ROBMOSYS-1FORC

ROQME

DEALING WITH NON-FUNCTIONAL PROPERTIES THROUGH GLOBAL ROBOT QUALITY OF SERVICE METRICS

The RoQME Data Visualization Toolbox: Installation & User Guide

v1.0_201904302112

Content

1 Introduction to Data Visualization in RoQME

This section describes how the different RoQME data (i.e., contexts, observations and properties) are graphically represented by the RoQME Visualization Toolbox. Each type of data is represented in a different way attending to its nature and taking into account readability criteria. It is worth mentioning that the RoQME Visualization Toolbox allows displaying any number of variables into each chart. Thus, in particular for multi-variable charts, selecting appropriate representations becomes particularly important in order to get graphical plots that are easy to read and interpret.

In order to illustrate how the different types of data managed by RoQME are graphically represented, we will use the example RoQME model included next:

```
1   roqme "IntralogisticsRoboticApplication"
2   param      MAX_V             : number
3   param      MAX_JOB_DONE      : number
4   property   Safety  reference  1
5   property   Performance reference  0.5
6   context    Bump              : eventType
7   context    Velocity          : number
8   context    PersonState       : boolean
9   context    JobState          : enum { NOT_STARTED, STARTED, COMPLETED, ABORTED }
10  context    RobotState        : enum {
11             IDLE_NOT_CHARGING,
12             IDLE_CHARGING,
13             BUSY_DRIVING_WITH_LOAD,
14             BUSY_DRIVING_EMPTY,
15             ERROR
16  }
17  context    TimeJobDone       : time := interval ( JobState::STARTED  -> JobState::COMPLETED )
18  observation O1 : Bump undermines Safety VERY_HIGH
19  observation O2 : Velocity > MAX_V and PersonState  undermines Safety VERY_HIGH
20  observation O3 : JobState::COMPLETED while ( TimeJobDone < MAX_JOB_DONE )  reinforces Performance HIGH
21  observation O4 : RobotState::ERROR undermines Performance
22  observation O5 : JobState::ABORTED undermines Performance
```

This was the model used in the RoQME demonstration developed on the Intralogistics Industry 4.0 Robot Fleet Pilot provided by the Ulm University of Applied Science (HSU).

1.1 QoS Metrics and Observations

The RoQME Tool-Chain generates one graphical representation per QoS metric (`property`) included in the input RoQME model. In this representations, the *x* axis represents time (hh:mm:ss) and the *y* axis represents the value of the corresponding property (always in the range [0, 1]). During the demonstration, the two properties included in the RoQME model (i.e., *Safety* defined in line 4, and *Performance* defined in line 5) were displayed as illustrated next in Figure 1 and Figure 2, respectively.

It is worth noting that relevant observations (i.e., those with impact on each property) are also displayed as points on the *x* axis (x=0). This helps understanding the (positive or negative, higher or lower) impact observations have on each property.

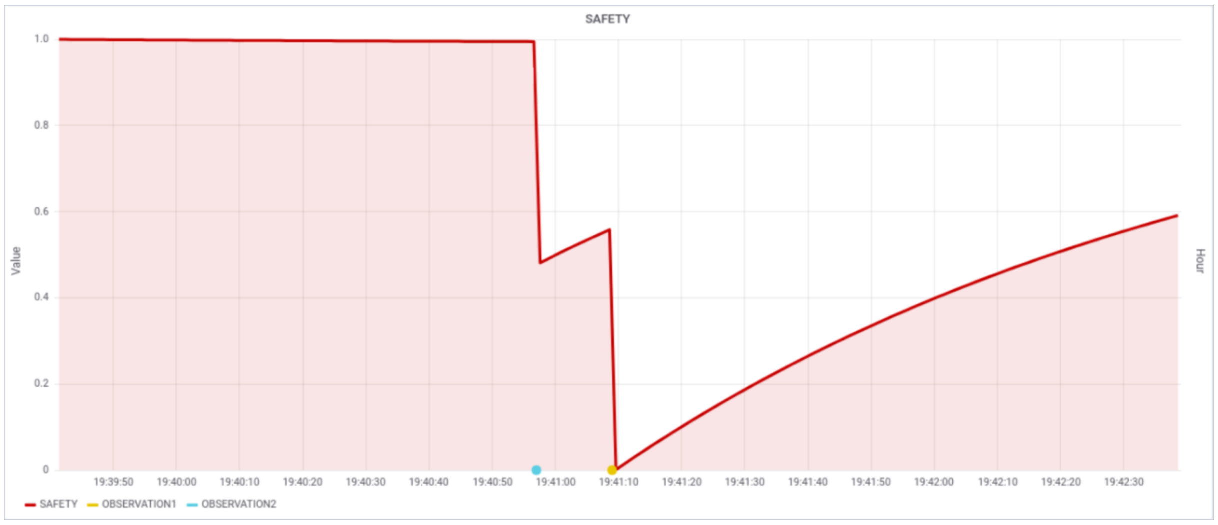


Figure 1. QoS metric defined on Safety and observations 1 and 2

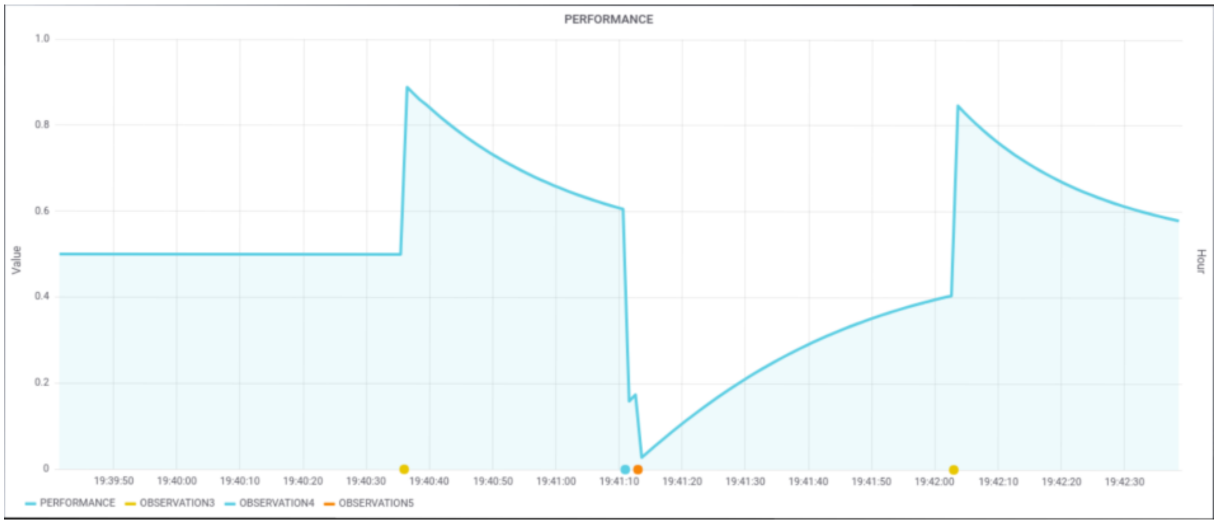


Figure 2. QoS metric defined on Performance and observations 3, 4 and 5.

Observations may be also displayed together with one or more context variables (see next section) to help understanding how particular situations (contexts) give raise to certain observations.

1.2 Contexts

RoQME contexts have different representations depending on their type, i.e., depending on whether they are declared as booleans (`boolean`), events (`eventtype`), enumerated values (`enum`) or numbers (`number`).

1.2.1 Discrete contexts

Contexts of type boolean (`boolean`), event (`eventtype`) and enumeration (`enum`) are managed as discrete variables and thus plotted as points. The *x* coordinate of each point represents time (hh:mm:ss), while the value of *y* depends on the type of the context being displayed: (1) boolean contexts (e.g., ***`PersonState`***, defined in line 8 in the example model) take values *y* = 0 (*false*) or *y* = 1 (true); (2) events (e.g., ***`Bump`***, defined in line 6 in the example model) are plotted at coordinate *y* = 2; and (3) enumerated values (e.g., ***`JobState`***, defined in line 9 in the example model) are plotted at *y* = 3 (see *Figure 3*). This makes it easier to distinguish among the different points displayed in the graph, i.e., booleans, events, enumerations and observations.

Each context of type boolean or event is assigned a different colour. For enumerated contexts, different colours are assigned to each enumeration literal. Thus, each literal is considered and plotted as an independent variable. For instance, consider the ***`JobState`*** enumerated context. This variable may take four possible values: ***`NOT_STARTED`***, ***`STARTED`***, ***`COMPLETED`***, and ***`ABORTED`***. Each of these values is considered an independent variable, displayed using a different colour, and identified with a different label at the bottom of the graph, namely: ***`JobState.NOT_STARTED`***, ***`JobState. STARTED`***, ***`JobState.COMPLETED`***, and ***`JobState.ABORTED`*** (see *Figure 3*).

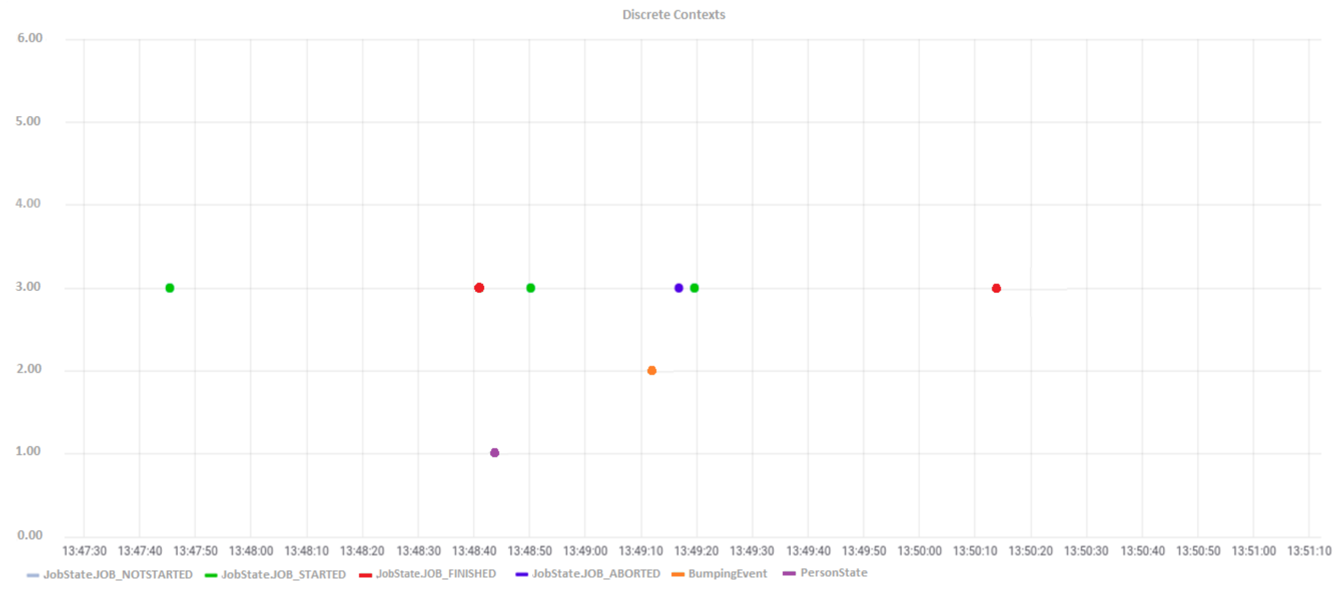


*Figure 3. Graph displaying some of discrete contexts included in the example model, namely: (1) the **`JobState`** `enum` context managed as four independent variables (one per literal), all of them displayed (using different colours) with value y = 3; (2) the **`Bump`** `eventtype` context, displayed with value y=2; and (3) the **`PersonState`** `boolean` context, displayed with values y = {0, 1}.*

As illustrated in *Figure 3*, during the demonstration:

- The ***JobState*** context took values ***JobState.JOB_STARTED*** (twice at 13:47.46 and 13:49:19), ***JobState.JOB_ABORTED*** (only once, at 13:49:16), and ***JobState.JOB_FNISHED*** (twice at 13:48.41 and 13:50:14);
- A ***Bump*** event was identified at 13:49:11; and
- The ***PersonState*** boolean context was set to true once (at 13:48:44) and never changed to false throughout the experiment.

1.2.2 Continuous contexts

Numeric contexts (**number**), e.g., the ***Velocity*** context included in the example RoQME model in line 7, are plotted as continuous variables. Numeric contexts may take values in quite different ranges. Thus, as the graph adapts the range of visualized data to the minimum and maximum values that need to be displayed, combining several continuous contexts, moving in very different ranges, into a single graph may lead to difficult to read and interpret plots. As a consequence and, for the sake of readability, the RoQME Tool-Chain generates an individual graphical representation for each of these contexts. The graph generated during the demonstration for the ***Velocity*** context is included next in *Figure 4*.

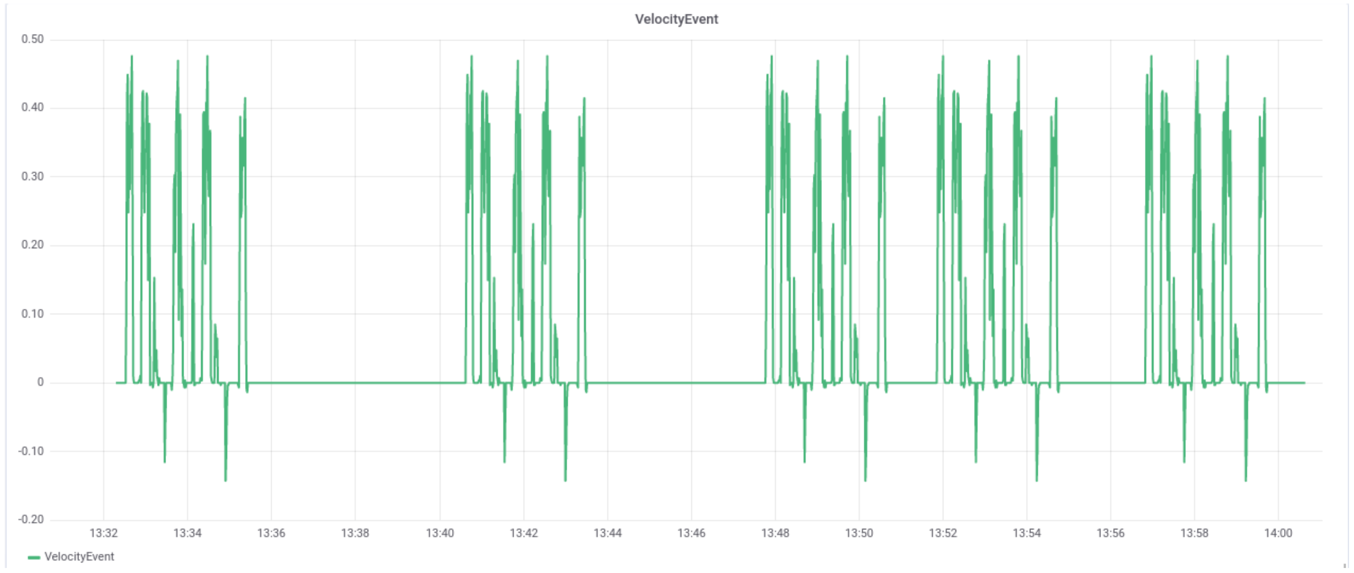


Figure 4. The ***Velocity*** ***number*** *context.*

2 Structure of the RoQME Data Visualization Toolbox

As described in the previous section, the RoQME Data Visualization Toolbox (DaViT) aims at graphically displaying the different types of data managed by RoQME at runtime, i.e., contexts, observations and QoS metrics (a.k.a. properties). These data are directly obtained from the *RoQME QoS Metrics Provider* component as DaViT is subscribed to the information published in the *RoQME Data Space* (based on DDS middleware) by the *Context Monitor*, the *Complex Event Processor* and the *Probabilistic Reasoner*. Apart from real-time data visualization, DaViT also allows recording these data so that they can be later visualized off-line. In fact, the DaViT recording tool (*Recorder*) stores the data obtained from the RoQME Data Space in InfluxDB[1], a non-SQL database that then feeds the visualization tool, based on Grafana[2]. In order to achieve platform-independence, both InfluxDB and Grafana are deployed using Docker[3].

Figure 5 illustrates the relationship existing among the different elements integrating DaViT (i.e., the *Recorder*, InfluxDB and Grafana) and the RoQME QoS Metric Provider Component that feeds DaViT with the input data to be visualized/stored.

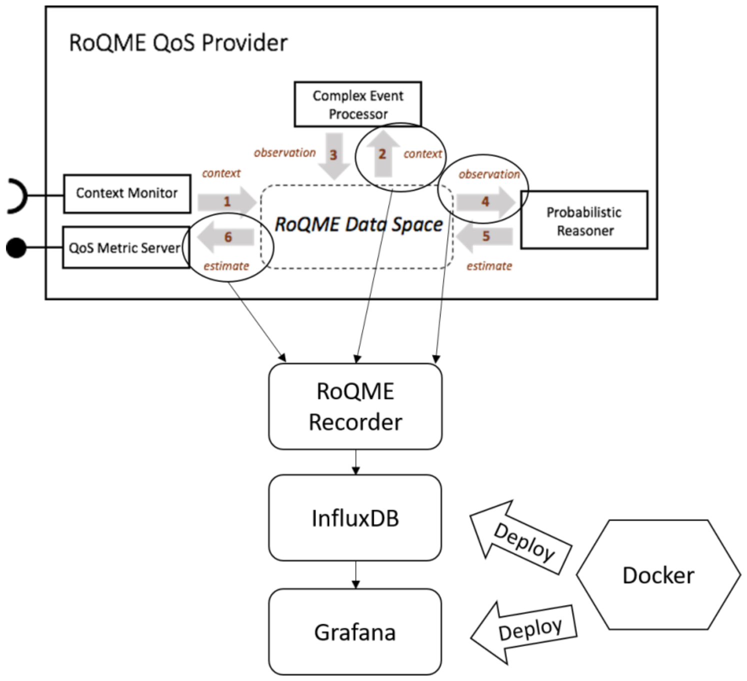


Figure 5. Relationship existing between the main DaViT elements and the RoQME QoS Metric Provider Component.

[1] **InfluxDB** (https://www.Influxdata.com/) is an open-source time series database, optimized for fast, high-availability storage and retrieval of time series data in fields such as operating monitoring, IoT and real-time analytics.

[2] **Grafana** (https://grafana.com/) is an open-source, general purpose dashboard and graph composer that runs as a web application.

[3] **Docker** (https://www.docker.com/) supports operating-system-level virtualization, allowing the execution, in a single operating system kernel, more lightweight than conventional virtual machines, of several software packages ("containers") bundled to their own tools, libraries and configuration files.

3 Installation and Configuration Instructions

This section details how to install and configure InfluxDB and Grafana (from Docker) and the RoQME Recorder.

3.1 Deploying InfluxDB and Grafana from Docker

The instructions included next describe the process that need to be followed to deploy InfluxDB and Grafana from Docker in Ubuntu.

Step 1: Install Docker[4].

- Update the apt package index.

```
$ sudo apt-get update
```

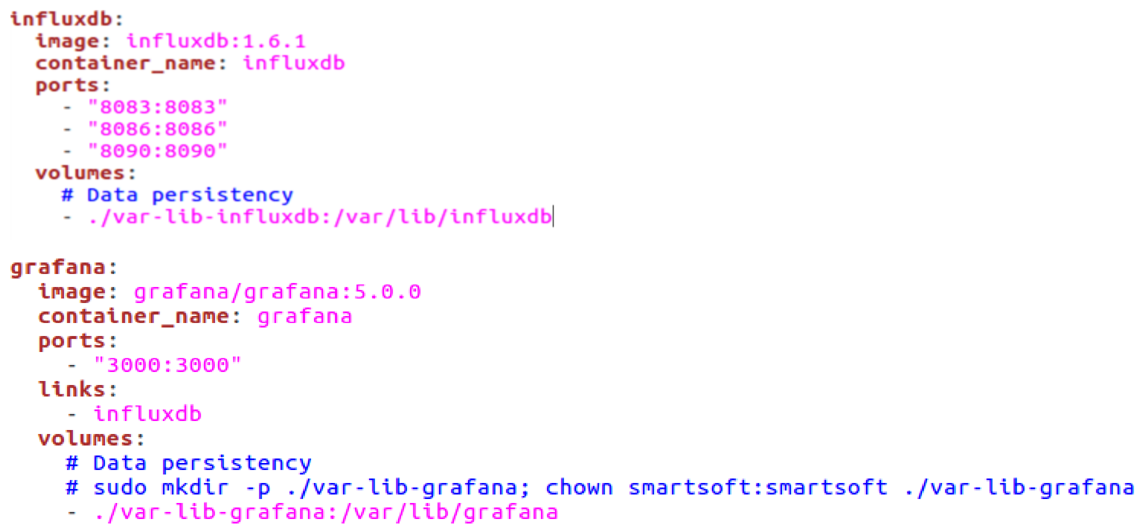
- Install the latest version of Docker CE, or go to the next step to install a specific version:

```
$ sudo apt-get install docker
```

- Install Docker-compose:

```
$ sudo apt-get install docker-compose
```

Step 2: Check the content of the Docker-componse.yml[5] file



```
influxdb:
  image: influxdb:1.6.1
  container_name: influxdb
  ports:
    - "8083:8083"
    - "8086:8086"
    - "8090:8090"
  volumes:
    # Data persistency
    - ./var-lib-influxdb:/var/lib/influxdb

grafana:
  image: grafana/grafana:5.0.0
  container_name: grafana
  ports:
    - "3000:3000"
  links:
    - influxdb
  volumes:
    # Data persistency
    # sudo mkdir -p ./var-lib-grafana; chown smartsoft:smartsoft ./var-lib-grafana
    - ./var-lib-grafana:/var/lib/grafana
```

[4] https://docs.docker.com/install/linux/docker-ce/ubuntu/#upgrade-docker-ce

[5] **IMPORTANT:** The Docker-compose.yml file is generated by the RoQME Tool-Chain and is available in the following project (local folder): QoSMetricsProvider/utils/viewer.

Step 3: Start docker-compose

- In the terminal, move to the yml file directory and execute the following command:

```
$ sudo docker-compose up
```

  InfluxDB and Grafana are deployed from Docker (the terminal window should display the following messages):

```
…
Starting influxdb
Starting grafana
```

- Now, InfluxDB and Grafana are ready to be used from DaViT. Note that Grafana will be available at: localhost:3000.
- In order to stop running Grafana and Influx execute the following commands:

```
$ sudo docker-compose stop
$ kill {PID}
```

3.2 Launching the RoQME Recorder

In order to visualize the data corresponding to a given experiment, either obtained in real-time from the *RoQME QoS Metrics Provider* component or from a dataset previously stored in InfluxDB, we first need to run the RoQME Recorder.

The RoQME Recorder is available in GitHub at: https://github.com/roqme/robmosys-roqme-itp/blob/master/src/roqme.generator/files/roqme-recorder/RoQMERecorder.jar.

In order to run the RoQME Recorder, we will need to execute the following commands:

```
$ cd RoQME_Recorder_Directory
$ java -jar RoQME_Recorder.jar
```

The RoQME Recorder looks as illustrated next in Figure 6.

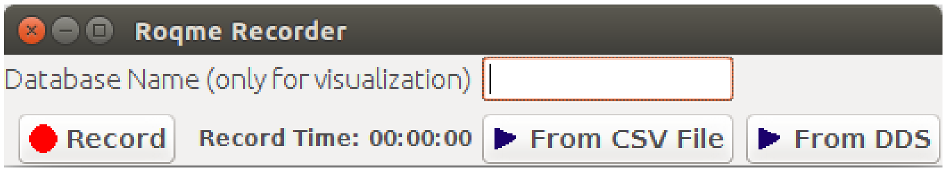


Figure 6. The RoQME Recorder

In order to start receiving and visualizing the data generated in real-time by a RoQME *QoS Metrics Provider* component, we must choose a name for the new database and click on the "From DDS" button. From that moment on, the Recorder will receive all the data published in the RoQME *Data Space* and store them in InfluxDB which will, in turn, make them available to Grafana. Grafana will then display the received data as illustrated in **¡Error! No se encuentra el origen de la referencia.**.

Further information on how to save and reproduce experiments with the RoQME Recorder can be found in Appendix A.

3.3 Configuring the Grafana Dashboard

Once the RoQME Recorder has been connected to a real-time or an off-line data source, being recorded in InfluxDB, we can add that dataset to Grafana and import the corresponding dashboard.

Step 1: Login into Grafana

After deploying Grafana from Docker (see Section 3.1, Step 3) it will be available at localhost:3000.

The Grafana login page looks like the one displayed in Figure 7. The default user and password is *admin*. Entering these credentials, we will access the Grafana control panel.

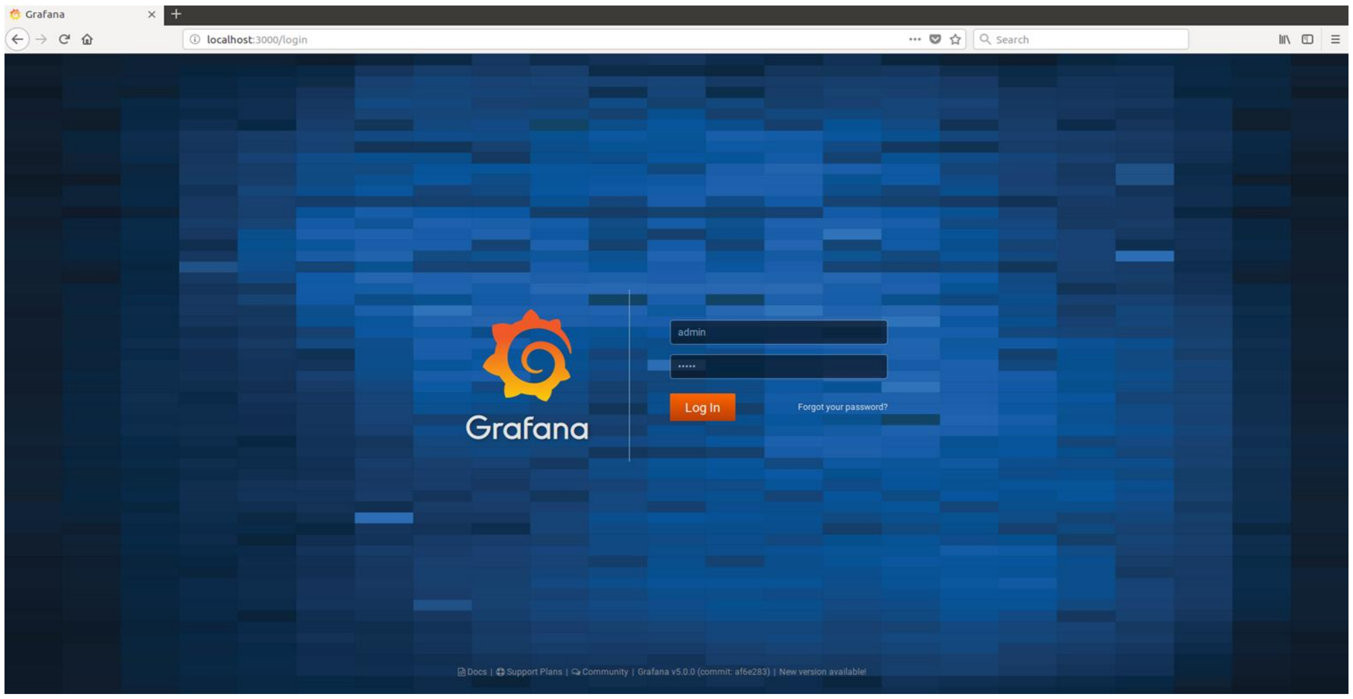
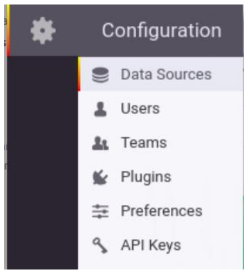

Grafana
localhost:3000/login
Search
Grafana
admin
•••••
Log In
Forgot your password?
Docs | Support Plans | Community | Grafana v5.0.0 (commit: af6e283) | New version available!

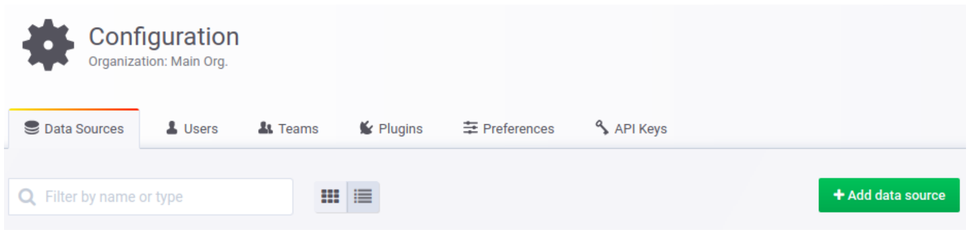
Figure 7. Grafana login page

Step 2: Adding and configuring a data source

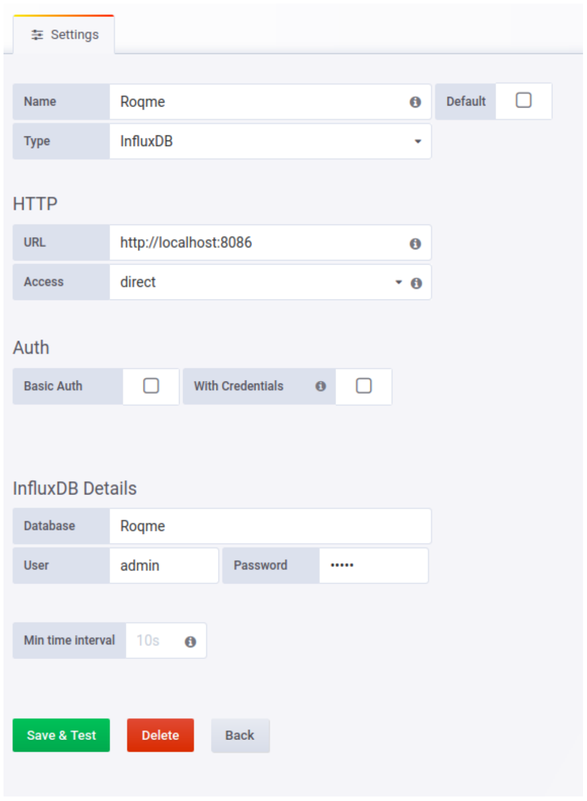
- In order to open the Data Source Panel, select: *Configuration* → *Data Sources*.



- Click on "+Add data source" (green button at the right-bottom corner of the Data Source Panel included below).



- A new "Settings" panel will be displayed (see the figure included below) enabling the configuration of the new data source.



Name	Roqme	Default	☐
Type	InfluxDB		
URL	http://localhost:8086		
Access	direct		
Basic Auth	☐	With Credentials	☐
Database	Roqme		
User	admin	Password	•••••
Min time interval	10s		

- Data source parameters must be configured as follows:

  Name: Roqme (or the name given to the database in RoQME Recorder)

  Type: InfluxDB

  HTTP

  URL: http://localhost:8086

  Access: direct (or browser, depending on the version of Grafana being used)

  InfluxDB Details

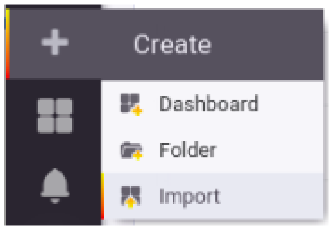
  Database: Roqme (or the name given to the database in RoQME Recorder)

  User: admin

  Password: admin
- To conclude the configuration of the data source, click on "Save & Test" (green button at the left-bottom corner of the "Settings" panel).
- If everything was OK, we will receive a confirmation message, or an error message otherwise.

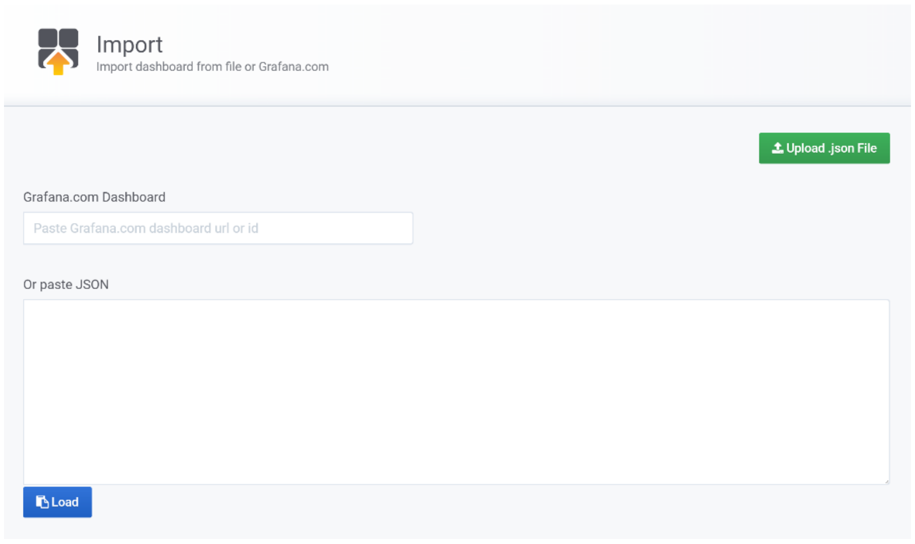
Step 3: Importing existing dashboards

- This option requires having a JSON file with a Grafana dashboard configuration. The RoQME Tool-Chain[6] generates such a JSON file for each RoQME model, describing the default Grafana dashboard. This dashboard will include: (1) one plot representing each QoS Metric (property) and the related observations; (2) one plot representing all the discrete contexts; and (3) one plot representing each continuous context (see Section 0 for further details).
- In order to open the Grafana Import Panel, select: Create (+) → Import.

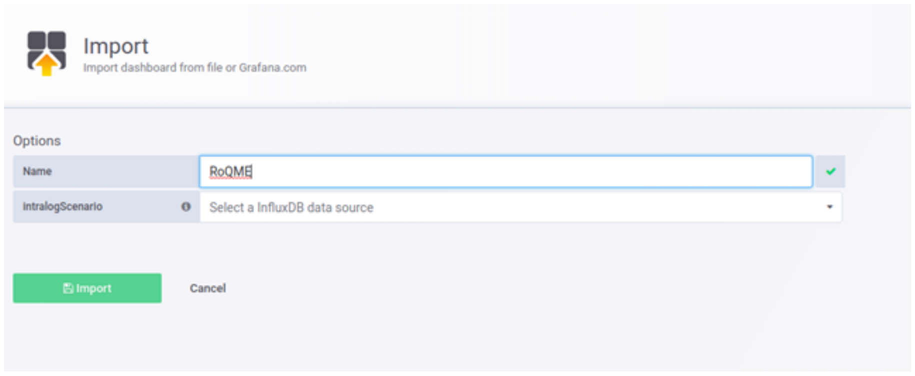


- In the “Import” panel (displayed below): (1) click on "Upload JSON File”; (2) select the JSON file generated by the RoQME Tool-Chain; and (3) press the “Load” button.

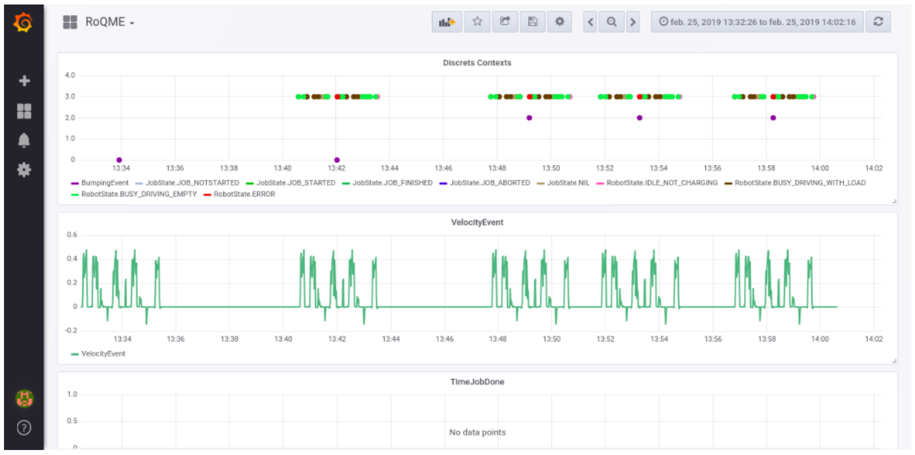
[6] **IMPORTANT:** It is worth mentioning that all the artifacts generated by the RoQME Tool-Chain related to the Data Visualization Toolbox (e.g., the JSON file associated to the generated Grafana dashboard or the Docker .yml file), are generated in the following project (local folder): QoSMetricsProvider/utils/viewer.


Import
Import dashboard from file or Grafana.com
Upload .json File
Grafana.com Dashboard
Paste Grafana.com dashboard url or id
Or paste JSON
Load

- Next, click on "Select a InfluxDB data source", select the name of data source previously created in step 2, and click on "Import" (green button on the left-bottom corner of the Import Panel).


Import
Import dashboard from file or Grafana.com
Options
Name
RoQME
IntralogScenario
Select a InfluxDB data source
Import
Cancel

- From this moment on, the data received from the InfluxDB data source is plotted in the Grafana dashboard (see the example included below).



Further information on how to create and configure Grafana dashboards and graphs can be found in Appendix B.

4 Known issues

After resetting the PC, it is possible that the Graphana port (3000) is marked as busy. If this is the case, we must kill the process using that port. In order to find the PID of that process, execute the following command in a Terminal window:

```
$ sudo top
```

Annotate the PID of the process that keeps the port 3000 busy (usually the *grafana-server* process) and kill it executing the following command:

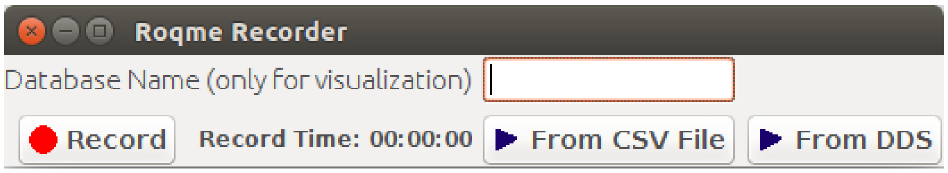
```
$ sudo kill {PID}
```

Appendix A: Saving & Reproducing Experiments with the RoQME Recorder

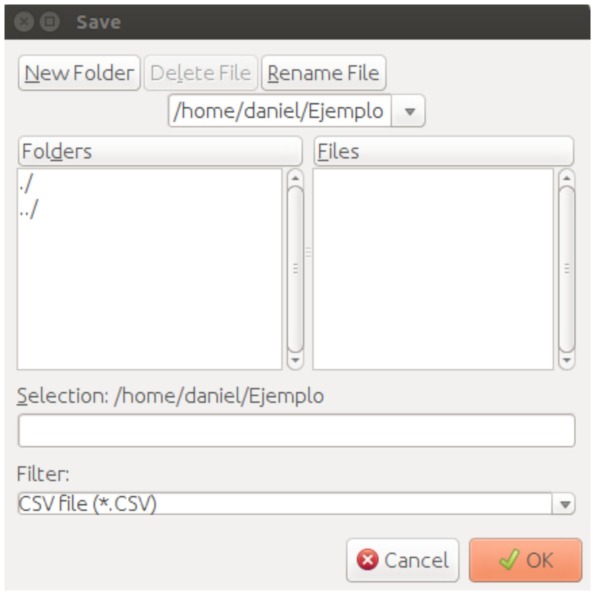
This appendix provides further information about the RoQME Recorder. As previously described in Section 3.2, the RoQME Recorder allows connecting the *RoQME QoS Metrics Provider* component (namely, the data published by its internal modules in its DDS-based RoQME Data Space) with InfluxDB which, in turn, makes these data available to Grafana for visualization purposes.

However, the RoQME Recorder also allows storing all the data gathered throughout an experiment, saving them in CSV format. The generated CSV file can then be used later to reproduce the experiment off-line.

As introduced in Section 3.2, the RoQME Recorder is provided as an executable JAR file, available in GitHub at: [https://github.com/roqme/robmosys-roqme-itp/blob/master/src/roqme.generator/files/roqme-recorder/ RoQMERecorder.jar](https://github.com/roqme/robmosys-roqme-itp/blob/master/src/roqme.generator/files/roqme-recorder/RoQMERecorder.jar). When executed, the RoQME Recorder looks as follows:



In order to save the RoQME data associated to a given experiment in a CSV file, you must click the "Record" button. The following panel will open, allowing you to select the target folder and file name for your CSV file.



As soon as you press the "OK" button, the caption of the “Record” button will be changed to “STOP” and all the data received from the *RoQME QoS Metrics Provider* component will be stored in the selected CSV file. Data will be saved in the CSV file until you press the "STOP" button.

Saving data in a CSV file is not incompatible with simultaneously publishing the real-time data in InfluxDB (by pressing the "From DDS" button) so that they can be visualized in Grafana, as previously detailed in Sections 3.2 and 3.3.

In the CSV file, each new data received from the RoQME Data Space will be inserted into a new line, including different information depending on the type of the data:

- **Contexts**: [Date],[ContextName],[Value],[TypeOfContext]
- **Observations**: [Date],[ObservationName],[BlankSpace],*Observation*
- **QoS Metric estimation**: [Date],[PropertyName],[Value],*Estimate*

All lines include 4 comma-separated values. The last one (i.e, the [TypeOfContext], or the literals *Observation and Estimate*) allow distinguishing among the different types of data stored in the CSV file.

In order to reproduce an experiment, previously stored in a CSV file, you must execute the RoQME Recorder and press the "From CSV File" button. You will be prompt to select the CSV file containing the data of the desired experiment. Note that only correct CSV files (i.e., structured according to the convention previously defined) are allowed.

Once you select a CSV file, you will be prompt to select a name for the InfluxDB database that will be created to store the data and make it available to Grafana. In case you do not provide a name for the database, it will be called "RoQME" by default.

IMPORTANT: Before reproducing any experiment, both InfluxDB and Grafana need to be deployed using Docker, as detailed in Section 3.1.

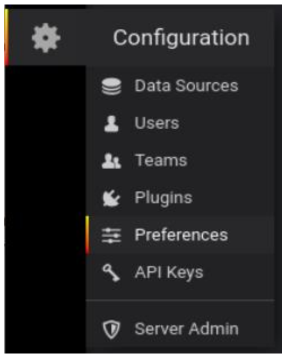
Appendix B: Grafana User Manual

This appendix provides further details about how to create, delete, modify and import dashboards, how to create and configure different graphs, how to manage existing data sources, and how to install new plugins in Grafana.

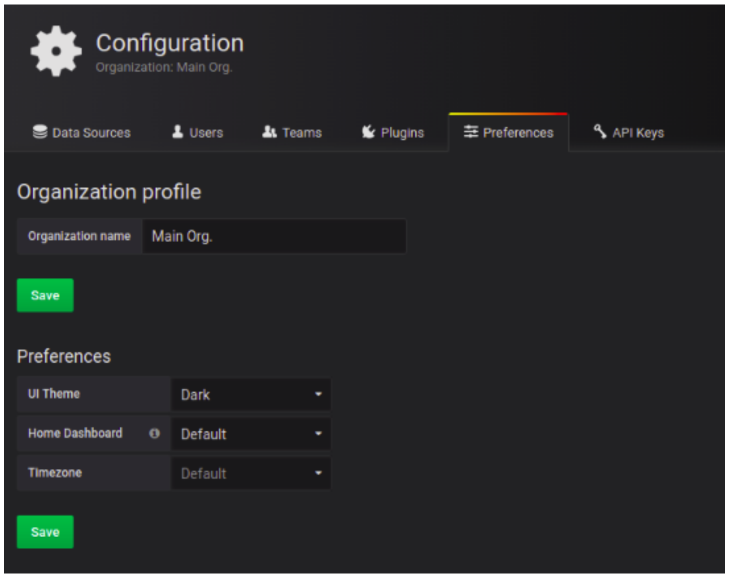
B.1. Changing the Grafana UI Theme

The Grafana User Interface (UI) can be easily customized (e.g., selecting a different background colour for all Grafana windows) by selecting and applying one out of the existing Grafana UI Themes. To change the UI Theme:

- Select the following menu options: *Configuration* → *Preferences*.



- The following "Configuration" panel will be displayed → Change the "UI Theme" e.g., from "Dark" to "Light" and click on the "Save" button. From that moment on, the Grafana windows will be displayed using a white (instead of a black) background.



Configuration
Organization: Main Org.

Organization profile

Organization name: Main Org.

Preferences

UI Theme: Dark

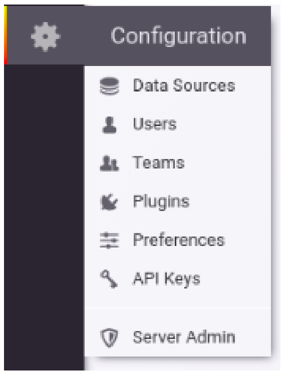
Home Dashboard: Default

Timezone: Default

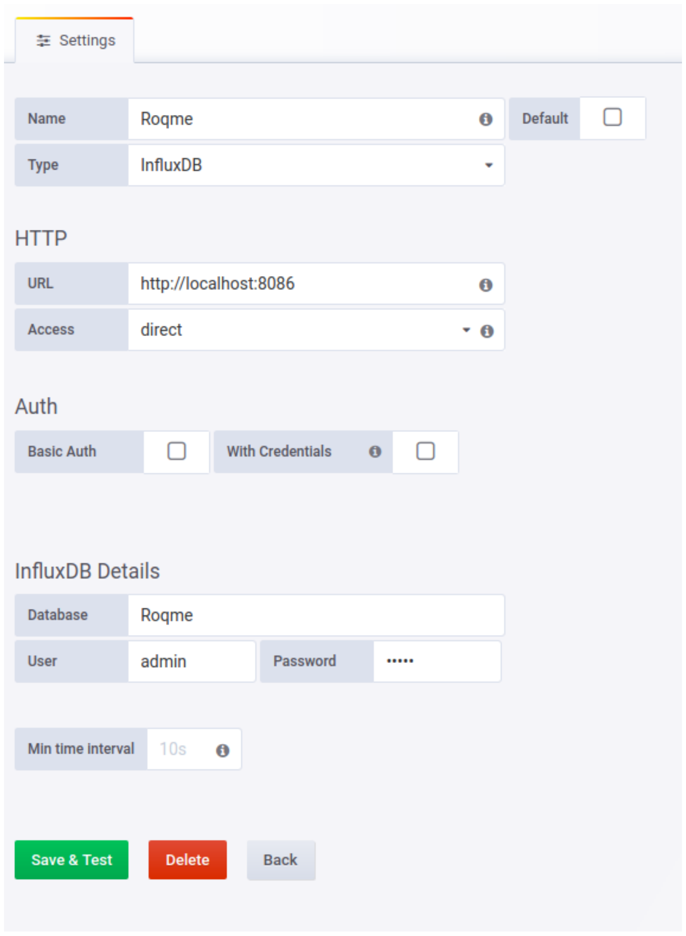
B.2. Adding a New Data Source

The dataset to be displayed in Grafana, needs to be configured at the very beginning.

- Select the following menu options: *Configuration → Data Sources*.



- Configure the different options available in the "Settings" panel as follows:



Name: Roqme
Default: ☐
Type: InfluxDB

HTTP

URL: http://localhost:8086
Access: direct

Auth

Basic Auth: ☐
With Credentials: ☐

InfluxDB Details

Database: Roqme
User: admin
Password: •••••

Min time interval: 10s

Name: Roqme (or the name given to the database in RoQME Recorder)

Type: InfluxDB

HTTP

URL: http://localhost:8086

Access: direct (or browser, depending on the version of Grafana being used)

InfluxDB Details

Database: Roqme (or the name given to the database in RoQME Recorder)

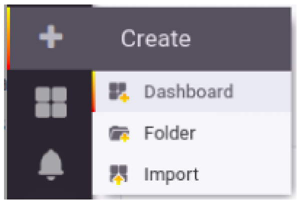
User: admin

Password: admin

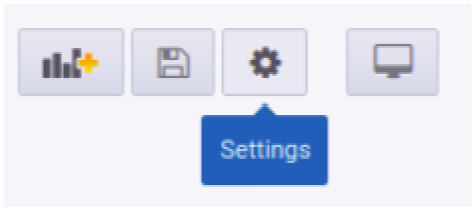
- To conclude the configuration of the data source, click on "Save & Test" (green button at the left-bottom corner of the "Settings" panel).
- If everything was OK, we will receive a confirmation message, or an error message otherwise.

B.3. Creating a New Dashboard

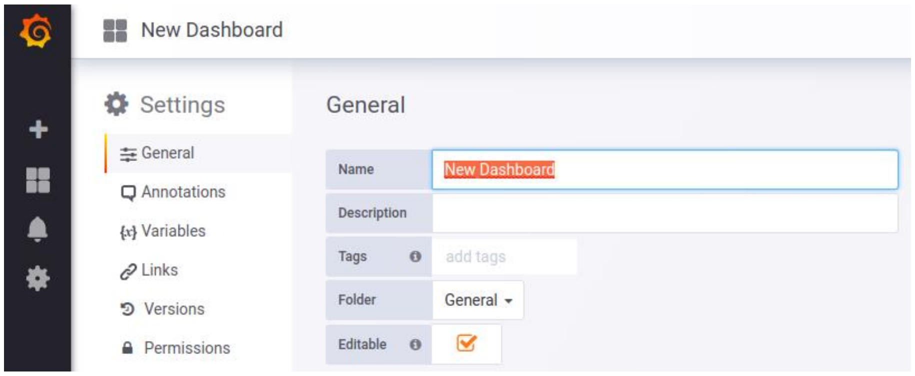
- Select the following menu options: *Create* → *Dashboard*.



- In order to set the name for the new dashboard click on the “Settings” button.

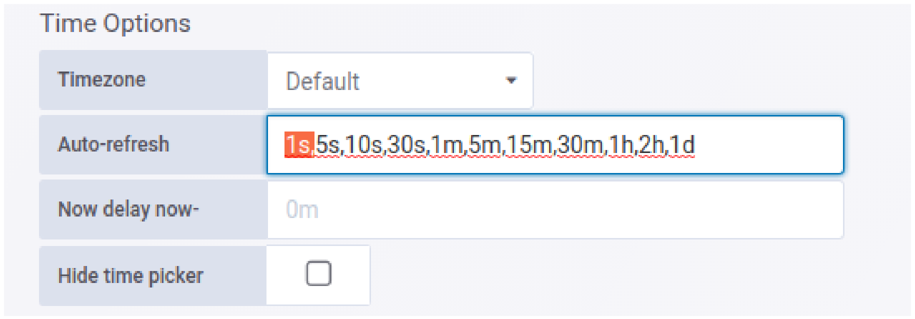


The following “New Dashboard” panel will be displayed, allowing you to set the name for the new dashboard (“Name” field in the “General” tab) and some additional configuration options.



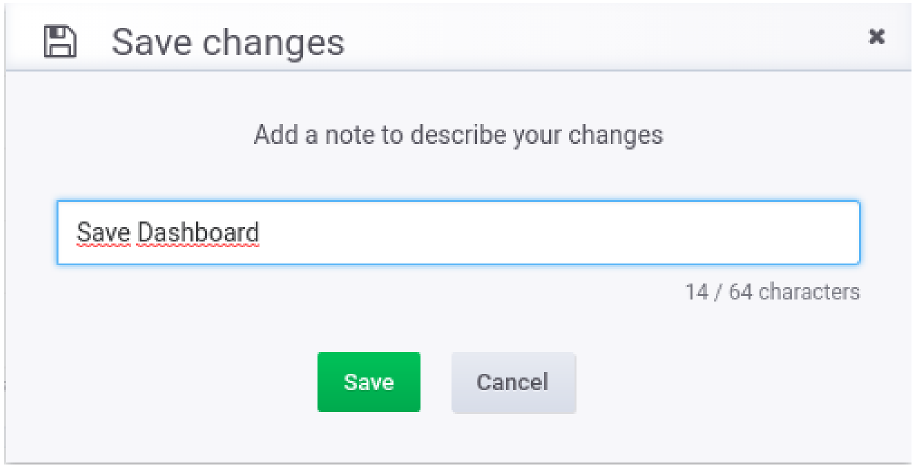
Name	New Dashboard
Description	
Tags	add tags
Folder	General
Editable	✓

- Scrolling down the "General" tab you can also see the available frequencies at which data may be refreshed. We recommend you to add the value "1s" to the list of values already available in the "Auto-refresh" field. See Section B.6 to learn how to select the most appropriate auto-refresh frequency from this list for each graph included in the active dashboard.



Timezone	Default
Auto-refresh	1s,5s,10s,30s,1m,5m,15m,30m,1h,2h,1d
Now delay now-	0m
Hide time picker	☐

- Finally, click on the "Save" button to apply the changes.



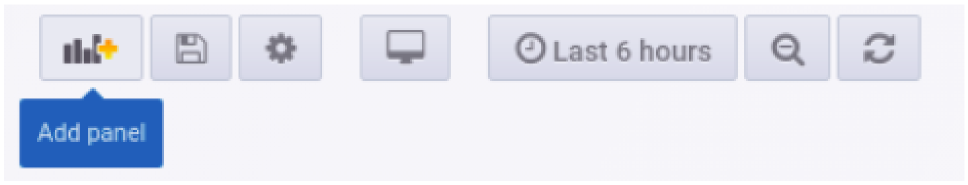
Add a note to describe your changes

Save Dashboard

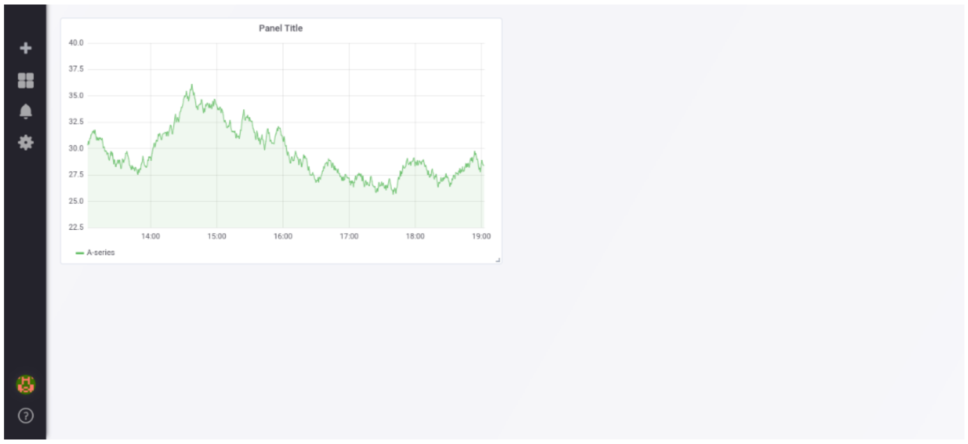
14 / 64 characters

B.4. Creating New Graphs

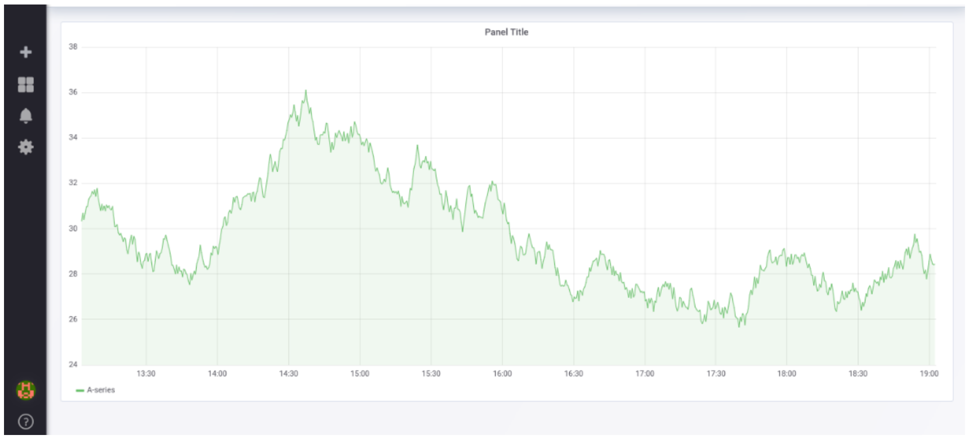
- In order to create new graphs to display one or more variables stored in an InfluxDB dataset, select the "Add Panel" menu option. **IMPORTANT:** graphs newly created will be automatically added to the active dashboard in Grafana.



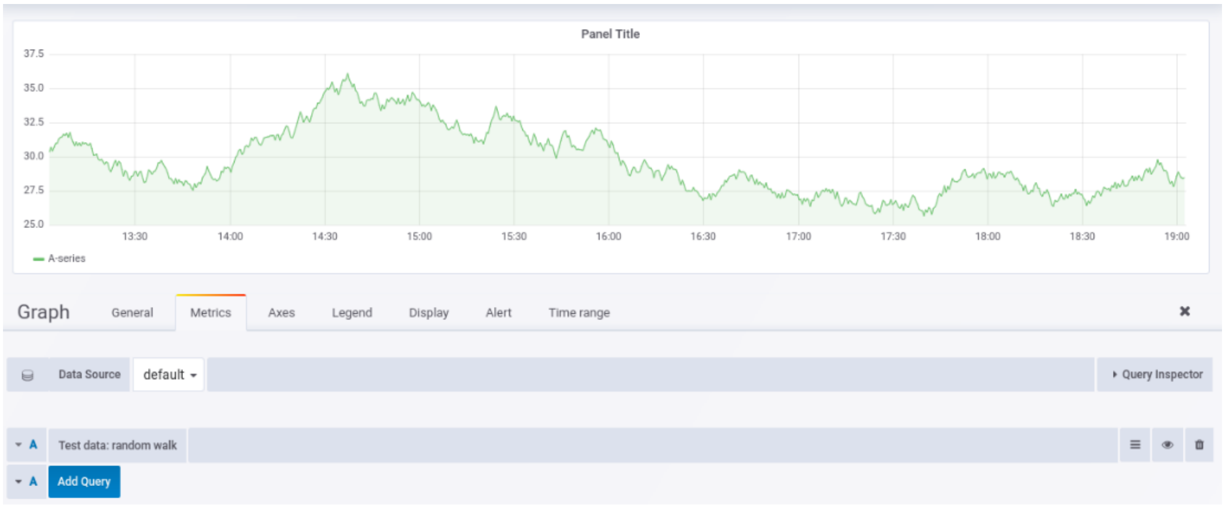
- Select your preferred type of graph. The graph displayed below is of type "Graph".



- When a new graph is created, its size and name will be initialized by default, and it will display random values.
    - In order to change the size of the new graph, click on "Panel Title" (at the top of the graph) → "View" and you will be allowed to resize the graph.



- In order to change the name of the graph, click on "Panel Title" → "Edit" → "General", and fill in the "Title" field.
- Finally, in order to visualize in the new graph one or more variables from an existing InfluxDB dataset, select: "Panel Title" → "Edit" → "Metric".
  - First, enter the name of an existing InfluxDB dataset in the "Data Source" field.

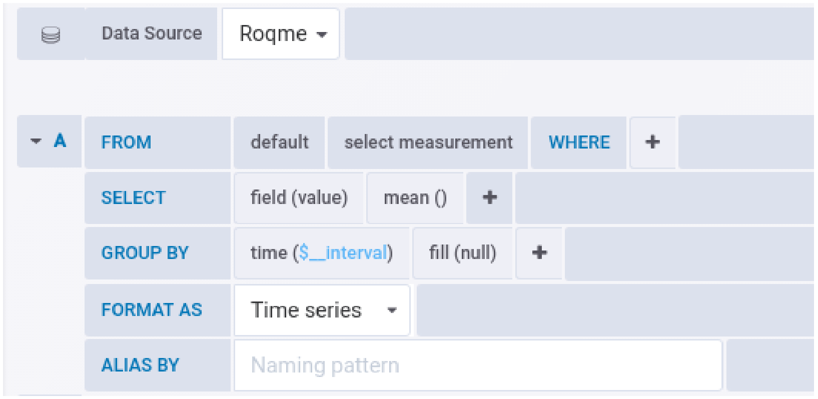


Data Source: default

A: Test data: random walk

  - After selecting a dataset (e.g., "Roqme" in the example included below), you will be allowed to select one or more variables from those available in the selected dataset. For each variable to be displayed in the new graph, you will need to add a query as follows:

    Click on the "Add Query" option available in the "Metrics" panel, and configure the following options:



Data Source: Roqme

A

FROM default select measurement WHERE +

SELECT field (value) mean () +

GROUP BY time ($__interval) fill (null) +

FORMAT AS Time series

ALIAS BY Naming pattern

For instance, in order to display the ***VelocityEvent*** numeric context, configure the previous parameters as follows:

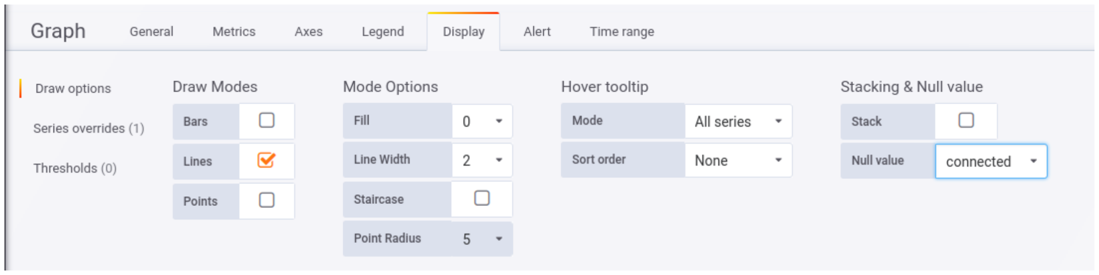
FROM:

- Replace "default" with one of the available options → in this case, select: ***RoqmeExampleDefaultRetentionPolicy***.
- Replace "select measurement" with one of the available options (i.e., "In_Context", "Observation" or "Estimate") → In this case, select ***In_Context***.
- Replace "field (value)" with one of the available options → In this case, select ***VelocityEvent***

ALIAS BY:

- Select a representative name for the magnitude being displayed → In this case, you can type, e.g., ***Velocity***.

○ For each new query added to the graph (i.e., for each variable selected from the InfluxDB dataset), you can configure some display options, e.g., you can indicate that you want the points representing the *Velocity* query, previously defined, to appear connected. This option can be activated by changing the value of the "Stacking & Null Value" option in the "Display" tab from "null value" to "connected".


Graph | General | Metrics | Axes | Legend | Display | Alert | Time range

Draw options
Series overrides (1)
Thresholds (0)

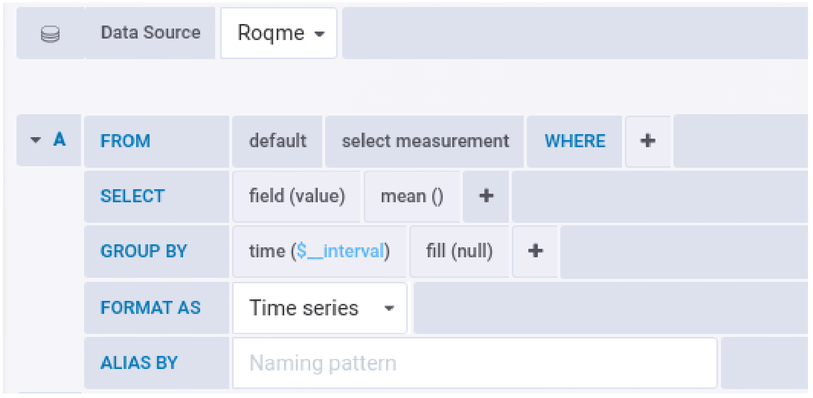
Draw Modes: Bars ☐, Lines ✓, Points ☐

Mode Options: Fill 0, Line Width 2, Staircase ☐, Point Radius 5

Hover tooltip: Mode All series, Sort order None

Stacking & Null value: Stack ☐, Null value connected

○ Let's see now how to configure discrete variables, usually displayed as points. As before, you will need to add the corresponding query by clicking on the "Add Query" option available in the "Metrics" panel. Let's see how to configure this query to display one of the RoQME observations (e.g., Observation 1):


Data Source: Roqme

A
FROM default select measurement WHERE +
SELECT field (value) mean () +
GROUP BY time ($__interval) fill (null) +
FORMAT AS Time series
ALIAS BY Naming pattern

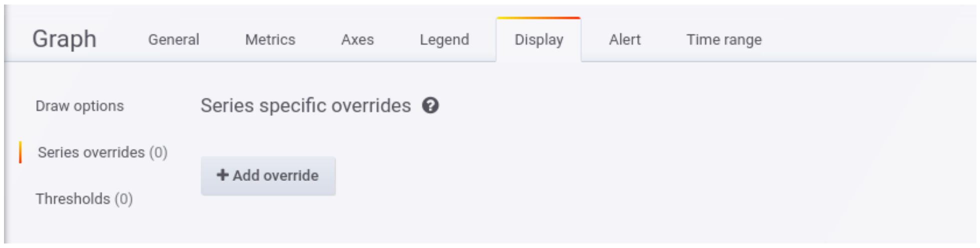
FROM:

- Replace "default" with one of the available options → in this case, select: ***RoqmeExampleDefaultRetentionPolicy***.
- Replace "select measurement" with one of the available options (i.e., "In_Context", "Observation" or "Estimate") → In this case, select ***Observation***.
- Replace "field (value)" with one of the available options → In this case, select ***Observation1***

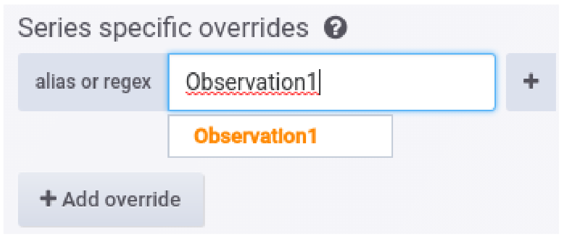
ALIAS BY:

- Select a representative name for the magnitude being displayed → In this case, you can type, e.g., ***Observation1***.

○ In order to configure this new query as a point, you will need to click on the "+Add override" button, available in the "Display" tab.



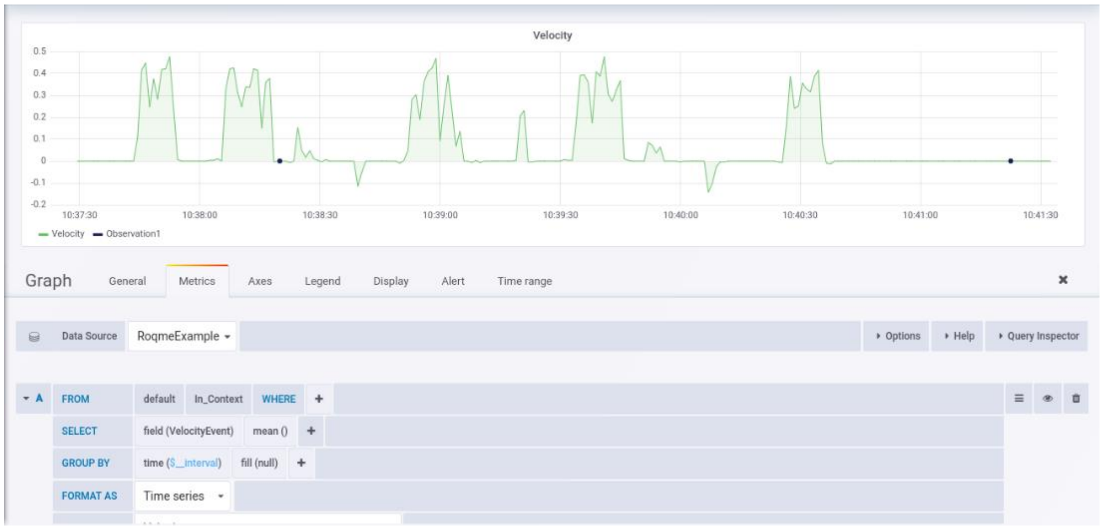
○ Enter the alias associated to the new query (i.e., ***Observation1***) in the "alias or regex" field.



○ Click on the "+" button next to the previous alias, and configure the following options:

Points: true
Null point mode: null

○ From this moment, apart from the ***Velocity*** continuous variable (represented as a line) you graph will also display the ***Observation1*** discrete variable (represented using points). The following picture illustrates how the resulting graph will look like.



Data Source RoqmeExample

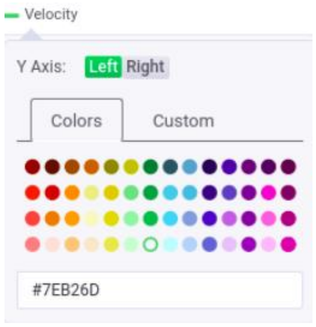
A FROM default In_Context WHERE +

SELECT field (VelocityEvent) mean () +

GROUP BY time ($__interval) fill (null) +

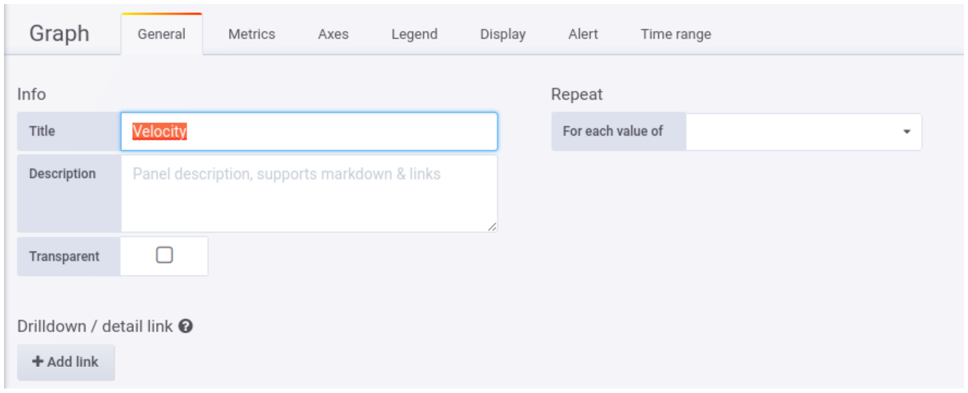
FORMAT AS Time series

- Besides, Grafana allows you to configure additional options, such as:
  - The colour associated to each variable → Click on the colour option associated to the alias, and select any colour either from those available in the predefined palette or by entering the corresponding hexadecimal RGB code.



#7EB26D

  - The title of the graph → Set the "Title" field available in the "General" tab as follows:



Title: Velocity

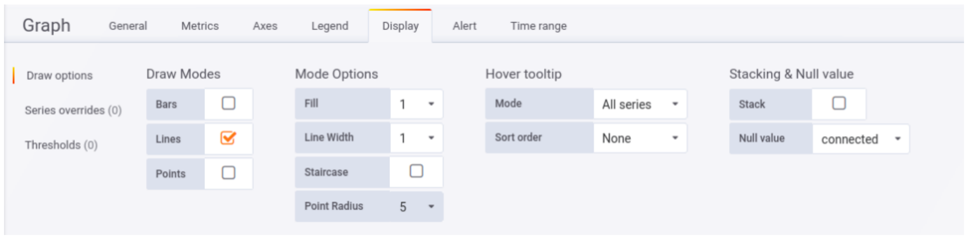
Description: Panel description, supports markdown & links

Transparent: ☐

For each value of

Drilldown / detail link

  - Different display options available in the "Display" tab:



Graph General Metrics Axes Legend Display Alert Time range

Draw options
Series overrides (0)
Thresholds (0)

Draw Modes
Bars ☐
Lines ☑
Points ☐

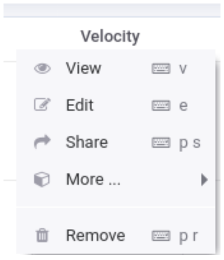
Mode Options
Fill 1
Line Width 1
Staircase ☐
Point Radius 5

Hover tooltip
Mode All series
Sort order None

Stacking & Null value
Stack ☐
Null value connected

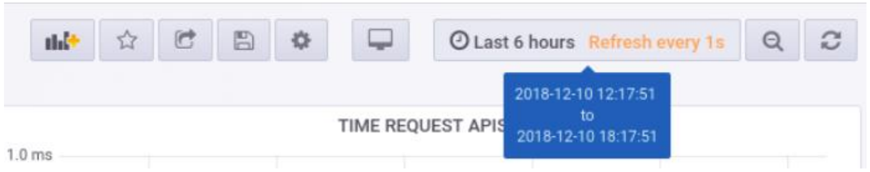
B.5. Removing Existing Graphs from the Active Dashboard

- You can remove any of the existing graphs from the active dashboard by clicking on "Panel Title" (at the top of the graph) and selecting the "Remove" option, available at the end.

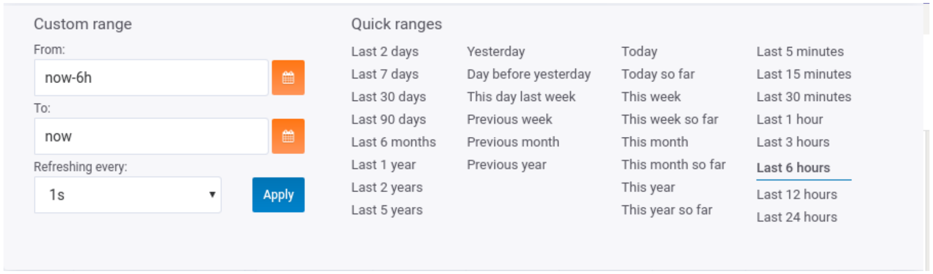


B.6. Configuring the Visualization Time Range

- Selecting appropriate time ranges for each graph is highly important in order to achieve easy to read and interpret visualizations. In most cases, we will prefer to refresh the visualized data as frequently as possible (e.g., every second). We can configure this and other time-related parameters by clicking on the button that displays the current time window (e.g., Last 6 hours) and refresh frequency (e.g., Refresh every 1s); see the figure included next.



- When we press the previous button, the following options appear:



Custom range
From: now-6h
To: now
Refreshing every: 1s
Apply

Quick ranges

Last 2 days	Yesterday	Today	Last 5 minutes
Last 7 days	Day before yesterday	Today so far	Last 15 minutes
Last 30 days	This day last week	This week	Last 30 minutes
Last 90 days	Previous week	This week so far	Last 1 hour
Last 6 months	Previous month	This month	Last 3 hours
Last 1 year	Previous year	This month so far	Last 6 hours
Last 2 years		This year	Last 12 hours
Last 5 years		This year so far	Last 24 hours

- The time window can be set using the "From" and "To" fields. In the previously figure, this parameters are set to:

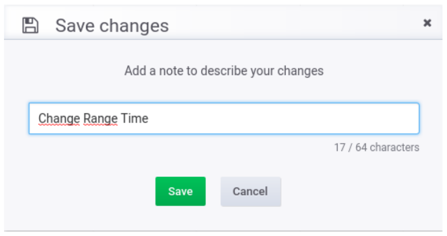
From: now-6h (from six hours ago)

To: now (until now)

You can change these values to cover any time window. You can use different types of time expressions, which may optionally include a reference to "now". Valid time expressions are: yyyy-mm-dd hh:mm:ss, Xd (X days), Xw (X weeks), Xy (X years), Xh (X hours), Xm (X minutes), Xs (X seconds), etc.

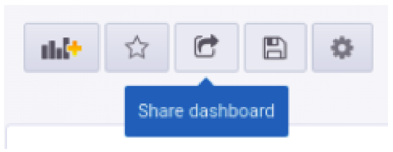
- Finally, you can also modify the refresh rate selecting one of the available values (see "Section B.3 - Auto-refresh" to learn how to extend the list of available refresh rates). Click on the "Apply" button to apply the changes.

IMPORTANT: Remember to save all the changes you made in your dashboard by selecting the "Save dashboard" menu option and clicking on the "Save" button.

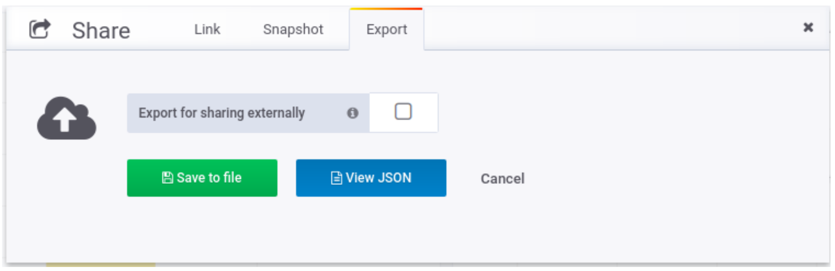


B.7. Importing, Exporting and Managing Existing Dashboards

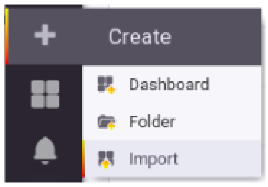
- **Exporting a Dashboard**: In order to export the active dashboard (including all its graphs), previously created in Grafana as described in Section B.3, click on the "Share Dashboard" menu option and move to the "Export" tab.



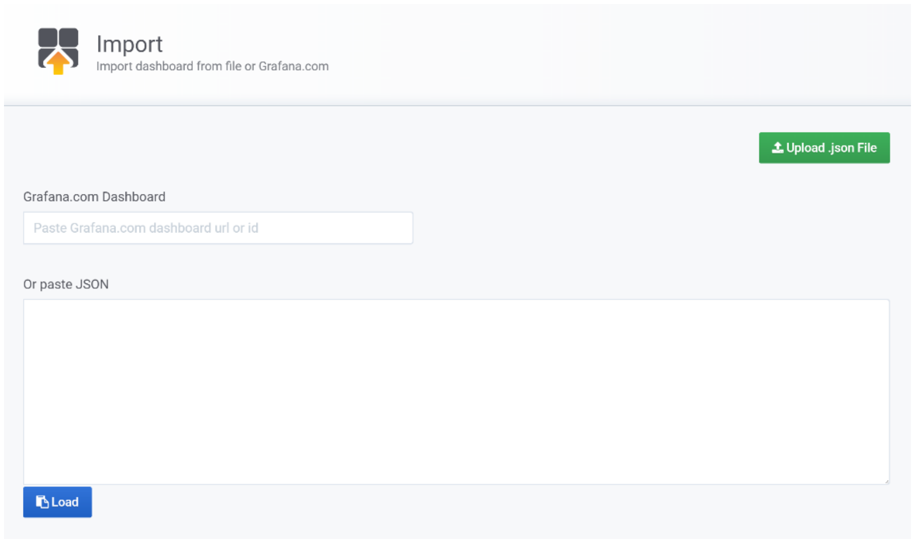
Click on the "Save to file" button to save the dashboard as a JSON file. In case you just want to see the JSON code, without saving it, click on the "View JSON" button.



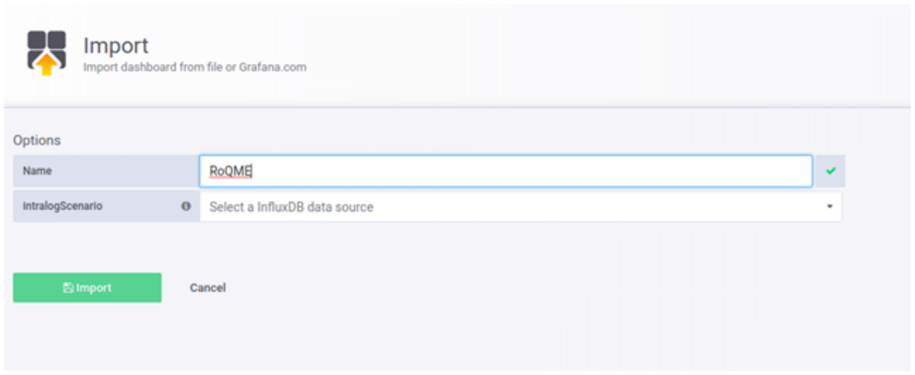
- **Importing a Dashboard**: You can import an existing dashboard (including all its graphs) into Grafana from an existing JSON file, e.g., from a JSON file obtained when exporting a dashboard as detailed in previous step, or from the JSON file generated by the RoQME Tool-Chain in the following local folder: QoSMetricsProvider/utils/viewer. In order to import an existing dashboard, select *Create (+)* → *Import*.



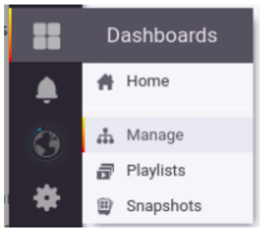
In the "Import" panel (displayed below): (1) click on "Upload JSON File"; (2) select the JSON file generated by the RoQME Tool-Chain; and (3) press the "Load" button.


Import
Import dashboard from file or Grafana.com
Upload .json File
Grafana.com Dashboard
Paste Grafana.com dashboard url or id
Or paste JSON
Load

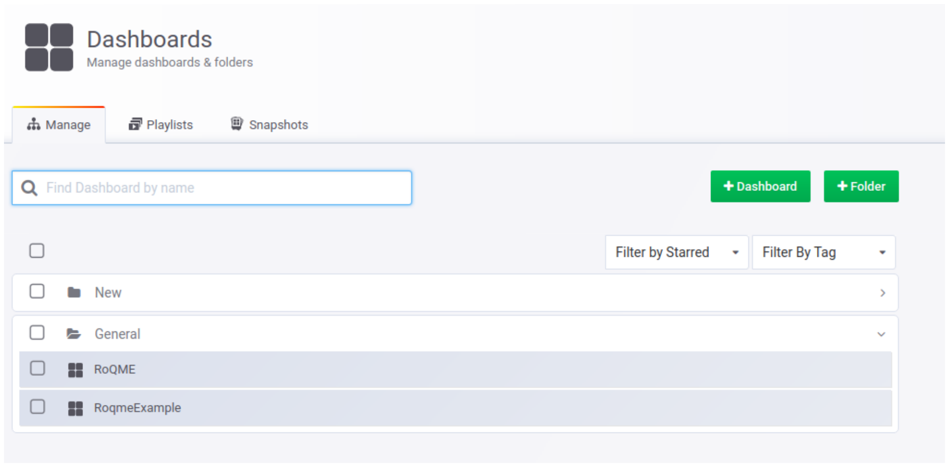
Next, click on "Select a InfluxDB data source", select the name of an existing InfluxDB dataset, and click on "Import" (green button on the left-bottom corner of the Import Panel).


Import
Import dashboard from file or Grafana.com
Options
Name
RoQME
IntralogScenario
Select a InfluxDB data source
Import
Cancel

- **Managing Existing Dashboards**: You can view the available dashboards by selecting *Dashboards → Manage*.



The following figure displays the list of all available dashboards. In that window, you can add and remove existing dashboards, and organize them into folders.


Dashboards
Manage dashboards & folders
Manage
Playlists
Snapshots
Find Dashboard by name
+ Dashboard
+ Folder
Filter by Starred
Filter By Tag
New
General
RoQME
RoqmeExample