Sensor-based Linked Open Rules

Creator	Amelie Gyrard (Eurecom - Insight - NUIG/DERI) Designed and implemented by Amélie Gyrard, she was a PhD student at Eurecom under the supervision of Prof. Christian Bonnet and Dr. Karima Boudaoud. Currently, LOVIoT is maintained since she is 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.
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@insight-centre.org
Google Group	https://groups.google.com/d/forum/m3-semantic-web-of-things
Created	June 10, 2016
Last updated	June 10, 2016
Status	⚠ Work in progress
Goal	This documentation enables understanding the S-LOR tool: • Deduce meaningful knowledge from sensor data • Reasoning engine • Dataset of interoperable rules

Table of contents

Table of figures

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

I. S-LOR Citations

Please do not forget to cite our S-LOR work:

- 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
- 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

II. Introduction

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.

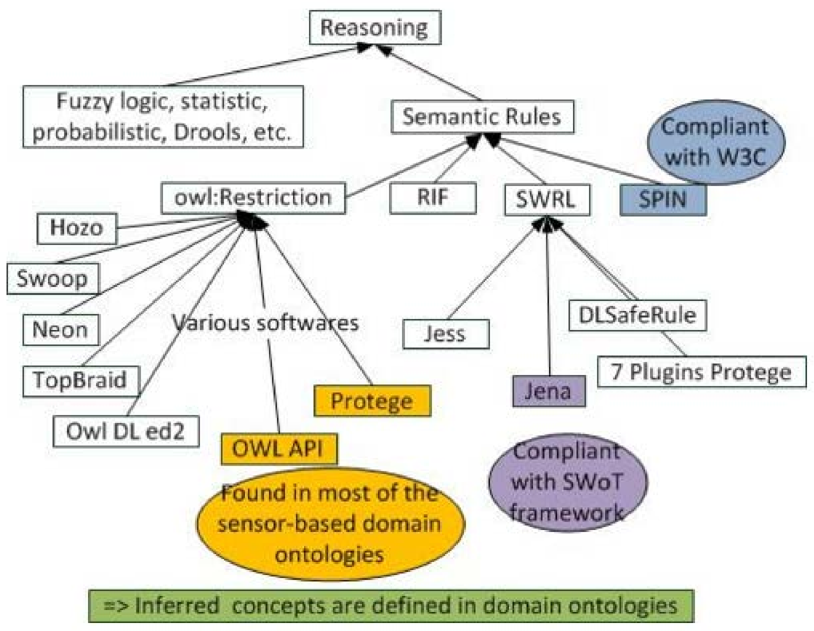

Reasoning
Fuzzy logic, statistic, probabilistic, Drools, etc.
Semantic Rules
Compliant with W3C
owl:Restriction
RIF
SWRL
SPIN
Hozo
Swoop
Neon
TopBraid
Owl DL ed2
Various softwares
Protege
OWL API
Jess
DLSafeRule
7 Plugins Protege
Jena
Found in most of the sensor-based domain ontologies
Compliant with SWoT framework
=> Inferred concepts are defined in domain ontologies

Figure 1 Interoperability issues regarding reasoning

TO DO: Explain owl restrictions, interoperability issues

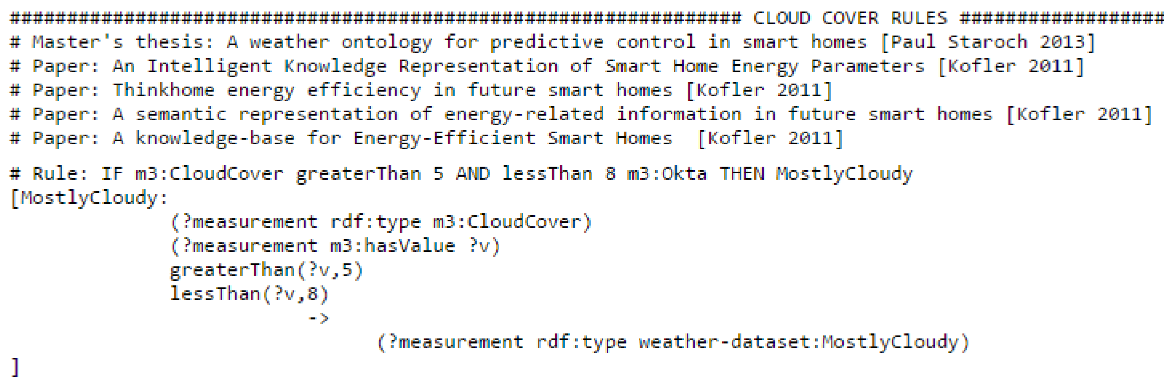
2. Sharing and reusing based approach

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.

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 2. 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".

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

[2] http://linkededitrules.org/

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 3. SPARQL CONSTRUCT RULE equivalent to Jena rules

4. Jena inference engine

SWRL

Logic-based reasoning

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

5. Algorithm

Algo:

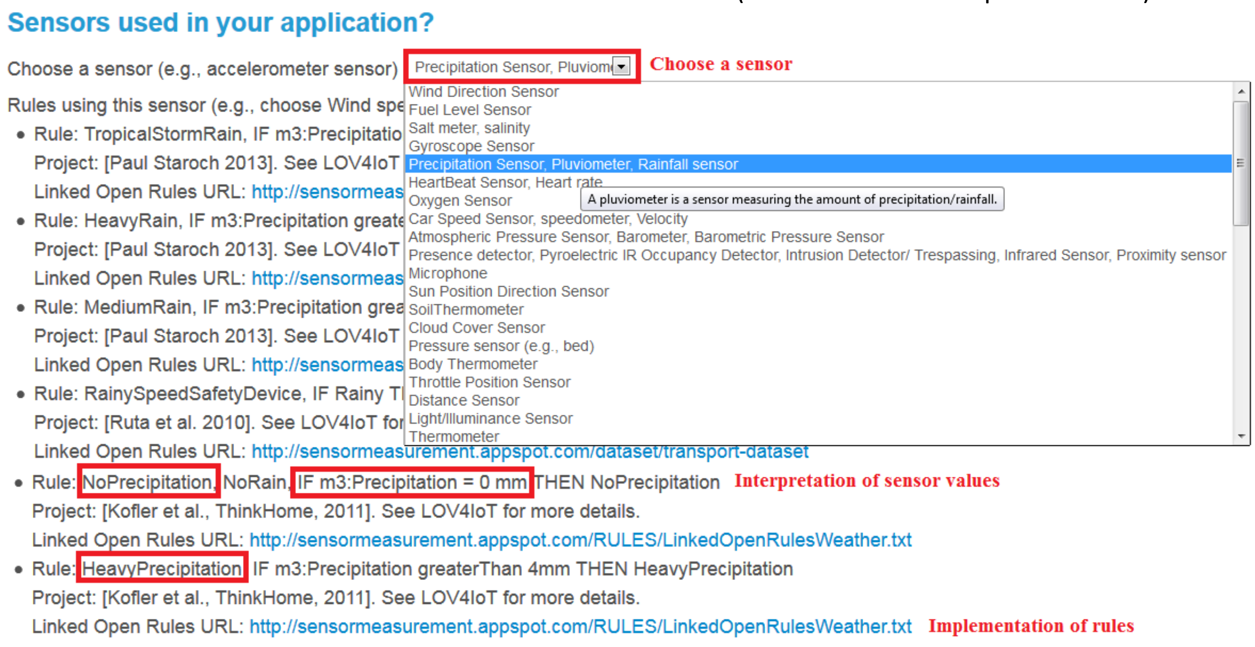
- Input: Dataset semantically annotated according to the FIESTA-IoT ontology including the M3-lite taxonomy
- Output: Dataset updated with more triples, High level abstraction
- Algo:
  - Load the dataset or triplestore
  - Load the rules of subset
  - Execute the reasoning engine
  - Update the dataset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

III. Interpreting IoT data with S-LOR

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 4 (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)


Sensors used in your application?

Choose a sensor (e.g., accelerometer sensor) Precipitation Sensor, Pluviom Choose a sensor

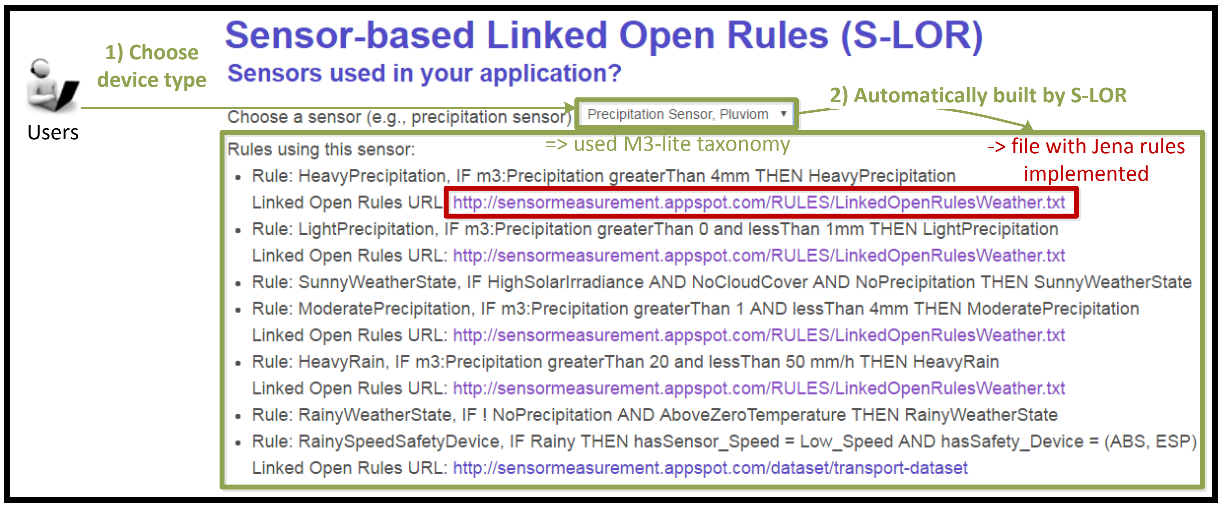
Rules using this sensor (e.g., choose Wind spe

- Rule: TropicalStormRain, IF m3:Precipitatio
  Project: [Paul Staroch 2013]. See LOV4IoT
  Linked Open Rules URL: http://sensormeas
- Rule: HeavyRain, IF m3:Precipitation greate
  Project: [Paul Staroch 2013]. See LOV4IoT
  Linked Open Rules URL: http://sensormeas
- Rule: MediumRain, IF m3:Precipitation grea
  Project: [Paul Staroch 2013]. See LOV4IoT
  Linked Open Rules URL: http://sensormeas
- Rule: RainySpeedSafetyDevice, IF Rainy T
  Project: [Ruta et al. 2010]. See LOV4IoT for
  Linked Open Rules URL: http://sensormeasurement.appspot.com/dataset/transport-dataset
- Rule: NoPrecipitation, NoRain, IF m3:Precipitation = 0 mm THEN NoPrecipitation Interpretation of sensor values
  Project: [Kofler et al., ThinkHome, 2011]. See LOV4IoT for more details.
  Linked Open Rules URL: http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt
- Rule: HeavyPrecipitation IF m3:Precipitation greaterThan 4mm THEN HeavyPrecipitation
  Project: [Kofler et al., ThinkHome, 2011]. See LOV4IoT for more details.
  Linked Open Rules URL: http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt Implementation of rules

Wind Direction Sensor
Fuel Level Sensor
Salt meter, salinity
Gyroscope Sensor
Precipitation Sensor, Pluviometer, Rainfall sensor
HeartBeat Sensor, Heart rate
Oxygen Sensor
A pluviometer is a sensor measuring the amount of precipitation/rainfall.
Car Speed Sensor, speedometer, Velocity
Atmospheric Pressure Sensor, Barometer, Barometric Pressure Sensor
Presence detector, Pyroelectric IR Occupancy Detector, Intrusion Detector/ Trespassing, Infrared Sensor, Proximity sensor
Microphone
Sun Position Direction Sensor
SoilThermometer
Cloud Cover Sensor
Pressure sensor (e.g., bed)
Body Thermometer
Throttle Position Sensor
Distance Sensor
Light/Illuminance Sensor
Thermometer

Figure 4. 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).


1) Choose device type
Users
2) Automatically built by S-LOR
=> used M3-lite taxonomy
-> file with Jena rules implemented
Precipitation Sensor, Pluviom

Sensor-based Linked Open Rules (S-LOR)

Sensors used in your application?

Choose a sensor (e.g., precipitation sensor)

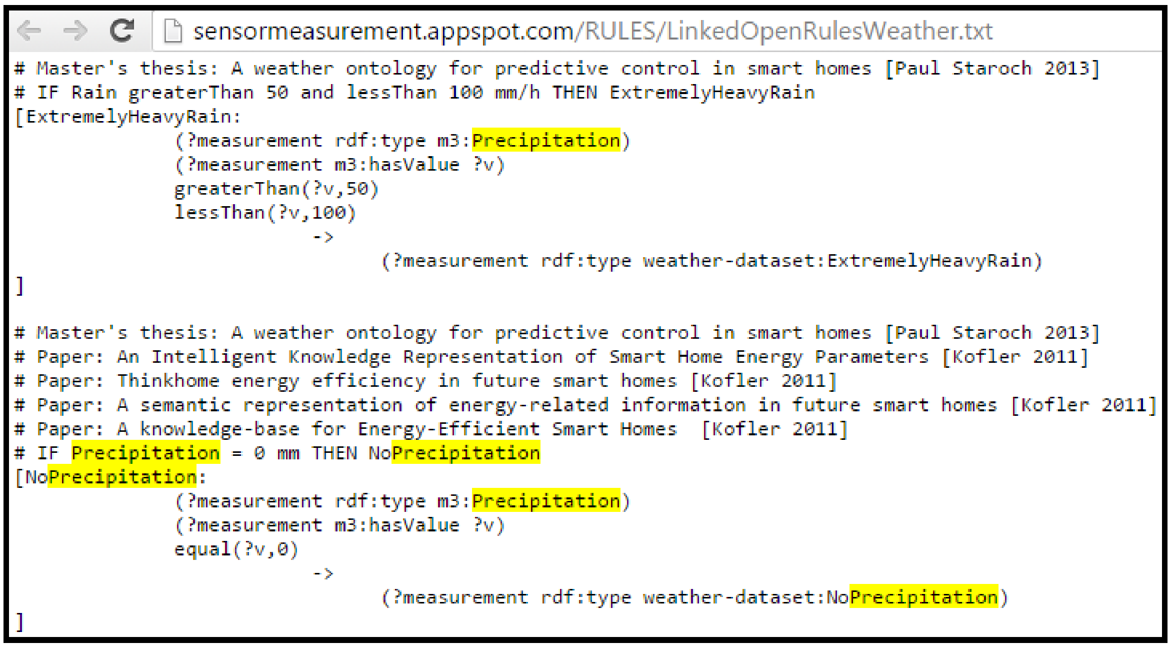
Rules using this sensor:

- Rule: HeavyPrecipitation, IF m3:Precipitation greaterThan 4mm THEN HeavyPrecipitation
  Linked Open Rules URL: http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt
- Rule: LightPrecipitation, IF m3:Precipitation greaterThan 0 and lessThan 1mm THEN LightPrecipitation
  Linked Open Rules URL: http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt
- Rule: SunnyWeatherState, IF HighSolarIrradiance AND NoCloudCover AND NoPrecipitation THEN SunnyWeatherState
- Rule: ModeratePrecipitation, IF m3:Precipitation greaterThan 1 AND lessThan 4mm THEN ModeratePrecipitation
  Linked Open Rules URL: http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt
- Rule: HeavyRain, IF m3:Precipitation greaterThan 20 and lessThan 50 mm/h THEN HeavyRain
  Linked Open Rules URL: http://sensormeasurement.appspot.com/RULES/LinkedOpenRulesWeather.txt
- Rule: RainyWeatherState, IF ! NoPrecipitation AND AboveZeroTemperature THEN RainyWeatherState
- Rule: RainySpeedSafetyDevice, IF Rainy THEN hasSensor_Speed = Low_Speed AND hasSafety_Device = (ABS, ESP)
  Linked Open Rules URL: http://sensormeasurement.appspot.com/dataset/transport-dataset

Figure 5. S-LOR demo

The users can click on the URL to get access to the file with already implemented Jena rules. Figure 13 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 6. Jena rules from the S-LOR Semantic Rule Repository

IV. S-LOR Code

S-LOR is a component of the M3 framework.

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

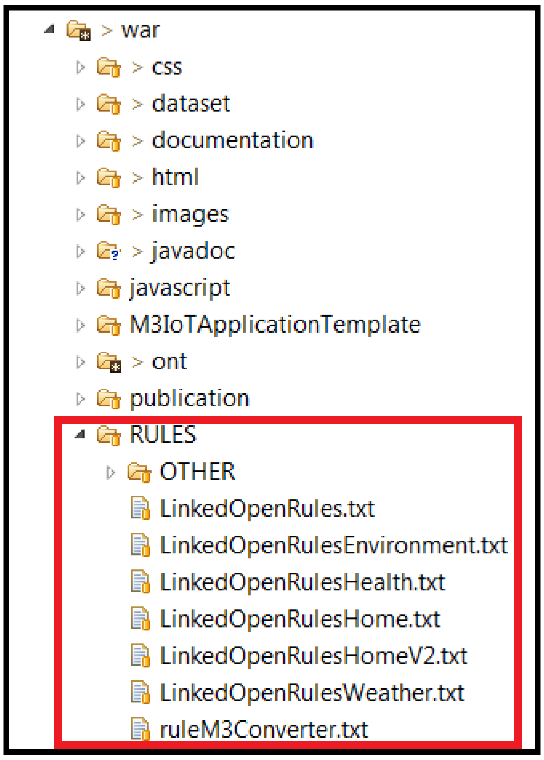
M3 documentation to set up the code:

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.



```
> war
    > css
    > dataset
    > documentation
    > html
    > images
    > javadoc
    javascript
    M3IoTApplicationTemplate
    > ont
    publication
    RULES
        OTHER
        LinkedOpenRules.txt
        LinkedOpenRulesEnvironment.txt
        LinkedOpenRulesHealth.txt
        LinkedOpenRulesHome.txt
        LinkedOpenRulesHomeV2.txt
        LinkedOpenRulesWeather.txt
        ruleM3Converter.txt
```

Figure 7. Rule directory

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

Several steps need to be achieved to interpret IoT data (see Figure 22):

- 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 8. Code example to interpret IoT data and get M3 suggestions

VI. 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

VII. 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.