

# TANGO-grafana

a study of the modifiability on Prometheus and Grafana

Matteo Di Carlo

---

## Writing an exporter

- Guideline available and very clear:
    - <https://prometheus.io/docs/practices/instrumentation/>
  - Python client library for development
    - [https://github.com/prometheus/client\\_python#gauges](https://github.com/prometheus/client_python#gauges)
-

# Writing an exporter for TANGO

- Read all the attributes from a TANGO control system
    - I have already made something similar in a smoke test in skampi:  
[https://gitlab.com/ska-telescope/skampi/-/blob/master/post-deployment/tests/smoke/test\\_devices.py](https://gitlab.com/ska-telescope/skampi/-/blob/master/post-deployment/tests/smoke/test_devices.py)
  - Result: 220 lines of code and more than 4000 attributes read in less than 4ss
    - Timeout set at 10ms!
-

# TANGO exporter

```
device_attribute{
```

```
    device="mid_csp_cbf/pssconfig/01",
```

```
    dim_x="1", dim_y="0",
```

```
    label="obsState",
```

```
    name="obsState",
```

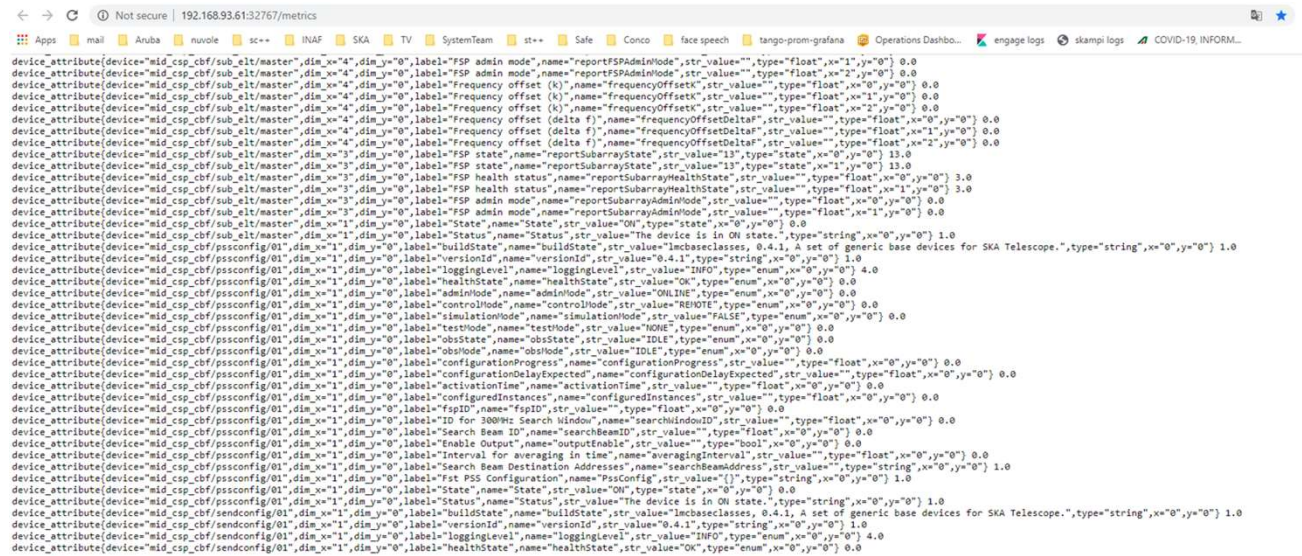
```
    str_value="IDLE",
```

```
    type="enum",
```

```
    x="0", y="0"
```

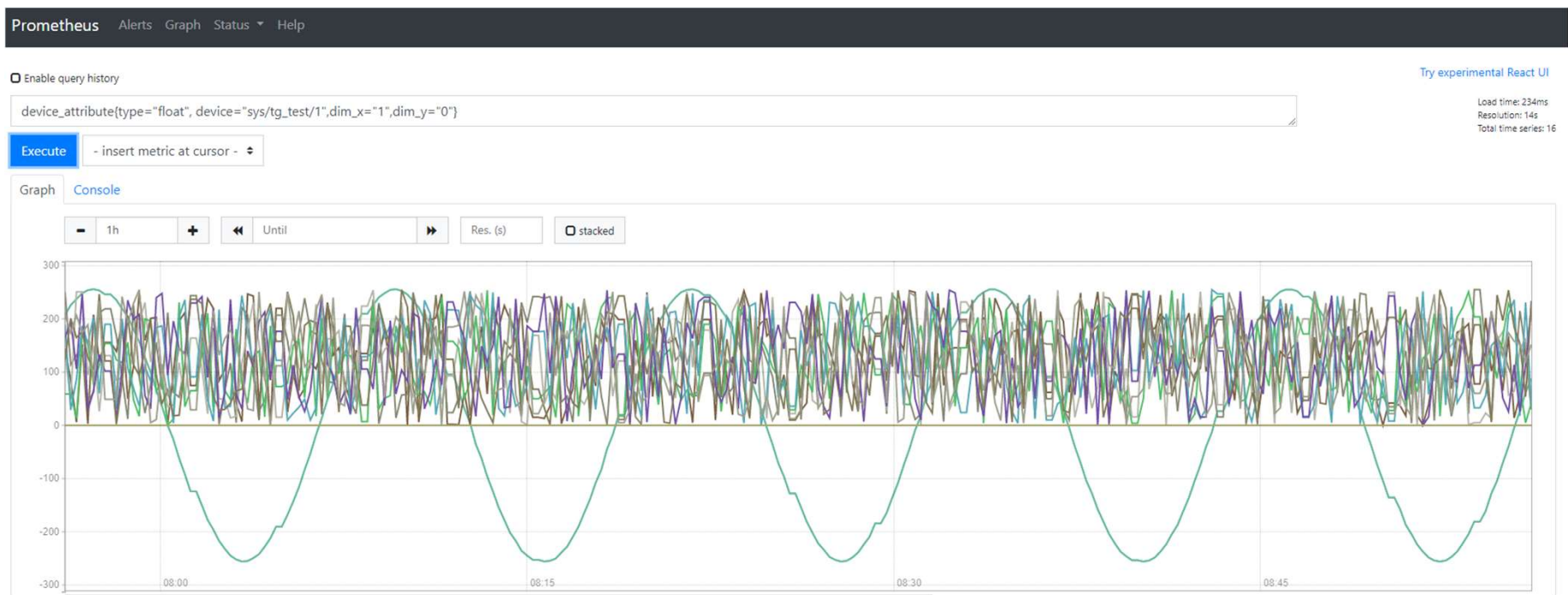
```
} 0.0
```

<http://192.168.93.69:32767/metrics>

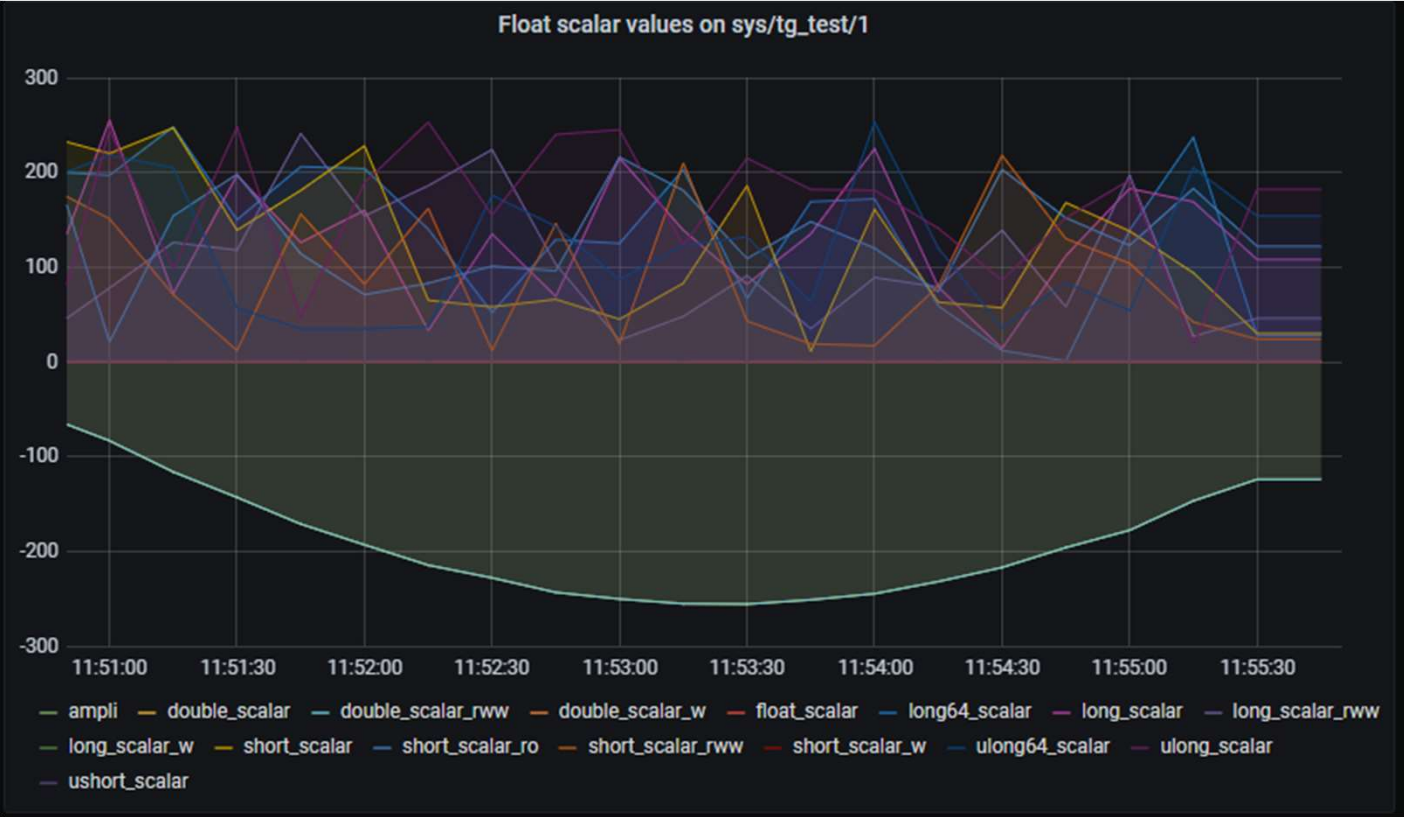


```
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="FSP admin mode",name="reportFSPAdminNode",str_value="",type="float",x="1",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="FSP admin mode",name="reportFSPAdminNode",str_value="",type="float",x="2",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="Frequency offset (k)",name="frequencyOffset",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="Frequency offset (k)",name="frequencyOffset",str_value="",type="float",x="1",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="Frequency offset (k)",name="frequencyOffset",str_value="",type="float",x="2",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="Frequency offset (delta f)",name="frequencyOffsetDeltaf",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="Frequency offset (delta f)",name="frequencyOffsetDeltaf",str_value="",type="float",x="1",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="4",dim_y="0",label="Frequency offset (delta f)",name="frequencyOffsetDeltaf",str_value="",type="float",x="2",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="3",dim_y="0",label="FSP state",name="reportSubarrayState",str_value="13",type="state",x="0",y="0"} 13.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="3",dim_y="0",label="FSP state",name="reportSubarrayState",str_value="13",type="state",x="1",y="0"} 13.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="3",dim_y="0",label="FSP health status",name="reportSubarrayHealthState",str_value="",type="float",x="0",y="0"} 3.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="3",dim_y="0",label="FSP health status",name="reportSubarrayHealthState",str_value="",type="float",x="1",y="0"} 3.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="3",dim_y="0",label="FSP admin mode",name="reportSubarrayAdminNode",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="3",dim_y="0",label="FSP admin mode",name="reportSubarrayAdminNode",str_value="",type="float",x="1",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="1",dim_y="0",label="State",name="State",str_value="ON",type="state",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/sub_elt/master",dim_x="1",dim_y="0",label="Status",name="Status",str_value="The device is in ON state.",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="buildState",name="buildState",str_value="Incbaseclasses, 0.4.1, A set of generic base devices for SKA Telescope.",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="versionId",name="versionId",str_value="0.4.1",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="loggingLevel",name="loggingLevel",str_value="INFO",type="enum",x="0",y="0"} 4.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="healthState",name="healthState",str_value="OK",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="adminNode",name="adminNode",str_value="ONLINE",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="controlMode",name="controlMode",str_value="REMOTE",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="simulationMode",name="simulationMode",str_value="FALSE",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="testMode",name="testMode",str_value="NONE",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="obsState",name="obsState",str_value="IDLE",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="obsNode",name="obsNode",str_value="IDLE",type="enum",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="configurationProgress",name="configurationProgress",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="configurationDelayExpected",name="configurationDelayExpected",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="activationTime",name="activationTime",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="configuredInstances",name="configuredInstances",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="FSPID",name="FSPID",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="ID for 300MHz Search Window",name="searchWindowID",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="Search Beam ID",name="searchBeamID",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="Enable Output",name="outputEnable",str_value="",type="bool",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="Interval for averaging in time",name="averagingInterval",str_value="",type="float",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="Search Beam Destination Addresses",name="searchBeamAddress",str_value="",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="Fst PSS Configuration",name="PssConfig",str_value="{}",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="State",name="State",str_value="ON",type="state",x="0",y="0"} 0.0
device_attribute{device="mid_csp_cbf/pssconfig/01",dim_x="1",dim_y="0",label="Status",name="Status",str_value="The device is in ON state.",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/sendconfig/01",dim_x="1",dim_y="0",label="buildState",name="buildState",str_value="Incbaseclasses, 0.4.1, A set of generic base devices for SKA Telescope.",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/sendconfig/01",dim_x="1",dim_y="0",label="versionId",name="versionId",str_value="0.4.1",type="string",x="0",y="0"} 1.0
device_attribute{device="mid_csp_cbf/sendconfig/01",dim_x="1",dim_y="0",label="loggingLevel",name="loggingLevel",str_value="INFO",type="enum",x="0",y="0"} 4.0
device_attribute{device="mid_csp_cbf/sendconfig/01",dim_x="1",dim_y="0",label="healthState",name="healthState",str_value="OK",type="enum",x="0",y="0"} 0.0
```

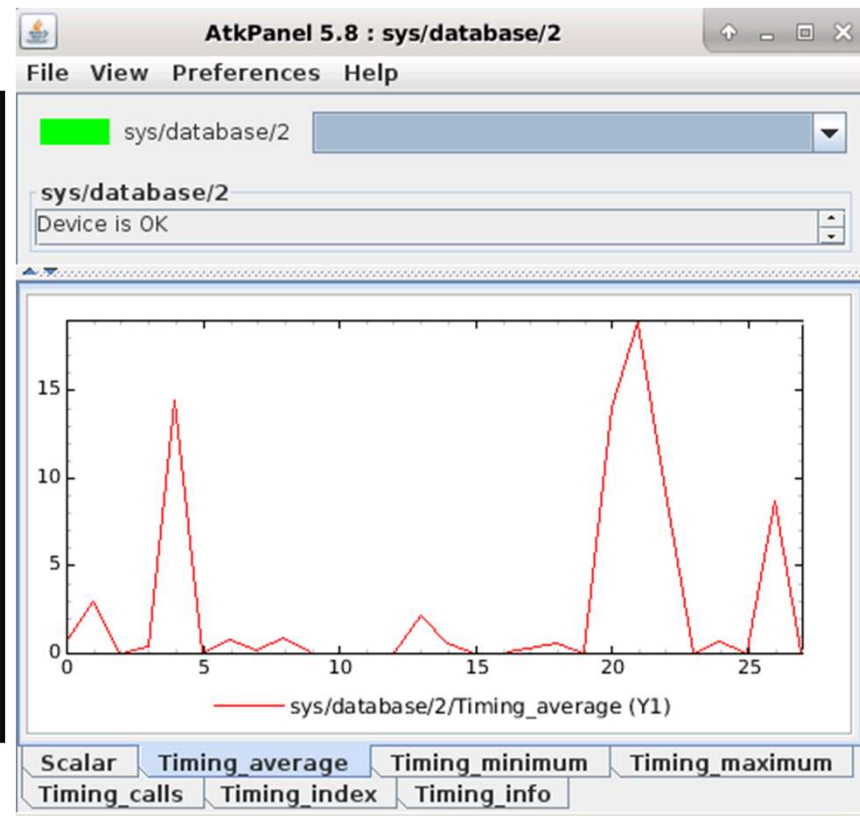
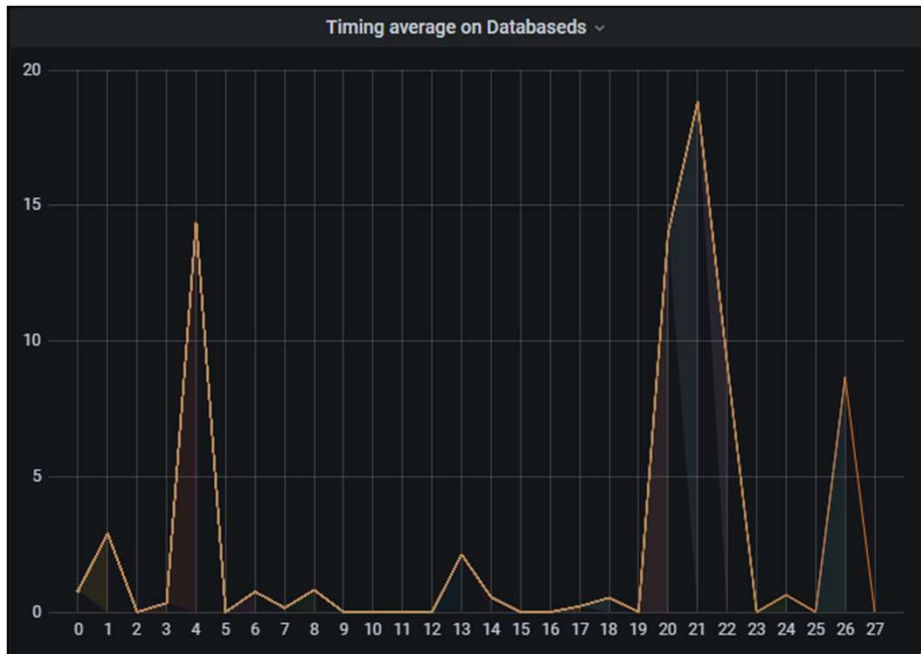
# Prometheus graph engine



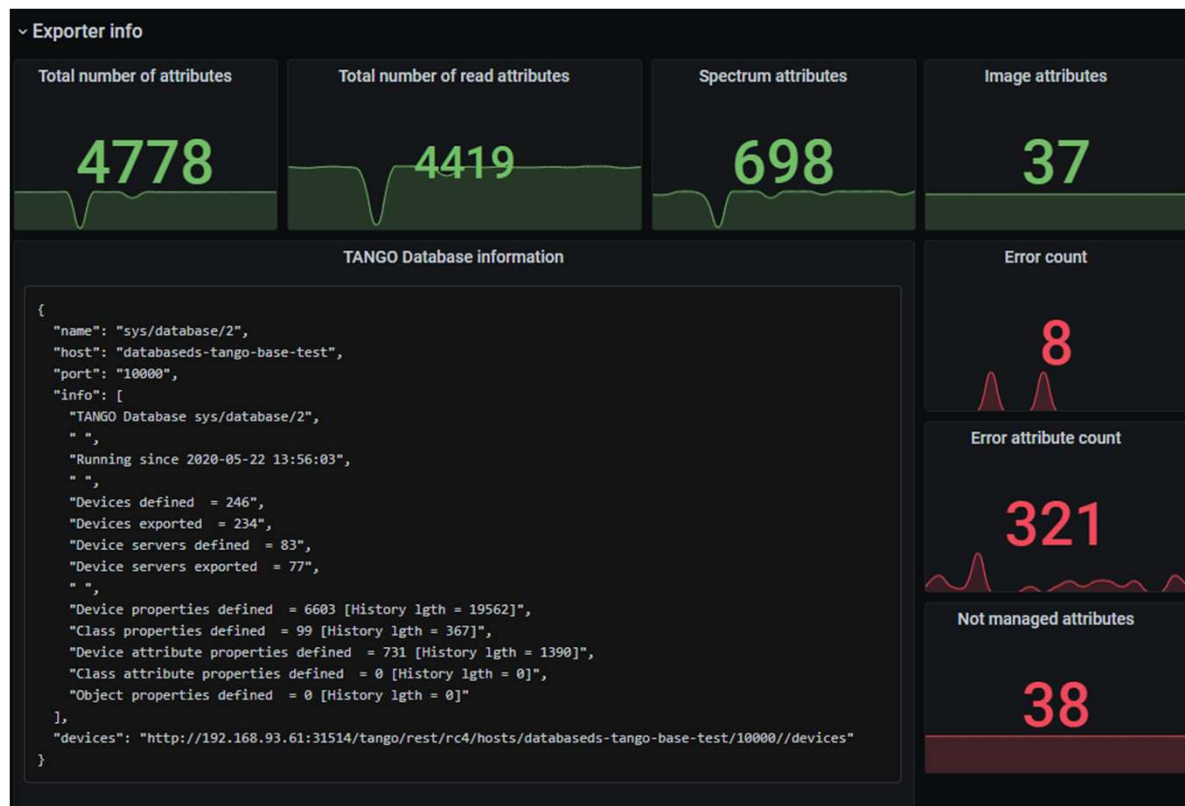
# TANGO scalar



# TANGO spectrum



# View of the system (with open source ajax plugin)



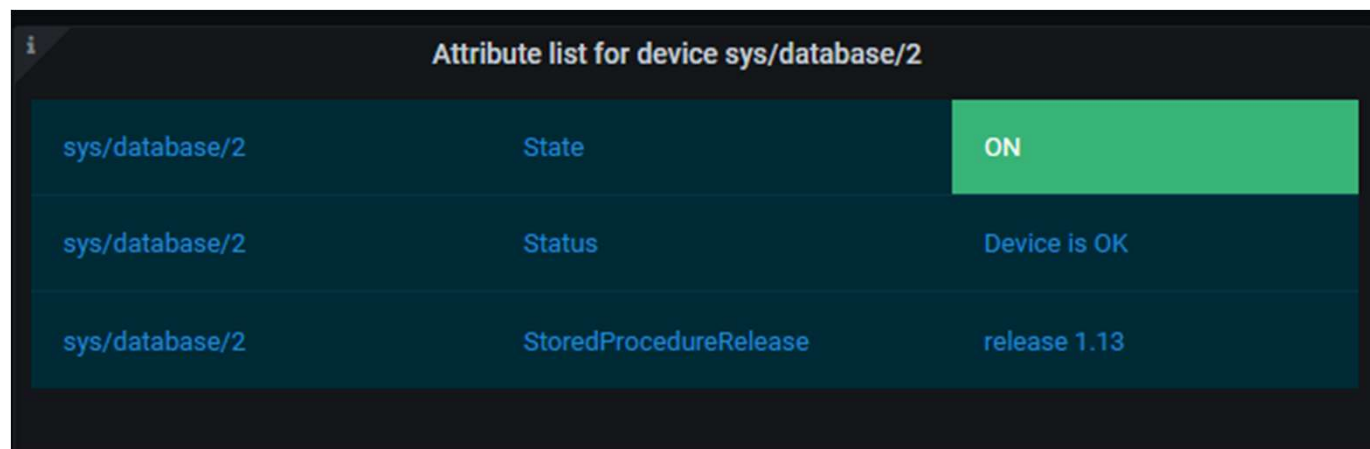


# What about displaying strings?

In prometheus and grafana, everything is a number.

Needs for a new plugin:

- Show attributes in a table form (label value in case of state/string/enum)



The screenshot shows a Grafana dashboard panel titled "Attribute list for device sys/database/2". It displays a table with three rows of attributes for the device "sys/database/2". The first row shows the "State" attribute with a value of "ON", which is highlighted in a green background. The second row shows the "Status" attribute with a value of "Device is OK". The third row shows the "StoredProcedureRelease" attribute with a value of "release 1.13".

Label	Value
sys/database/2	State ON
sys/database/2	Status Device is OK
sys/database/2	StoredProcedureRelease release 1.13

# Anatomy of a grafana plugin

React project composed by at least 2 files:

- *Plugin.json*:
  - when grafana starts it looks for those files and load them according to the specified configuration
- *Module.ts*:
  - expose the implementation



# So far so good

Very easy to create new exporters

Not that easy for grafana plugin:

- Typescript
- The react model of thinking



# Monitoring and Control panel?

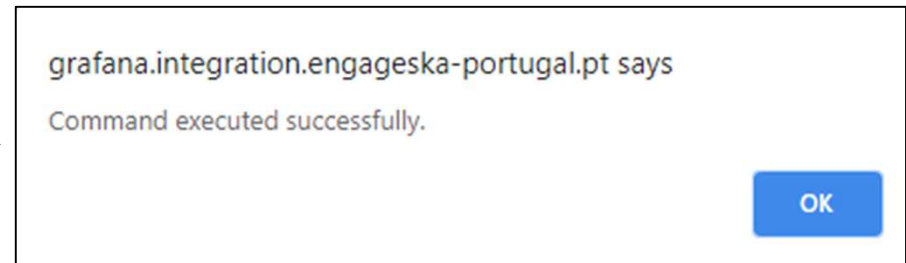
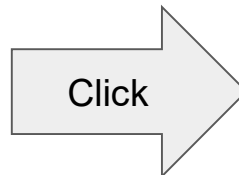
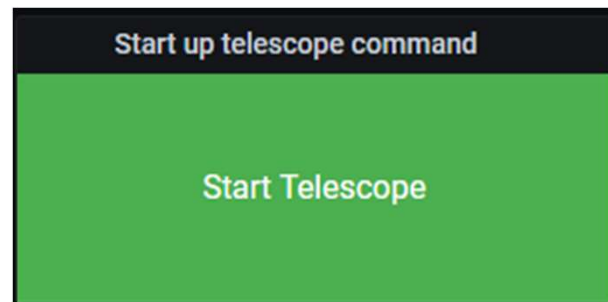
While it is quite easy to create a new monitoring plugin and dashboards, it isn't the same for the control counterpart

- CORS problems
- Parameter panel can be tricky with complex parameter



# TANGO-Command plugin

Based on TANGO-GQL backend



# TANGO-Command Edit panel

Panel

> Visualization

Display

Tango GQL web url  
Tango GQL web url  
http://192.168.93.61:5000/mutation

Username  
Username  
user1

Password  
Password  
abc123

TangoGQL mutation  
TangoGQL mutation  
{query:"mutation {\n executeCommand(device: \'ska\_mid/tm\_central/central\_node\', command: \'StartupTelescope\') {\n ok\n output\n message\n }\n}"}

Button Text  
Button Text  
Start Telescope

Links  
+ Add link

Repeat options  
Repeat by variable  
Repeat this panel for each value in the selected variable. This is not visible while in edit mode. You need to go back to dashboard and then update the variable or reload the dashboard.  
Choose



## Demo 2: TANGO-Grafana



# Thanks!

Prometheus and Grafana are highly modifiable

- Created a TANGO dashboard in less than one sprint!
  - Source code (needs refactoring) <https://github.com/Matteo04052017/TANGO-grafana>:
    - TANGO-exporter and helm chart to deploy in k8s
    - TANGO-Attribute plugin
    - TANGO-Command plugin
    - TANGOGQL-proxy (flask) for enabling CORS in the TANGO-command plugin

