayed in Listing 1.4 for the event shown in Fig. 1 would be the following:

1. *Hamburg* is saved into the variable *?city*
2. The related total population (*1,796,077*) of this *?city* is read from the knowledge base, here DBpedia, and is saved into the variable *?population*
3. The value of the *?population* is compared to the specified *1.5 million*
4. The *?population* is higher than 1.5 million that is why the query returns *True*, otherwise it would have returned *False*. If the *?city* would not have a *population* value in the DBpedia ontology, the *?population* variable would be empty and the query would return *False*.

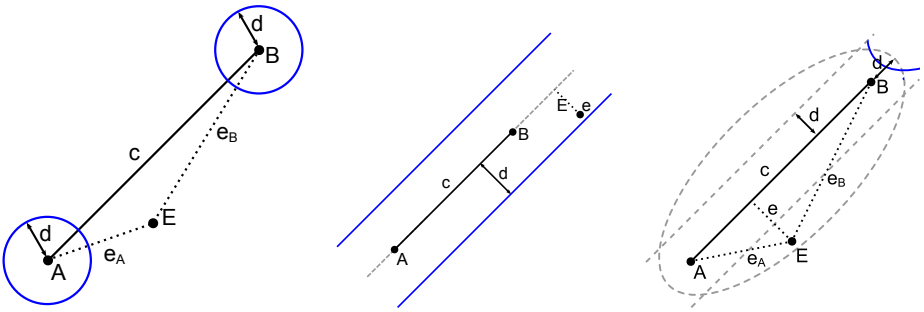
As the semantic subpattern query for the example event in Listing 1.1 returns *True*, it passes the event to a CEP evaluation module. In this module, the CEP subpattern is evaluated for the event in the traditional CEP manner. If the criteria of the CEP subpattern are fulfilled, the user is informed of a match with his SCEP query. Details on how this is prototypically implemented are described in Section 5.

4 Identifying Transportation-Related Events via SCEP

In the domain of logistics, the locality of events and their distance to transportation routes are of special interest, as these events might be relevant to the

[3] See `http://www.w3.org/TR/sparql11-query/#ask`

transportation plan and its execution (e.g., a flood event in a nearby location could affect the transportation plan). We assume that a transportation route is stored in an ontology consisting of route segments that have a start and an end point, which in turn are places with geographical coordinates (in latitude and longitude format). We want to determine, whether an event is in a given distance to a route (i.e., whether it is relevant for transportation). Each route segment can be checked independently. An event is transportation-relevant, if it is nearby at least one segment. Fig. 2 shows three methods that determine whether an event is relevant for a route. These methods were selected due to the simplicity of their calculation and are described as follows.



(a) Point-to-point distance. (b) Point-to-line distance. (c) Ellipse inclusion.

Fig. 2. Three different ways of finding nearby events (E) to a route (A, B). (a) Distance between two points. (b) Distance of a point to a line. (c) Ellipse inclusion and distance of a point to a line.

Distance between two points. The event is considered transportation-related, if the location of the event is closer to any of the points on the route than a defined distance d. Assuming that the geographical points on the route are dense, the error introduced by this simple solution can be acceptable. However, as Fig. 2a shows, if the distance between the points on the route is high compared to d, a relevant event E lying between two route points might not be detected.

Distance of a point to a line. The event is considered transportation-related, if the location of the event is closer to the line between two consecutive points on a route than distance d. However, as Fig. 2b highlights, events that are close enough to the line between two points are marked as relevant even if the events are too far away from the two points on that line.

Ellipse inclusion. An event is considered transportation-related, if it lies within the ellipse spanned by the two points and the defined distance d with the formula $a + b < c + 2 * d$, and if the distance of the event's location to the line between the two route points is less than distance d (cf. previous method). The addition of the ellipse constraint ensures that we do not include events far from the route segments.

With current CEP querying, the task to find transportation-related events to a route would require the query designer to know all the relevant transportation plans, the geographical points on these plans and their order on the route. He could then compute the distance according to the before mentioned methods. Although the approach introduced by Zhou et al. [3] provides the knowledge about transportation plans and their segments in a convenient way, a user would still have to manually create a query that computes the distance of an event to any of the segments in the transportation plans.

Therefore, the presented approach can be extended by a convenient built-in function that serves exactly this purpose. A query example with the built-in function is shown in Listing 1.5. As displayed, users can filter for specific event types like *floods* and define, at what distance to the route they are to be included. The example sets a distance threshold of 50km for floods nearby the transportation route of plan *LSP_1*. In practice, domain experts need to determine distance thresholds for the inclusion of events.

Listing 1.5. Usage of the built-in function `nearby` provided by the prototype

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/resource/>
PREFIX logistics: <http://example.org/logistics#>
SELECT type, location
FROM WorldWeatherStream as $event
{$event rdf:type dbp:Flood.
$event dbo:place ?location.
logistics:LSP_1 dbo:plan ?plan.
filter($nearby(?location, ?plan, 50))}
```

On query registration, the query is automatically translated to a more complex query including the necessary distance calculations[4]. Note that the rewritten query comprises more than forty additional lines of SPARQL code in that example. A query designer without the proposed method based on SCEP would have to manually write that amount of additional code to enable the search of nearby events for transportation plans. Our generated queries are less error-prone, as they are generated from a shorter – and therefore, more understandable – query. Furthermore, queries are also less error-prone, since the code is generated automatically and individual mistakes can be avoided. Besides, once the improved calculation of nearby search is implemented with the built-in method `nearby`, all queries using the method benefit from that improvement automatically.

5 Implementation/Architecture

The concepts introduced in the previous sections were implemented in a prototype[5] based on *Python*[6] for server-side implementation and *JavaScript* on the

[4] The rewritten query stub can be found at `http://bpt.hpi.uni-potsdam.de/pub/Public/EPP/nearby_function.txt`

[5] Instructions and downloads are available at: `http://bpt.hpi.uni-potsdam.de/Public/EPP#Semantic_Extension`

[6] See `http://www.python.org/`

client side. The prototype , depicted in Fig. 3, provides a client-server architecture, serving a web interface that can be accessed through a common web browser. In Fig. 3, the web interface is marked in gray, denoting that it has been exchanged or added compared to the initial architecture, provided by Zhou et al. [3]. This holds for all other components that are depicted in out architecture as well. The SCEP engine includes an Resource Description Framework (RDF) Server, namely *Virtuoso RDF Triple Store*[7] which holds the ontologies necessary for the use cases of the logistics domain mentioned before. It allows for querying these ontologies through a web interface as well as programmatically. The semantic engine is build around an existing CEP engine further described by Herzberg et al. [14]. The CEP engine supports simple filtering, sequence and aggregation patterns over event streams. It also allows for sliding time window aggregations. It offers a web interface as well, allowing for the registration of queries as well as the sending of events. The prototype we developed in this paper provides additional semantic filtering functionality around the CEP engine in order to enable SCEP querying on incoming event streams. These additional modules, shown in Fig. 3, serve the following purposes:

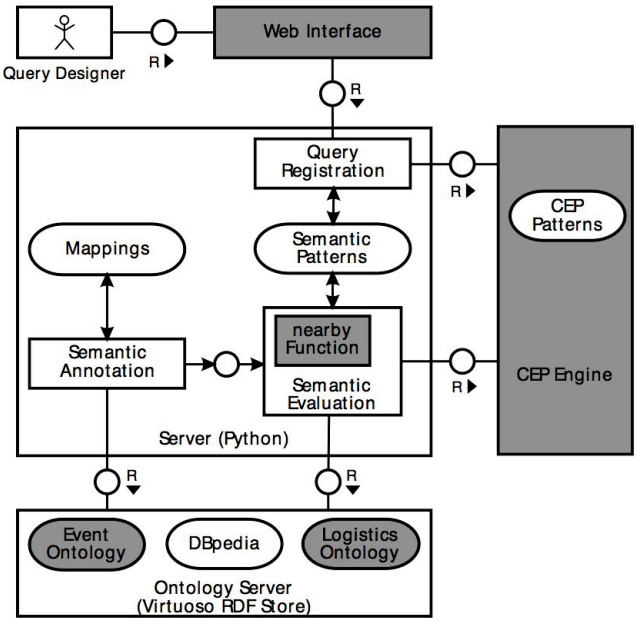


Fig. 3. Architectural Component Overview

The Query Registration module processes the user queries entered through the web interface of the SCEP engine. They are split into their semantic and their CEP pattern, the latter of them is registered in the CEP engine for

[7] See `http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSTriple`

CEP evaluation later on. The semantic subpattern is stored in the server, shown in the interface and evaluated on incoming event streams that match the defined stream source in the CEP subpattern part.

The Ontological Library holds the defined knowledge bases in OWL[8] format. They capture the relations between semantic entities and rules defined for them. Events are linked into these ontologies by pointing to any of their entities. The *Event Ontology* reflects the *semantic event model* mentioned before.

The Semantic Annotation module materializes semantic events from incoming raw events arriving on a registered event stream source. Mapping files describe the correlation between input event attributes and the semantic event properties and entities. These mapping files can be static, direct mappings or dynamic mapping files that use semantic annotation techniques in order to find the best matching semantic properties and entities for incoming event attributes. For the prototype, the direct mappings are implemented.

The Semantic Evaluation module processes incoming semantic events and evaluates them according to the workflow shown in Section 3.3. Therefore, the registered semantic subpatterns that match the stream source of the incoming event are evaluated for it with the help of the RDF Server. If a semantic subpattern evaluates to *True* , the raw event associated to the semantic event is passed on to the CEP Engine where the event is evaluated with the corresponding CEP pattern of the semantic subpattern.

Finally, the CEP engine evaluates the passed raw events and performs the actions registered in the CEP pattern when identifying a pattern match. For performance reasons, if no registered semantic subpatterns associated to the event's stream exist, the raw events are not semantically annotated but passed on to the CEP engine directly.

6 Experiment

The prototype presented in this paper basically relies on the concepts introduced by Zhou et al. [3] and would therefore yield no new insights concerning event throughput performance. However, the built-in function for nearby search serving as a logistics specific extension may introduce new computational complexity.

Therefore, the median response time of a query similar to the one depicted in Listing 1.5 was measured with the help of the prototype introduced in Section 5. The evaluation routes where given by a list of real-world transportation routes containing different numbers of geographical points describing them. The different routes contain 4, 8, 36, 187, 201, 388, 744, 1520, 3040, 6080, 12160, 24320, 48640, and 97280 points, where 97280 points can describe a route of approximately 16,000 kilometers with a GPS coordinate for every 170 meters. All routes were stored in the Virtuoso Server in RDF triple format. For every route,

[8] See `http://www.w3.org/2004/OWL/`

a query evaluating incoming events against the route was registered in the prototype. Only one query was registered at a time in the system in order to avoid dependencies between the response times of the queries. Each query was then evaluated three times with a set of 400 randomly created events which were fed into the system sequentially. The average of the measured response times builds the final value for every query.

Fig. 4 displays the behavior of a query containing a nearby search. The response times themselves are rather high compared to the prototype build by Zhou et al. [3] due to the client-server architecture of the prototype (see Section 5). The nearby search query however indicates a linear relationship between points on the route and the response time of the query. When doubling the number of points on the transportation route from 48,640 to 97,280, the response time grows by a factor of approximately 1.6. Thus, the overhead produced by a nearby search can be well estimated and stays in a feasible range when increasing the accuracy by introducing more points along a route. Further work on decreasing the complexity while preserving the accuracy of the nearby search could yield better results concerning computational overhead in the future.

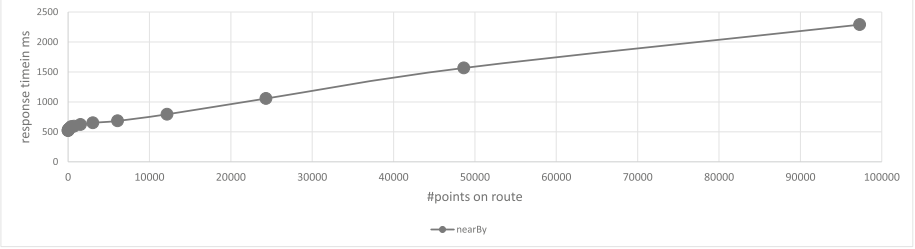


Fig. 4. Response time of a query containing the *nearBy* function for different numbers of points on a transportation route

7 Related Work

Recently, the fusion of complex event processing with semantic background knowledge as presented in this paper has been widely researched [2,3,8,15,16]. Anicic et al. [17] proposed EP-SPARQL, an extension of the SPARQL language that allows to process streams of data that are temporally related. They introduced new language constructs that allow for queries not only on stored background knowledge but also the time relations between incoming events. With ETALIS, Anicic et al. [8] provided a rule-based deductive system that acts as a semantic event processing engine and uses EP-SPARQL. Event queries are written in SPARQL and enriched by the temporal operators introduced in EP-SPARQL. All background knowledge and event queries are transformed into Prolog rules and executed in a Prolog engine. While this approach is independent from existing, traditional CEP engines, the approach presented in this paper leverages their expressibility and performance.

Teymourian et al. [7] proposed a modular ontology model and architectural vision for SCEP. In their architecture, the semantic knowledge base includes ontologies and inferencing rules. They use a rule-based engine in order to process the incoming event data and to evaluate the user queries. However, they do not leverage existing CEP engines and their expertise and performance in the field of event processing. Zhou et al. [3] built on the architectural model shown in [7], employing a state of the art CEP engine instead of a rule-based engine and extending the approach by the ability to evaluate historical event data as well. They enable semantic event queries that can evaluate past, present, and future event data. The approach presented in this paper builds on the proposed architecture, event model, and query definitions. It explores the applicability of the given approach for the domain of logistics, uses a recently build CEP engine and extends the given approach by logistics specific ontologies and a built-in function.

The need for special functionality in the domain of logistics originates in the typical use cases of this domain like *tracking and tracing*. It has been introduced by van Dorp [5] and Shamsuzzoha et al. [6], highlighting the benefits and possibilities of real-time monitoring of transportations for the domain of logistics. The presented approach employs an example from the tracking and tracing use cases and outlines the effectiveness and expressibility of semantic CEP queries in this domain.

In order to work on semantic background knowledge, it has to be provided in the form of ontological knowledge. Approaches by Lian et al. [10] and Hoxha et al. [11] have introduced how such ontologies can be established and which concepts are relevant in the domain of logistics. The work presented in this paper bases on concepts introduced in the DBpedia ontology like companies, means of transportation, and geographical regions. The necessary extension points for this ontology to work in the domain of logistics are minimal, this is why no logistics specific ontology was used in the presented approach. However, more complex use cases may imply the use of a specific ontology rather than a top-level ontology like the DBpedia.

8 Conclusion

In this paper, the applicability of SCEP for the domain of logistics is evaluated. The presented approach clearly shows that the use of domain-specific knowledge in logistics can lead to expressive and convenient CEP query creations. As shown, typical logistics use cases can benefit from the capabilities provided by semantic web technologies with only a few adaption when it comes to semantic event processing. The paper also introduces a prototype that leverages the advantages of an existing CEP engine. Furthermore, the work outlines a novel built-in functionality for logistics based on event query rewriting which can easily be integrated in any similar SCEP approach. Nonetheless, the use of domain-specific knowledge also introduces computational overhead compared to state of the art CEP which has to be tackled by efficient implementations and

caching mechanisms. Future work will focus on further logistics use cases and their characteristics and needs in terms of CEP. Beyond that, the applicability of other approaches to the logistics domain needs to be evaluated.

References

1. Luckham, D.C.: The Power of Events. Addison-Wesley (2002)
2. Teymourian, K., Rohde, M., Paschke, A.: Fusion of Background Knowledge and Streams of Events. In: Proc. of the 6th ACM International Conference on Distributed Event-Based Systems (DEBS), pp. 302–313 (2012)
3. Zhou, Q., Simmhan, Y., Prasanna, V.: SCEPter: Semantic Complex Event Processing over End-to-end Data Flows. Technical report, Technical Report 12-926. Computer Science Department, University of Southern California (2012)
4. Walavalkar, O.B.: Streaming Knowledge Bases. ProQuest (2007)
5. Van Dorp, K.J.: Tracking and Tracing: A Structure for Development and Contemporary Practices. Logistics Information Management 15(1), 24–33 (2002)
6. Shamsuzzoha, A., Helo, P.T.: Real-time Tracking and Tracing System: Potentials for the Logistics Network. In: Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management, pp. 22–24 (2011)
7. Teymourian, K., Paschke, A.: Enabling Knowledge-based Complex Event Processing. In: Proc. of the 2010 EDBT/ICDT Workshops, vol. 37, pp. 1–37. ACM (2010)
8. Anicic, D., Rudolph, S., Fodor, P., Stojanovic, N.: Stream Reasoning and Complex Event Processing in ETALIS. Semantic Web 3(4), 397–407 (2012)
9. Zhou, Q., Simmhan, Y., Prasanna, V.: Towards an Inexact Semantic Complex Event Processing Framework. In: Proc. of the 5th ACM International Conference on Distributed Event-based Systems (DEBS), pp. 401–402 (2011)
10. Lian, P., Park, D.W., Kwon, H.C.: Design of Logistics Ontology for Semantic Representing of Situation in Logistics. In: Proc. of the 2nd Workshop on Digital Media and its Application in Museum & Heritages, pp. 432–437. IEEE (2007)
11. Hoxha, J., Scheuermann, A., Bloehdorn, S.: An Approach to Formal and Semantic Representation of Logistics Services. In: Proc. of the Workshop on Artificial Intelligence and Logistics (AILog), pp. 73–78 (2010)
12. Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D.: Semantic Annotation, Indexing, and Retrieval. Web Semantics: Science, Services and Agents on the World Wide Web 2(1) (2011)
13. Demers, A., Gehrke, J., Panda, B., Riedewald, M., Sharma, V., White, W.M., et al.: Cayuga: A General Purpose Event Monitoring System. In: CIDR (2007)
14. Herzberg, N., Meyer, A., Weske, M.: An Event Processing Platform for Business Process Management. In: Proc. of the 17th IEEE International EDOC Conference (2013)
15. Crapo, A., Wang, X., Lizzi, J., Larson, R.: The Semantically Enabled Smart Grid. In: Proc. of the Grid-Interop Forum, vol. 2009, pp. 177–185 (2009)
16. Barbieri, D.F., Braga, D., Ceri, S., Grossniklaus, M.: An Execution Environment for C-SPARQL Queries. In: Proc. of the 13th International Conference on Extending Database Technology, pp. 441–452. ACM (2010)
17. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: EP-SPARQL: A Unified Language for Event Processing and Stream Reasoning. In: Proc. of the 20th International Conference on World Wide Web, pp. 635–644. ACM (2011)