Interpreting IoT Data with Sensor-based Linked Open Rules (S-LOR)

Creator	Amelie Gyrard (Eurecom - Insight - NUIG/DERI, Ecole des Mines Saint-Etienne) Designed and implemented by Amélie Gyrard, she was a PhD student at Eurecom under the supervision of Prof. Christian Bonnet and Dr. Karima Boudaoud. S-LOR was maintained since she was a post-doc researcher at Insight within the IoT unit led by Dr. Martin Serrano and involved in the FIESTA-IoT (Federated Interoperable Semantic IoT/Cloud Testbeds and Applications) H2020 project. S-LOR is still maintained since she is a post-doc researcher at Ecole des Mines Saint-Etienne (EMSE) and within the Connected Intelligence, Knowledge Representation and Reasoning team.
Send Feedback	Do not hesitate to ask for help or give us feedback, advices to improve our tools or documentations, fix bugs and make them more user-friendly and convenient: **amelie.gyrard@emse.fr**
Platform URL	http://linkedopenreasoning.appspot.com/
Documentation URL	http://linkedopenreasoning.appspot.com/documentation/SLORDocumentation.pdf (to check latest updates)
Google Group	https://groups.google.com/d/forum/m3-semantic-web-of-things
Last updated	June 2017 • APIs, web services (relevant for the IC 2017 tutorial) • Refactoring and various improvements June 2016 • Refactoring code (SLORWS.java, slor.js, renaming files, etc.) • Update the documentations • Explain better the web services + screenshots • Update the Javadoc
Created	June 2016
Status	Work in progress
Goal	This documentation enables understanding the S-LOR tool: • Interpreting IoT Data • APIs, RESTfful web services • Deduce meaningful knowledge from sensor data • Reasoning engine • Dataset of interoperable rules
Technologies	M3 ontology M3 framework LOV4IoT LOV4IoT RDF dataset

	Jena Rule Engine Jena Framework

Table of contents

I. SLOR Web Services....4
1. Getting all rules related to a specific sensor....4
2. Getting all projects employing a specific sensor 5
II. SLOR Architecture 6
III. Why SLOR?....6
1. Reusing domain knowledge from LOV4IoT 6
2. Sharing and reusing based approach....7
3. Dataset of interoperable rules....8
4. Jena inference engine 9
5. Algorithm 9
IV. DEMOS & GUIs....9
1. Demo....9
V. S-LOR Code....11
1. WAR/RULES....12
VI. Code example: Interpreting IoT data and getting M3 suggestions 12
1. Getting a template....13
2. Loading M3 domain knowledge....13
3. Executing rules....13
4. Executing SPARQL query....14
5. Finishing the application....14
6. Code summary 15
VII. S-LOR Limitations 15
VIII. S-LOR Citations....16
IX. References 16

Table of figures

FIGURE 1. WEB SERVICE TO GET RULES FOR A SPECIFIC SENSOR 5
FIGURE 2. WEB SERVICE TO GET PROJECT USING A SPECIFIC SENSOR 6
FIGURE 3. INTEROPERABILITY ISSUES REGARDING REASONING 7
FIGURE 4. RULE EXAMPLE IMPLEMENTED FOR BEING COMPLIANT WITH THE JENA FRAMEWORK 8
FIGURE 5. SPARQL CONSTRUCT RULE EQUIVALENT TO JENA RULES 8
FIGURE 6. FINDING RULES TO INTERPRET SENSOR DATA WITH S-LOR 10
FIGURE 7. S-LOR DEMO 10
FIGURE 8. JENA RULES FROM THE S-LOR SEMANTIC RULE REPOSITORY 11
FIGURE 9. SLORWS.JAVA FILE 11
FIGURE 10. RULE DIRECTORY 12
FIGURE 11. CODE EXAMPLE TO INTERPRET IOT DATA AND GET M3 SUGGESTIONS 15

Terms and acronyms

IoT	Internet of Things (IoT)
LOV	Linked Open Vocabularies
LOV4IoT	Linked Open Vocabularies for Internet of Things
M3 framework	Machine-to-Machine Measurement (M3) framework
S-LOR	Sensor-based Linked Open Rules
Jena	A framework to build Semantic Web Applications

I. SLOR Web Services

To contribute or understand the web services:

Getting all rules related to a specific sensor	http://sensormeasurement.appspot.com/slor/rule/{sensorType} E.g. http://sensormeasurement.appspot.com/slor/rule/BodyThermometer sensorType should be compliant with the classes referenced with M3 ontology
Getting all projects employing a specific sensor	http://sensormeasurement.appspot.com/slor/{sensorType} E.g. http://sensormeasurement.appspot.com/slor/BodyThermometer sensorType should be compliant with the classes referenced with M3 ontology

1. Getting all rules related to a specific sensor

http://sensormeasurement.appspot.com/slor/rule/{sensorType}

The parameter sensorType should be compliant with the classes referenced with M3 ontology.

The web service returns all rules using this sensor type. It provides “IF THEN ELSE” rules readable for human and URLs for getting the Jena rule implementations.

Example: http://sensormeasurement.appspot.com/slor/rule/BodyThermometer


```xml
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="ruleLabel"/>
    <variable name="ruleComment"/>
    <variable name="m2mappli_label"/>
    <variable name="lorUrl"/>
  </head>
  <results>
    <result>
      <binding name="ruleLabel">
        <literal xml:lang="en">ModerateHypothermia</literal>
      </binding>
      <binding name="ruleComment">
        <literal xml:lang="en">
          IF m3:BodyTemperature greaterThan 28 m3:DegreeCelsius AND lessThan 32 m3:DegreeCelsius THEN ModerateHypothermia
        </literal>
      </binding>
      <binding name="lorUrl">
        <uri>
          http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesHealth.txt
        </uri>
      </binding>
    </result>
    <result>
      <binding name="ruleLabel">
        <literal xml:lang="en">NormalFever</literal>
      </binding>
      <binding name="ruleComment">
        <literal xml:lang="en">
          IF m3:BodyTemperature greaterThan 36 m3:DegreeCelsius AND lessThan 38 m3:DegreeCelsius THEN NormalFever
        </literal>
      </binding>
      <binding name="lorUrl">
        <uri>
          http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesHealth.txt
        </uri>
      </binding>
    </result>
```


Figure 1. Web service to get rules for a specific sensor

2. Getting all projects employing a specific sensor

The web service returns all projects using this sensor type. The projects are those referenced within the LOV4IoT RDF dataset.

Example: http://sensormeasurement.appspot.com/slor/BodyThermometer


```xml
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="m2mappli"/>
    <variable name="m2mappli_label"/>
    <variable name="m2mappli_comment"/>
    <variable name="m2mdevice"/>
    <variable name="onto"/>
    <variable name="techno"/>
    <variable name="techno_label"/>
    <variable name="techno_comment"/>
    <variable name="securityMechanism"/>
    <variable name="security_mechanism_label"/>
    <variable name="security_mechanism_comment"/>
    <variable name="domain"/>
    <variable name="domain_label"/>
  </head>
  <results>
    <result>
      <binding name="m2mappli">
        <uri>http://sensormeasurement.appspot.com/ont/m3/Ko2006</uri>
      </binding>
      <binding name="m2mappli_label">
        <literal xml:lang="en">[Ko et al. 2006]. See LOV4IoT for more details.</literal>
      </binding>
      <binding name="m2mappli_comment">
        <literal xml:lang="en">
          Paper: Ontology-based context-aware service engine for U-Healthcare
        </literal>
      </binding>
      <binding name="m2mdevice">
        <uri>
          http://sensormeasurement.appspot.com/m3#BodyThermometer
        </uri>
      </binding>
      <binding name="domain">
        <uri>http://sensormeasurement.appspot.com/m3#Health</uri>
      </binding>
      <binding name="domain_label">
        <literal xml:lang="en">Healthcare</literal>
      </binding>
    </result>
```


Figure 2. Web service to get project using a specific sensor

II. SLOR Architecture

See paper: Sensor-based Linked Open Rules (S-LOR): An Automated Rule Discovery Approach for IoT Applications and its use in Smart Cities. 3rd International ACM Smart City Workshop (AW4city) in conjunction with 26th International World Wide Web Conference (WWW 2017), April 3-7 2017, Perth, Australia. Amelie Gyrard, Martin Serrano, Soumya Kanti Datta, Joao Bosco Jares, Muhammad Intizar Ali

PDF: http://www.eurecom.fr/fr/publication/5144/download/comsys-publi-5144.pdf

III. Why SLOR?

Sensor-based Linked Open Rules (S-LOR) is an approach to share and reuse the rules to interpret IoT data [2]. They provide interoperable datasets of rules compliant with the Jena framework and inference engine.

1. Reusing domain knowledge from LOV4IoT

The rules have been written manually but extracted from the Linked Open Vocabularies for Internet of Things (LOV4IoT) dataset, an ontology/dataset/rule catalogue designed by domain experts in various applicative domains relevant for IoT such as healthcare, agriculture, smart home, smart city, etc.

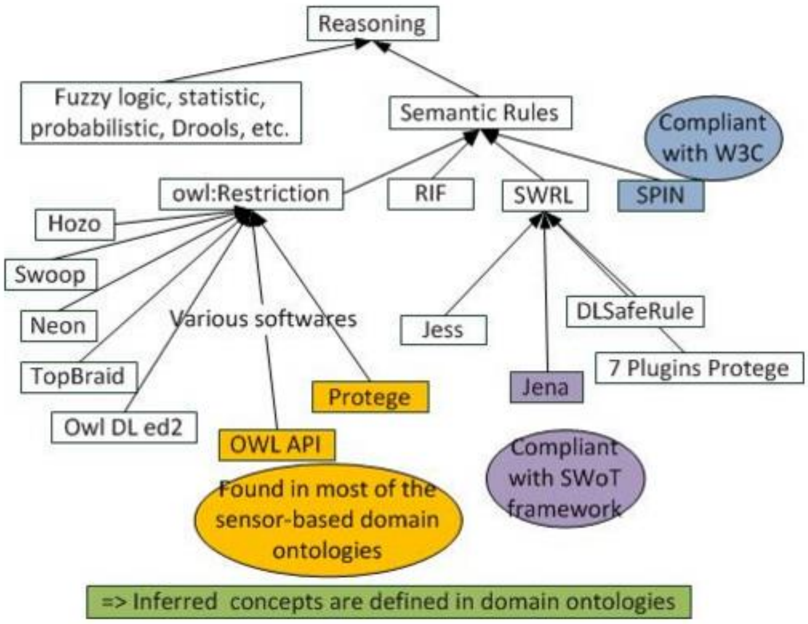


Figure 3. Interoperability issues regarding reasoning

TO DO: Explain owl restrictions, interoperability issues

2. Sharing and reusing based approach

S-LOR has a common vision with the following approaches:

- The **BASIL (Building APIs SImpLy)**[1] framework combines REST principles and SPARQL endpoints in order to benefit from Web APIS and Linked Data approaches [1]. BASIL reduces the learning curve of data consumers since they query web services exploiting SPARQL endpoints. The main benefit is that data consumers do not need to learn the SPARQL language and related semantic web technologies.
- **Linked Edit Rules (LER)**[2] [3] is a recent approach similar to the Sensor-based Linked Open Rules (S-LOR) to share and reuse the rules associated to the data. This work has been not applied to the context of IoT. LER is more focused on checking consistency of data (e.g., a person's age cannot be negative, a man cannot be pregnant and an underage person cannot process a driving license). LER extends the RDF Data Cube data model by introducing the concept of EditRule. The implementation of LER is based on Stardog's rule reasoning to check obvious consistency.

[1] http://basil.kmi.open.ac.uk/app/

[2] http://linkededitrules.org/

3. Dataset of interoperable rules

Data has been unified thanks to the M3 taxonomy, a cornerstone component for building a dataset of interoperable rules. The picture shows the implementation of the rule based on the M3 taxonomy: the hierarchy of quantity kinds and units.

```
################################################################ CLOUD COVER RULES ##################
# Master's thesis: A weather ontology for predictive control in smart homes [Paul Staroch 2013]
# Paper: An Intelligent Knowledge Representation of Smart Home Energy Parameters [Kofler 2011]
# Paper: Thinkhome energy efficiency in future smart homes [Kofler 2011]
# Paper: A semantic representation of energy-related information in future smart homes [Kofler 2011]
# Paper: A knowledge-base for Energy-Efficient Smart Homes  [Kofler 2011]

# Rule: IF m3:CloudCover greaterThan 5 AND lessThan 8 m3:Okta THEN MostlyCloudy
[MostlyCloudy:
              (?measurement rdf:type m3:CloudCover)
              (?measurement m3:hasValue ?v)
              greaterThan(?v,5)
              lessThan(?v,8)
                          ->
                                (?measurement rdf:type weather-dataset:MostlyCloudy)
]
```

Figure 4. Rule example implemented for being compliant with the Jena framework

Since we are using the Jena framework, within this project, intuitively we use the Jena inference engine and Jena rules for the implementation.

After the implementation, we realized that the same rules can be built using the SPARQL query language with the keyword “CONSTRUCT”.

Both methods have the same goal updating the knowledge graphs or triplestore with additional information (more triples).

SPARQL construct encourages interoperability since SPARQL is a W3C recommendation.

```
CONSTRUCT {?m naturopathy:hasDisease nat:Flu} WHERE {
?m rdf:type m3:BodyTemperature.
?m m3:hasValue ?v.
FILTER ( ?v > 38 ).
}
```

Figure 5. SPARQL CONSTRUCT RULE equivalent to Jena rules

4. Jena inference engine

SWRL

Logic-based reasoning

https://jena.apache.org/documentation/inference/

5. Algorithm

Algo:

- Input: Dataset semantically annotated according to the FIESTA-IoT ontology including the M3-lite taxonomy
- Output: Dataset updated with more triples, High level abstraction
- Algo:
    - Load the dataset or triplestore
    - Load the rules of subset
    - Execute the reasoning engine
    - Update the dataset or triplestore with more triples

IV. DEMOS & GUIs

1. Demo

Go to this web page: http://www.sensormeasurement.appspot.com/?p=swot_template

- ➔ Select a sensor to find all rules interpreting sensor values as depicted in Figure 6 (e.g., Precipitation)
- ➔ The demonstration will show all rules related to the sensor chosen by the user to interpret sensor values.
  (e.g., if precipitation = 0 mm/h then NoPrecipitation)
- ➔ You have both the rule for humans and for machines (click on the LinkedOpenRules link)

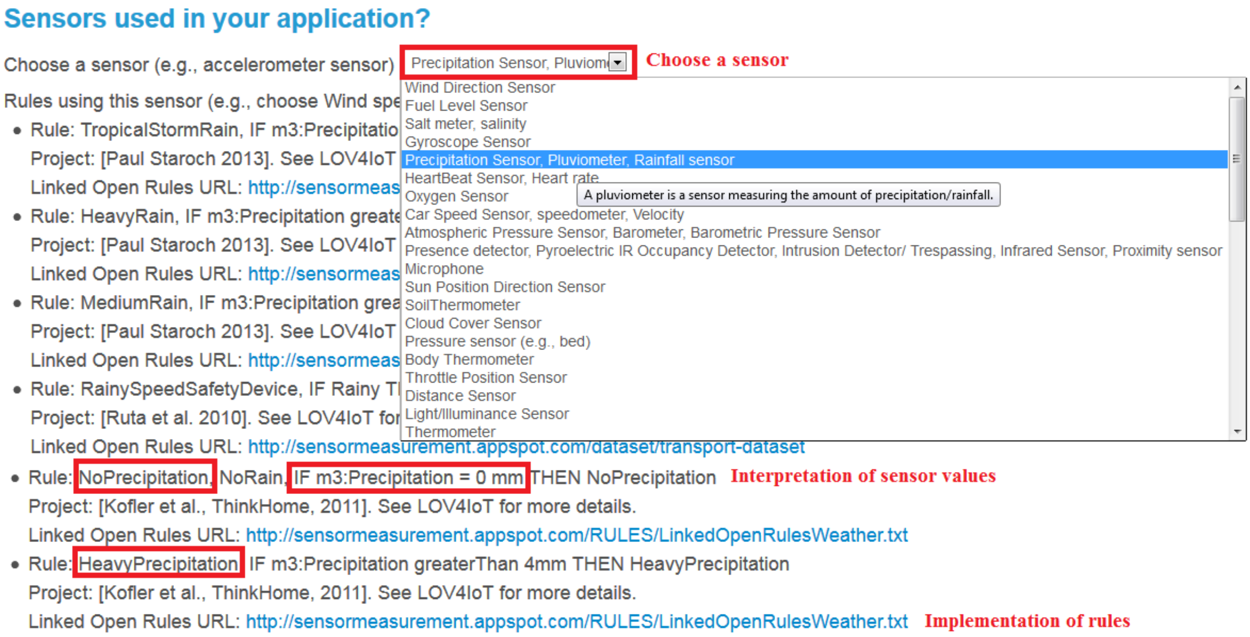


Figure 6. Finding rules to interpret sensor data with S-LOR

The users can choose a device type from the drop-down list. This drop-down list queries in the back-end the M3 ontology (V1) or M3-lite taxonomy (V2).

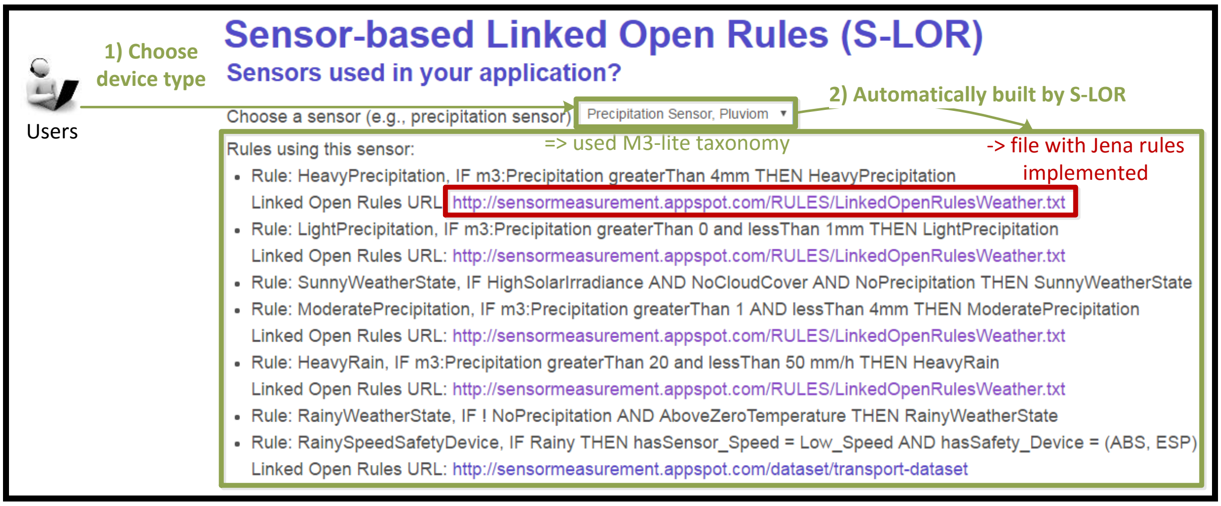
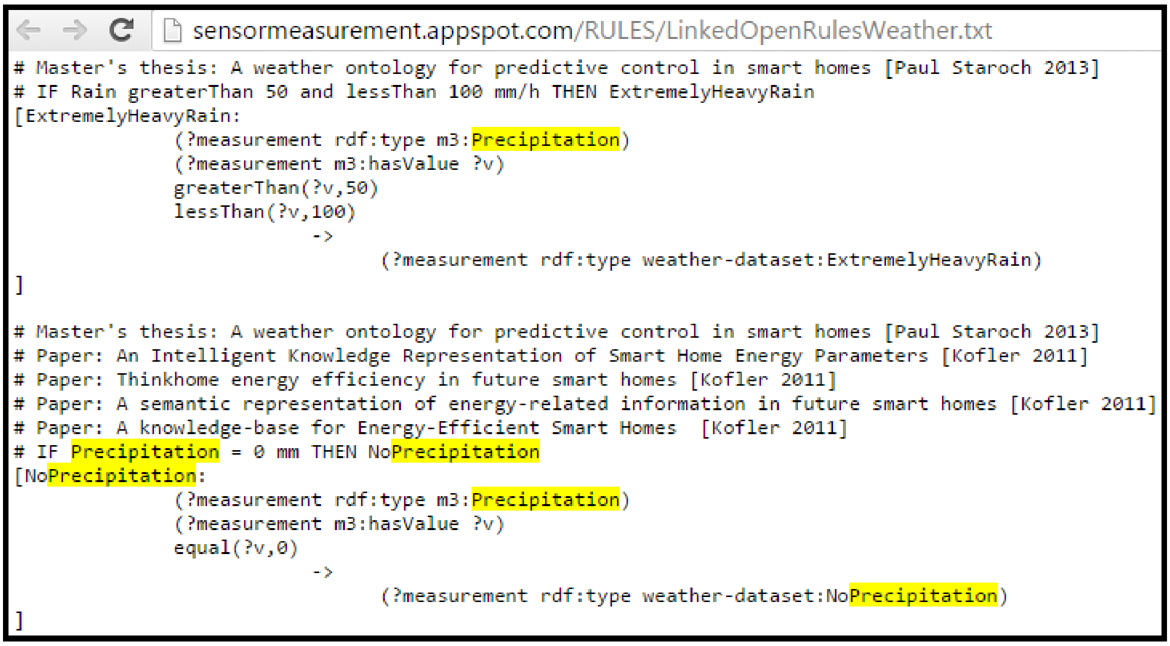


Figure 7. S-LOR demo

The users can click on the URL to get access to the file with already implemented Jena rules. Figure 8 shows two rules:

- IF precipitation measurement >50 and < 100 mm/h THEN **Extremely Heavy Rain**
- IF precipitation measurement = 0 mm/h THEN **No Precipitation**.



sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt

```
# Master's thesis: A weather ontology for predictive control in smart homes [Paul Staroch 2013]
# IF Rain greaterThan 50 and lessThan 100 mm/h THEN ExtremelyHeavyRain
[ExtremelyHeavyRain:
              (?measurement rdf:type m3:Precipitation)
              (?measurement m3:hasValue ?v)
              greaterThan(?v,50)
              lessThan(?v,100)
                          ->
                               (?measurement rdf:type weather-dataset:ExtremelyHeavyRain)
]

# Master's thesis: A weather ontology for predictive control in smart homes [Paul Staroch 2013]
# Paper: An Intelligent Knowledge Representation of Smart Home Energy Parameters [Kofler 2011]
# Paper: Thinkhome energy efficiency in future smart homes [Kofler 2011]
# Paper: A semantic representation of energy-related information in future smart homes [Kofler 2011]
# Paper: A knowledge-base for Energy-Efficient Smart Homes  [Kofler 2011]
# IF Precipitation = 0 mm THEN NoPrecipitation
[NoPrecipitation:
              (?measurement rdf:type m3:Precipitation)
              (?measurement m3:hasValue ?v)
              equal(?v,0)
                          ->
                               (?measurement rdf:type weather-dataset:NoPrecipitation)
]
```

Figure 8. Jena rules from the S-LOR Semantic Rule Repository

V. S-LOR Code

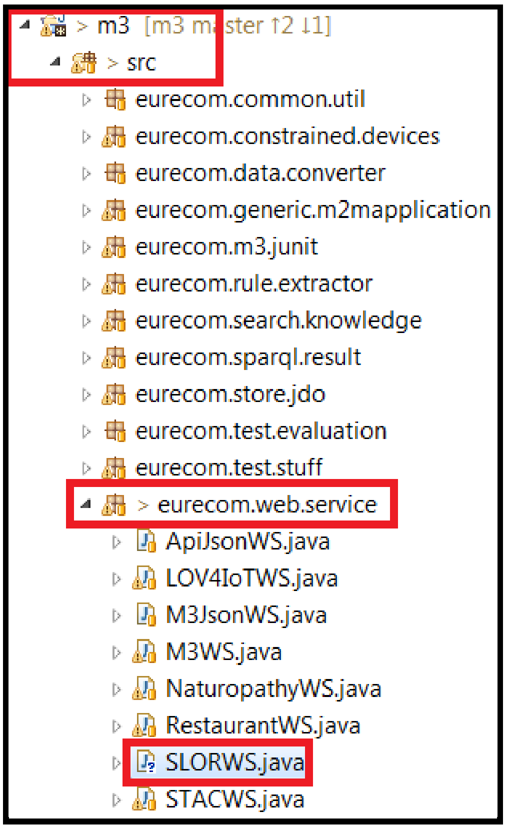


Figure 9. SLORWs.java file

S-LOR is a component of the M3 framework.

M3 on Github: https://github.com/gyrard/M3Framework

M3 documentation to set up the code:

http://sensormeasurement.appspot.com/documentation/M3DeveloperDocumentation.pdf

1. WAR/RULES

You will find in this directory all the interoperable rules that we designed.

This is the Sensor-based Linked Open Rules (S-LOR) tool. The SWoT generator has predefined-templates to build semantic-based IoT application. The templates will referenced these pre-defined set of rules classified by domains.

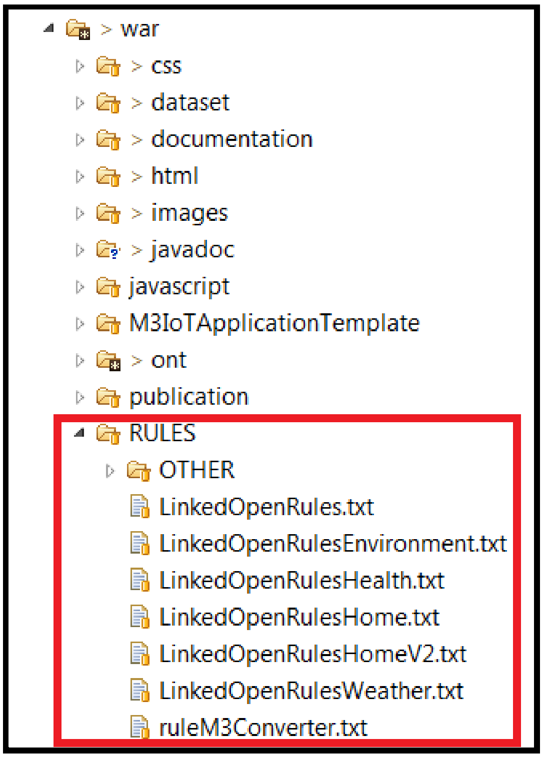


Figure 10. Rule directory

VI. Code example: Interpreting IoT data and getting M3 suggestions

Several steps need to be achieved to interpret IoT data (see Figure 11):

- Loading M3 ontologies, datasets which have been generated in the M3 template.
- Loading M3 data.= which has been generated by the M3 converter.
- Interpreting IoT data using the Jena reasoned

- Executing the M3 SPARQL query which has been generated in the M3 template
- Parse the result and build the user interface , control actuators or send notification, etc.

1. Getting a template

http://sensormeasurement.appspot.com/?p=m3api

See documentation (section Tutorial):
http://sensormeasurement.appspot.com/documentation/M3APIDocumentation.pdf

2. Loading M3 domain knowledge

Jena tutorial:

http://jena.apache.org/tutorials/rdf_api.html

Code example:

```
// STEP 1: Loading M3 domain knowledge and m3_data

Model model = ModelFactory.createDefaultModel();

InputStream in = new FileInputStream(PATH_FILE + m3_data);

// m3_data has been generated with the M3 converter

model.read( in, fileURL );//read all ontologies generated in the M3 template (.owl)

model.read( in, fileURL );//read all datasets generated in the M3 template (.rdf)

in.close();
```

3. Executing rules

Jena tutorial:

http://jena.apache.org/documentation/inference/

Code example:

```
// STEP 2: Interpreting M3 data
```

```
Reasoner reasoner = new GenericRuleReasoner(Rule.rulesFromURL(PATH_FILE +
LinkedOpenRules*.txt));

// LinkedOpenRules*.txt: rules generated in the M3 template

reasoner.setDerivationLogging(true);

InfModel infModel = ModelFactory.createInfModel(reasoner, model); //apply the reasoner

// infModel has been updated with high-level abstraction
```

4. Executing SPARQL query

Jena tutorial:

http://jena.apache.org/tutorials/rdf_api.html

Code example:

```
// STEP 3: Getting M3 suggestions

//          Executing the SPARQL query:

Query query = QueryFactory(m3_sparql); // m3_sparql has been generated in the M3 template

ResultSet results = QueryExecutionFactory.create(m3_sparql, model)

String m3_suggestions = ResultSetFormatter.asXMLString(results)
```

5. Finishing the application

The main task of the develop is to design a user-friendly interface or control actuators, etc. according to the high-level abstractions deduce by M3 or the M3 suggestions provided by M3.

Code example:

```
 // STEP 4: Parsing and displaying m3_suggestions to build the IoT application

// or control actuators, alerting, etc.
```

6. Code summary

```
// STEP 1: Loading M3 domain knowledge and m3_data
Model model = ModelFactory.createDefaultModel();
InputStream in = new FileInputStream(PATH_FILE + m3_data);
// m3_data has been generated with the M3 converter
model.read( in, fileURL );//read all ontologies generated in the M3 template (.owl)
model.read( in, fileURL );//read all datasets generated in the M3 template (.rdf)
in.close();

// STEP 2: Interpreting M3 data
Reasoner reasoner = new GenericRuleReasoner(Rule.rulesFromURL(PATH_FILE + LinkedOpenRules*.txt));
// LinkedOpenRules*.txt: rules generated in the M3 template
reasoner.setDerivationLogging(true);
InfModel infModel = ModelFactory.createInfModel(reasoner, model); //apply the reasoner
// infModel has been updated with high-level abstraction

// STEP 3: Getting M3 suggestions
//  Executing the SPARQL query:
Query query = QueryFactory(m3_sparql); // m3_sparql has been generated in the M3 template
ResultSet results = QueryExecutionFactory.create(m3_sparql, model)
String m3_suggestions = ResultSetFormatter.asXMLString(results)

// STEP 4: Parsing and displaying m3_suggestions to build the IoT application
// or control actuators, alerting, etc.
```

Figure 11. Code example to interpret IoT data and get M3 suggestions

VII. S-LOR Limitations

S-LOR has some limitations:

- S-LOR works only with simple sensors such as thermometer, rainfall sensors. Some more complicated sensors such as camera provide images that cannot be proceed by S-LOR. For this reasoning, an objective is integrating the KAT toolkit based on machine leaning techniques to deal with more complicated sensors.
- How the Semantic Rule repository can be automatically updated with new rules provided by the experimenters (knowledge producers). Adding a new rule in the repository is easy. However, dealing with redundancy and overlapping rules is more complicated. For this, we need to check correctness and completeness of rules. Correctness and completeness have been checked manually.
- Redesign the Jena rules as SPARQL construct since SPARQL construct to encourage interoperability since SPARQL is a W3C recommendation.

TO DO:

Correctness and completeness have been checked manually.

Add table example:

http://sensormeasurement.appspot.com/documentation/NomenclatureSensorData.pdf

VIII. S-LOR Citations

Please do not forget to cite S-LOR:

- Sensor-based Linked Open Rules (S-LOR): An Automated Rule Discovery Approach for IoT Applications and its use in Smart Cities. 3rd International ACM Smart City Workshop (AW4city) in conjunction with 26th International World Wide Web Conference (WWW 2017), April 3-7 2017, Perth, Australia. Amelie Gyrard, Martin Serrano, Soumya Kanti Datta, Joao Bosco Jares, Muhammad Intizar Ali
- Enrich Machine-to-Machine Data with Semantic Web Technologies for Cross-Domain Applications. IEEE World Forum on Internet of Things (WF-IoT), Seoul, Korea, March 6-8, 2014 Amelie Gyrard, Christian Bonnet and Karima Boudaoud
- Book Chapter: A Review of Tools for IoT Semantics and Data Streaming Analytics. Book: The Building Blocks of IoT Analytics - Internet-of-Things Analytics. Martin Serrano and Amelie Gyrard
- Demo paper: Helping IoT application developers with Sensor-based Linked Open Rules 7th International Workshop on Semantic Sensor Networks, in conjunction with the 13th International Semantic Web Conference (ISWC) October 19-23, 2014, Riva del Garda, Trentino Italy. Amelie Gyrard, Christian Bonnet and Karima Boudaoud

IX. References

[1] Enrico Daga, Luca Panziera, and Carlos Pedrinaci. A basilar approach for building web apis on top of sparql endpoints. 2015.

[2] Amelie Gyrard, Christian Bonnet, and Karima Boudaoud. Helping IoT application developers with sensor-based linked open rules. In *SSN 2014, 7th International Workshop on Semantic Sensor Networks in conjunction with the 13th International Semantic Web Conference (ISWC 2014), 19-23 October 2014, Riva Del Garda, Italy*, 10 2014.

[3] Albert Merono-Penuela, Christophe Gueret, and Stefan Schlobach. Linked edit rules: A web friendly way of checking quality of rdf data cubes. In *3rd International Workshop on Semantic Statistics co-located with 14th International Semantic Web Conference (ISWC 2015)*, 2015.