



Taurus Status Report

Martí Caixal, Emilio Morales, Miquel Navarro, Carlos Pascual (on-leave), Jose Ramos, Sergi Rubio and

Zbigniew Reszela

on behalf of ALBA Controls Section

27-29.06.2023

37th Tango Community Meeting 2023

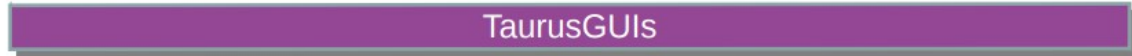


Agenda

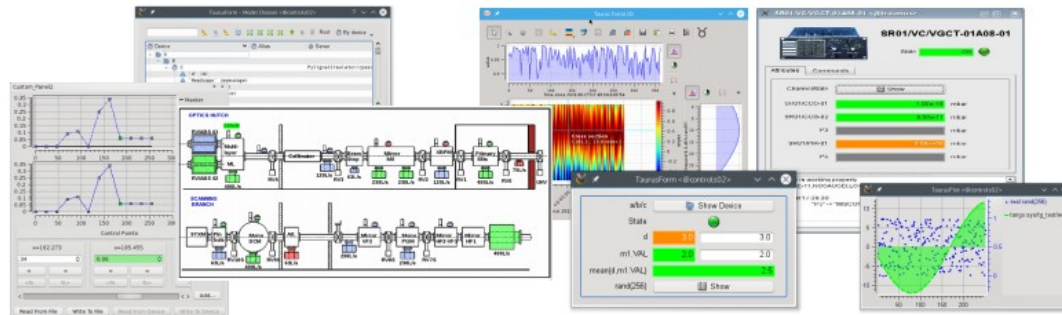
- New features in Taurus
 - Archiving support in TaurusTrend
 - Multi models
- Taurus performance optimization
- Taurus Community

Introduction to Taurus

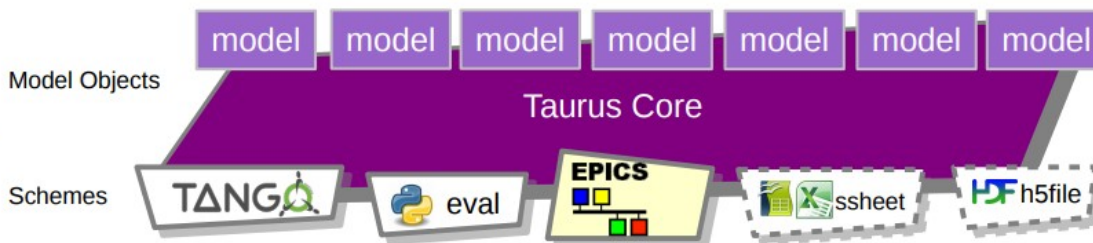
TaurusGUIs



Taurus Qt Widgets



Taurus Core



External Hardware and data sources

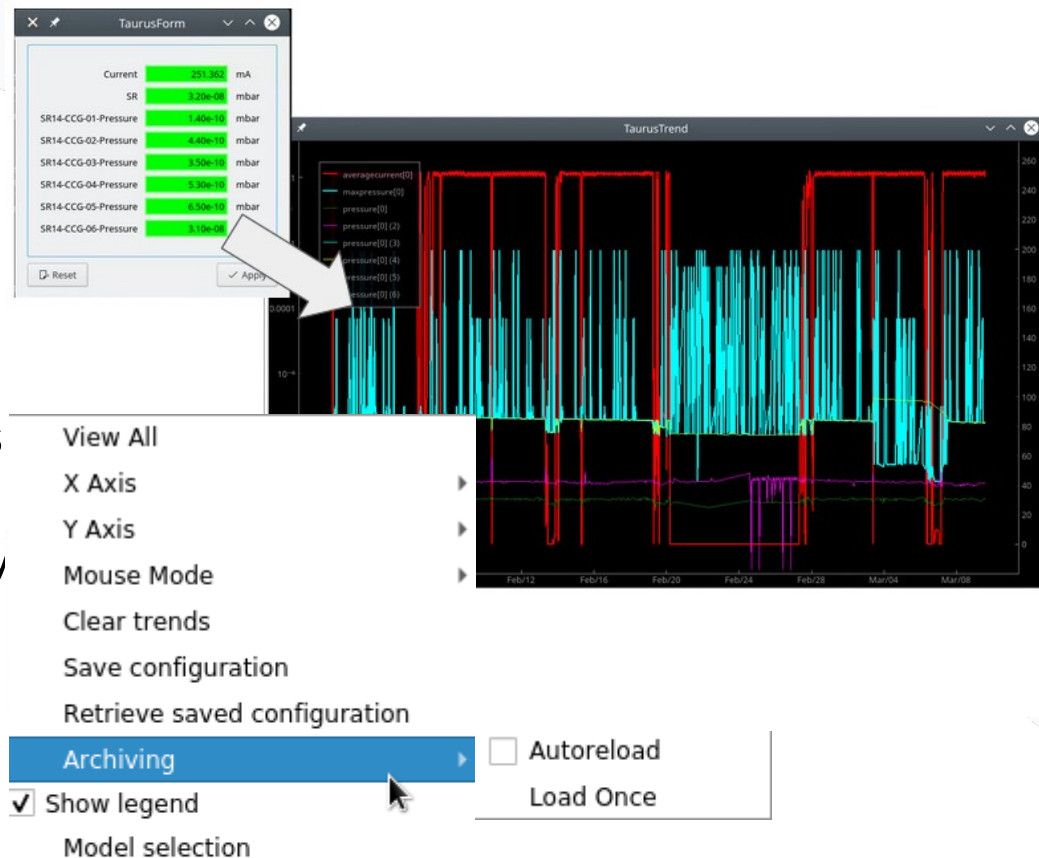


Agenda

- New features in Taurus
 - Archiving support in TaurusTrend
 - Multi models
- Taurus performance optimization
- Taurus Community

Trend of archiving data

- Widget displays either archived data or current data:
 - Autoreload by just changing the time scale
 - Load Once on user request
- Attributes are just drag & drop from other widgets or applications into the plot
- Underneath uses pyhdbpp library that supports both MySQL and TimeScaleDB, easy to extend for other backends
- Thanks to Jose Ramos and Sergi Rubio (ALBA)!

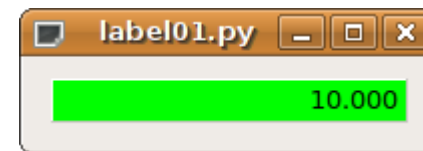
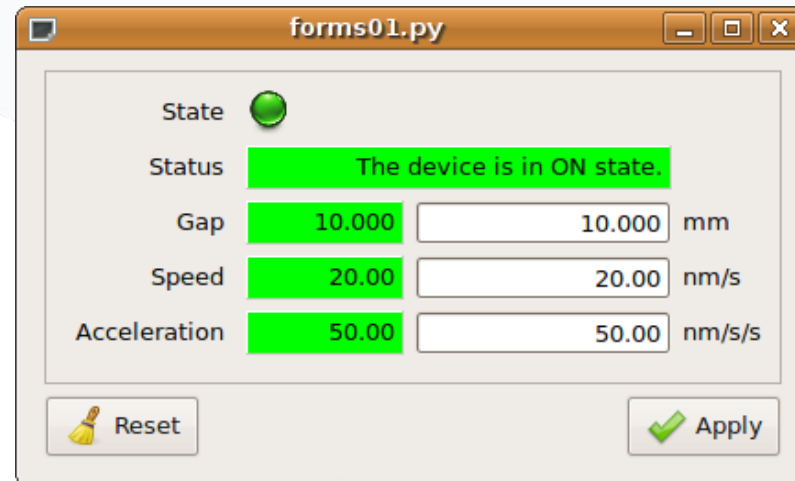


Agenda

- New features in Taurus
 - Archiving support in TaurusTrend
 - **Multi models**
- Taurus performance optimization
- Taurus Community

Multi models (TEP20) - problem

- Complex widgets:
 - "model container"
e.g. TaurusForm, TaurusPlot, etc.
 - "model composer"
e.g. TaurusPlotDataItem, custom TaurusLabel, etc.
- Previous solution - custom implementations and cluttered API e.g.
 - addModel(), insertModel(), updateModels(), etc.
 - setModel() for the Y-axis and setXModel(), getXModelName(), etc. for the X-axis



Multi models (TEP20) - solution

- Implementation done on the TaurusBaseComponent class level (base class of all widgets)
- support multi-models by assigning a "key" to each model e.g.

```
w.setModel("tango:sys/tg_test/1/state", key="bg")  
w.getModelObj(key="bg")  
# -> TangoAttribute(tango://db:10000/sys/tg_test/1/state)
```
- Widget declares which keys are supported
- Backwards compatible implementation
- Implementation done just on the base classes level – no built-in widgets use this new API yet.
- Thanks to Carlos Pascual (ALBA, on-leave)

Multi models (TEP20) - example

```

6  class PowerMeter2(Qt.QProgressBar, TaurusBaseComponent):
7      """A Taurus-ified QProgressBar with separate models for value and color"""
8
9      # setFormat() defined by both TaurusBaseComponent and QProgressBar. Rename.
10     setFormat = TaurusBaseComponent.setFormat
11     setBarFormat = Qt.QProgressBar.setFormat
12
13     modelKeys = ["power", "color"] # support 2 models (default key is "power")
14     _template = "QProgressBar::chunk {background: %s}" # stylesheet template
15
16     def __init__(self, parent=None, value_range=(0, 100)):
17         super(PowerMeter2, self).__init__(parent=parent)
18         self.setOrientation(Qt.Qt.Vertical)
19         self.setRange(*value_range)
20         self.setTextVisible(False)
21
22     def handleEvent(self, evt_src, evt_type, evt_value):
23         """reimplemented from TaurusBaseComponent"""
24         try:
25             if evt_src is self.getModelObj(key="power"):
26                 self.setValue(int(evt_value.rvalue.m))
27             elif evt_src is self.getModelObj(key="color"):
28                 self.setStyleSheet(self._template % evt_value.rvalue)
29         except Exception as e:
30             self.info("Skipping event. Reason: %s", e)
31
32
33
34
35
36
37     w = PowerMeter2()
38     w.setModel("eval:Q(60+20*rand())") # implicit use of key="power"
39     w.setModel("eval:['green','red','blue'][randint(3)]", key="color")

```

Full example: https://taurus-scada.org/devel/custom_widgets.html#multi-model-support-model-composer

Multi models (TEP20) - example

```

6  class PowerMeter2(Qt.QProgressBar, TaurusBaseComponent):
7      """A Taurus-ified QProgressBar with separate models for value and color"""
8
9      # setFormat() defined by both TaurusBaseComponent and QProgressBar. Rename.
10     setFormat = TaurusBaseComponent.setFormat
11     setBarFormat = Qt.QProgressBar.setFormat
12
13     modelKeys = ["power", "color"] # support 2 models (default key is "power")
14     _template = "QProgressBar::chunk {background: %s}" # stylesheet template
15
16     def __init__(self, parent=None, value_range=(0, 100)):
17         super(PowerMeter2, self).__init__(parent=parent)
18         self.setOrientation(Qt.Qt.Vertical)
19         self.setRange(*value_range)
20         self.setTextVisible(False)
21
22     def handleEvent(self, evt_src, evt_type, evt_value):
23         """reimplemented from TaurusBaseComponent"""
24         try:
25             if evt_src is self.getModelObj(key="power"):
26                 self.setValue(int(evt_value.rvalue.m))
27             elif evt_src is self.getModelObj(key="color"):
28                 self.setStyleSheet(self._template % evt_value.rvalue)
29         except Exception as e:
30             self.info("Skipping event. Reason: %s", e)
31
32
33
34
35
36
37     w = PowerMeter2()
38     w.setModel("eval:Q(60+20*rand())") # implicit use of key="power"
39     w.setModel("eval:['green','red','blue'][randint(3)]", key="color")

```



Full example: https://taurus-scada.org/devel/custom_widgets.html#multi-model-support-model-composer

Multi models (TEP20) - example

```

6  class PowerMeter2(Qt.QProgressBar, TaurusBaseComponent):
7      """A Taurus-ified QProgressBar with separate models for value and color"""
8
9      # setFormat() defined by both TaurusBaseComponent and QProgressBar. Rename.
10     setFormat = TaurusBaseComponent.setFormat
11     setBarFormat = Qt.QProgressBar.setFormat
12
13     modelKeys = ["power", "color"] # support 2 models (default key is "power")
14     _template = "QProgressBar::chunk {background: %s}" # stylesheet template
15
16     def __init__(self, parent=None, value_range=(0, 100)):
17         super(PowerMeter2, self).__init__(parent=parent)
18         self.setOrientation(Qt.Qt.Vertical)
19         self.setRange(*value_range)
20         self.setTextVisible(False)
21
22     def handleEvent(self, evt_src, evt_type, evt_value):
23         """reimplemented from TaurusBaseComponent"""
24         try:
25             if evt_src is self.getModelObj(key="power"):
26                 self.setValue(int(evt_value.rvalue.m))
27             elif evt_src is self.getModelObj(key="color"):
28                 self.setStyleSheet(self._template % evt_value.rvalue)
29         except Exception as e:
30             self.info("Skipping event. Reason: %s", e)
31
32
33
34
35
36
37     w = PowerMeter2()
38     w.setModel("eval:Q(60+20*rand())") # implicit use of key="power"
39     w.setModel("eval:['green','red','blue'][randint(3)]", key="color")

```



Full example: https://taurus-scada.org/devel/custom_widgets.html#multi-model-support-model-composer

Multi models (TEP20) - example

```
6 class PowerMeter2(Qt.QProgressBar, TaurusBaseComponent):
7     """A Taurus-ified QProgressBar with separate models for value and color"""
8
9     # setFormat() defined by both TaurusBaseComponent and QProgressBar. Rename.
10    setFormat = TaurusBaseComponent.setFormat
11    setBarFormat = Qt.QProgressBar.setFormat
12
13    modelKeys = ["power", "color"] # support 2 models (default key is "power")
14    _template = "QProgressBar::chunk {background: %s}" # stylesheet template
15
16    def __init__(self, parent=None, value_range=(0, 100)):
17        super(PowerMeter2, self).__init__(parent=parent)
18        self.setOrientation(Qt.Qt.Vertical)
19        self.setRange(*value_range)
20        self.setTextVisible(False)
21
22    def handleEvent(self, evt_src, evt_type, evt_value):
23        """reimplemented from TaurusBaseComponent"""
24        try:
25            if evt_src is self.getModelObj(key="power"):
26                self.setValue(int(evt_value.rvalue.m))
27            elif evt_src is self.getModelObj(key="color"):
28                self.setStyleSheet(self._template % evt_value.rvalue)
29        except Exception as e:
30            self.info("Skipping event. Reason: %s", e)
31
32
33
34
35
36
37    w = PowerMeter2()
38    w.setModel("eval:Q(60+20*rand())") # implicit use of key="power"
39    w.setModel("eval:['green','red','blue'][randint(3)]", key="color")
```



Full example: https://taurus-scada.org/devel/custom_widgets.html#multi-model-support-model-composer

Agenda

- New features in Taurus
 - Archiving support in TaurusTrend
 - Multi models
- Taurus performance optimization
- Taurus Community

Current limitations

- Long GUI startup time
 - “big” GUIs e.g. for accelerator control
 - GUIs connecting to unavailable devices (timeouts) e.g. for beamline control
- Taurus polling issues when attributes give timeouts or are “slow”
- Benchmark developed and base line measured
- “Slow” and timeout attributes are the worse cases due to bugs triggering multiple reads (especially those using events)
- But there are also real scalability issues...

Taurus Attribute on application startup

```
label = TaurusLabel()
label.setModel("a/b/c/d")
# model.getValueObj()
```

```
# subscribe to
ATTR_CONF_EVENT
# read in event callback - #1275
a = taurus.Attribute("a/b/c/d")

# subscribe to CHANGE_EVENT
# read (Tango internal)
# if no events → add to polling
a.addListener(callback)

# getValueObj()
a.read() # cache=True
```

Refactor for the scalability issues – app startup

Case Study:

RF plant – 5 devices with 577 attributes (500 with events, 77 w/o events)

PyTango subscription to CHANGE_EVENT of 500 attrs (read: 1.4 s)	4.1 s
PyTango subscription to ATTR_CONF_EVENT of 414 attrs	3.1 s
PyTango read of 77 attrs w/o events	0.1 s
Taurus startup (Attribute() + addListener() + getValueObj()) of 577 attrs	15.4 s
Taurus Labels startup of 414 attrs	17.6 s
Taurus Form startup of 414 attrs	27.2 s

Table presented on the Tango SIG – Taurus Workshop contains an error in Taurus startup time

Refactor for the scalability issues – app startup

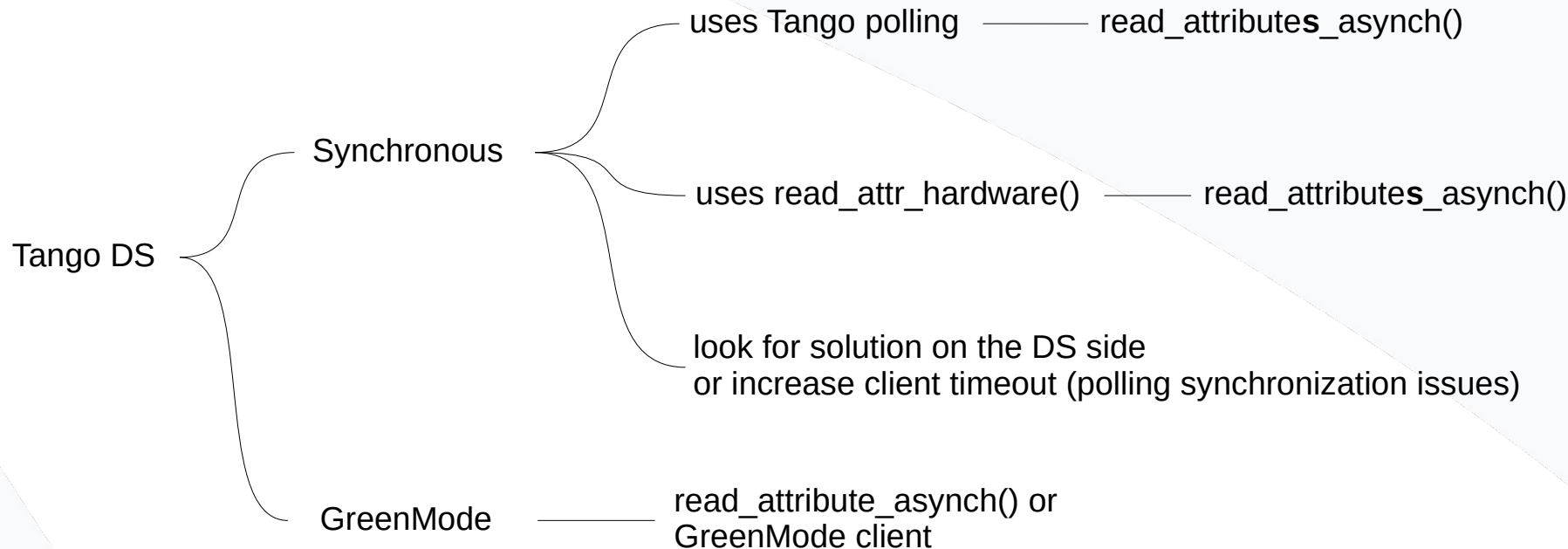
- Discuss with Tango Community (TangoTickets#33) the following ideas:
 - Add to Tango: **subscribe_event(read=False)**
 - In Taurus: when all models are set trigger one shot: `read_attributes_async()` and `read_attributes_reply()` for attributes with events
 - Add to Tango: **subscribe_events_asynch()** and **subscribe_events_replies()**
 - In Taurus: `setModel()` create Attribute object but does not subscribe to events.
 - In Taurus: when all models are set trigger subscriptions
 - **Unlock** `subscribe_event()` to take a profit of **GreenMode**:
 - From our first tests we don't see any improvements with using GreenMode

Taurus Attribute(s) on Taurus polling

```
for dev, attrs in devs.items()
    # read_attributes_async(attrs)
    req_id = dev.poll(attrs, async=True)
    req_ids[dev] = attrs, req_id

for dev, (attrs, req_id) in req_ids.items():
    # self.read_attributes_reply(req_id)
    dev.poll(attrs, req_id=req_id)
```

How Taurus polling fits to different Tango DS?



Taurus Performance Optimization - Roadmap

- Discuss with Tango Community (TangoTickets#33) the eventual improvements in subscription to events
- Transform benchmark into automatic tests (core - 80% & Qt - 20%)
- Fix bugs to minimize startup time of worse case scenarios timeout and “slow” attributes
- Profile taurus core and try to optimize as much as possible (40-50%)
- Refactor for the scalability issues
 - Option 1: use newly added Tango API (eventually Green Mode for subscribe events)
 - Option 2: natively integrate Delayed Subscriber (GUI appears faster (using polling) and subscriptions done in background)

Agenda

- New features in Taurus
 - Archiving support in TaurusTrend
 - Multi models
- Taurus performance optimization
- **Taurus Community**

Not so long time ago...

taurus-org > taurus > Issues > #1140

Open

Issue created 2 years ago by Carlos Pascual Owner

Close issue



The boat is sinking...

This is a quite special bug report. It is about a community issue

IMHO, we should seriously re- think the organization of the Taurus development, and specially the core members (@taurus-org/integrators). Currently the taurus management (integration, planning, steering,...) is *de-facto* concentrated almost exclusively on a single institute (ALBA)... and mostly a single person (me). This is neither healthy for the Taurus Project nor for me.

I assume the responsibility for this situation because at some point, 2-3 years ago, I perceived that the Taurus community had gotten enough momentum to be self-sustained, and therefore I neglected some community grooming to devote more time to other pressing issues..., this lead to decreased involvement from other institutes... and then more burden on ALBA... and then less time for community grooming... and this spiraled down to the current situation.

In contrast, let's look at the case of the Sardana community (which naturally should overlap a lot with the Taurus one). Thanks undoubtedly to the great amount of effort put in the last years by @reszelaz in building the community, it is now healthier than ever (i.e spiraling up instead of down). Big lesson here.

So, what to do?

Without discussing specific measures, I see the following general aspects in which we could improve:

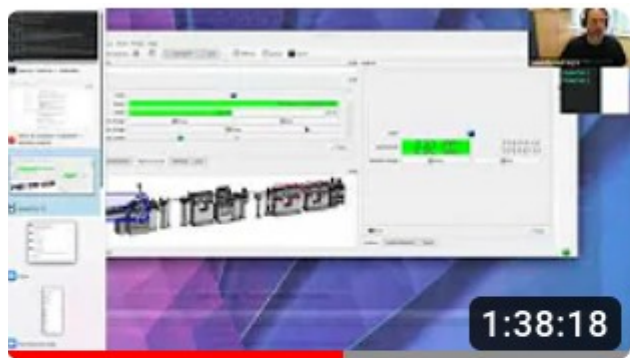
- increasing the commitment from the large institutes that use Taurus (e.g. securing some fixed amount of person-hours from each institute).
- opening the governance. In 2013 we started with quite a formal structure: a Memorandum of Understanding, rules for the officially approving releases, etc. This was done in fear of introducing instability on a critical infrastructure component, but IMHO a Bazar-like approach would have been better.
- In line with the previous point, encouraging casual contributors to participate and become integrators. Make the "core team" more permeable and dynamic. Lower the barrier for becoming (and ceasing to be) an integrator...
- improving communication: re-think about channels of communication ...

I leave it here.

Let's use this issue as the place for discussion, proposals, etc

Please do not leave this issue unwatched and without comments... the irony of it would kill me... ;)

Taurus Webinar & Carlos Farewell



**Tango Kernel Webinar #5 -
Taurus - Part 2**

129 views • 1 year ago



**Tango Kernel Webinar #5 -
Taurus - Bonus during break**

55 views • 1 year ago



**Tango Kernel Webinar #5 -
Taurus - Part 1**

180 views • 1 year ago

Videos: <https://www.youtube.com/@tango-controls>

Materials: [https://gitlab.com/taurus-org/taurus/-/wikis/Taurus-webinar-\(for-maintainers\)](https://gitlab.com/taurus-org/taurus/-/wikis/Taurus-webinar-(for-maintainers))

Tango SIG - Taurus Workshop



Overview
Timetable
Contribution List
Registration
Participant List
Contact

In the [TANGO community](#), TAURUS is the de facto standard for those searching for a powerful python-based solution for CLIs and GUIs. Initially developed at ALBA Synchrotron (Spain), TAURUS is also a successful open-source project widely adopted by major scientific facilities for their daily operation.

After 15 years of constant development, TAURUS has reached a key point in its history. The emergence of web-based solutions tends to challenge the existence of traditional GUIs frameworks by inviting them to redefine their role or to justify their maintenance beyond the catalog of existing applications. The historical TAURUS maintainers team is also changing and could offer new opportunities in terms of involvement in the life cycle of the project.

The proposed 2 days event will be organised around two topics. The first day will be dedicated to **feedback and GUI strategy for the future**. During the associated sessions, speakers and attendees will have the opportunity to share their experience with TAURUS and explain their strategy and thoughts regarding the future of their graphical interfaces. Please note that feedback from both computing and operation teams would be appreciated. The second day will be focused on the **organisational and technical future of TAURUS** itself.

Regarding the organisation details, this hybrid TAURUS workshop will be hosted by the ESRF and held on **March 14th & 15th 2023**. Speakers and attendees are welcome on site but the sessions will be also accessible remotely.



Starts 14 Mar 2023, 09:15
Ends 15 Mar 2023, 12:30
Europe/Paris

ESRF
MD-1-21
71 avenue des Martyrs, 38000 Grenoble

Indico: <https://indico.esrf.fr/event/76/>

27-29/06/2023

37th Tango Community Meeting 2023

- First of all, thanks to the ESRF for the organization of the workshop!
- Several issues from SOLARIS reminded
- Type hinting e.g. typing, traits, stubs, etc. suggested.
- QuickGUI (define GUIs in YAML) developed at MAX IV
- Polish taurus_pyqtgraph
- Silx for 2D plotting
- Consult with ATK, QTango performance problems

Most of them still on our TODO list.

Recent Merge Requests (last 2 months)


Unofficial Release 5.1.6 #1259 · created 1 week ago by Emilio Morales	Merged ✓ Approved 3 updated 1 week ago
py311-deprecation-fix #1258 · created 3 weeks ago by Antonio Bartalesi	Merged ✓ Approved 3 updated 1 week ago
Run testsuite on conda-3.10 and conda-3.11 #1257 · created 1 month ago by Zbigniew Reszela	Merged ✓ 1 updated 3 weeks ago
Adapt tests to PyTango 9.4.0 with regard to attr w_value #1256 · created 1 month ago by Zbigniew Reszela	Merged ✗ 1 updated 3 weeks ago
Fix UR.parse_units() usage to follow its API #1255 · created 1 month ago by Zbigniew Reszela	Merged ✓ Approved 2 updated 3 weeks ago
Correct bug converting enums to str. #1254 · created 2 months ago by Emilio Morales	Merged ✓ Approved 3 updated 3 weeks ago
Fix pint package in bullseye image #16 · created 1 week ago by Benjamin Bertrand ¹ bullseye	Merged ✓ Approved 3 updated 1 week ago
Switch to micromamba image #15 · created 1 month ago by Benjamin Bertrand	Merged ✗ Approved 7 updated 1 month ago
Draft: Build docker images with conda with py-3.11 and py-3.10 #14 · created 1 month ago by Zbigniew Reszela	Closed ⓘ 1 updated 1 month ago
Full tango names on legend #107 · created 2 weeks ago by Jose A. Ramos	✗ 0 updated 2 weeks ago
Datainspector improvements #106 · created 3 weeks ago by Antonio Bartalesi	✓ 0 updated 3 weeks ago
Put linter job to different stage in CI/CD #105 · created 3 weeks ago by Jakub Kowalczyk	✗ 0 updated 3 weeks ago
Statistics tool #104 · created 3 weeks ago by Jakub Kowalczyk	✗ 0 updated 3 weeks ago
Fix compatibility with pyqtgraph 0.13.2 #103 · created 1 month ago by Marti C	Merged ✓ Approved 9 updated 1 month ago
Changed way of mutating y shape and forcing a read after extending buffers to... #102 · created 4 months ago by Jose A. Ramos	Merged ✓ Approved 1 updated 2 months ago
Fix compatibility with pyqtgraph 0.13 #101 · created 5 months ago by Zbigniew Reszela	Merged ✓ Approved 2 updated 2 months ago

Taurus - 6


Taurus Docker - 3




Taurus PyQtGraph - 7

Recent Merge Requests (last 2 months)

Emilio Morales (ALBA)	Merged  Approved  3 updated 1 week ago	Taurus - 6
Antonio Bartalesi (MAX IV)	Merged  8 Approved  1 3 updated 1 week ago	
Run testsuite on conda-3.10 and conda-3.11 11257 · created 1 month ago by Zbigniew Reszela	Merged  1 updated 3 weeks ago	
Adapt tests to PyTango 9.4.0 with regard to attr w_value 11256 · created 1 month ago by Zbigniew Reszela	Merged  1 updated 3 weeks ago	
Fix UR.parse_units() usage to follow its API 11255 · created 1 month ago by Zbigniew Reszela	Merged  8 Approved  2 updated 3 weeks ago	
Emilio Morales (ALBA)	Merged  Approved  3 updated 3 weeks ago	
Benjamin Bertand (MAX IV)	Merged  8 Approved  3 updated 1 week ago	
Benjamin Bertand (MAX IV)	Merged  Approved  7 updated 1 month ago	
Draft: Build docker images with conda with py-3.11 and py-3.10 14 · created 1 month ago by Zbigniew Reszela	Closed  1 updated 1 month ago	
Jose Ramos (ALBA)	 0 updated 2 weeks ago	Taurus PyQtGraph - 7
Antonio Bartalesi (MAX IV)	 0 updated 3 weeks ago	
Jakub Kowalczyk (MAX IV; S2)	 0 updated 3 weeks ago	
Jakub Kowalczyk (MAX IV; S2)	 0 updated 3 weeks ago	
Marti Caixal (ALBA)	Merged  Approved  9 updated 1 month ago	
Jose Ramos (ALBA)	Merged  Approved  1 updated 2 months ago	
Fix compatibility with pyqtgraph 0.13 1101 · created 5 months ago by Zbigniew Reszela	Merged  8 Approved  2 updated 2 months ago	

First Taurus project follow-up meeting

From Guifré Cuní 

To tauruslib-devel@lists.sourceforge.net , tauruslib-users@lists.sourceforge.net , info@tango-controls.org  12:49

Subject **[Tango-info] Taurus project first follow-up meeting - invitation & poll**

Dear Community,

We are thrilled to announce the first follow-up meeting for the Taurus project, which aims to enhance the Taurus Community and build upon the decisions made during the Tango SIG - Taurus Workshop [1]. We cordially invite you to join us for this important gathering.

To ensure that we accommodate everyone's availability, we kindly request you to participate in the availability poll [2]. The poll will remain open until the end of June, after which we will finalize the meeting date.

For your reference, the tentative agenda can be found here [3].

We look forward to your active participation and valuable contributions.

Best regards,

Guifré Cuní on behalf of the Controls Section of the ALBA Synchrotron

[1] Tango SIG - Taurus Workshop Indico: <https://indico.esrf.fr/event/76>
[2] Availability poll: <https://framadate.org/emnyoVynZdVSoPna>
[3] Tentative agenda: https://gitlab.com/taurus-org/taurus-followup/-/blob/main/next_meeting/AGENDA.md

Conclusions

- There are two recent major enhancements to Taurus:
 - Archiving support in TaurusTrend
 - Multi Models
- We have identified and understood the Taurus performance issues.
- Taurus performance optimization project is in progress at ALBA.
- Efforts to revive the Taurus Community are ongoing.

Thank You!