



User Manual

StreamJess v0.1

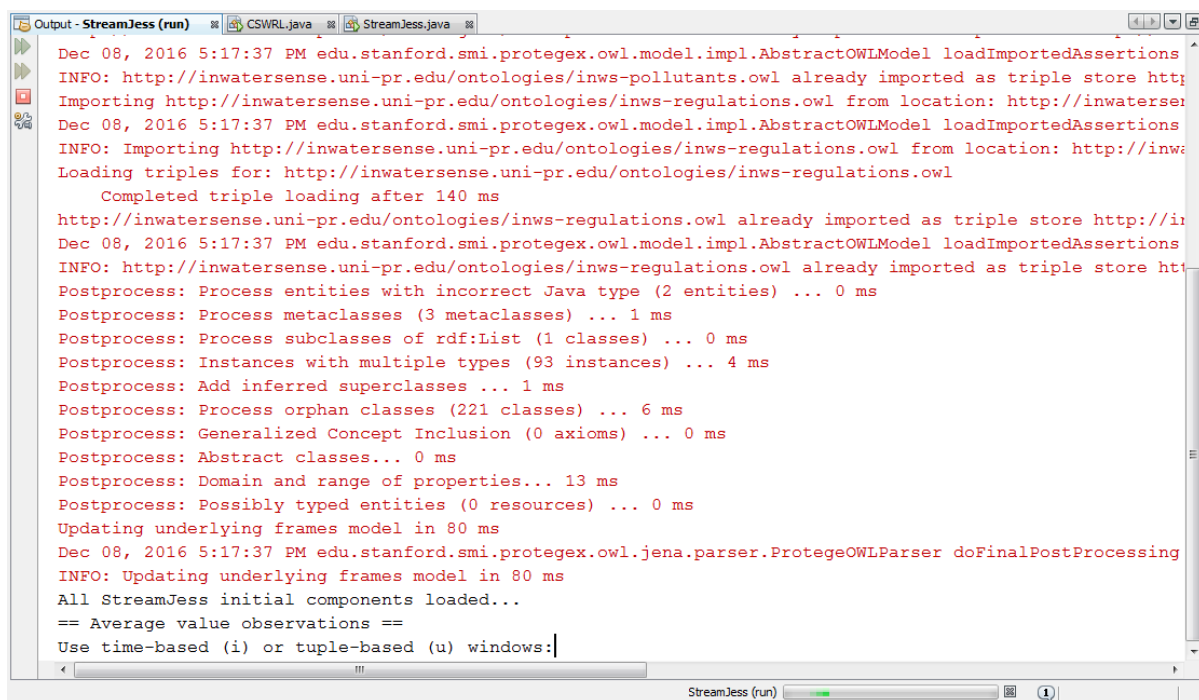
Table of Contents

I.	Installation	3
II.	Getting started examples.....	3
2.1.	Stream Generator default settings	3
2.2.	Jess reasoning	4
III.	User-defined examples	5
3.1.	Change data	5
3.2.	Change the model	5
3.3.	Change the Jess rules	5
	References	5

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 a copy of the InWaterSense ontology Protege project including all ontology modules from the Download section. Download and import all the jar libraries into your project including: C-SPARQL¹ v0.9.6, Jess² v7.1p2, Jess Tab³ v1.7 and Protégé⁴ v3.5.

Open *main\StreamJess.java* and replace the InWaterSense ontology project path with your local copy of the ontology (modify the value of *inwsOntologies* variable). Moreover, modify the path location of Jess rules to match the one of your local disc i.e. the *inwsRules* variable value. Make sure you are connected to the Internet for loading correctly the ontology modules and run the application. If everything was correctly loaded the following screenshot should appear:



```
Dec 08, 2016 5:17:37 PM edu.stanford.smi.protege.owl.model.impl.AbstractOWLModel loadImportedAssertions
INFO: http://inwatersense.uni-pr.edu/ontologies/inws-pollutants.owl already imported as triple store http://inwatersense.uni-pr.edu/ontologies/inws-pollutants.owl
Importing http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl from location: http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl
Dec 08, 2016 5:17:37 PM edu.stanford.smi.protege.owl.model.impl.AbstractOWLModel loadImportedAssertions
INFO: Importing http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl from location: http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl
Loading triples for: http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl
Completed triple loading after 140 ms
http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl already imported as triple store http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl
Dec 08, 2016 5:17:37 PM edu.stanford.smi.protege.owl.model.impl.AbstractOWLModel loadImportedAssertions
INFO: http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl already imported as triple store http://inwatersense.uni-pr.edu/ontologies/inws-regulations.owl
Postprocess: Process entities with incorrect Java type (2 entities) ... 0 ms
Postprocess: Process metaclasses (3 metaclasses) ... 1 ms
Postprocess: Process subclasses of rdf:List (1 classes) ... 0 ms
Postprocess: Instances with multiple types (93 instances) ... 4 ms
Postprocess: Add inferred superclasses ... 1 ms
Postprocess: Process orphan classes (221 classes) ... 6 ms
Postprocess: Generalized Concept Inclusion (0 axioms) ... 0 ms
Postprocess: Abstract classes... 0 ms
Postprocess: Domain and range of properties... 13 ms
Postprocess: Possibly typed entities (0 resources) ... 0 ms
Updating underlying frames model in 80 ms
Dec 08, 2016 5:17:37 PM edu.stanford.smi.protege.owl.jena.parser.ProtegeOWLParser doFinalPostProcessing
INFO: Updating underlying frames model in 80 ms
All StreamJess initial components loaded...
== Average value observations ==
Use time-based (i) or tuple-based (u) windows:
```

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 (BOD₅), pH, Arsenic, Benthic Invertebrate Fauna, Chromium III, "Cyanide", Diazinon, Dimethoate,

¹ <http://streamreasoning.org/resources/c-sparql>

² <http://www.jessrules.com/>

³ <http://www.jessrules.com/jesswiki/view?JessTab>

⁴ <http://protege.stanford.edu/>

Fluoride, Total Ammonia and Glyphosate. Furthermore, it identifies the potential sources of pollution induced by BOD₅ 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 and setting up the Jess's working memory by mapping the required classes as specified by *initiateINWSrules.jess* file, the user is presented with a dialog for choosing the type of the windows for the average value observations and for individual ones. The streaming of observations starts immediately after choosing the windows type and each second new observation is produced. 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 *StreamJess.java*.

2.2. Jess reasoning

After C-SPARQL processes the first window results, the Jess engine acts to do the inference based on the registered rules. Namely, based on C-SPARQL design principle each query attaches new result formatter. Except for formatting the output, StreamJess uses the function *RDFResultsFormatter(r)*, where *r* is the instance of the Rete engine, to also do rule-based reasoning. For each new query result, firstly, new assertions are published into the knowledge base and then the Rete engine runs to do the rule-based reasoning. The rules' output gets printed out on the console as depicted in the following screenshot:

```

Output - StreamJess (run)  CSWRL.java  StreamJess.java  INWS1RDFStreamTestGenerator1.java  RDFResultsFormatter.java  RDFResultsFormatter.java...
+++++++ 5 new C-SPARQL result(s) at SystemTime=[1481229314937] ++++++
(StreamJess)
MODERATE status detected: BOD64182 In:http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#ms1 On:Thu
Pollution source: http://inwatersense.uni-pr.edu/ontologies/inws-pollutants.owl#Fish_farming
(StreamJess)
GOOD/HIGH status detected: Arsenic28642 In:http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#ms1 On
(StreamJess)
MODERATE status detected: Arsenic30170 In:http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#ms4 On
(StreamJess)
MODERATE status detected: BOD3444 In:http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#ms4 On:Thu
Pollution source: http://inwatersense.uni-pr.edu/ontologies/inws-pollutants.owl#Landfill_sites
Pollution source: http://inwatersense.uni-pr.edu/ontologies/inws-pollutants.owl#Fish_farming
(StreamJess)
GOOD status detected: BOD54040 In:http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#ms3 On:Thu Dec
http://inwatersense.uni-pr.edu/stream#obs_9157 http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://in
http://inwatersense.uni-pr.edu/stream#obs_9157 http://purl.oclc.org/NET/ssnx/ssn#observationResult http://
http://inwatersense.uni-pr.edu/stream#so_9157 http://purl.oclc.org/NET/ssnx/ssn#hasValue http://inwaterse
http://inwatersense.uni-pr.edu/stream#obs_9157 http://purl.oclc.org/NET/ssnx/ssn#observedBy http://inwat
# 10: WQ: BiochemicalOxygenDemand Val: 1.078 Loc: ms3 Time: 1481229316379

+++++++ 1 new C-SPARQL result(s) at SystemTime=[1481229316384] ++++++
(StreamJess)
MODERATE status detected: pH55953 In:http://inwatersense.uni-pr.edu/ontologies/inws-core.owl#ms3 On:Thu
http://inwatersense.uni-pr.edu/stream#obs_2482 http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://in
http://inwatersense.uni-pr.edu/stream#obs_2482 http://purl.oclc.org/NET/ssnx/ssn#observationResult http://
http://inwatersense.uni-pr.edu/stream#so_2482 http://purl.oclc.org/NET/ssnx/ssn#hasValue http://inwaterse

```

III. User-defined examples

This section contains information for users willing to write their own examples in StreamJess.

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 Jess 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 path value of *inwsOntologies*, configure appropriately the stream generator and write appropriate Jess rules.

3.3. Change the Jess rules

Using the InWaterSense model, users are open to add other Jess rules for monitoring or finding potential sources of pollution. One can write a Jess rule, save it as a *.jess* file in the *jessRules* folder of the project and write a Jess command like follows:

```
r.batch(inwsRules + "xxx.jess");
```

Place this line right after creating the Rete 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.