



Taurus performance and technical roadmap

Martí Caixal, Emilio Morales, Miquel Navarro and
Zbigniew Reszela
on behalf of ALBA Controls Section

14-15/03/2023

Taurus Workshop; ESRF



OUTLINE

- Taurus application startup and polling
- Known problems/limitations
- Taurus Performance Optimization - Roadmap

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
```

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)
```

Known problems/limitations

Benchmark

- Attributes with and w/o events
 - Fast attribute – read takes 0 s
 - Exception attribute – read raises an exception immediately
 - Slow attribute – read takes 2 s
 - Timeout attribute – read takes > 3 s (default timeout = 3 s)
- Attribute redistributed in one Device and multiple Devices
 - Tango serialization monitor
 - `read_attributes_async()` & `read_attributes_reply()`

Fast attribute on application startup

- Attribute w/o events
 - Reads: ATTR_CONF_EVENT callback
 - Startup time (1 attr): **~100 ms**; (3 attrs*): **~130 ms** (+15 ms per attr)
- Attribute with events
 - Reads: ATTR_CONF_EVENT callback + subscription
 - Startup time (1 attr): **~100 ms**; (3 attrs*): **~130 ms** (+15 ms per attr)

* from 3 different devices

Exception attribute on application startup

- Attribute w/o events
 - Reads: ATTR_CONF_EVENT callback
 - Startup time (1 attr): **~100 ms**; (3 attrs*): **~140 ms** (+20 ms per attr)
- Attribute with events
 - Reads: ATTR_CONF_EVENT callback + subscription
 - Startup time (1 attr): **~100 ms**; (3 attrs*): **~140 ms** (+20 ms per attr)

* from 3 different devices

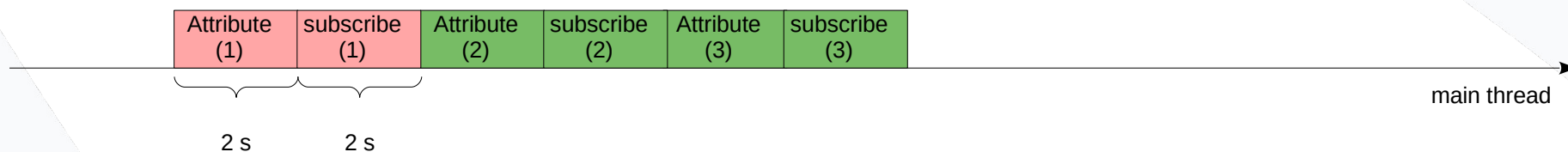
Slow attribute on application startup

- Attribute w/o events
 - Reads: ATTR_CONF_EVENT callback
 - Startup time (1 attr): **~2 s**
 - Startup time (3 attrs from 3 different devices): **~6 s**



Slow attribute on application startup

- Attribute with events
 - Reads: ATTR_CONF_EVENT callback + subscription
 - Startup time (1 attr): **~4 s**
 - Startup time (3 attrs from 3 different devices): **~12 s**

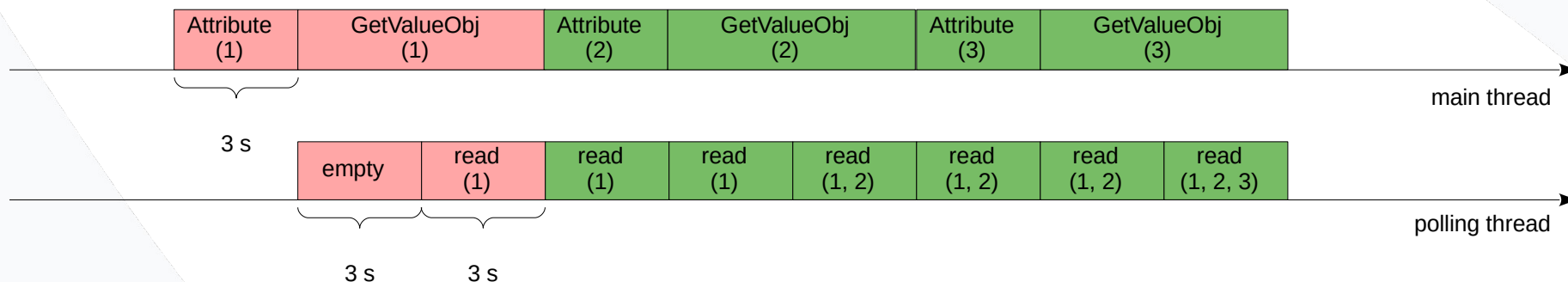


Slow attribute considerations

- Polling:
 - Few Slow attributes affect all the other attributes from the same device.

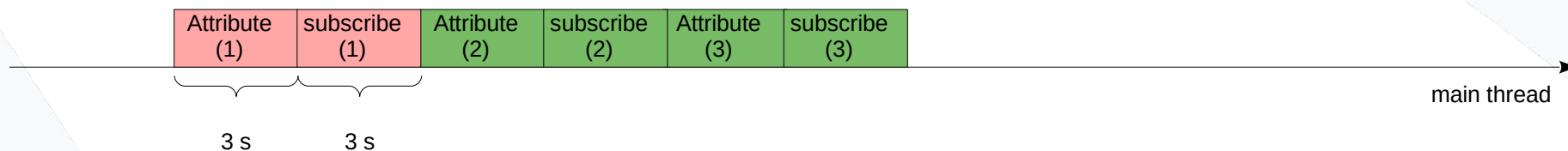
Timeout attribute on application startup

- Attribute w/o events
 - Reads: ATTR_CONF_EVENT callback + polling (due to expiration of cache)
 - Bug in polling (#1278) affects startup time – extra **3 s**
 - Startup time (1 attr): **~9 s**
 - Startup time (3 attrs from 3 different devices): **~27 s**



Timeout attribute on application startup

- Attribute with events
 - Reads: ATTR_CONF_EVENT callback + subscription
 - Startup time (1 attr): **~6 s**
 - Startup time (3 attrs from 3 different devices): **~18 s**



Timeout attribute considerations

- Expiration period of 3 s is not enough – we never reuse the cache
- Polling: one Timeout attribute affects all other attributes from the same device.

Taurus Performance Optimization - Roadmap

Taurus Performance Optimization - Roadmap

- Open TEP21 – “Taurus startup and polling performance optimization”
- Develop benchmark using automatic tests
 - Detailed: on the taurus.Attribute level (~80%)
 - High level: on the Taurus widget level (~20%)
- Fix bugs to minimize startup time with current design. Let's target:
 - Attr w/o events – one polling cycle (3 s)
 - Attr with events – sum of attribute read times (at most one read per attribute)
- **Refactor for the scalability issues**

Scalability issues

- Timeout and Slow attributes does not scale well:
 - Startup time of attributes with events
 - Polling with timeout/slow attributes
- Find solution on the DS side:
 - Improve Taurus polling performance by optimizing DS with `read_attr_hardware()`
 - Configure Tango polling on the DS – immediate reads (source: CACHE)
- DS and Taurus Tango scheme in Green Mode – to be studied

Refactor for the scalability issues – app startup

Case Study:

RF plant – 1 device with 414 attributes (389 with events, 25 w/o events)

PyTango subscription to CHANGE_EVENT of 389 attrs	2 s
PyTango subscription to ATTR_CONF_EVENT of 414 attrs	1.2 s
PyTango read of 414 attrs	1 s
Taurus startup (Attribute() + addListener() + getValueObj()) of 414 attrs	6 s
Taurus Labels startup of 414 attrs	10 s
Taurus Form startup of 414 attrs	16 s

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_asynch()` 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

Refactor for the scalability issues – app startup

- Natively integrate *DelayedSubscriber* feature
 - Start with all attributes in polling mode.
 - Later subscribe gradually to events and disable polling.

```
from taurus.qt.qtcore.util.emitter import DelayedSubscriber

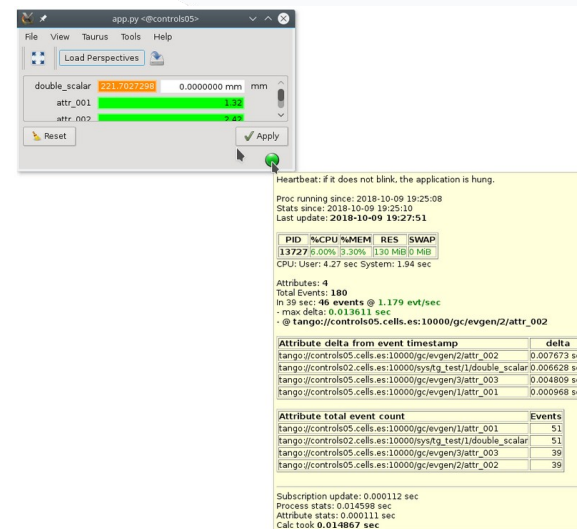
taurus.Factory('tango').set_tango_subscribe_enabled(False)
app = taurus.qt.qtgui.application.TaurusApplication()

# create and initialize your GUI

subscriber = DelayedSubscriber('tango', parent=app, sleep=10e3)
sys.exit(app.exec_())
```

Other improvements

- To be removed: automatic Taurus polling activation on API_EventTimeout
- Issues in Tango when unsubscribing from events in `__del__()` of objects with circular references.
- Memory leak in some GUIs – to be investigated
- Eliminate deadlocks in TaurusPollingTimer
- APM (Application Performance Monitoring) and more feedback to the user



The image shows a screenshot of a Taurus GUI window titled 'app.py -@controls05'. The GUI displays three attributes: 'double_scalar' with a value of 0.000000 mm, 'attr_001' with a value of 1.32, and 'attr_002' with a value of 0.25. There are 'Reset' and 'Apply' buttons at the bottom of the GUI.

Below the GUI screenshot is a monitoring window with the following text:

Heartbeat: if it does not blink, the application is hung.
 Proc running since: 2018-10-09 19:25:08
 Stats since: 2018-10-09 19:25:10
 Last update: 2018-10-09 19:27:51

PID	%CPU	%MEM	RES	SWAP
113227	0.00%	0.30%	110 MB	0 MB

CPU: User: 4.27 sec System: 1.94 sec

Attributes: 4
 Total Events: 180
 In 39 sec: 46 events @ 1.179 evt/sec
 - max delta: 0.013911 sec
 - @ tango://controls05.cells.es:10000/gc/evgen2/attr_002

Attribute	delta
tango://controls05.cells.es:10000/gc/evgen2/attr_002	0.007673 sec
tango://controls02.cells.es:10000/sys/tg_test/1/double_scalar	0.006628 sec
tango://controls05.cells.es:10000/gc/evgen/3/attr_003	0.004809 sec
tango://controls05.cells.es:10000/gc/evgen/1/attr_001	0.000968 sec

Attribute	total event count	Events
tango://controls05.cells.es:10000/gc/evgen/1/attr_001	51	51
tango://controls02.cells.es:10000/sys/tg_test/1/double_scalar	51	51
tango://controls05.cells.es:10000/gc/evgen/3/attr_003	39	39
tango://controls05.cells.es:10000/gc/evgen/2/attr_002	39	39

Subscription update: 0.000112 sec
 Process stats: 0.014598 sec
 Attribute stats: 0.000111 sec
 Calc took 0.014867 sec

Conclusions

- Taurus has a rich set of GUI features
- Taurus core has a simple and intuitive API
- Some obvious issues were identified
 - Solutions will probably require changes in just a few lines of code
- Comprehensive benchmark tests:
 - Crucial in the performance optimization process
 - Helps avoid regressions
- ALBA already started the performance optimization project
 - 4 engineers will be focused on it in the following 3 months
 - You are more than welcome to join it!

Thank You