GAIA Recommendation Engine Developer's manual

CNIT

04/08/2017

draft

Executive Summary

This document provides a documentation of the GAIA's recommendation engine.
The documentation is for developers of modules of the recommendation engine and of applications using the APIs offered by the engine.
This is a draft document.

Table of content

1 Rule documentation

Hereafter we describe the GAIA Rule classes that have been specified so far for using the Recommendation Engine in the GAIA project. For each rule class we provide a textual description, input parameters, suggestion, behavior and output message provided as notification (through WebSocket connection) or logged event.

1.1 Generic GaiaRule

Description

All the rules inherit from the GaiaRule Class. This class exposes all the services (loaded from the application context) needed by the inheriting rules:

- WebSocket Service: for sending notifications over a websocket channel
- Event Service: for events management
- Measurement Repository: as a cache for the measurements
- Building Database Service: for building information management
- Rule Database Service: management of the rules
- Weather service: for retrieving weather history and forecast
- Metadata Service: provides metadata, for instance the schedules/activity calendars for the school/areas

Input parameters

String	name	Sparkworks relative URI of the sensor or numeric resourceID for Active Power	REQUIRED
String	suggestion	Textual suggestion	OPTIONAL
Long	fireInterval	Interval in seconds between two fires	OPTIONAL
String	fireCron	Cron expression to limit the firing of the rule to a specific time instant or period See 5.1 for more information.	OPTIONAL
String	description	Description of the rule	OPTIONAL

Suggestion

Some rules have a default suggestion used if no custom suggestion is provided.
It is possible to use placeholders in the textual suggestion in the form ${*FIELD_NAME*}.
For example let's say the rule provides as output the value in a field named *value* you can write a suggestion like: "the sensor measured a value of ${value} that is too high".
If the *value* variable contains *45.32* the placeholder will be replaced, generating the following suggestion:
"the sensor measured a value of 45.32 that is too high".

Behavior

The GaiaRule defines the default behavior of a rule.
Default **fire** behaviour:

- checks if the rule should be executed by checking both:
  - if the interval between the latest fire and the current instant is greater than *fireInterval*
  - if the current date and time are within the range expressed by *fireCron*
- if the (condition) is verified then the corresponding action is triggered

Default **action**: log the event and send notification to the /recommendations/{school_id} endpoint

Output

The default output is a notification sent through the websocket channel (see 3 Notification) and an event logged into the database (see 4 Events).
Hereafter a schematic description of the content:

Basic Event	Basic Notification
{ @rid: string, timestamp: long, rule: string, values: object }	{ timestamp: long, timestamp: string, name: string, ruleId: string, ruleClass: string, ruleName: string, suggestion: string, description: string, area: long, school: object, values: object }
Values includes all the fields annotated with @LogMe(event=true) (default is true))	Values includes all the fields annotated with @LogMe(notification=true) (default is true))

A list a possible values is given for each rule class

Notes

In the rules described hereafter, the parameter names that follow the pattern "*_*uri* " have to be filled with a string representing the URI of the resource (relative to the sparkworks endpoint) or directly the numeric resource id (represented as a string e.g. "123") when creating the corresponding instance.

1.2 Composite Rules

Description

A CompositeRule is a rule whose behavior depends on the conditions of the linked rules.
The available implementations of this kind of rules are:

AnyCompositeRule	Implement a logical OR among rules' conditions
AllCompositeRule	Implement a logical AND among rules' conditions
RepeatingRule	Trigger the linked rule if the condition is verified more than N times consecutively

How to add rule to a Composite Rule

The general way is described in 2.3

1.2.1 AnyCompositeRule

Class: AnyCompositeRule

Description

This rule implementes a logical OR among the conditions of the linked rules, i.e. the composite is triggered only if at least one of the conditions of the linked rules is true.

Behavior

When fired, the rule iterates over the linked rules and tests their conditions. If one or more conditions are true the action method of the parent rule (the composite) is invoked. The children rules are never fired, only their condition is checked, thus the actions of these rules are never executed.

Simplified composition

It is possible to create a composite in a simplified way using a POST request to
`/area/{aid}/rules/composite`
With the following body:

```
{
"name": "NameOfTheComposite",
"operator": "OR", //or "AND"
"suggestion": "This is a suggestion", //or "AND"
"rules": [
    "class": "SimpleThresholdRule",
    "fields": {
        "name": "MyRuleInstance1",
        "description": "This is a description",
        "operator": ">",
        "suggestion": "You can save energy!",
        "threshold": 8.0,
        "uri": "gaia-prato/gw1/Geom/1F/53/temp"
    },
    "class": "SimpleThresholdRule",
    "fields": {
        "name": "MyRuleInstance2",
        "description": "This is a description",
        "operator": "<",
```

```
        "suggestion": "You can save energy!",
        "threshold": 56.0,
        "uri": "gaia-prato/gw1/Geom/1F/53/humid"
    },
]
}
```

The request in the example will create a AnyCompositeRule linked with 2 SimpleThresholdRules, it will be triggered when at least one of the children rules is true encoding the expression (humid < 56.0 OR temp > 8.0) .

1.2.2 AllCompositeRule

Class: AllCompositeRule

Description

This rule implements a logical AND among the conditions of the linked rules, i.e. the composite is triggered only if all the conditions of the linked rules is true.

Behavior

When fired the composite rule iterates over the linked rules and tests their conditions. If all the conditions are true the action method of the composite is invoked. The children rules are never fired, only their condition is checked, thus the actions of these rules are never executed.

Simplified instantiation

See rule 1.2.1. Use operator “AND” instead of “OR”.

1.2.3 RepeatingRule

Class: RepeatingRule

Description

The rule checks the condition of the linked rule and stores how many times consecutively it is true.
When the child rule is true for more than *threshold* times the action of the linked rule is triggered.

Behavior

When fired, the condition of the linked rule is checked and, if true, a counter is increased.
Then the value of the counter is checked and if it is greater or equal the threshold the action method of the child rule is invoked. The counter value is stored in the database.

Input parameters

See also the parameters for the GaiaRule from which all the rules inherit

Long	threshold	Threshold for the counter value	REQUIRED

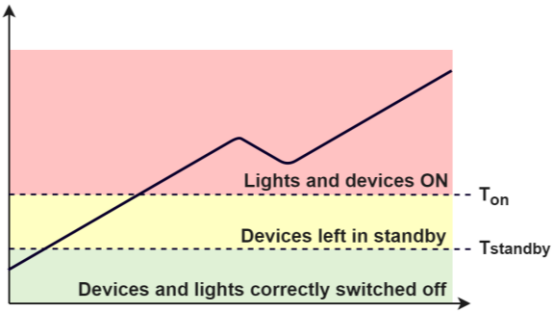
Note

Only ONE rule should be linked to the RepeatingRule composite.

1.3 Don’t waste energy

Class: EnergyWasting

Description



This rule aims at reducing energy consumption waste when an area is not in use (no persons inside).
For instance, in a classroom it means that lights and other equipments should be turned off when leaving. In a laboratory the lighting and the electronic equipment (TVs, DVD players, computers, battery chargers, etc.) should be turned off instead of being left on standby when not in use.
The rule evaluates the following conditions:

a) If the room is not occupied and the active power measurements is greater than T_{on}, a notification is delivered to suggest to switch off lights and/or devices.
b) if the room is not occupied and the active power measurements is greater than $T_{standby}$ and less than T_{on} the event is logged (if the event happens too often the building manager is notified).

Behavior

If the room is occupied return. Else, check the power value, if it is over the *on_threshold* then send a notification to the BM and log the event (normal behaviour). If it is less than the on_threshold but over the *standby_threshold* then log the event and, if in the latest *interval* hours the rule has been triggered more than *n times* then send a notification to the BM.

Input parameters

See also the parameters for the GaiaRule from which all the rules inherit

String	power_uri	Sparkworks relative URI of the sensor or numeric resourceID for Active Power	REQUIRED
String	occupancy_uri	Sparkworks relative URI of the sensor or numeric resourceID for the PIR sensor. If empty or not present the schedule associated with the relative area will be used.	*OPTIONAL
Double	standby_threshold	Standby threshold	REQUIRED
Double	on_threshold	When the ActivePower exceeds this threshold the devices are considered ON	REQUIRED
Integer	interval	Interval in hours in which check the latest triggered events Default: 1	OPTIONAL
Integer	times	Threshold for the latest triggered event Default: 3	OPTIONAL
-	SCHEDULE	*If the occupancy sensor is not available, the schedule of the area is needed	*OPTIONAL

Output

See GaiaRule

Double	power_value	The latest power value measured

Notes

If the system detects that the power required is greater than T_{on} then a notification is sent
If the system detects devices have been left in standby $T_{standby}$ then it logs the event

If the system detects the previous event happening too often (more than *times* in the latest *interval* hours) than a notification is sent

1.4 Exploit natural light

Class: ExploitNaturalLight

Description

The rule checks if the artificial lighting can be switched off without compromising the users' comfort.
Exploiting natural light helps to reduce electricity consumption and decrease eyes fatigue at the end of the day.
Sunlight is also important to maximally synchronize human circadian rhythms.

Behavior

To evaluate the user comfort in terms of luminosity, we need to compare the current luminosity measurement (internal and/or external) with a given threshold (if the measurement is above the threshold, the user comfort is achieved just with natural light).
It is therefore important to define an appropriate value of this threshold in the specific context where this rule is applied (school and area in the school).
If the external luminosity measurement is available, the threshold should be defined considering the position of the room and the variability of the sunlight during the day.
If the room internal luminosity is available, the threshold value should consider that the lights are ON when the rule will be triggered, so you have the sum of artificial lighting plus natural light.
The previous luminosity values can be used together.
Here you can find a reference table for the indoor lighting levels for ordinary activities.
More information can be found here.

Input parameters

See also the parameters for the GaiaRule from which all the rules inherit

String	power_uri	Sparkworks relative URI of the sensor or numeric resourceID for Active Power	REQUIRED
String	luminosity_uri	Sparkworks relative URI of the sensor or numeric resourceID for internal or external luminosity	REQUIRED
String	occupancy_uri	Sparkworks relative URI of the sensor or numeric resourceID for the PIR sensor. If not filled the schedule associated with the relative area will be used	*OPTIONAL
Double	power_threshold	Threshold over which the lights are supposed to be ON	REQUIRED
Double	luminosity_threshold	Threshold over which there should be a good visual comfort. Related to the type of sensor internal / external	REQUIRED
-	SCHEDULE	*If the occupancy sensing is not available we need at least the schedule of the area	*OPTIONAL

Output

See GaiaRule

Double	luminosity_value	
Double	power_value	

Example

The HALL LIGHTING of the Prato school usually consumes 400 W when the building is closed (lights off) and about 6 kW when the school is open (all the lights ON). So we can set a threshold of 5 kW to identify the LIGHTS ON condition.
The figure below shows the internal luminosity profile in the HALL during a typical day. A sensed value of 500 Lux empirically gives a reasonable comfort for this area, so this value can be used as threshold for internal luminosity. This value depends on the position of the sensor, the variability of the luminosity during the day and on the main activities in the area.

1.5 Power Factor

Class: PowerFactor

Description

PowerFactor is the measure of the efficiency of the power being used. A power factor of 1 would mean 100% of the supply is being used efficiently, whereas a power factor of 0.5 indicates that the use of the power is very inefficient. In some countries (e.g. Italy) if the power factor is below a given threshold, the consumer may be required to pay some penalties.
The rule uses 2 thresholds to send different notification according to the gravity of the situation.

Behavior

The rule logs events when the power factor falls below a threshold. The events are recorded to be later retrieved and analysed by the building manager. This information helps deciding if some corrective actions are needed (e.g. install a capacitor).

Input parameters

See also the parameters for the GaiaRule from which all the rules inherit

String	pwf_uri	Sparkworks relative URI of the sensor or numeric resourceID for the Power factor	REQUIRED
Double	pwf_threshold	Threshold for the power factor Default: 0.95	OPTIONAL
Dobule	pwf_lowerthreshold	Lower threshold for the power factor Default: 0.70	OPTIONAL
Integer	windowLength	Number of days used to compute the average power factor value Default: 10 Max: 40	OPTIONAL
String	suggestion_low	Textual suggestion sent when the value is below the pwf_lowerthreshold	OPTIONAL
String	suggestion_base	Textual suggestion sent when the value is below the pwf_threshold	OPTIONAL

Output

See GaiaRule

Double	average_pwf	The average of the pwf value within the specified window

1.6 Holiday shutdown

Class: HolydayShutdown

Description

A significant amount of energy can be wasted during the school holidays (and over weekends and bank holidays) because equipment is not switched off.
Before a holiday period, a tour of all areas should be made, to check (and turning off) all nonessential equipment, for instance:

- Computer monitors left in standby
- Photocopiers / Printers in sleep mode
- Point-of-use water heaters

Behavior

The rule generates a message for the building manager delivered before the holiday to remind him/her to check and switch off devices before the vacation period. The rule is more effective if the building manager has an updated checklist of actions to do.

Input parameters

See also the parameters for the GaiaRule from which all the rules inherit

Long	timeBeforeInHours	The number of hours before the event the system notifies the user	REQUIRED
-	SCHEDULE	The schedule of the area stored in the building db	REQUIRED

Output

See GaiaRule

Example

If the school is closed the week from 14/08 to 18/08 you can set up this rule with a *timeBeforeInHours* of 48
In this way a notification is sent 48 hours before the school closes reminding to switch off all the not required device instead of leaving them in standby etc.

1.7 Schedule Reminder Rule

Class: ScheduleReminderRule

Description

A general rule to schedule a textual reminder into the system.

Behavior

The rule generates a notification for the building manager delivered the specified amount of hours before the date represented in the cron strings.

Input parameters

See also the parameters for the GaiaRule from which all the rules inherit

Long	timeBeforeInHours	The number of hours before the event you want to be notified	REQUIRED
List of Strings	cron	A list of one or more cron strings E.g. [“* * * 15 8 ? *”,”* * * ? * SAT *”]	REQUIRED

Output

See GaiaRule

Example

This rule can be used to remind the building manager to change the rotation direction of the ceiling fans depending on the heating system status and season.

1.8 Temperature Forecast

Class: TemperatureForecast

Description

Heating system may not be able to heat well when temperature suddenly decreases, therefore in such days it is better to wear warmer clothes instead of strain the heating system.
The aim of this scenario is to warn people about the likely sudden decrease of the temperature in the next days, suggesting to wear warmer clothes and suggesting the building manager to decrease the comfort temperature of the heating system by one degree. This should save money and decrease CO_2 emissions. If the building manager is allowed to set the thermostat he/she can modify the temperature when warned by the system, if this cannot be done (e.g. in Prato) the suggestion will be only related to wearing warmer clothes.

Behavior

The condition for this rule can be expressed as follows:
First check if the external temperature of today is the “winter range” (e.g. below 12°).
Then check the difference between the average external temperature of today and the forecast of the temperature for the day after, if the difference is negative and greater than a certain value (e.g. 5°) the rule is triggered.

Input parameters

Double	threshold	Threshold for the temperature difference Default: 8.0	OPTIONAL
String	ext_temp_uri	If a weather station is available provide the identifier for the external temperature. If not provided, a web weather service is to be used to estimate the temperature.	OPTIONAL
Double	coordinates	The rule need the *lat* and *lon* coordinates inside the json field of the building	REQUIRED

Output

Double	temp_today	The temperature of today (from sensors or from the web service)
Dobule	temp_forecast	The temperature forecast (from sensors or from the web service)

1.9 Simple Threshold Rule

Class: SimpleThresholdRule

Description

This rule is fully customizable by the user. It evaluates a simple expression in the form:

```
{value}      {operator}   {threshold}
temperature        ≥            32.0
```

Valid operators are:

>	Greater than
<	Less than
==	Equal*
>=	Greater or equal
<=	Less or equal

*floating point numbers with a difference below 0.001 are considered equal

Input parameters

String	uri	The resource URI or numeric id (represented as string)	REQUIRED
Double	threshold	The threshold	REQUIRED
String	operator	A valid operator	REQUIRED

Output

Double	value	The latest reading of the resource identified by *uri*

Example

Rule to receive an alert when the temperature measured by *gaia-prato/gw1/weather/temp* is below 3.0 degrees.
uri: "*gaia-prato/gw1/weather/temp*"
threshold: 3.0
operator: "<"

1.10 Comfort Index

Class: ComfortIndex

Description

Heat Index specifies how the human body experiences temperature and is based on subjective measurements. However, it is only meaningful above 40% RH and 25 °C. At high temperature and humidity conditions, the comfort factor can be controlled with the concept of Heat Index. The Heat Index in °C is given by:

$$HI = c_{00} + c_{10}t + c_{01}U_w + c_{11}tU_w + c_{20}t^2 + c_{02}U_w^2 + c_{21}t^2U_w + c_{12}tU_w^2 + c_{22}t^2U_w^2$$

- < 30°C: no discomfort
- 30 – 40°C: some discomfort
- 40 – 45°C: great discomfort
- > 45°C: dangerous
- > 54°C: heat stroke imminent
-

Behavior

Input parameters

String	temperature_uri	The resource URI representing the temperature sensor	REQUIRED
String	humidity_uri	The resource URI representing the rel. humidity sensor	REQUIRED
Double	threshold	The threshold Default: 30.0	OPTIONAL

Output

Double	temp	The value of the temperature	
Double	humid	The value of the relative humidity	
Double	index	The computed comfort index	

1.11 CO2 Level

Class: CO2Level

Description

The aim of the rule is to maintain a good level of CO_2 inside an area. This is done by monitoring the level of $CO_2$and, when it exceeds the comfort level (e.g. 1000 ppm), a notification is triggered.
The notification may suggest to open the windows to facilitate the air change (if the external temperature is fair in relation to the internal temperature) or to open the door for a while.
High CO_2 level can influence negatively the comfort, students' learning capabilities and the average school attendance. Furthermore poor air ventilation can raise the risk of illnesses.

Behavior

The condition for this rule can be expressed as follows:
First check if the CO2 inside an area exceeds the comfort level (e.g. 1000 ppm).
The type of suggestion to be sent to users is chosen by evaluating the difference between external and internal temperature, if available (if the difference is below a threshold, open the window, otherwise open the door).

Input parameters

String	co2_uri	The resource URI representing the CO_2 sensor	REQUIRED
String	in_temp_uri	The resource URI representing the indoor temp. sensor	OPTIONAL
String	out_temp_uri	The resource URI representing the outdoor temp. sensor	OPTIONAL
Double	co2_threshold	Default: 1000.0	OPTIONAL
Double	temp_diff_thresh	The threshold (parts per milion) Default: 5.0	OPTIONAL
String	suggestion_base	Temperature measurements not available Default: High CO2 level! Enhance the airflow in the room!	OPTIONAL
String	suggestion_over	Temperature difference over temp_diff_thresh Default: High CO2 level! Open the door to let some clean air in!	OPTIONAL
String	suggestion_below	Temperature difference belowtemp_diff_thresh Default: High CO2 level! Open the window to let some clean air in!	OPTIONAL

Output

Double	co2_value	The value of CO_2 measurements	
Double	temp_diff	The indoor/outdoor temperature difference (if available)	

1.12 Expression Rule

Class: ExpressionRule

Description

This rule evaluates a user defined expression which contains:

- Numbers (floating point)
- Operators (see the following table)
- User defined variables / parameters
- Measurements retrieved from the platform

The expression must return a logical value 1.0 stands for TRUE and 0.0 stands for FALSE

In order to use a parameter you need to specify it in the rule fields (using data type double)
If you add a double field named “threshold” you can use its value inside the expression

To access a measurement you need to specify the resource id or the URI as a string field by following this pattern: desired_name_uri
For instance if you want to access the measurement (identified by the URI gaia/gw1/temp) using temperature as the variable name you can add a field (string) in the rule with the name *temperature_uri* and filling it with the URI gaia/gw1/temp.
It this way you are able to use the variable temperature inside the expression, that will be replaced by the latest value of the measurement.

Note: all the numeric fields of the rule can be used inside the expression

Valid operators:

Operator	Operands	Description	
+	2	Addition	
-	2	Subtraction	
*	2	Multiplication	
/	2	Division	
^	2	Exponentiation	
%	2	Modulo	
&&	2	Logical AND	(returns 1.0 if TRUE, 0.0 if FALSE)
||	2	Logical OR	(returns 1.0 if TRUE, 0.0 if FALSE)
>	2	Greater	(returns 1.0 if TRUE, 0.0 if FALSE)
<	2	Lesser	(returns 1.0 if TRUE, 0.0 if FALSE)
==	2	Equal	(returns 1.0 if TRUE, 0.0 if FALSE)
<=	2	Greater or Equal	(returns 1.0 if TRUE, 0.0 if FALSE)
>=	2	Lesser or Equal	(returns 1.0 if TRUE, 0.0 if FALSE)

Behavior

Input parameters

String	expression	The resource URI representing the temperature sensor	REQUIRED
String	*_uri	URI of the resource you want to use	OPTIONAL
Double	*	Custom parameters name (starting with a letter/underscore)	OPTIONAL

Output

Map	variable2uri	User defined URI	
Map	fields	Additional user defined variables/parameters	

Examples

Expression	Fields

`temp > threshold + bias`	`"temp_uri" : "155262"`
	`"bias" : 23.0`
	`"threshold" : 15.0`

`power < min_t || power > max_t`	`"power_uri" : "gaia/gw1/area3/actpw"`
	`"min_t" : 10000.0`
	`"max_t" : 500000.0`

`humid*c1 + temp*humid*c3 - temp/humid + temp*(c2-humid) > threshold`	`"humid_uri" : "gaia/gw1/area3/humid"`
	`"temp_uri" : "gaia/gw1/area3/temp"`
	`"c1" : 48545.0`
	`"c2" : 1221.0`
	`"c3" : 1.123`
	`"threshold" : 35.0`

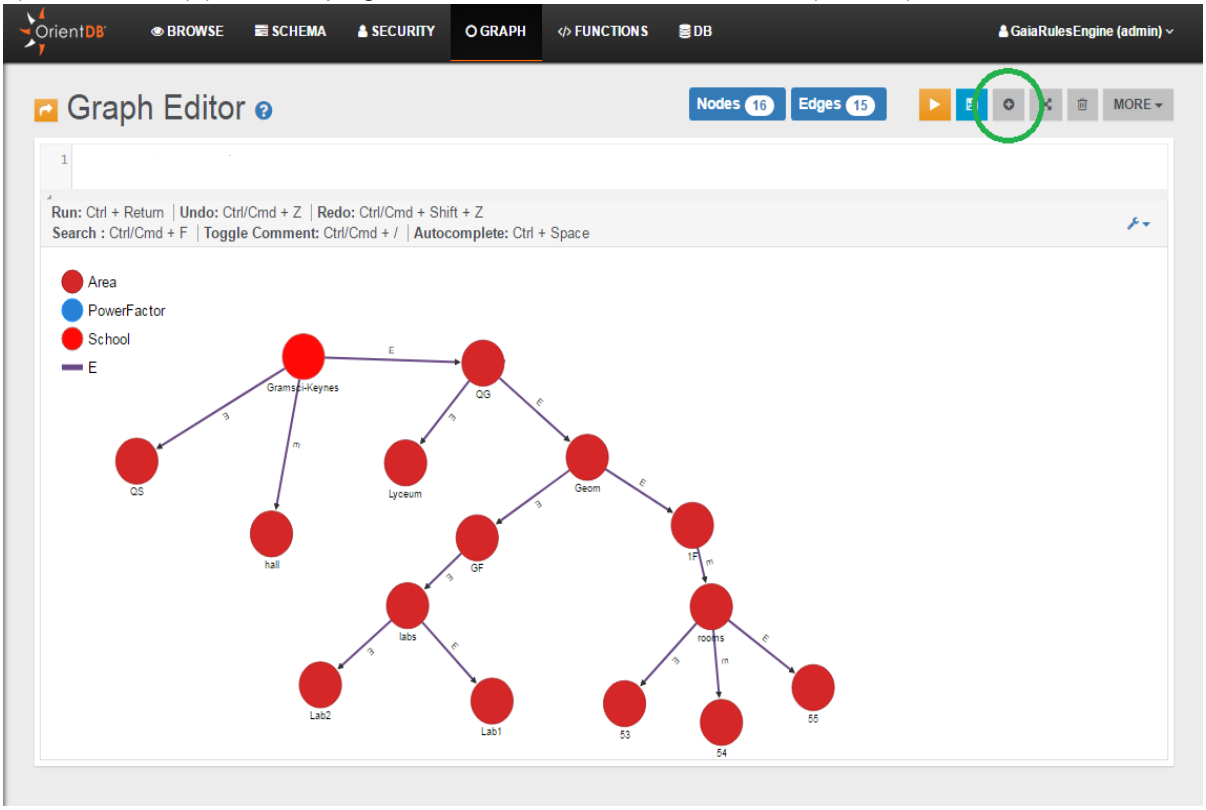
2 How to add a rule

2.1 Using OrientDB GUI

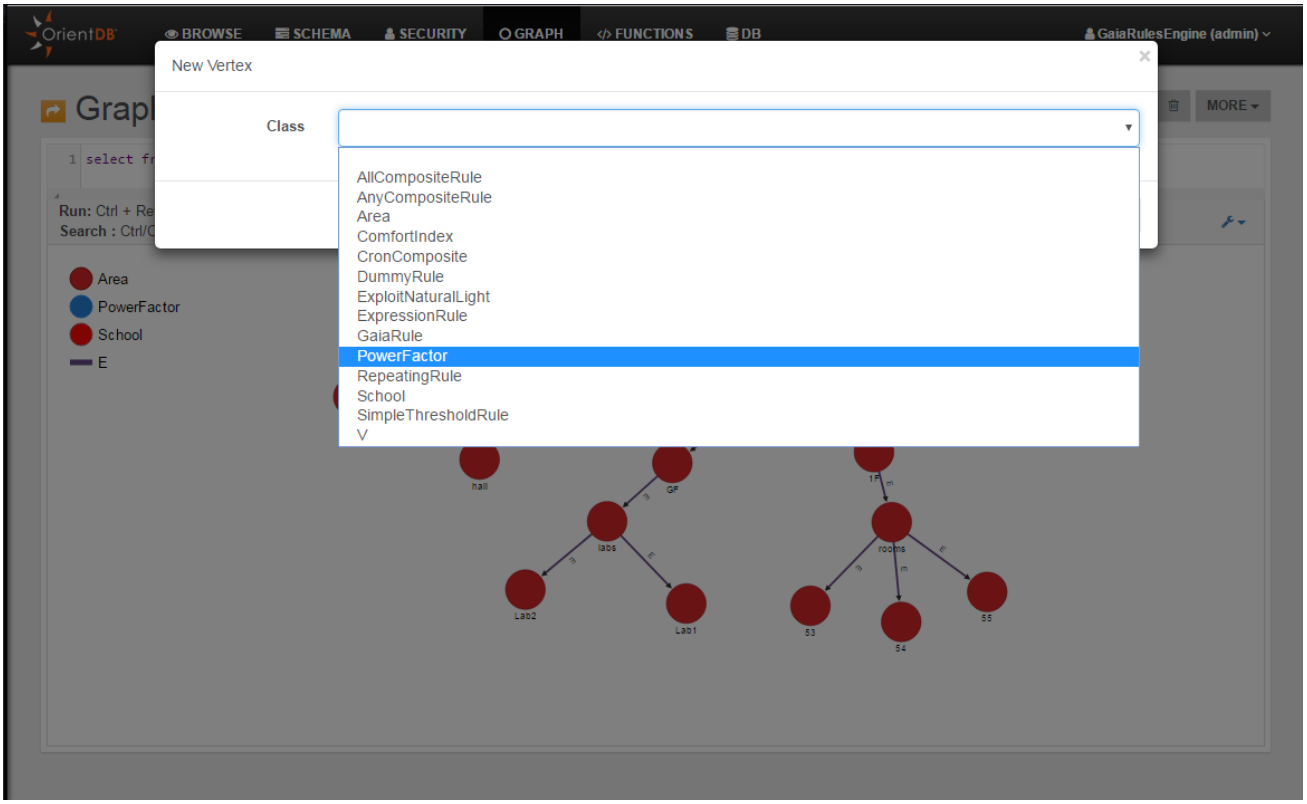
OrientDB Web UI: http://150.140.5.63:2480

1) Login
2) Go to the graph tab
3) Query what you need, for instance
   - query a school by name: *select from School where name = "MySchoolName"*
   - query a school by id: *select from School where aid = 12456*
   - query the whole building tree at once including the rules (complex query)
     *select from (traverse * from (select from School where aid = 155076))*

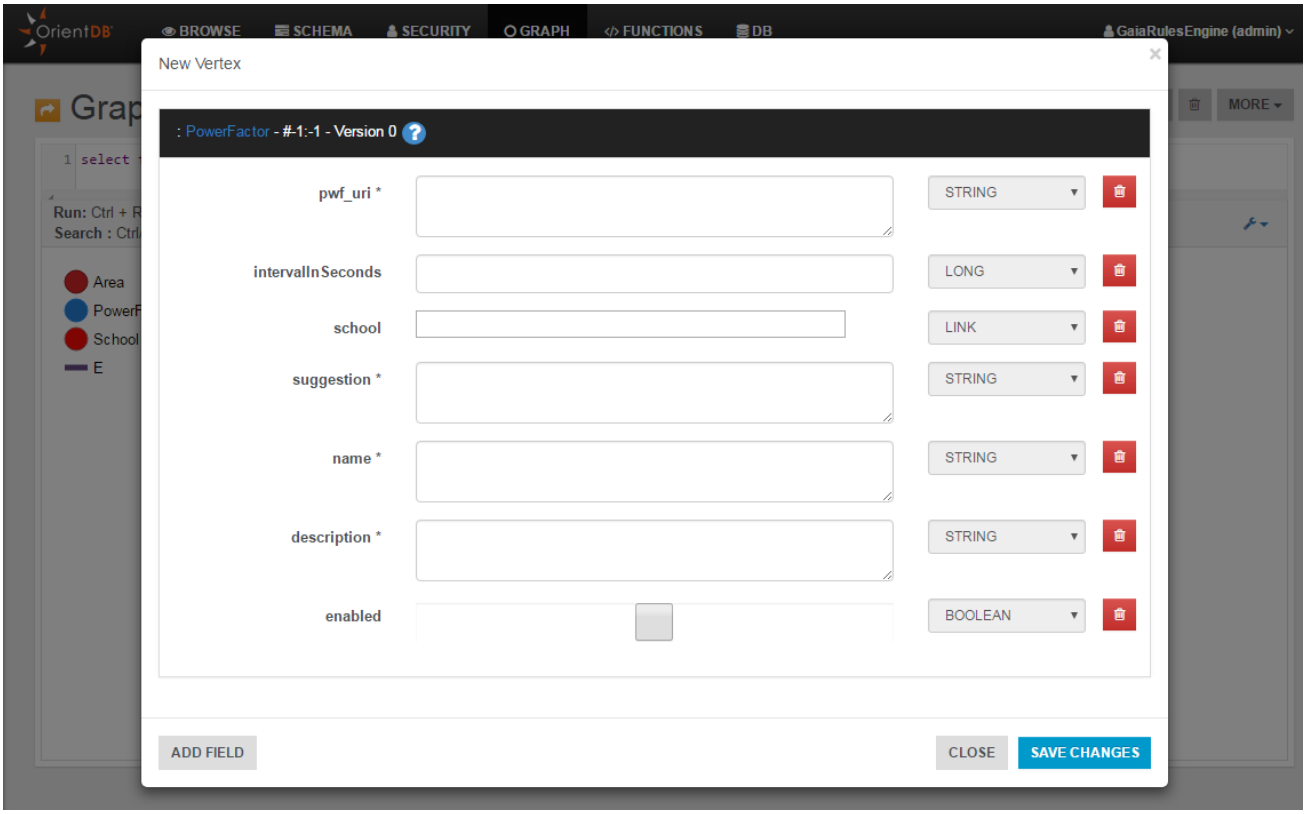
4) Click on the (+) button top right of the screen to add a new vertex (i.e. rule)



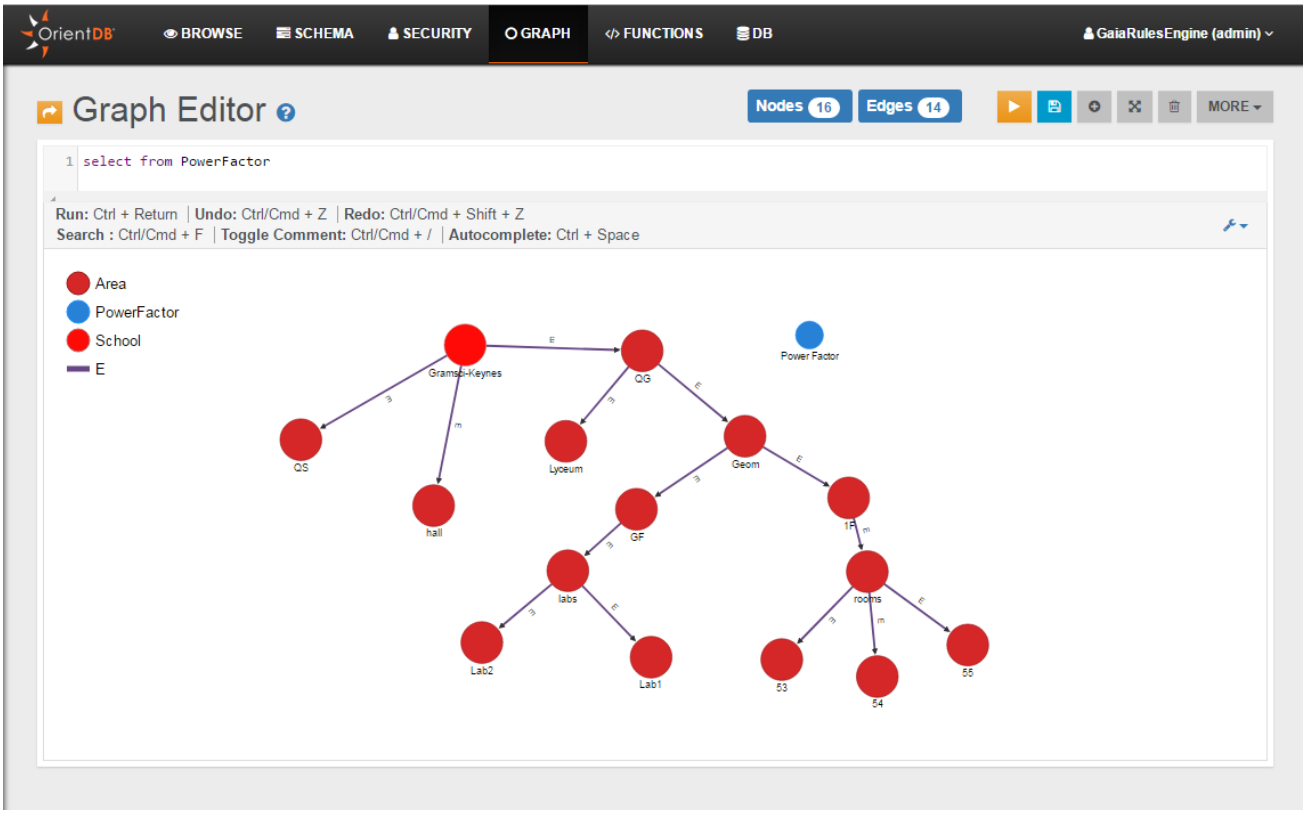
5) Select the class of the rule you want to add (e.g. Power Factor)



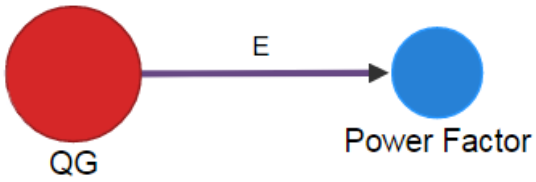
6) Fill in the fields, paying attention on the required fields marked with *



7) Click “Save Changes”, the new vertex should appear in the GUI



8) Click on the area to which you want to connect the rule and then click on  to link the area with newly created Power Factor rule. Link FROM the area TO the rule.
Click Next and Create Edge.



9) The rule will be loaded starting from the next iteration

2.2 Using REST API

To create a new rule POST a json object as specified below to the

`/areas/{aid}/rules`

endpoint substituting {aid} with the identifier of the area you want to link the rule with.
The object contains the class of the rule (SimpleThresholdRule) and the fields (remember that some of them are mandatory).
The documentation of the APIs is available at this link: https://recommendations.gaia-project.eu/docs/.

Generic RuleDTO Object

POST /areas/{aid}/rules

```
{
   "Class": Class of the rule,
   "fields":{
      "name": Custom name ,
      "description": Description,
      "suggestion":Textual suggestion
   }
}
```

For editing or deleting a rule you can use **PUT** and **DELETE** HTTP methods on /rules/{rid}
Where {rid} is the identifier of the rule in the form "#nn:n" e.g. "#23:5"
The user can edit/delete only rules that have been added through this API.

The server will accept also the identifier without the "#" prefix, if you want to use it remember to convert it to a valid HTML entity (link %23).

The Rule Engine exposes utility API for testing the rule:

- /rules/{rid}/condition: for evaluating the condition of a rule
- /rules/{rid}/fire: for firing a rule (check condition and, if true, invoke action)
- /building/{id}/trigger: force firing of the rules connected to a building
-

Hereafter an example using the SimpleThresholdRule.

SimpleThresholdRule Request

POST /areas/{aid}/rules

```
{
   "class": Class of the rule,
   "fields":{
      "name": Custom name ,
      "description": Description,
      "operator": {<,>,==,>=,<=},
      "suggestion":Textual suggestion,
      "threshold": Threshold,
```

```
{
    "class": "SimpleThresholdRule",
    "fields": {
        "name": "MyRuleInstance",
        "description": "This is a description",
        "operator": ">",
        "suggestion": "You can save energy!",
        "threshold": 10.0,
```

"uri": URI or Resource ID } }	"uri": "gaia-prato/gw1/Geom/1F/53/temp" } }

Response

The server will validate the fields by create a test instance of the rule. If it is valid the instance is connected with the area and persisted on the database. It will be fired from the next iteration.
The server will respond with a similar json object with the *rid* additional field representing the identifier of the rule.

```
{
    "rid": "#65:3",
    "class": "SimpleThresholdRule",
    "fields": {
        "name": "MyRuleInstance",
        "description": "This is a
description",
        "operator": ">",
        "suggestion": "You can save energy!",
        "threshold": 10.0,
        "uri": "gaia-
prato/gw1/Geom/1F/53/temp"
    }
}
```

3 Notifications

3.1 WebSocket

The notifications generated by the rules are delivered through a websocket channel in the form of STOMP messages. In the current implementation each school has its own channel: */recommendations/{schoolId}.*
A basic implementation is available at *http://150.140.5.63:8080/demo/index.html*
Fill the *School* text field on the top-left with the id of the building/school you want to receive notification from (e.g., 155076) and press connect.
Our websocket implementation uses STOMP as an application level WebSocket sub-protocol.
Fallback options are supported in order to simulate the WebSocket API where necessary based on the SockJS protocol. The SockJS endpoint is */gs-guide-notification*
More information can be found here.
Hereafter an example of notification:


```
{
    "timestamp": 1498659671498,
    "school": {
        "rid": "#45:47",
        "name": "1ο Γυμνάσιο Ν. Φιλαδέλφειας",
        "type": null,
        "aid": 144242,
        "path": "1ο Γυμνάσιο Ν. Φιλαδέλφειας"
    },
    "area": {
        "rid": "#5:7",
        "name": "Room 2",
        "type": "classroom",
        "aid": 12346,
        "metadata": {"sqmt": 5000.0}
    },
    "ruleClass": "SimpleThresholdRule",
    "ruleName": "name",
    "ruleId": "#63:7",
    "values": {
        "name": "name",
        "threshold": 32,
        "uri": "0013a200409c168b/0xd1b/temp",
        "value": 34.3,
        "operator": ">"
    },
    "description": null,
    "suggestion": "suggestion",
    "type": "info"
}
```

4 Events

4.1 Events logging

The default behavior of a rule when its condition is true is to log this event into the database.
The event contains the main information needed for reconstructing what happened, redundant information are not stored into the database because they can be retrieved through further queries using the APIs.
See APIs for further information about how to access to the events.


```
{
    "timestamp": 1498659671498,
    "rule": "#5:7",
    "values": {
        "name": "name",
        "threshold": 32,
        "uri": "0013a200409c168b/0xd1b/temp",
        "value": 34.3,
        "operator": ">"
    }
}
```

5 Appendix

5.1 Cron expression

Field Name	Allowed Values	Allowed Special Characters
Seconds	0-59	, - * /
Minutes	0-59	, - * /
Hours	0-23	, - * /
Day of month	1-31	, - * ? / L W
Month	1-12 or JAN-DEC	, - * /
Day of week	1-7 or SUN-SAT	, - * ? / L #
Year	empty, 1970-2099	, - * /

***** (“all values”) - used to select all values within a field. For example, “” in the minute field means *“every minute”.

? (“no specific value”) - useful when you need to specify something in one of the two fields in which the character is allowed, but not the other. For example, if I want my trigger to fire on a particular day of the month (say, the 10th), but don’t care what day of the week that happens to be, I would put “10” in the day-of-month field, and “?” in the day-of-week field. See the examples below for clarification.

- used to specify ranges. For example, “10-12” in the hour field means “the hours 10, 11 and 12”.

, used to specify additional values. For example, “MON,WED,FRI” in the day-of-week field means “the days Monday, Wednesday, and Friday”.

Example

** * 8-14 10-15 8 ? 2017* → From August 10th 2017 to August 15th 2017

The rule is fired (the action triggered only if also the condition is true) only if we are in this interval.