User Manual

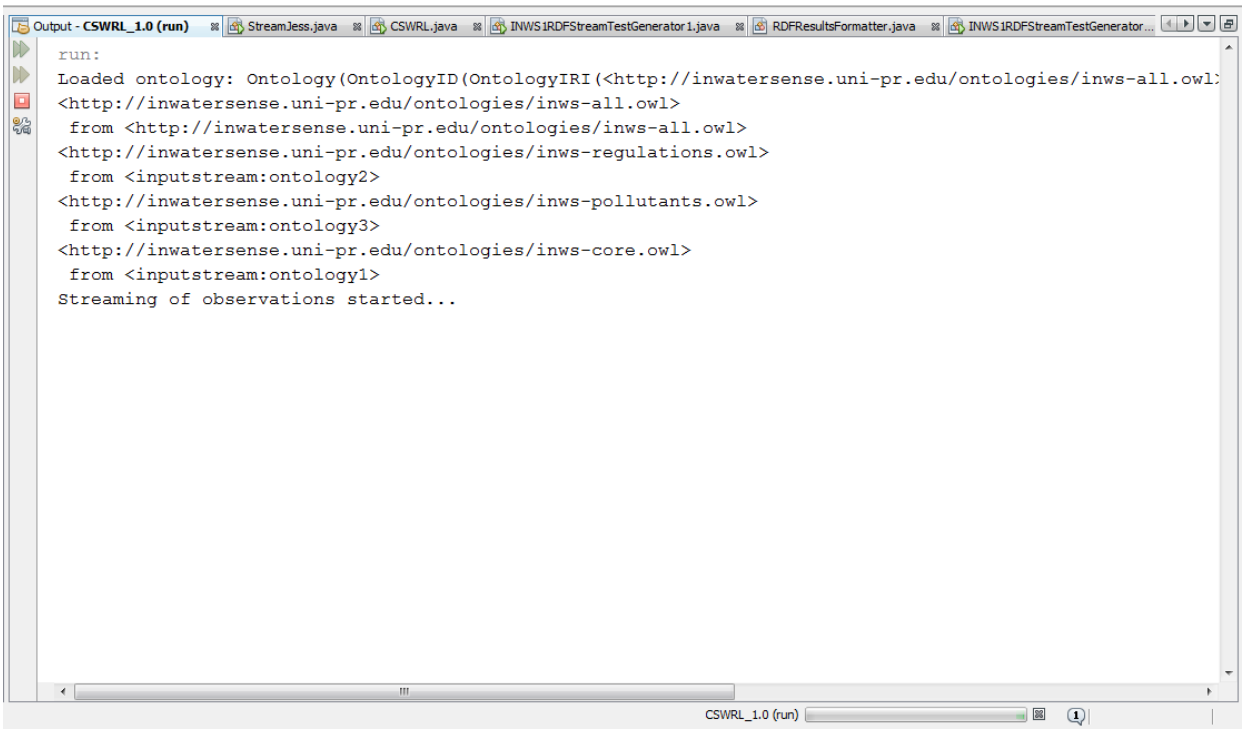
C-SWRL v0.1

Table of Contents

I. Installation

Download application's source distribution from the Download section. Unzip the zip file into your local folder. Import the project into your NetBeans. Download and import all the jar libraries into your project including: C-SPARQL[1] v0.9.6, OWL API[2] v4.0.2, SWRLTab[3] v1.0, SWRL API Drools Engine[4] v1.0 and JUnit[5] v4.10.

Open *main\StreamJess.java* and replace the InWaterSense ontologies paths with your local copies of the ontologies (modify the values of variables: *coreOntoSystemPath*, *regulOntoSystemPath* and *pollOntoSystemPath*). Run the application. If everything was correctly set the following screenshot should appear:



II. Getting started

2.1. Dataset and Stream Generator default settings

The default stream generator (*INWSRDFStreamTestGenerator.java*) produces sensor simulated values on 70 different measurement sites for 11 water quality parameters. Our implemented WQM example includes WFD monitoring of the following water quality parameters: Biochemical Oxygen Demand (BOD_5), pH, Arsenic, Benthic Invertebrate Fauna, Chromium III, "Cyanide", Diazinon, Dimethoate,

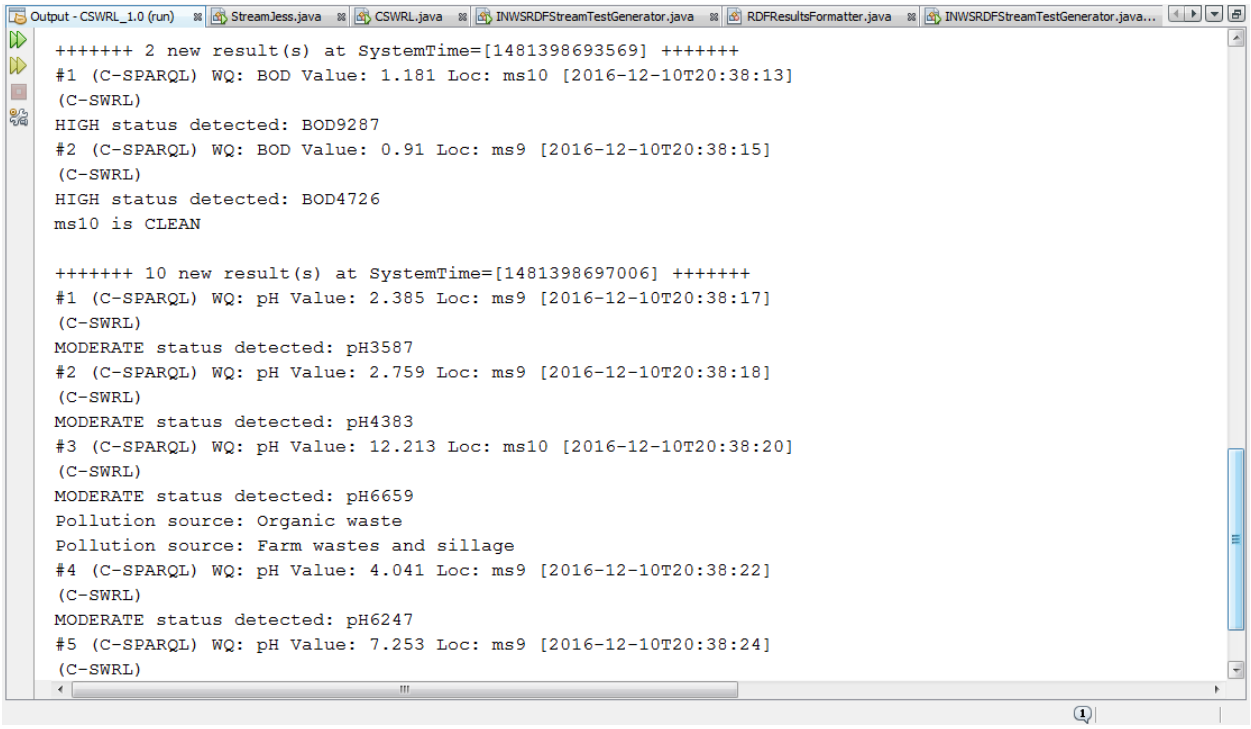
[1] http://streamreasoning.org/resources/c-sparql
[2] http://owlapi.sourceforge.net/
[3] https://github.com/protegeproject/swrltab
[4] https://github.com/protegeproject/swrlapi-drools-engine
[5] http://junit.org/junit4/

Fluoride, Total Ammonia and Glyphosate. Furthermore, it identifies the potential sources of pollution inducted by BOD_5 and pH moderate values. All values are randomly generated. Two C-SPARQL queries are encoded in the application's main file *StreamJess.java*. One for considering observations one by one, which according to Water Framework Directive (WFD) [1] is the case of pH observations and one for average value observations, which include all other water quality observations according to WFD.

After correctly loading all the ontology modules, the streaming of observations starts by generating new observation each second. Of course, this is a modifiable parameter which can be set by changing the milliseconds values in *Thread.sleep(1000).* Moreover, the user can also register multiple streamers, which can be enabled by setting the value of variable *NUM_STREAMS* in *CSWRL.java*.

2.2. SWRL reasoning

After C-SPARQL processes the first window results, the SWRL Drools engine acts to do the inference based on the registered rules. Based on C-SPARQL design principle each query attaches new result formatter implemented by the function *RDFResultsFormatter(ruleEngine, newOnto, reasoner, prefixManager, manager, owlDataFactory, queryEngine)*, where: *ruleEngine* is the instance of the Drools engine; *newOnto* is the OWL ontology instance; *reasoner* is the OWL reasoner; *prefixManager* is the ontology prefix manager; *manager* is an instance of *OWLOntologyManager*; *owlDataFactory* provides a reference to a data factory from an OWLOntologyManager; and *queryEngine* is the SQWRL query engine instance. For each new query result, firstly, the last pollution status of the current measurement site gets removed. Secondly, new observation information is published into the knowledge base. Finally, the Drools engine runs to do the rule-based reasoning. The rules' output gets printed out on the console as depicted in the following screenshot:



III. User-defined examples

This section contains information for users willing to write their own examples in C-SWRL.

3.1. Change the dataset

In case one would like to use different water quality monitoring data, it will be required to stop the streamer(s) and find appropriate middleware for importing real-time stream data and annotate them appropriately based on the InWaterSense ontology pattern. In cases of importing stream data annotated differently from InWaterSense ontology descriptions then he/she should provide their own model, write their own SWRL rules and potentially write appropriate Java codes. Of course, in these cases we prefer the users to contact us.

3.2. Change the model

If one would like to use other model than InWaterSense ontology, then he/she would have to change the variable paths value of InWaterSense ontology, configure appropriately the stream generator and write appropriate SWRL rules.

3.3. Change the SWRL rules

Using the InWaterSense model, users are open to add other SWRL rules for monitoring or finding potential sources of pollution. One can write a SWRL rule based on the following suggested format:

```
ruleEngine.createSWRLRule("rule_name", "rule_text");
```

`rule_name` is a user chosen name of the rule, while `rule_text` is the text of the rule written in SWRL syntax. Place this line right after creating the Drools engine, but not after the running of C-SPARQL engine.

References

[1] Directive 2000/60/EC of the European Parliament and of the Council of Europe of 23 October 2000 establishing a frame-work for Community action in the Field of water quality O.J. L327/1, 2000.