C-SWRL: SWRL for Reasoning over Stream Data

Edmond Jajaga
Department of Computer Science
South East European University
Tetovë, Macedonia
e.jajaga@seeu.edu.mk

Lule Ahmedi
Department of Computer Engineering
University of Prishtina
Prishtinë, Kosova
lule.ahmedi@uni-pr.edu

***Abstract*—Semantic technologies have been extensively used for integrating stream data applications. However, using SWRL, which has become the de facto standard rule language in Semantic Web, has never been used in stream data applications. Its open world assumption and monotonic nature makes SWRL powerless for doing continuous inference over stream data. For example, using aggregate functions on a particular window of streams cannot be expressed in SWRL. C-SPARQL is a framework which supports continuous querying over data streams. We introduce here C-SWRL, a unified Semantic Web stream reasoning system that further supports continuous reasoning over stream data. C-SWRL utilizes C-SPARQL filtering and aggregation of RDF streams to enable closed-world and time-aware reasoning with SWRL rules. Moreover, the non-monotonic behavior is supported with the use of OWLAPI constructs. The system is presented by means of examples in water quality monitoring.**

Keywords-stream data; Semantic Web; reasoning; SWRL; rules

I. INTRODUCTION

Sensor measurements, social networks, health monitoring, smart cities and other massive data sources are continuously producing massive amount of data called stream data. Stream data are defined as unbounded sequences of time-varying data elements [6]. Reasoning with these kinds of data with Semantic Web techniques has eventually contributed in a new research area called Stream Reasoning (SR). The aim to derive high level knowledge from low level data streams is one of the challenging requirements which cannot be easily satisfied with the classic solutions for data stream and complex event processing and with reasoning engines for static data [23]. The W3C RDF Stream Processing Community Group has set their mission to define common model for producing, transmitting and continuously querying Resource Description Framework (RDF) Streams. RDF streams are a sequence of RDF triples that are continuously produced and annotated with a timestamp [11]. However, even though different works exist (e. g. ETALIS [16], StreamRule [14] etc.), rule-based reasoning over RDF streams still remains vastly unexplored.

This paper proposes a unified Semantic Web approach for rule-based reasoning over stream data complementing state of the art query processing engine C-SPARQL [11] with the W3C proposed Semantic Web rule language i.e. the Semantic Web Rule Language (SWRL).

Semantic technologies have proved evidence of efficient implementations on stream data domains [1]. Firstly, the Web Ontology Language (OWL) has been widely used for modeling stream data domains, e.g., the SSN ontology [36]. Secondly, querying these knowledge bases has been merely done by SPARQL extensions e.g. C-SPARQL [11], EP-SPARQL [12], etc. Although layering different rule systems over ontologies has already been suggested [3], using Semantic Web rule languages, SWRL [5] and the Rule Interchange Format (RIF) [25], over stream data has to the best of our knowledge not been considered to date. Thus, as described in our previous works [1-4], there is an inherent need for a Semantic Web unified rule system capable of reasoning with stream data. In line with this vision, we have previously developed the INWATERSENSE (hereinafter referred to as INWS) ontology [2], an expert system [3] demonstrating its usage and StreamJess [4], a production rules system for stream reasoning. In this paper, we describe Continuous SWRL (or simply C-SWRL), a SWRL system for reasoning with stream data. It utilizes C-SPARQL definition of RDF streams and windows that further supports non-monotonic and time-aware reasoning on stream data.

The system was validated with simulated data in the water quality monitoring (WQM) domain, but it is developed for use within the InWaterSense project with real data. InWaterSense is an EU funded research project aimed to apply recent advanced practices stemming from ICT in WQM for healthy environment, and strengthen Kosovo's capacity in research in national priority sectors of environment and ICT. An intelligent wireless sensor network (WSN) for monitoring surface water quality has been deployed in a river in Kosova and is further being enriched with more intelligent behavior like is the contribution presented in this paper.

The paper is organized as follows. Section 2 describes C-SWRL prototypical design and implementation. System validation is presented in Section 3 through examples in the domain of WQM. Related works take place on Section 4. Finally, the paper closes with conclusion and future plans.

II. C-SWRL

As depicted in Figure 1, C-SWRL uses C-SPARQL output data as input for SWRL to infer and assert new knowledge to ontologies. Firstly, sensor provided RDF streams filtering and aggregation is done by C-SPARQL. Secondly, based on C-SPARQL output data, OWLAPI [27] constructs are invoked for asserting new OWL individuals in a temporary class holding all observation's information. Finally, these individuals are processed by SWRL rules loaded at application startup. These rules mainly fall into two broad categories:

- *monitoring rules*, rules for continuous classification of water bodies based on in situ observations, and
- *investigation rules*, which fire after monitoring rules detect any critical status. The information of sources of pollution stored into the pollutants ontology is used to prejudge the causer of the pollution.

In another domain, say medicine, the monitoring rules may continually classify the human's health status, while the investigation ones may try to identify the potential sources of the disease in cases of critical status detection.

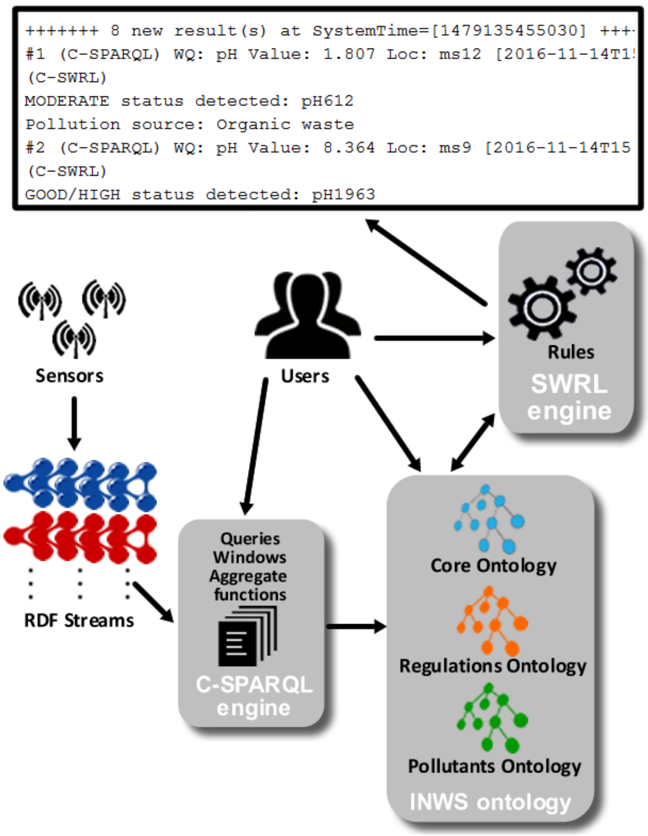

Figure 1. C-SWRL conceptual architecture

A. Input data

Input data feeding the system are two folds: domain-specific and stream data. Static or "slowly" changing data include description facts of a specific domain e.g. river names, measurement sites, sensor devices etc. Stream data in C-SWRL are sensor provided data formatted as RDF streams. Sensors continually transmit observational data including: observed parameter name and value and observation location and time.

B. C-SPARQL

RDF streams feed the C-SPARQL query engine. Based on the registered queries, the engine will output the results, which in turn will be published on the knowledge base. In general, triples of values are produced: the water quality name, the location of measurements and the calculated average value. According to these values, new instances will be added in the INWS ontology, described in the next sub-section.

C. INWS ontology

The INWS ontology [2] consists of three modules: core, regulations and pollutants ontology. The core module includes description of observation entities. The regulation ontology models different regulation authorities' standards for monitoring systems e.g. the Water Framework Directive (WFD), which represents EU's framework for WQM. The pollutants module models pollutants entities and the sources of pollution.

D. SWRL engine

After processing each window the knowledge base becomes modified and thus new inferences should be made. In order to enable SWRL to reason over stream data three approaches were considered:

- Extending SWRL with stream data reasoning features,
- Translating SWRL to another rule system which supports stream data reasoning and
- Layering SWRL on top of another system to fill the gaps of SWRL in support of stream data reasoning.

Extending SWRL with stream data reasoning features is very expensive since neither of non-monotonicity, closed-world or time-aware reasoning are supported. State-of-the-art SWRL extensions may support one, but fail on another feature. For example, JNOMO [20] is a SWRL extension for enabling non-monotonic reasoning, but it does not support time-aware reasoning. JNOMO [20] is also an example of translating SWRL into Jena [7]. These kind of approaches do not deal with the different nature of stream data and they also have the potential of losing information while translating the constructs.

Given the drawbacks if approaching any of the previous two options, it was decided to layer SWRL over an existing SR system such as C-SPARQL. C-SPARQL is specifically designed for stream data applications. It supports closed-world and time-aware reasoning on stream data. However, as a query language, it is not intended to have any effect on the underlying ontology.

In C-SWRL, SWRL reasoning is implemented with SWRLAPI [29] methods. Registered monitoring rules detect the newly published observation data and classify the observation into appropriate status based on WFD standards e.g. good, high or moderate. Whenever a moderate status becomes detected the investigation rules fire to assert the polluted site and potential sources of pollution. Since this process is continuous and iterative, to avoid reasserting of individuals into appropriate classes, the temporary observation class needs to be cleared at each window processing. This was done by using the OWLAPI's `removeAxiom` construct. The same construct was used to enable system's non-monotonic behavior. Namely, SWRL's ability to assert new information in conjunction with OWLAPI's one to remove information enables the modification of the measurement site's pollution status. At each window processing, which processes an observation on a particular measurement site, the last known pollution status gets removed from the knowledge base (by OWLAPI constructs) and a new status is inferred based on the SWRL rules. In particular, this was managed through the object property `isPolluted` relating measurement sites with one of the instances true or false. Thus, one can query for measurement sites' state at any time of C-SWRL running application. Moreover, every time a measurement site gets polluted a new instance of the class `PollutedSite` is asserted related with time and pollutants information.

C-SWRL is implemented in Java following the availability of Java codes of C-SPARQL, OWLAPI and SWRLAPI. The system is open for loading different SSN-based domain ontologies, write appropriate C-SPARQL queries and SWRL rules. Moreover, instead of C-SPARQL and SWRL, with less effort different SPARQL-like query processing engines coupled with different rule languages can be integrated, respectively. The system is publicly available on http://streamreasoning.uni-pr.edu/. The link contains the source code, installation instructions and getting started tutorial.

III. VALIDATION

C-SWRL is validated in a typical water quality monitoring scenario based on WSN. We assume that sensors are deployed in different measurement sites at different times. They continually emit water quality measured values. C-SWRL will (1) classify the water body into the appropriate status according to WFD regulations [39, 15] and (2) identify the potential sources of pollution if the values are out of the allowed standard. The system was validated against a number of water quality parameters, but for brevity, we will demonstrate the cases of Biochemical Oxygen Demand (BOD_5) observations. Like most of water quality parameter observations, BOD_5 observations are classified based on the average value of measurements within a time interval, except pH ones which are considered one by one [15]. The validation examples run at the same time over the same RDF streams which are filtered out by different C-SPARQL queries. RDF streams generator runs in background simulating sensor measurements on, arbitrarily set, three measurement sites: ms10, ms11 and ms12. BOD_5 measurements appear on ms10 and ms11. The streaming rate is arbitrarily set to one stream per second. A single RDF stream holds information of the measured value, water quality name, observation time and location and the device providing the observation. Figure 2 illustrates a screenshot of the C-SWRL console output of the running example.

A WFD rule for classifying BOD_5 observations looks as follows: If BOD_5 measurements in mg O2/l is less than 1.3 (mean), then river belongs to "high" status of oxygen condition; if it is less than 1.5 then river belongs to "good" status of oxygen condition; otherwise the river belongs to "moderate" status of oxygen condition [15]. Potential sources of pollution from which BOD_5 discharges could arise include: contaminated land, farm wastes and silage, fish farming, effluent discharges from sewage treatment works, landfill sites and urban storm water discharges [10].

After processing the first window of RDF streams a new BOD_5 average value is calculated by the following C-SPARQL query:

```
REGISTER STREAM AvgObservations AS
PREFIX inwsc: <http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#>
PREFIX ssn: <http://purl.oclc.org/NET/ssnx/ssn#>
PREFIX dul: <http://www.loa-cnr.it/ontologies/DUL.owl#>
SELECT ?qo ?loc (AVG(?dv) AS ?avg)
FROM STREAM <http://inwatersense.uni-pr.edu/stream> [RANGE 20s STEP 20s]
FROM <http://inwatersense.uni-pr.edu/ontologies/inws-core.owl>
WHERE {
?o ssn:qualityOfObservation ?qo .
?o ssn:observationResult ?r .
?r ssn:hasValue ?v .
?v dul:hasDataValue ?dv .
  ?o inwsc:observationResultLocation ?loc .
  FILTER (?qo != inwsc:pH)
}
GROUP BY ?qo ?loc
```

```
+++++++ 2 new result(s) at SystemTime=[1470774422368] +++++++
#1 (C-SPARQL) WQ: BOD Value:1.503 Loc: ms11 [2016-08-09T22:27:02]
(C-SWRL) MODERATE status detected: BOD4595
Pollution source: Urban stormwater discharges
Pollution source: Landfill sites
#2 (C-SPARQL) WQ: BOD Value:1.312 Loc: ms10 [2016-08-09T22:27:03]
(C-SWRL) GOOD status detected: BOD3936

ms10 is CLEAN
ms11 is POLLUTED
```

Figure 2. An output excerpt of BOD_5 monitoring on C-SWRL

The query runs against the input RDF streams in the time frame of 20 seconds, sliding the window by 20 seconds. The chosen time frame is arbitrary and the user can change its values as desired. It produces triples of values: the water quality name (`?qo`), the location of measurements (`?loc`) and the calculated average value (`?avg`). The triples are filtered out to exclude pH ones and are firstly grouped by the water quality name and then by the measurement site. At every 20 seconds new RDF streams enter into the window and old ones exit. An output of a window processing of this query is depicted in the lower part of Figure 2, namely on the lines marked with '#' symbol followed by an order number and (C-SPARQL) label. Namely, C-SPARQL has outputted two results.

At every query execution, for each new triple (`?qo`, `?loc`, `?avg`), a new individual of a temporary INWS class `tmpObservation` is asserted into the ontology using OWLAPI constructs. This individual indicates a new input observation has arrived. Following the INWS ontology design this individual is associated through:

- `ssn:qualityOfObservation` with the water quality parameter name i.e. `?qo`,
- `observationResultLocation` property with `?loc`,
- `ssn:observationResult` with new `ssn:SensorOutput` instance, which in turn is related with a new `ssn:ObservationValue` instance through `ssn:hasValue` property, which finally is associated with the observation's average value `?avg` through `dul:hasDataValue`.
- `ssn:observationResultTime` with the system's timestamp

Next, the SWRL rule engine is executed firing the registered SWRL monitoring rules. These rules include the following ones for BOD_5 WFD classification (user-defined prefixes are omitted for brevity):

```
1. tmpObservation (?x) ^
qualityOfObservation(?x,BiochemicalOxygenDemand
) ^ observationResult(?x, ?y) ^ hasValue(?y,
?e) ^ hasDataValue(?e,?z) ^
swrlb:greaterThan(?z, 1.3) ^ swrlb:lessThan(?z,
1.5) -> GoodBODMeasurement(?x) ^
tmpGoodBODMeasurement(?x) ^ isPolluted(?ms,
false) ^ Observation(?x)
```

```
2. tmpObservation (?x) ^
qualityOfObservation(?x,BiochemicalOxygenDemand
) ^ observationResult(?x, ?y) ^ hasValue(?y,
?e) ^ hasDataValue(?e,?z) ^ swrlb:lessThan(?z,
1.3) -> HighBODMeasurement(?x) ^
tmpHighBODMeasurement(?x) ^ isPolluted(?ms,
false) ^ Observation(?x)
```

```
3. tmpObservation (?x) ^
qualityOfObservation(?x,
BiochemicalOxygenDemand) ^
observationResult(?x, ?y) ^ hasValue(?y, ?e) ^
hasDataValue(?e,?z) ^ swrlb:greaterThan(?z,
1.5) -> ModerateBODMeasurement(?x) ^
tmpModerateBODMeasurement(?x) ^ isPolluted(?ms,
true) ^ Observation(?x)
```

Figure 3. An output excerpt of the BOD_5 monitoring on C-SWRL

The first rule matches the individuals (`?x`) of the temporary class related to BOD_5 measurements and checks its average value. If it is between 1.3 and 1.5 then the status is "good" i.e. the individual is asserted as of type `GoodBODMeasurement`. The same matching is done with the second and third rule respectively. For the second one the average value is checked to be lower than 1.3 for its classification. If so, the status is "high" i.e. the individual is asserted as of type `HighBODMeasurement`. In the third rule the average value is checked to be greater than 1.5 for classifying in "moderate" status i.e. class `ModerateBODMeasurement`. A temporary class `tmpModerateBODMeasurement` is used for investigation of sources of pollution. In the first and second rule the respective temporary classes are used for displaying the calculated status to the user interface. In each RHS of the rules the temporary observation individual gets stored in the class Observation as per historical data records. Moreover, the `isPolluted` object property is used to maintain the current state of the measurement site. It is set to 'false' in the cases of "good" and "high" statuses while it is set to 'true' when detecting "moderate" status. In the running example the firing of rules has produced one "moderate" and one "good" status, as illustrated in the lower part of Figure 2 i.e. the lines starting with (C-SWRL) label followed by the detected status information. Since, the first C-SPARQL calculated average value is 1.503 which is greater than 1.5 the third rule has fired asserting new individuals in `ModerateBODMeasurement` and `tmpModerateBODMeasurement`.

New individual in the class `tmpModerateBODMeasurement` will cause to fire the following investigation rule, which is also registered at application startup:

```
4. tmpModerateBODMeasurement(?x) ^
observationResultTime(?x, ?t) ^
observationResultLocation(?x, ?ms) ^ has-
SourcesOfPollution(?ms, ?pollsrc) ^
potentialPollutant(?pollsrc,
BiochemicalOxygenDemand) ->
foundPollutionSources(?x, ?pollsrc)
```

This rule binds the "moderate" status observations (`?x`) with measurement site's (`?ms`) nearby BOD_5 sources of pollution (`?pollsrc`) extracted from the knowledge base. The observations (`?x`) satisfying the LHS clauses will become related with the matching pollution sources. These results will be displayed to the user interface right after the "moderate" status detection like is shown on the first C-SPARQL result in Figure 2. From the Figure we can observe that the potential sources of pollution caused on ms11 are "urban storm water discharges" and "landfill sites".

At the end of each window processing and reasoning, the current status of the sites are queried and printed out. On Figure 2, we can observe that the last statuses for measurement sites ms10 and ms11 are "clean" and "polluted", respectively.

The monitoring and investigation rules for all the water quality parameters are the same as the ones for BOD_5, described previously. Of course, the threshold values are different according to WFD regulations. The query `AvgObservations` will match all the water quality observations, except pH ones. For pH observations new query similar to `AvgObservations` was written. Namely, no aggregation function is used in the SELECT statement and thus no grouping is needed. The FILTER clause uses the equal symbol rather than the unequal one.

IV. RELATED WORKS

State-of-the-art rule-based systems for reasoning over stream data mainly fall into two broad categories: hybrid and homogeneous approaches [1]. In the former one the reasoning is done by interfacing existing rule reasoner with existing ontology reasoner, while in the latter one both ontologies and rules are embedded into the same logical language without making a priori distinction between the rule predicates and the ontology predicates [13].

A. *Hybrid approaches*

Hybrid approaches layer different non-DL rule systems on top of ontologies like: production rules, CEP, LP, answer set programming (ASP), etc. In our previous work [1], we described in more detail about each one of these approaches and their pros and cons. In general, hybrid solutions have achieved the desired system behavior. However, these approaches mainly suffer from translation and reasoner issues and potential side-effects occurrence. In these approaches, the ontology is translated into the corresponding formalisms of the underlying rule system. A drawback of this translation is that a possible loss of information may occur. Since the ontology and the rules are treated separately then a rule engine and a DL reasoner will run concurrently [9]. As argued in [9], some inferences would no longer be derived after separating OWL and rules. Furthermore, when adding a new rule a possible side-effect may occur.

A similar approach to C-SWRL is followed by StreamRule [14], the pioneer of coupling stream processing with ASP non-monotonic reasoning. Even though the approach is still much more prototypical it demonstrates how non-monotonic and time-aware reasoning can be integrated into a unique platform for stream data reasoning. The continuous rule feature is implemented through separate steps. Namely, stream filtering and aggregation is done through a stream query processor such as CQELS [31], while OClingo [32] is used to enable non-monotonic reasoning. In C-SWRL we use C-SPARQL for filtering and aggregation purposes, and OWLAPI for non-monotonic reasoning. Even though that CQELS outperforms C-SPARQL [38], we preferred C-SPARQL following its advantage to use nested aggregations and negation [37, 38]. Moreover, we plan to support temporal operators, which lack any support in CQELS [37]. Another feature difference between StreamRule and C-SWRL is the historical data management, which is one of the key requirements of SR tools [8]. C-SWRL keeps evidence of every previous environment state. For example, one can query the ontology for a particular measurement site's pollution status of the past. OClingo feeds back the reasoning results into Java runtime for further processing or display, while in C-SWRL, the results are deployed back into the knowledge base through the OWLAPI's `saveOntology` function and thus the memory gets released and the data are available for query and retrieval.

Recently, [21] proposed another non-monotonic ASP-based SR system, which provides support for C-SPARQL query engine. The system supports reasoning even in incomplete information cases through negation as failure feature, but like StreamRule it does not support historical data management. Moreover, the reasoning results are returned as JSON objects to the corresponding web socket clients, while in C-SWRL the reasoning results are returned as standard RDF data populating corresponding ontology classes.

ETALIS [16] together with EP-SPARQL [12] enables CEP with stream reasoning. Even though ETALIS offers reasoning on time and location spaces it does not implement the windows feature. Time-based windows are supported through its wrapper EP-SPARQL, but complicated aggregations within windows are not supported [38]. Moreover, there is no support for triple-based windows too.

B. *Semantic Web approaches*

In the literature this approach is also referred to as interaction of ontologies and rules with tight semantic integration [13]. Even though using SWRL with OWL has distinct advantages, these approaches mainly suffer from limited expressiveness or undecidability [13]. In C-SWRL, the required expressivity is extended by C-SPARQL and OWLAPI functions. Additionally, works described in [17], [18] and [19] prove that decidability can be retained by the so-called DL-safe rules. For example, retaining decidability in [17] is done through restricting the interface between OWL and rules.

State of the art homogeny approaches, like the ones described in [33, 34], do not make any distinction between stream and static data, while also lack implementation. They prove that SWRL can be used to infer new and approximate knowledge in stream data domains. However, their approach does not consider time-aware and non-monotonic reasoning. Recently, a SPARQL extension [24] that uses CONSTRUCT/WHERE clauses to express rules has been proposed. Yet again this approach does not consider non-monotonic reasoning. The works presented in [30, 35] describe a Rete-based [28] approach of RDFS entailment rules for producing data in a continuous manner. Although supporting time-aware and incremental reasoning, the approach does not deal with non-monotonic and closed-world reasoning. JNOMO [20] shows how SWRL can be extended to embrace non-monotonicity, CWA and NAF. However, it does not deal with stream data, while inclusion of temporal reasoning is envisioned as per future works.

V. CONCLUSION AND FUTURE WORKS

Until recently most of the SR research has been dedicated on ontology and query processing developments. Dealing with stream reasoning issues through query processing is not enough. Our work goes beyond the query processing achievements and thus focusing on rule level implications of stream data. SWRL, on its own, lacks the required expressivity level to reason over stream data. The main contribution of this paper is in establishing a unique Semantic Web rule system capable for expressive reasoning over stream data. In this vision, we developed C-SWRL which layers SWRL on top of C-SPARQL to enable time-aware, closed-world and non-monotonic reasoning on stream data domains. For non-monotonic reasoning purposes, C-SWRL uses SWRL together with OWLAPI constructs to modify the knowledge base.

We are currently evaluating the examples presented here in Drools, for which we shall conduct a thorough performance evaluation and thus analyze the scalability issues. Our initial findings show that evaluating C-SWRL proves difficult due to the nature of our system, code availability of related systems and published evaluation results. Regarding the stream processing level it has been discovered that C-SPARQL yields considerably lower through-put compared to JTALIS and CQELS [22]. Thus, our main evaluation concern remains the stream reasoning component. We agree with Barbieri et al. [26] urgency for development of specialized reasoners for stream data applications. We also plan to evaluate C-SWRL against our previously developed Jess system, StreamJess [4].

REFERENCES

[1] E. Jajaga, L. Ahmedi and L. Abazi-Bexheti, Semantic Web trends on reasoning over sensor data, in: 8th South East European Doctoral Student Conference, Greece, 2013.

[2] L. Ahmedi, E.Jajaga and F. Ahmedi, An ontology framework for water quality management, in: Ó. Corcho, C. A. Henson, and P. M. Barnaghi, ed., SSN@ISWC, Sydney, 2013, pp. 35-50.

[3] E. Jajaga and L. Ahmedi, An expert system for water quality monitoring based on ontology, in: 9th Metadata and Semantics Research Conference, Manchester, UK, September 9-11, 2015, pp. 89-100.

[4] E. Jajaga, L. Ahmedi and F. Ahmedi: StreamJess: a stream reasoning framework for water quality monitoring. J. of Metadata, Semantics and Ontologies, in press.

[5] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, SWRL: A Semantic Web rule language combining OWL and RuleML, 2004.

[6] E. Della Valle, S. Ceri, D. F. Barbieri, Daniel Braga, and A. Campi, A first step towards stream reasoning, in: Proc. Future Internet Symposium (FIS 08), Springer, 2008, pp. 72–81.

[7] B. McBride, Jena: implementing the RDF model and syntax specification, in: Proc. at Semantic Web Workshop (WWW), 2004.

[8] A. Margara, J. Urbani, F. van Harmelen and H. Bal, Streaming the web: reasoning over dynamic data, Web Semantics: Science, Services and Agents on the World Wide Web, 25(0), (2014), 24 – 44.

[9] J. Mei, E. P. Bontas: Reasoning Paradigms for SWRL-Enabled Ontologies Protégé With Rules, in: Workshop held at the 8th International Protégé Conference, Madrid, Spain, 2005.

[10] Sources of Pollution, Foundation for Water Research, Information Note FWR-WFD16, 2005.

[11] D. F. Barbieri, D. Braga, S. Ceri, E. Della Valle and M. Grossniklaus, C-SPARQL: a continuous query language for RDF data streams, International Journal of Semantic Computing 04-01 (2010), 3–25.

[12] D. Anicic, P. Fodor, S. Rudolph and N. Stojanovic, EP-SPARQL: a unified language for event processing and stream reasoning, in: WWW 2011, 2011, pp. 635–644.

[13] T. Eiter, G. Ianni, A. Polleres, R. Schindlauer and H. Tompits: Reasoning with rules and ontologies, in: P. Barahona, F. Bry, E. Franconi, N. Henze, U. Sattler (Eds.), Reasoning Web, Second International Summer School 2006, Tutorial Lectures, LNCS, vol. 4126, Springer, pp. 93–127, 2006.

[14] A. Mileo, A. Abdelrahman, S. Policarpio and M. Hauswirth, StreamRule: a nonmonotonic stream reasoning system for the semantic web, in: W. Faber, D. Lembo, ed., RR 2013, LNCS, Springer, Heidelberg, 2013, vol. 7994, pp. 247–252.

[15] European Communities Environmental Objectives (Surface Waters) Regulations, 2009.

[16] D. Anicic, P. Fodor, S. Rudolph, R. Stuhmer, N. Stojanovic, R. Studer: A Rule-Based Language for Complex Event Processing Reasoning, in: Proceedings of the Fourth International Conference on Web reasoning and rule systems, pp. 42-57, Springer-Verlag Berlin, Heidelberg, 2010.

[17] B. Motik, U. Sattler and R. Studer: Query Answering for OWL-DL with rules, Journal of Web Semantics, 3(1), pp. 41–60, 2005.

[18] F. M. Donini, M. Lenzerini, D. Nardi and A. Schaerf: AL-log: Integrating Datalog and Description Logics, J. of Intelligent Information Systems, 10(3), pp. 227–252, 1998.

[19] S. Heymans, D. V. Nieuwenborgh and D. Vermeir: Nonmonotonic Ontological and Rule-Based Reasoning with Extended Conceptual Logic Programs, in: Proc. Second European Semantic Web Conference (ESWC 2005), vol. 3532, pp. 392–407, Springer Verlag, 2005.

[20] J. M. A. Calero, A. M. Ortega, G. M. Perez, J. A. B. Blaya and A. F. G. Skarmeta, A non-monotonic expressiveness extension on the semantic web rule language, J. Web Eng., 11(2), pp. 93–118, 2012.

[21] M. I. Ali, N. Ono, M. Kaysar, Z. U. Shamszaman, T.-L. Pham, F. Gao, K. Griffin and A. Mileo: Real-time Data Analytics and Event Detection for IoT-enabled Communication Systems, J. of Web Semantics: Science, Services and Agents on the World Wide Web, July, 2016.

[22] D. Le-Phuoc, M. Dao-Tran, M.-D. Pham, P. Boncz, T. Eiter and M. Fink: Linked stream data processing engines: facts and figures, in: The Semantic Web – ISWC 2012. Springer, pp. 300–312, 2012.

[23] E. Della Valle, D. Dell'Aglio, A. Margara: Tutorial: Taming Velocity and Variety Simultaneously in Big Data with Stream Reasoning, in: The 10th ACM International Conference on Distributed and Event-Based Systems, Irvine, USA, June 20-24, 2016.

[24] J. Anderson, T. Athan and A. Paschke: Rules and RDF Streams - A Position Paper, in: Proceedings of the RuleML 2016 Challenge, Doctoral Consortium and Industry Track hosted by the 10th International Web Rule Symposium (RuleML 2016), New York, USA, July 6-9, 2016.

[25] H. Boley, M.Kifer, P.-L. Patranjan, and A. Polleres, Rule interchange on the web, in: G. Antoniou, U. Aßmann, C. Baroglio, S. Decker, N. Henze, P.-L. Patranjan, R. Tolksdorf, ed., Reasoning Web, LNCS, Springer, Heidelberg, 2007, vol. 4636, pp. 269–309.

[26] D. Barbieri, D. Braga, S. Ceri, E. Della Valle, M. Grossniklaus: Stream Reasoning: Where We Got So Far, in: Proceedings of the 4th International Workshop on New Forms of Reasoning for the Semantic Web: Scalable and Dynamic (NeFoRS), 2010.

[27] M. Horridge and S. Bechhofer, The OWL API: A Java API for working with OWL 2 ontologies, in: OWLED 2009, 6th OWL Experienced and Directions Workshop, Chantilly, Virginia, 2009.

[28] C. L. Forgy, Rete: A fast algorithm for the many pattern/many object pattern match problem, Artificial Intelligence 19 (1) (1982), 17 – 37.

[29] M. J. O'Connor, H. Knublauch, S. W. Tu, B. Grossof, M. Dean, W. E. Grosso, and M. A. Musen, Supporting rule system interoperability on the Semantic Web with SWRL, in: 4th International Semantic Web Conference (ISWC), Galway, Ireland, Springer Verlag, LNCS 3729, 2005, pp. 974-986.

[30] S. Tallevi-Diotallevi, S. Kotoulas, L. Foschini, F. Lecue and A. Corradi, Real-time urban monitoring in Dublin using semantic and stream technologies, in: The Semantic Web ISWC 2013, H. Alani, L. Kagal, A. Fokoue, P. Groth, C. Biemann, J. Parreira, L. Aroyo, N. Noy, C. Welty and K. Janowicz, ed., vol. 8219 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 178–194.

[31] D. Le-Phuoc, M. Dao-Tran, J. Xavier Parreira and M. Hauswirth, A native and adaptive approach for unified processing of linked streams and linked data, in: The Semantic Web–ISWC 2011, 2011, pp. 370–388.

[32] M. Gebser, T. Grote, R. Kaminski, P. Obermeier, O. Sabuncu, and T. Schaub, Answer set programming for stream reasoning, in: CoRR, 2013.

[33] W. Wei and P. Barnaghi, Semantic annotation and reasoning for sensor data, in: Smart Sensing and Context, 2009, pp.66-76.

[34] C. Keßler, M. Raubal and C, Wosniok, Semantic rules for context-aware geographical information retrieval, in: P. Barnaghi, ed., European Conference on Smart Sensing and Context, EuroSSC 2009, LNCS, Springer, 2009, vol. 5741, pp. 77–92.

[35] R. Albeladi, K. Martinez and N. Gibbins, Incremental rule-based reasoning over RDF streams: An expression of interest, in: RDF Stream Processing Workshop at the 12th Extended Semantic Web Conference, Portoroz, Slovenia, 2015.

[36] M. Compton, P. Barnaghi, L Bermudez, R. GarcíaCastro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. A. Henson, A. Herzog, V. A. Huang, K. Janowicz, W. D. Kelsey, D. L. Phuoc, L. Lefort, M. Leggieri, H. Neuhaus, A. Nikolov, K. R. Page, A. Passant, A. P. Sheth and K. Taylor, The SSN ontology of the W3C semantic sensor network incubator group, Journal of Web Semantics 17 (2012), 25–32.

[37] N. Lanzanasto, S. Komazec and I. Toma, Deliverable D4.8: Reasoning over real time data streams, ENVISION Consortium 2009-2012.

[38] D. Le-Phuoc, M. Dao-Tran, M.-D. Pham, P. Boncz, T. Eiter, and M. Fink, Linked stream data processing engines: facts and figures, in: The Semantic Web–ISWC 2012, Springer, 2012, pp. 300–312.

[39] Method statement for the classification of surface water bodies, v2.0 (external release), Monitoring Strategy v2.0, July 2011.