5GCity:  
 Monitoring System Description



Summary

[1. Scope 2](#_Toc532389603)

[2. System description 3](#_Toc532389604)

[3. Metrics 4](#_Toc532389605)

[4. Front End 5](#_Toc532389606)

[5. Web Gui 6](#_Toc532389607)

[5.1. Add a metric 6](#_Toc532389608)

[5.2. Add nodes to be monitored 7](#_Toc532389609)

[5.3. Add a service 8](#_Toc532389610)

[6. API REST 10](#_Toc532389611)

# Scope

This document provides a quick view of monitoring system implemented within the 5GCity framework, along with the family metrics and the GUI interface.

For the installation and configuration issues see the README.md file in the 5GCity GitHub.

# System description

The monitoring system within the 5Gcity framework is based on:

* Prometheus - an open source, metrics-based monitoring system and alerting - (version 2.3.2);
* Grafana (version 5.2.4) - that supports querying Prometheus - is used for the graphic representation of the collected data from the instrumented nodes composing the monitored system;
* Node Exporter (version 0.16.0): exposes a wide variety of hardware- and kernel-related metrics in order to allow Prometheus to display and/or collect specific "system" metrics;
* FrontEnd: it’s a custom java application to manage the configuration of the instrumented objects; the Linux nodes can typically be grouped into services to combine the data related to metrics and/or related to applications.

The Front End application is based on (open) jdk1.8 and Wildfly14.1.0.Final.

# Metrics

In the first issued release (Oct 2018), the scraping is foreseen to the Linux nodes, through the metrics available to Node Exporter, imported by Prometheus and viewed through Grafana on an ad hoc Dashboard.

NOTE: Linux node exporter must be installed and running on each node that has to be monitored:

<https://github.com/prometheus/node_exporter/releases/download/v0.16.0/node_exporter-0.16.0.linux-amd64.tar.gz>

The following metrics will be available:

* Up\_\* -> it is about the status of the node i.e. the reachability of the node\_exporter process;
* node\_cpu\_\* -> some metrics related to CPU usage;
* node\_memory\_\* -> some metrics related to RAM usage;
* node\_filesystem\_\* -> some metrics related to file system usage;
* node\_disk\_\* -> some metrics related to the I/O disk usage;
* node\_network\_\* -> some metrics to determine the transmission network quality.

# Front End

On the monitoring system, by the FrontEnd, the monitored nodes/services have to be configured in order to get the measurements using the API Rest or Web GUI.

To do this, on the FrontEnd will be necessary to configure:

* at least one "Metric" - NODE - related to the node Exporter metrics;
* all the nodes (Nodes) to be  monitored;
* at least one service (Service) grouping the configured nodes and the NODE metric

# Web Gui

About the Web GUI in the following a short manual is provided.

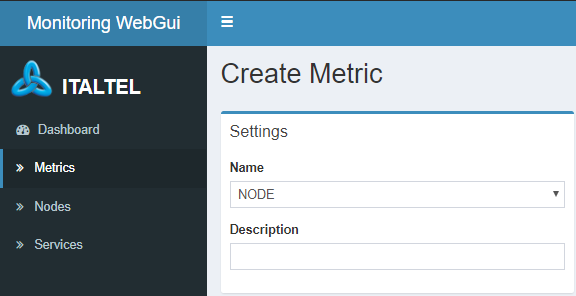
First connect by http to the ‘folder’ where the monitoring system is installed using port 8888 (default) or the one configured ad hoc if it is different from default; in the follows this port is defined <FEPort>

http://<IPFrontEnd>:<FEPort>/FrontEnd

## Add a metric

To Add the “metric” related to the Node Family, set *Name=NODE (see figure below)*:

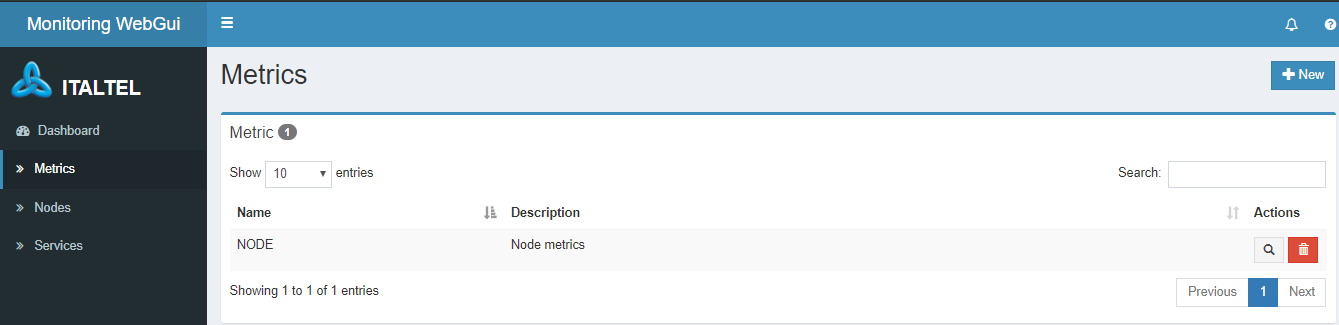
* Open Metric -> New
* Insert Name and Description (optional)
* Save



Every new added metric will be displayed in the metric list, that is: Metrics.

For every metric is possible to:

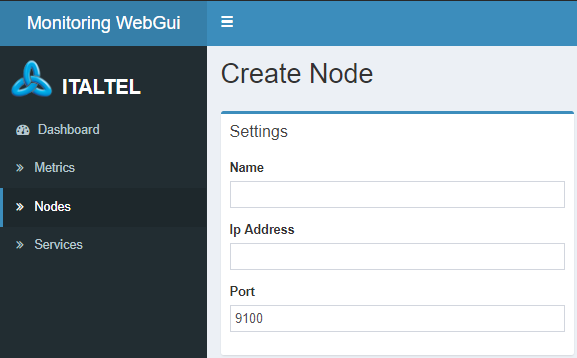
* Get a detailed visualization
* Remove the metric itself



## Add nodes to be monitored

To add nodes to be monitored by the monitoring system:

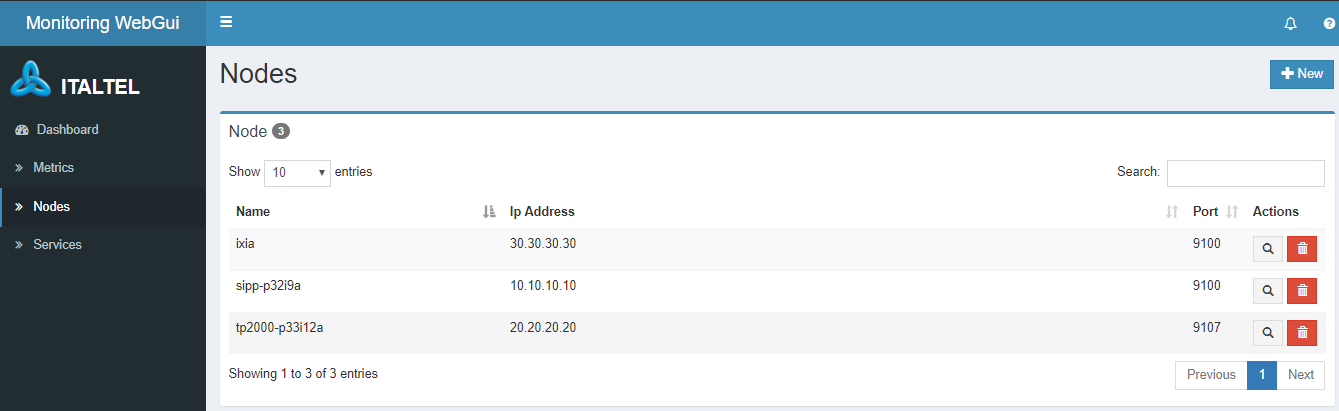
* Open Node; for every new node -> New
* Insert Name and address IP; the item port has its own default (9100)
* Save



Every new added node will be displayed in the node list, that is: Nodes.

For every node is possible to:

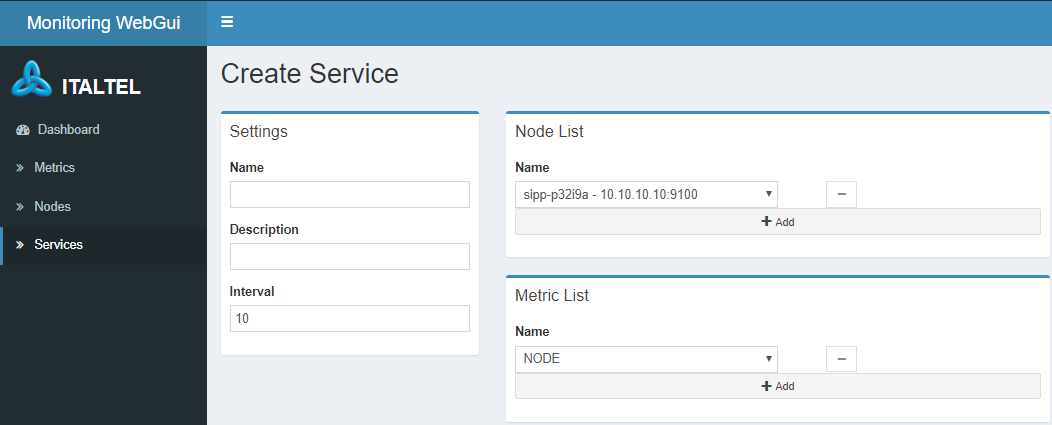
* Get a detailed visualization
* Remove the node itself



## Add a service

To add a service:

* Open the item Service ; for every new service -> New
* Insert Name , Description (optional), Interval (default=10s time scrape interval)
* Insert the nodes to be associated with the service (the first is already associated; use + or - signs to modify)
* Insert the metric to be associated to the (added) service (the first one is already associated; use + e - to modify)
* Save



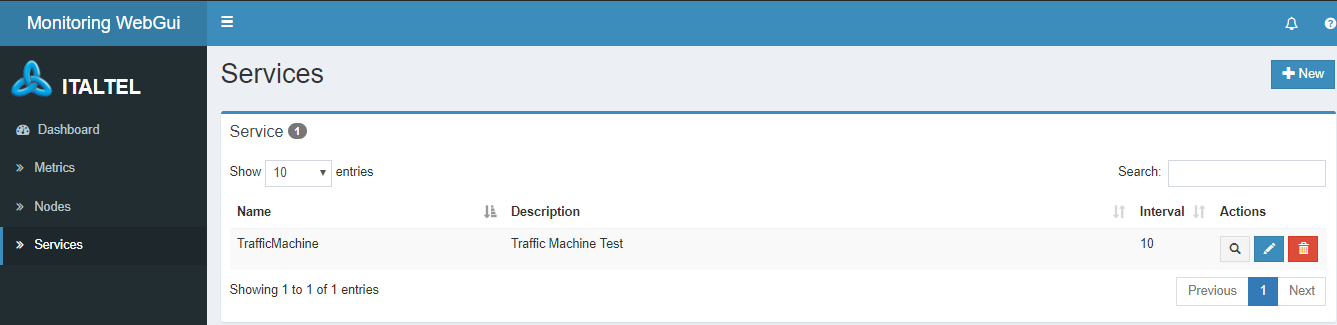
Every new added service will be displayed in the service list, that is: Services.

For every service is possible to:

* Get a detailed visualization
* Clear the service data
* Remove the service itself

For a service the following items can be modified:

* Description
* Interval
* Nodes
* Metrics



# API REST

In addition to the GUI it is possible to insert / modify / remove the data of the monitoring FrontEnd objects by the API rest

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Object** | **Url** | **Method** | **Description** | **Body** |
| Metric | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/metric | PUT | Create | Yes |
| Metric | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/metric | GET | List all the objects | No |
| Metric | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/metric/<name> | GET | READ a single object | No |
| Metric | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/metric/<name> | DELETE | REMOVE a single object | No |
|  |  |  |  |  |
| Node | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/node | PUT | Create | Yes |
| Node | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/node | GET | List all the objects | No |
| Node | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/node/<name> | GET | READ a single object | No |
| Node | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/node/<name> | DELETE | REMOVE a single object | No |
|  |  |  |  |  |
| Service | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/service | PUT | Create | Yes |
| Service | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/service | GET | List all the objects | No |
| Service | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/service/<name> | GET | READ a single object | No |
| Service | http://<IPFrontEnd>:<FEPort>/FrontEnd/rest/fe/service/<name> | DELETE | REMOVE a single object | No |
| Service | http://<IPFronEnd>:<FEPort>/FrontEnd/rest/fe/service | POST | MODIFY  a single object | Yes |

In the following same examples of Body are illustrated:

* **Create a Metric**

{name: "NODE", desc: "Metriche di Sistema"}

* **Create a Node**

{name: "nodeTest", ip: "10.10.10.10", port: "9100"}

* **Create/Modify service**

{name: "serviceTest", description: "test description", interval: "10",

nodes: [{name: "nodeTest"}],

metrics: [{name: "NODE"}]}