Quick comparison of Webjive vs Grafana

2020-05-26
Giorgio Brajnik

Introduction

This short memo is triggered by the demo done by Matteo Di Carlo on a solution based on Tango devices, Prometheus, Grafana. I see that as a potential alternative to Webjive.

Outline of the architecture

As far as I can tell the architecture involving Grafana comprises:

- a Tango server
- Prometheus, a tool that collects timed data points from “exporters” and stores them into an efficient time series database. Each metric in the database is basically a timestamped tuple with whatever attribute we want, a float value and a millisecond timestamp. For example a metric can be made of a collection of <timestamp, name of device, name of attribute, value>. There can be more than one metric in the database.
- an exporter, written in python or c++ that uses Tango deviceproxy. The exporter is an ad-hoc http server that Prometheus reads; it can do polling on selected devices/attributes/states.
- Grafana, the editor and runner of visual dashboards.
- Some existing and some yet-to-be-developed plugins in React for Grafana.
- TangoGQL (or TangoREST) to let dashboards send commands to devices.

Extension points include:

- Grafana plugins (in React)
- Prometheus exporters (Python, c++, …)

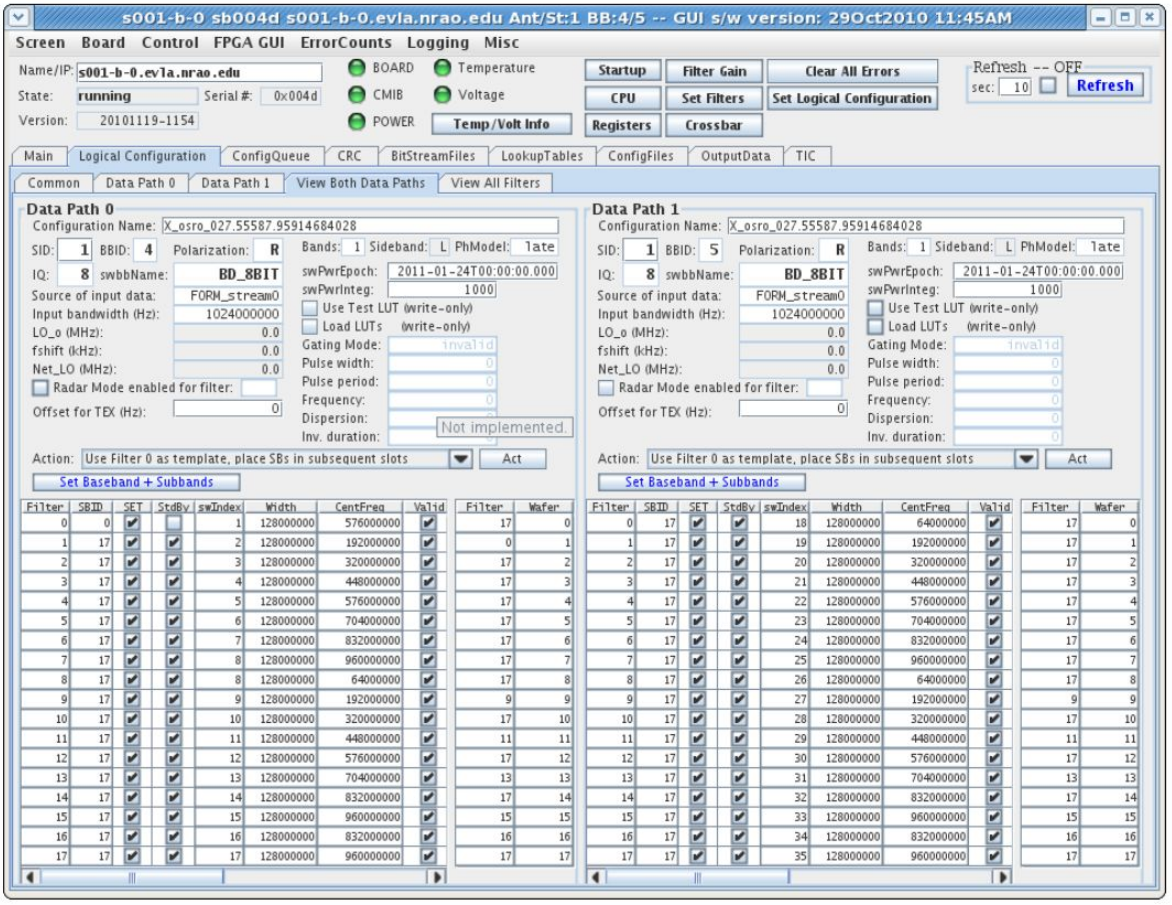
Pro’s and con’s wrt WebJive

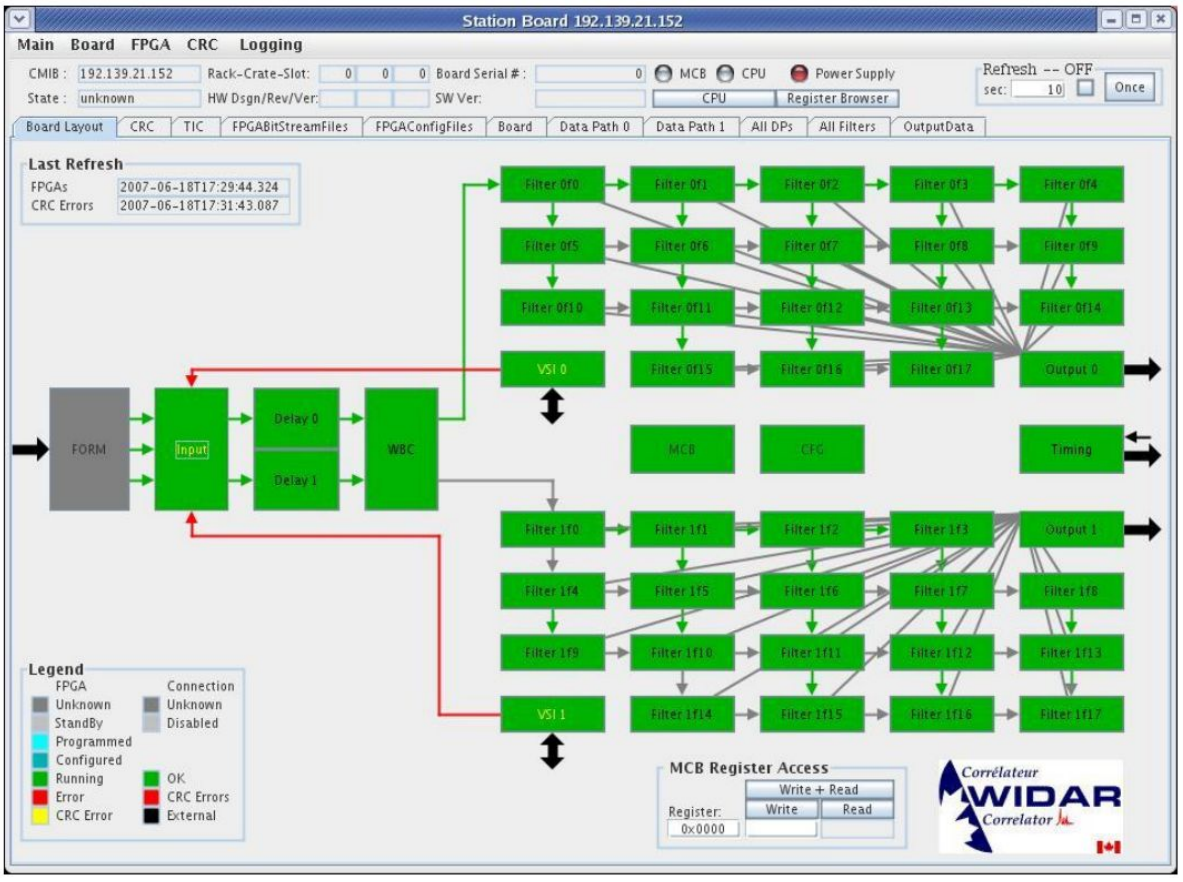
	WebJive	Grafana, Prometheus + REST
web-based	✅	✅
user management	minimal	✅
dashboard management: access privileges, show/hide, folders	minimal	✅
dashboard navigation	todo	✅
performance assessment	partially done (js dispatcher to reassess)	to do
plugins (in React)	none	✅
plugin repository	none	✅
attractive look & feel of running dashboards	none	✅
flexible grid layout	none	✅
rich set of widgets/charts	not there yet	✅
usability of dashboard editor	poor	✅
conditional styles and expressions	very limited	✅
open source	✅	✅ Apache, MIT
introspection on Tango devices/server	✅	❌ limited to what can be expressed as

		Prometheus metrics
Extendability of data that can be displayed		✅ by writing new Prometheus exporters
ability to send commands	✅	❌ needs the Ajax plugin and an http server
interactive synoptic viewer	none	✅ seems to be provided by existing plugin
import/export of dashboards	todo	✅
dropdown menu	todo	✅ available in grafana/ui (beta)
visual grouping of widgets	todo	✅ available in grafana/ui (beta)
structural grouping of widgets	todo	✅ available in grafana/ui (beta)
table	todo	✅ available in grafana/ui (beta)
alerts/alarms	todo	✅ grafana has tools, via http, slack, ... prometheus has push-mode components
log viewer	todo	✅ with existing plugin
digital assets (logo, color theme, L&F)	todo	✅ probably by developing a “Grafana app”

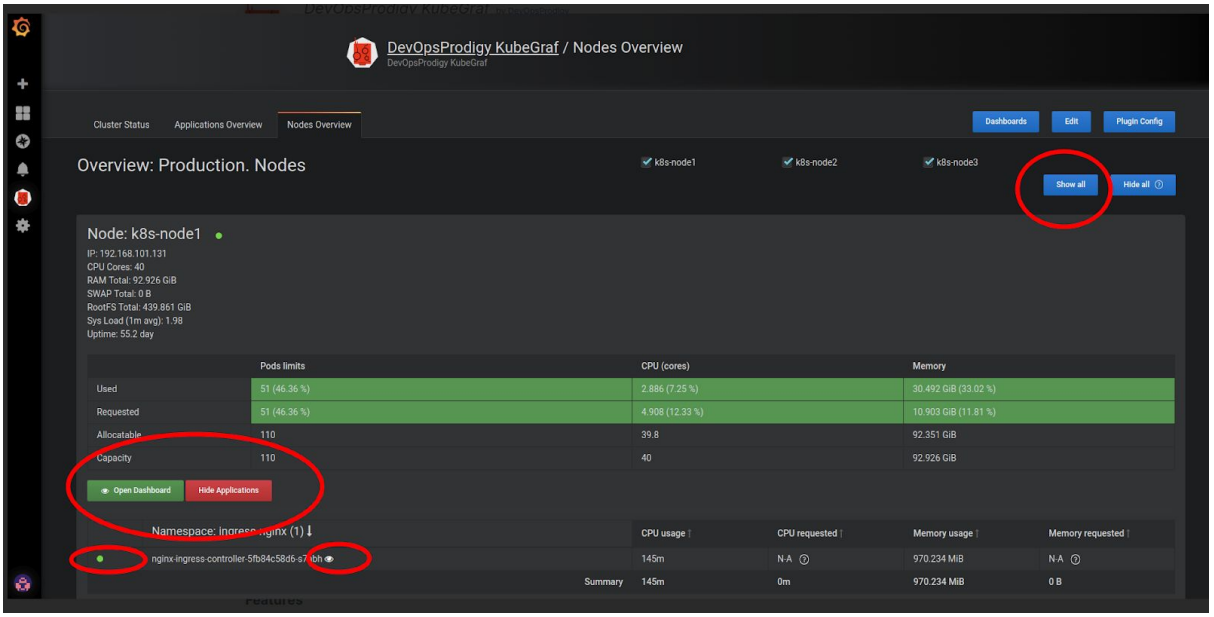
Where we want to go

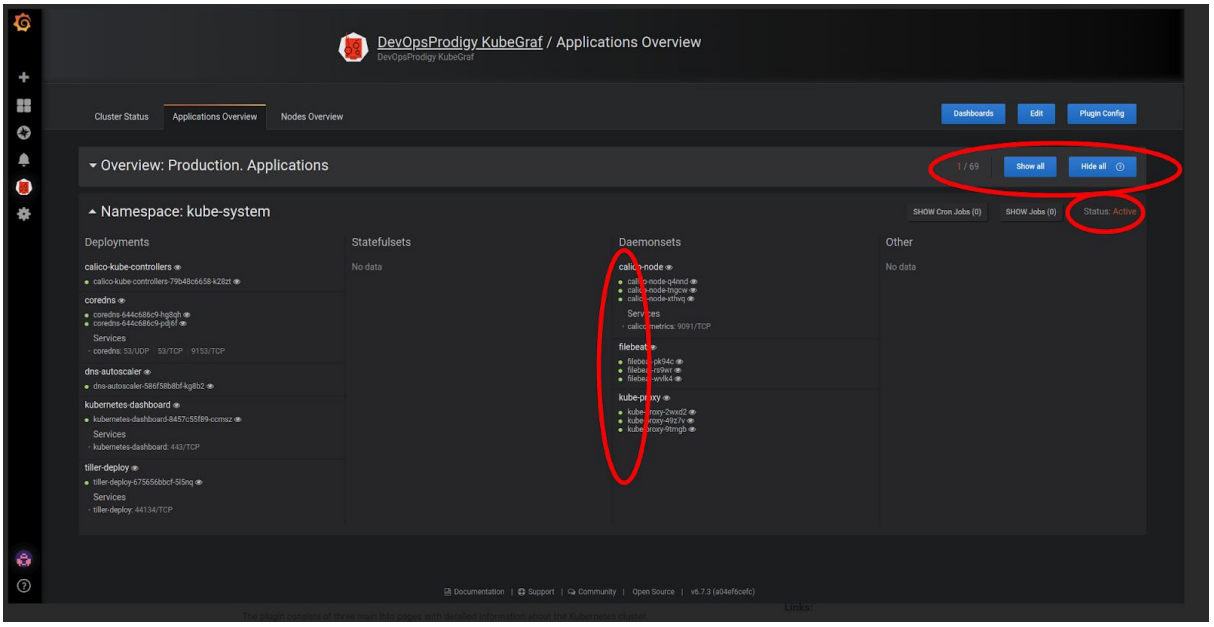
Examples:

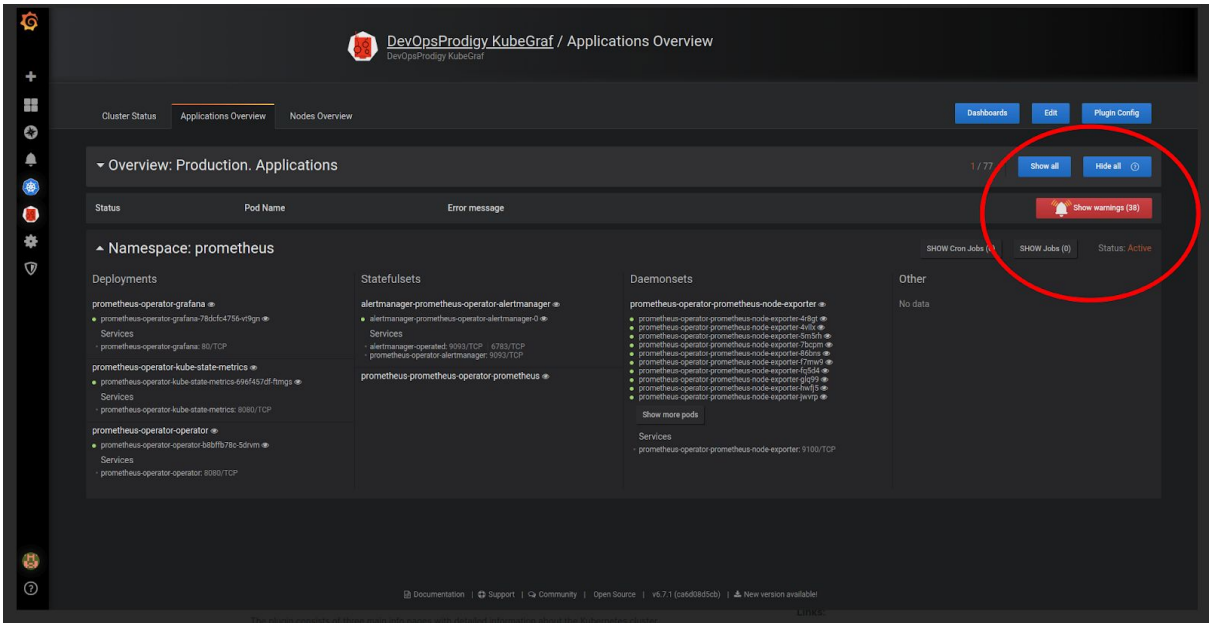


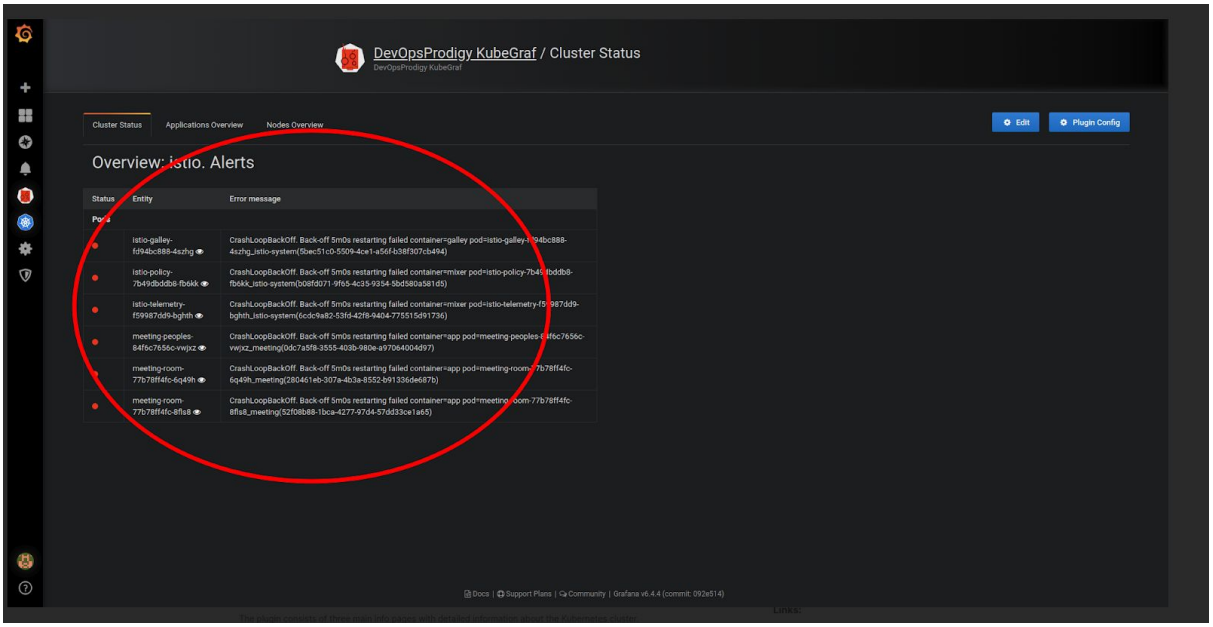


What seems to be possible now


DevOpsProdigy KubeGraf / Nodes Overview
DevOpsProdigy KubeGraf
Cluster Status
Applications Overview
Nodes Overview
Dashboards
Edit
Plugin Config
Overview: Production. Nodes
k8s-node1
k8s-node2
k8s-node3
Show all
Hide all
Node: k8s-node1
IP: 192.168.101.131
CPU Cores: 40
RAM Total: 92.926 GiB
SWAP Total: 0 B
RootFS Total: 439.861 GiB
Sys Load (1m avg): 1.98
Uptime: 55.2 day
Pods limits
CPU (cores)
Memory
Used
51 (46.36 %)
2.886 (7.25 %)
30.492 GiB (33.02 %)
Requested
51 (46.36 %)
4.908 (12.33 %)
10.903 GiB (11.81 %)
Allocatable
110
39.8
92.351 GiB
Capacity
110
40
92.926 GiB
Open Dashboard
Hide Applications
Namespace: ingress-nginx (1)
CPU usage
CPU requested
Memory usage
Memory requested
nginx-ingress-controller-5fb84c58d6-s7
145m
N-A
970.234 MiB
N-A
Summary
145m
0m
970.234 MiB
0 B

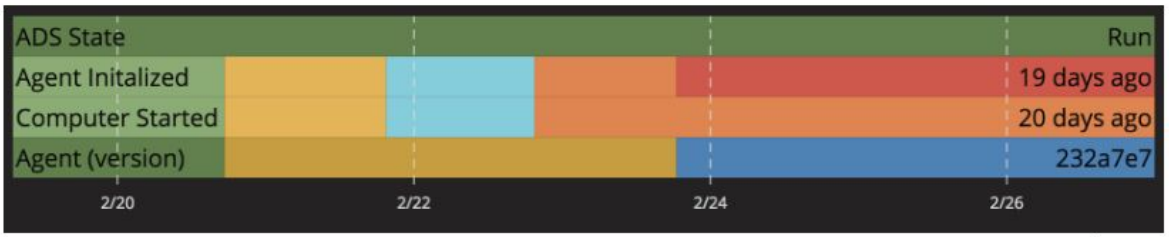

DevOpsProdigy KubeGraf / Applications Overview
DevOpsProdigy KubeGraf
Cluster Status
Applications Overview
Nodes Overview
Dashboards
Edit
Plugin Config
Overview: Production. Applications
1 / 69
Show all
Hide all
Namespace: kube-system
SHOW Cron Jobs (0)
SHOW Jobs (0)
Status: Active
Deployments
Statefulsets
Daemonsets
Other
No data
calico-kube-controllers
coredns
Services
dns-autoscaler
kubernetes-dashboard
tiller-deploy
filebeat
kube-proxy
Documentation
Support
Community
Open Source

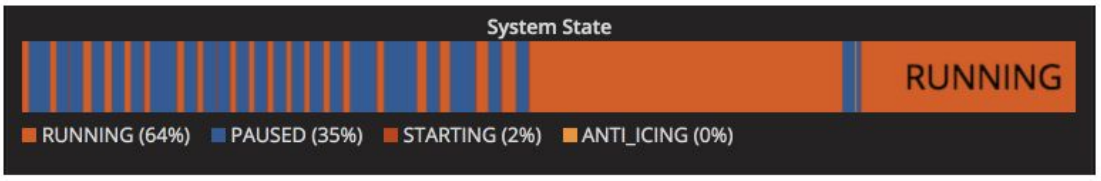

DevOpsProdigy KubeGraf / Applications Overview
DevOpsProdigy KubeGraf
Cluster Status
Applications Overview
Nodes Overview
Dashboards
Edit
Plugin Config
Overview: Production. Applications
1 / 77
Show all
Hide all
Status
Pod Name
Error message
Show warnings (38)
Namespace: prometheus
SHOW Cron Jobs (0)
SHOW Jobs (0)
Status: Active
Deployments
prometheus-operator-grafana
prometheus-operator-grafana-78dcfc4756-vt9gn
Services
prometheus-operator-grafana: 80/TCP
prometheus-operator-kube-state-metrics
prometheus-operator-kube-state-metrics-696f457df-fmgs
Services
prometheus-operator-kube-state-metrics: 8080/TCP
prometheus-operator-operator
prometheus-operator-operator-b8bffb78c-5dlvm
Services
prometheus-operator-operator: 8080/TCP
Statefulsets
alertmanager-prometheus-operator-alertmanager
alertmanager-prometheus-operator-alertmanager-0
Services
alertmanager-operated: 9093/TCP 6783/TCP
prometheus-operator-alertmanager: 9093/TCP
prometheus-prometheus-operator-prometheus
Daemonsets
prometheus-operator-prometheus-node-exporter
prometheus-operator-prometheus-node-exporter-4r8gt
prometheus-operator-prometheus-node-exporter-4vllx
prometheus-operator-prometheus-node-exporter-5m5fh
prometheus-operator-prometheus-node-exporter-7bcpm
prometheus-operator-prometheus-node-exporter-868ns
prometheus-operator-prometheus-node-exporter-f7mw9
prometheus-operator-prometheus-node-exporter-fcj5d4
prometheus-operator-prometheus-node-exporter-glq99
prometheus-operator-prometheus-node-exporter-hwfj5
prometheus-operator-prometheus-node-exporter-jwvrp
Show more pods
Services
prometheus-operator-prometheus-node-exporter: 9100/TCP
Other
No data
Documentation
Support
Community
Open Source
v6.7.1 (ca6d08d5cb)
New version available!

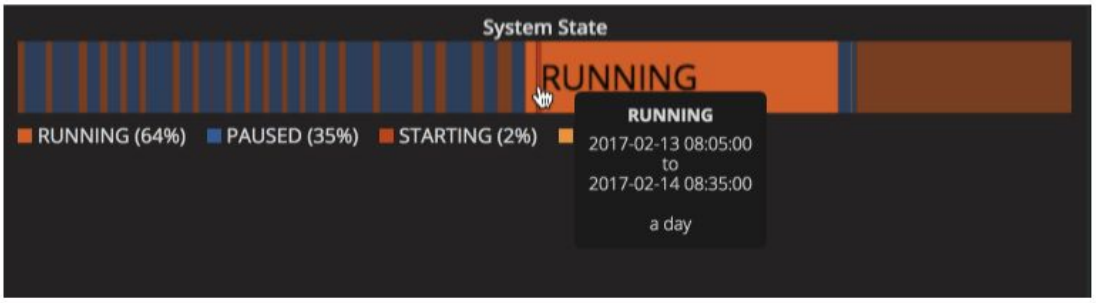


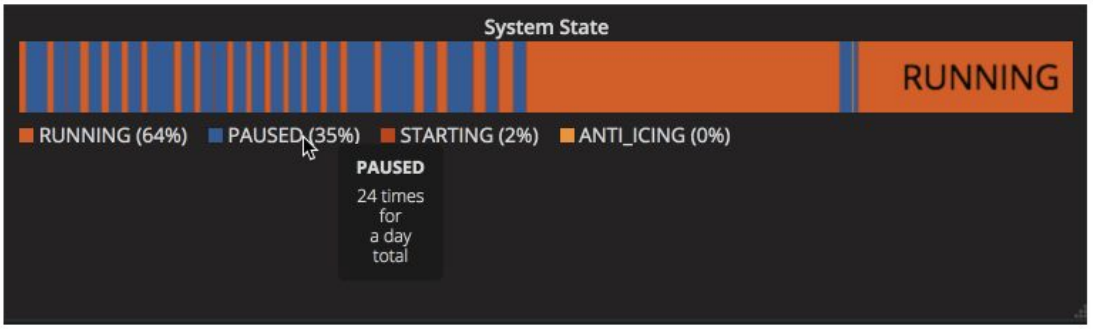
This page consists of three main info pages with detailed information about the Kubernetes cluster.

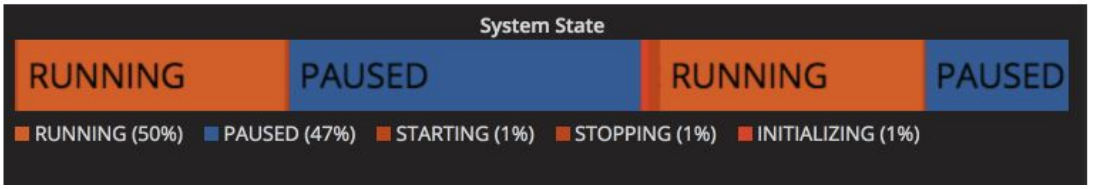
Links:

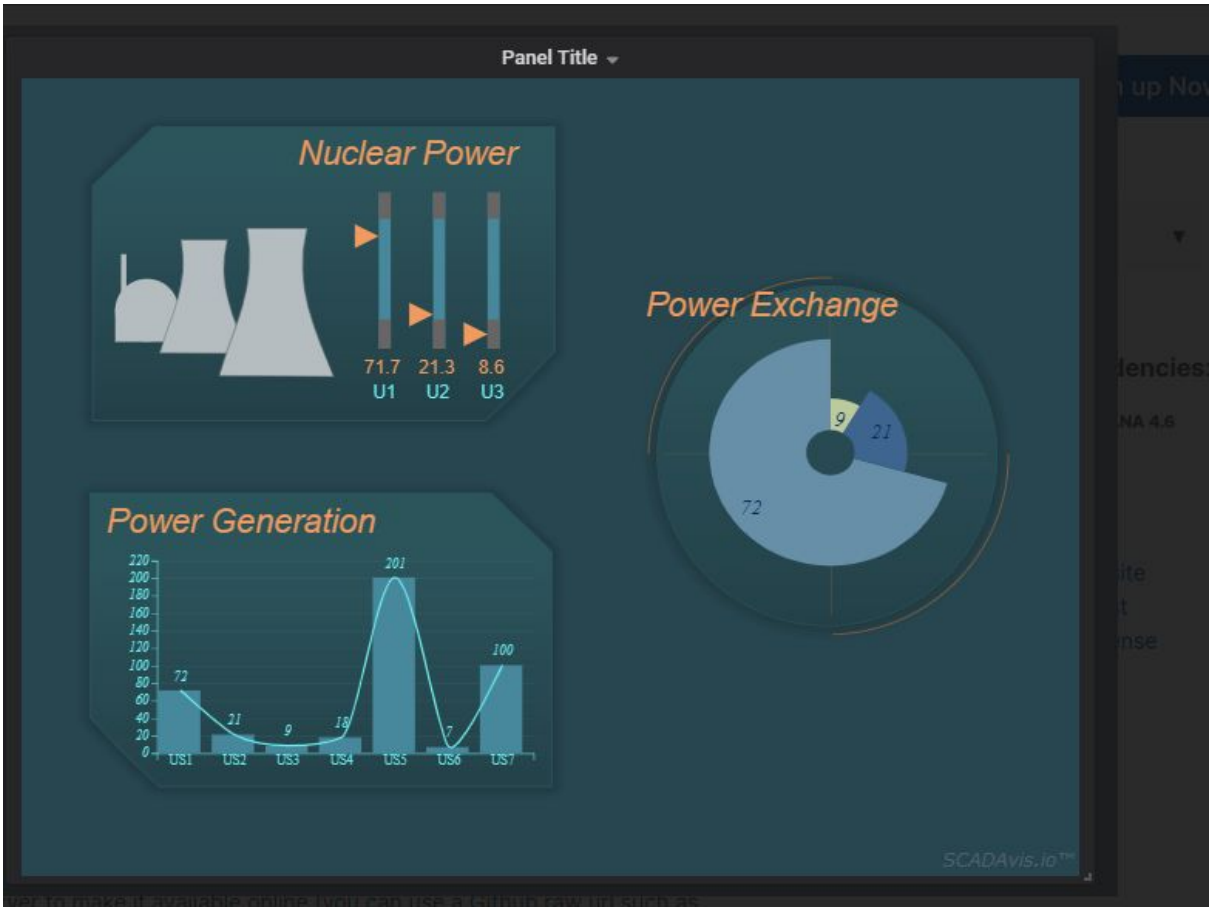

ADS State
Run
Agent Initalized
19 days ago
Computer Started
20 days ago
Agent (version)
232a7e7
2/20
2/22
2/24
2/26

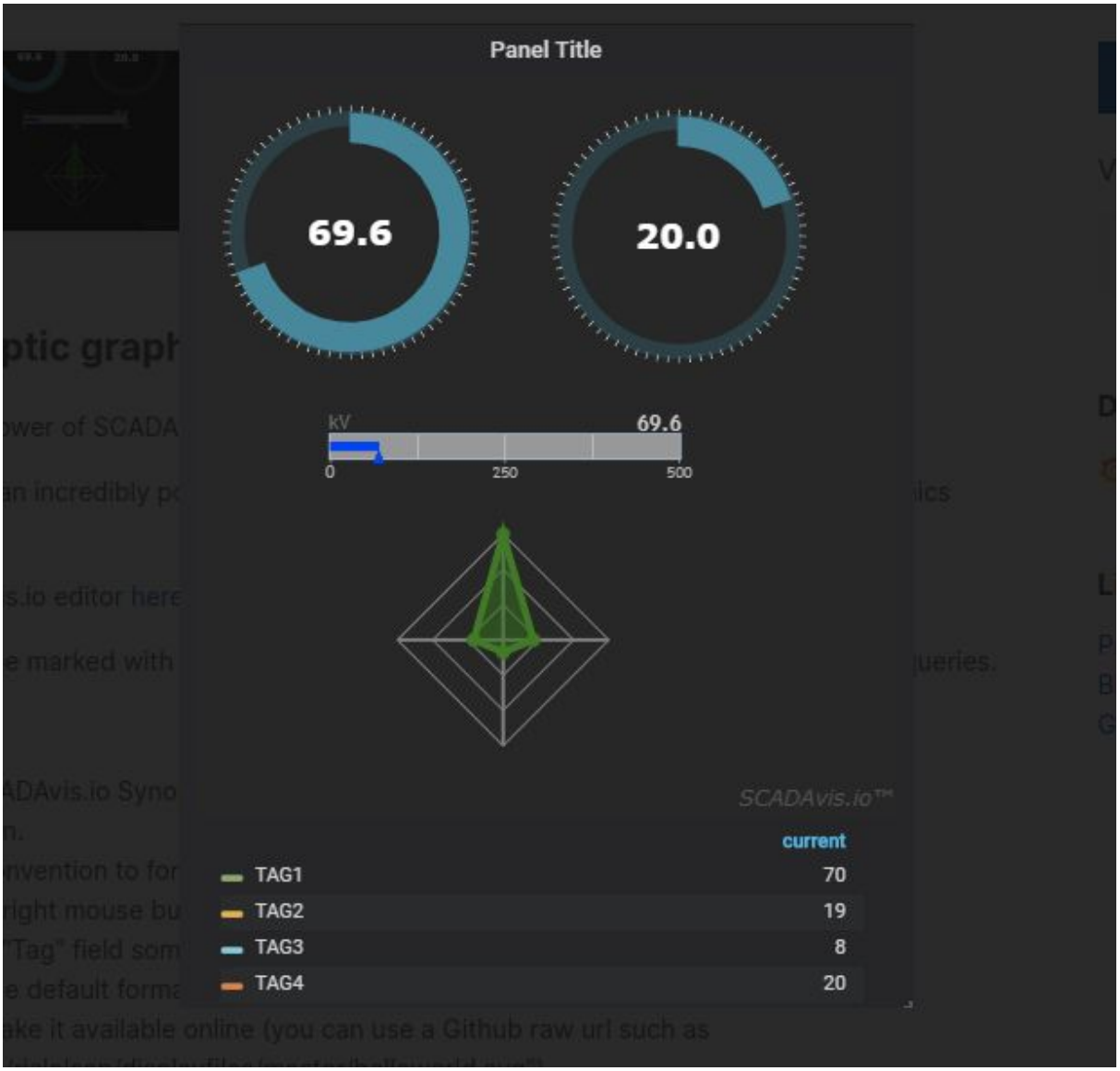

System State
RUNNING
RUNNING (64%)
PAUSED (35%)
STARTING (2%)
ANTI_ICING (0%)


System State
RUNNING
RUNNING (64%)
PAUSED (35%)
STARTING (2%)
RUNNING
2017-02-13 08:05:00
to
2017-02-14 08:35:00
a day

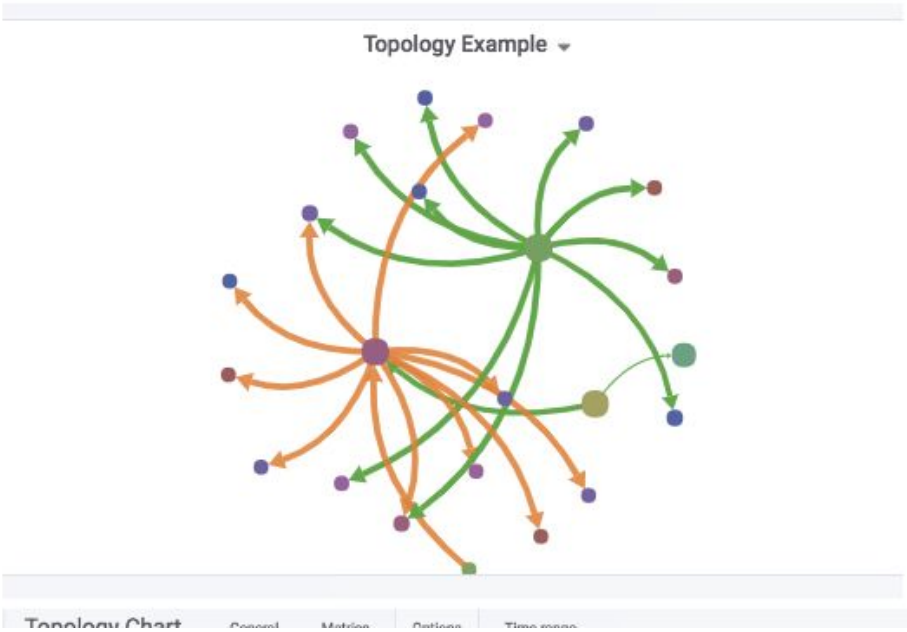

System State
RUNNING
RUNNING (64%)
PAUSED (35%)
STARTING (2%)
ANTI_ICING (0%)
PAUSED
24 times
for
a day
total


System State
RUNNING
PAUSED
RUNNING
PAUSED
RUNNING (50%)
PAUSED (47%)
STARTING (1%)
STOPPING (1%)
INITIALIZING (1%)

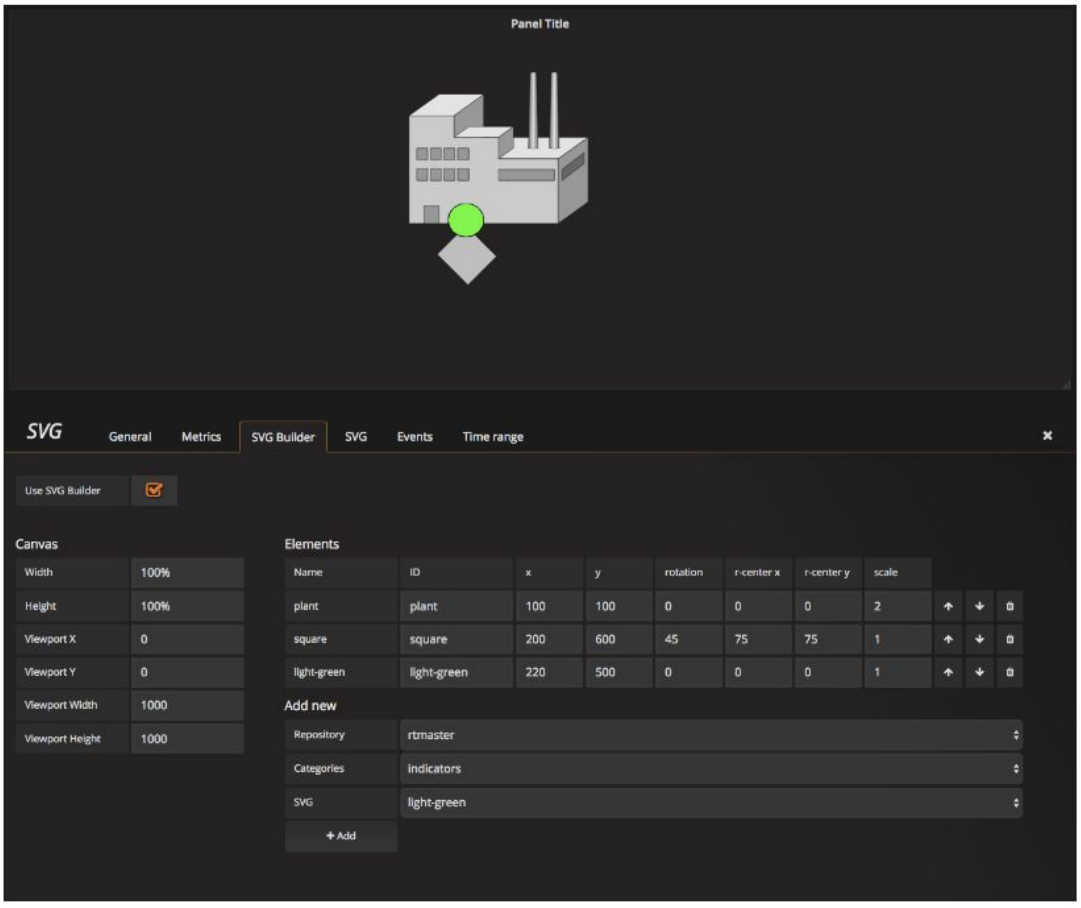

Panel Title
Nuclear Power
71.7
21.3
8.6
U1
U2
U3
Power Exchange
9
21
72
Power Generation
220
200
180
160
140
120
100
80
60
40
20
0
72
21
9
18
201
7
100
US1
US2
US3
US4
US5
US6
US7
SCADAvis.io™


Panel Title
69.6
20.0
kV
69.6
0
250
500
SCADAvis.io™
current
TAG1
70
TAG2
19
TAG3
8
TAG4
20

Topology Panel is an Sigma.js-based plugin for Elasticsearch and InfluxDB.

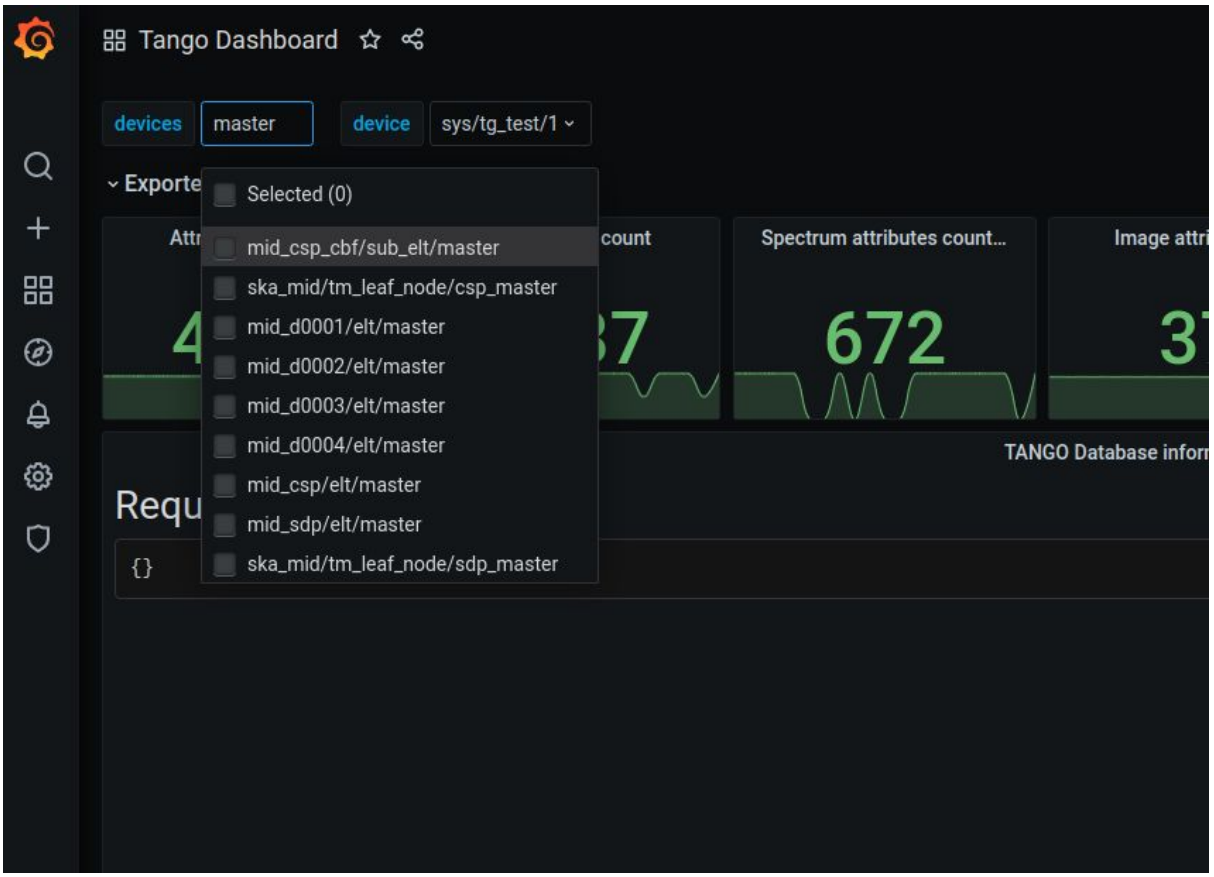


SVG Builder



“Navigating” through Tango devices:

uses a simple combo box:



The question is “pivot or persevere”.

Shall we continue working on Webjive as planned? or shall we explore the potential of this “new” solution? If so, shall we do it in PI7?

Giorgio thinks that:

1) The solution based on Grafana is more mature and tested than Webjive.
2) It yields GUIs that are visually more attractive, with a richer set of widgets from which to choose.
3) The dashboard editor provides many of the features that are missing and planned for Webjive.
4) It needs more development in the area related to controlling devices and in the area of associating Tango devices/attributes/states to widgets.
5) We will have more difficulties in changing the internals (of Grafana, Prometheus) in case something will appear not to be working (for example in terms of performance bottlenecks).

6) We will need to redo the performance assessment.
7) We will need to carefully assess the reliability of exporters.
8) The Tango community will have a web-based GUI toolkit in any case.

All things considered, Giorgio thinks that the current goal in PI7 (building attractive, connected and complex dashboards) could be achieved also with this new solution, instead of using Webjive. If we can achieve that, then this fact could be the demonstration that such a solution works.
After all we already have the basic building blocks (Grafana, Prometheus, TangoGQL, an exporter, an example of a button widget).
We need to focus on building the needed dashboards and 1-2 new plugins for Grafana for entering commands.
In doing this work we can assess the possible difficulties due to the poor Tango introspection capabilities, but for the moment live with what is now possible with Prometheus and Grafana.

Decision

After a discussion between Giorgio and Nick:

1. PERSEVERE with webjive.
2. Explore potential strong negative risk with Grafana + Prometheus that has to do with inability to sustain a high data rate for monitoring (the 1000 updates in less than a second) and a small latency when issuing commands. See feature https://jira.skatelescope.org/browse/SP-1089.