Taurus 4

Taurus4: TEP3
Taurus Core→ Tango Independent

- ***TEP3 objective:***
  - Make Tango dependency optional for Taurus
  - Generic Taurus Core accepting any scheme without forcing PyTango:
    - In order to open the way, in the future, to the inclusion of new schemes a part from Tango: Eval, Epics, MS_Environment, SPEC...

Taurus4: TEP14 Core refactoring Quantities and Configuration

- Highly related with TEP3 (refactoring of taurus-core)
  - Have a cleaner and **scheme-agnostic API**
  - Allow future creation of new schemes
- **Merge Attribute Configuration into Attribute**
  - AttrConfigValue and the AttrConfig objects disappear
- **Add Quantities (Pint)**: Handle unit conversion, etc.
- New **tests** and usage of **TDD** in some cases (more than 300 tests)
- Following TEP12, new enums introduced in TEP14 have been created as **Python enums** (e.g. TaurusDevState: Ready, NotReady, Undefined)
- Usage of **fragments** in URI names (model # fragment):
  - scheme://authority/this/is/model/identifier#fragment
    - eg: tango://controls01:10000/a/b/c/attr#label
    - eg: tango:a/b/c/attr#label

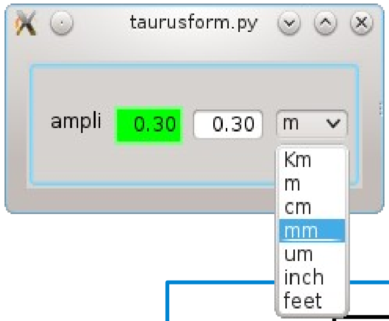
TEP14: core refactoring

Use Quantities for **int** and **float** types:

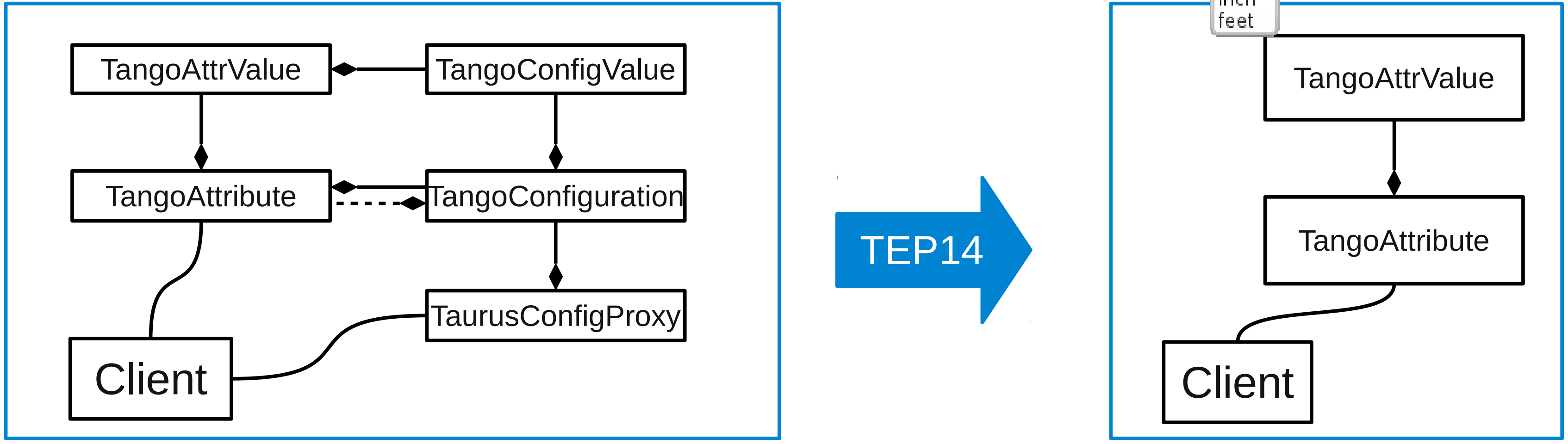
	0D	1D	ND
str	str	seq<str>	seq<seq<...<str>>>
bool	bool numpy.bool	ndarray (dtype=bool)	ndarray (dtype=bool)
int / float	pint.Quantity	pint.Quantity	pint.Quantity

http://pint.readthedocs.org

```
>>> a = taurus.Device('sys/tg_test/1').ampl
>>> a
<Quantity(0.3, 'meter')>
>>> print(a)
0.3 meter
>>> print(a.to('mm'))
300.0 millimeter
```



Merge attribute and configuration objects



Taurus4: TEP3- TEP14 details

- Deprecate and/or rename methods (decorator):

  - **@tep14_deprecation(alt='getFullName')**
  def getDisplayValue() → (from taurus.core.tango.tangodatabase)

  - Use **getDeviceProxy** instead of getHWObj

- **Move** from Taurus to Tango: addListener, eventReceived...

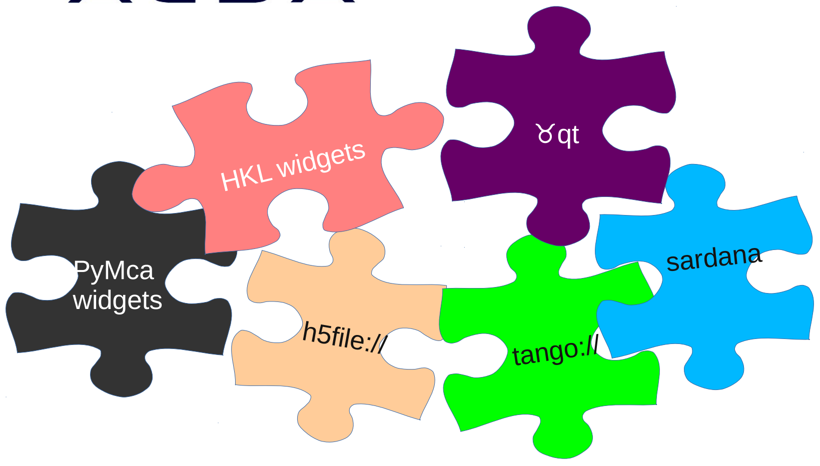
- Use **properties**
  - Property **description** instead of getDescription
  - Property **state** instead of getSWState

- **TaurusDevState** enum instead of TaurusSWDevState

Taurus4: TEP13 Plugins

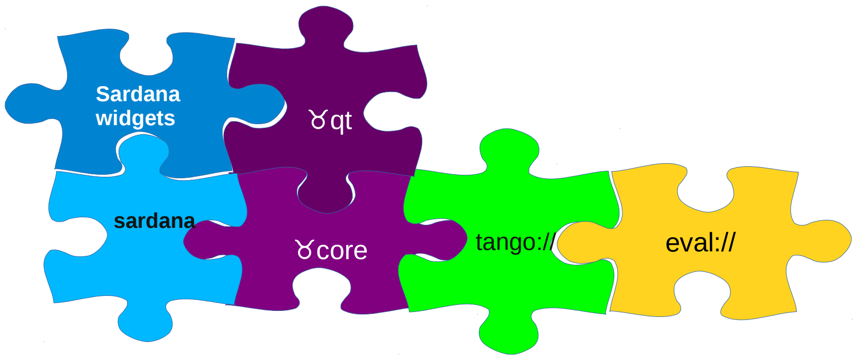
- **Why:**
  - Plugins: extensible system
  - Modular, Customizable and Flexible
  - Ease the maintenance at the long term
  - Ease the installation by reducing dependencies
  - Ease the collaborations and code contributions

Taurus4: TEP13 Plugins



...

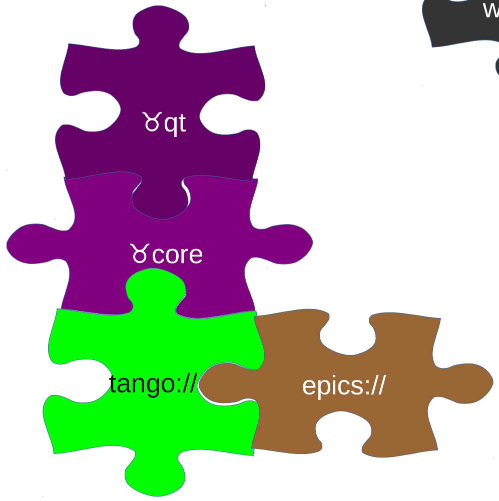
http://sf.net/p/tauruslib/wiki/TEP13



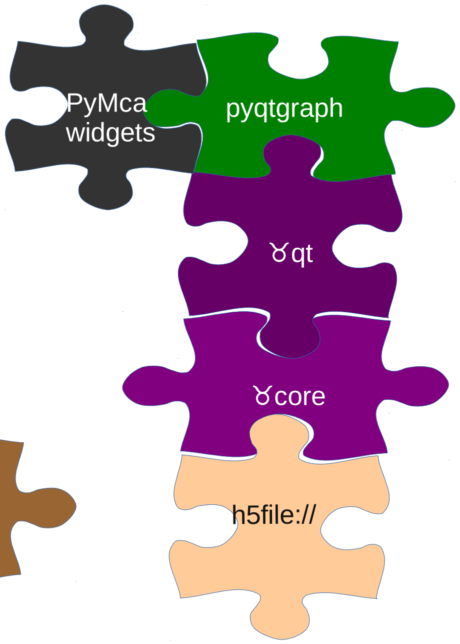
***Example**: Taurus+Sardana as we use it now in ALBA*

Plugins will make Taurus...

- Light: most dependencies optional
- Extendable for user specific need
- Taurus usable as a library for data analysis GUIs



***Example**: Controlling a mixed Tango+EPICS environment*

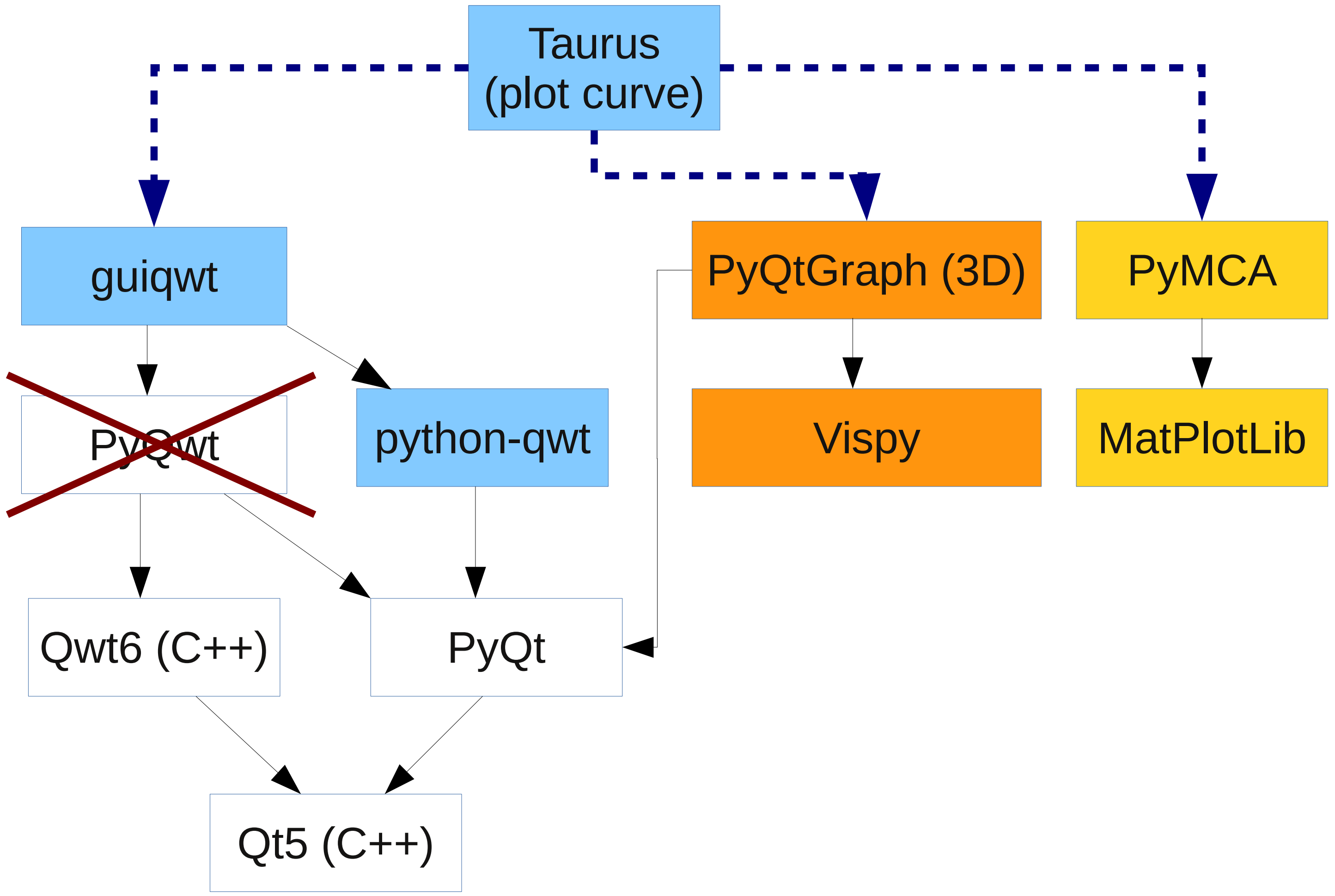


***Example**:Taurus for Data Analysis (no control system)*

Taurus4: TEP13 Plugins

- **Some plugin system options:**
  - stevedore:
    - based on setup tools entry points
    - systematic approach: discovery/enabling/importing/integrating...
    - well documented: *http://docs.openstack.org/developer/stevedore/*
  - yapsy (yet another plugin system)
- **Entry points** and some of its present/future plugins:
  - **Schemes**: Tango, eval, h5file, MS_env...
  - **Widgets**: taurusplot, taurustrend, taurusform...
  - **Codecs**: Json, pickle, zip...
  - **External**: qt, argparse, pint, enum, unittest
  - Icons
  - ...

Taurus4: Graphic libraries



Taurus4: Graphic libraries

- Thus, which way to follow? **Options:**
  - **guiqwt** with **python-qwt**: python library substituting PyQwt bindings (maintained by only one individual). Only supports 2D.
    - **Pros:** Comfort → low adaptation work
    - **Cons:** Low future projection → Only supports 2D
  - **PyQtGraph** depending on Vispy (collaboration): will support 3D
    - **Pros:** Future projection → Supports 3D, performance
    - **Cons:** code adaptation
  - **PyMCA** depending on **MatPlotLib**
    - **Pros:** maintenance benefits → working close to the creator of PyMCA
    - **Cons:** Performance concerns for dynamic plots, code adaptation

Taurus4: Questions & Comments

- Taurus4 will introduce backwards incompatibilities
- Testing Taurus4 in your institutions will be very valuable
- Taurus4 will allow the integration of new schemes
- Taurus4 will ease the collaborations thanks to opening the door to plugin development

Integrations

- Integrations: TEP3, TEP14...
- Who and when will work on plugins
- Decision about graphic library: PyQtGraphy, vispy...