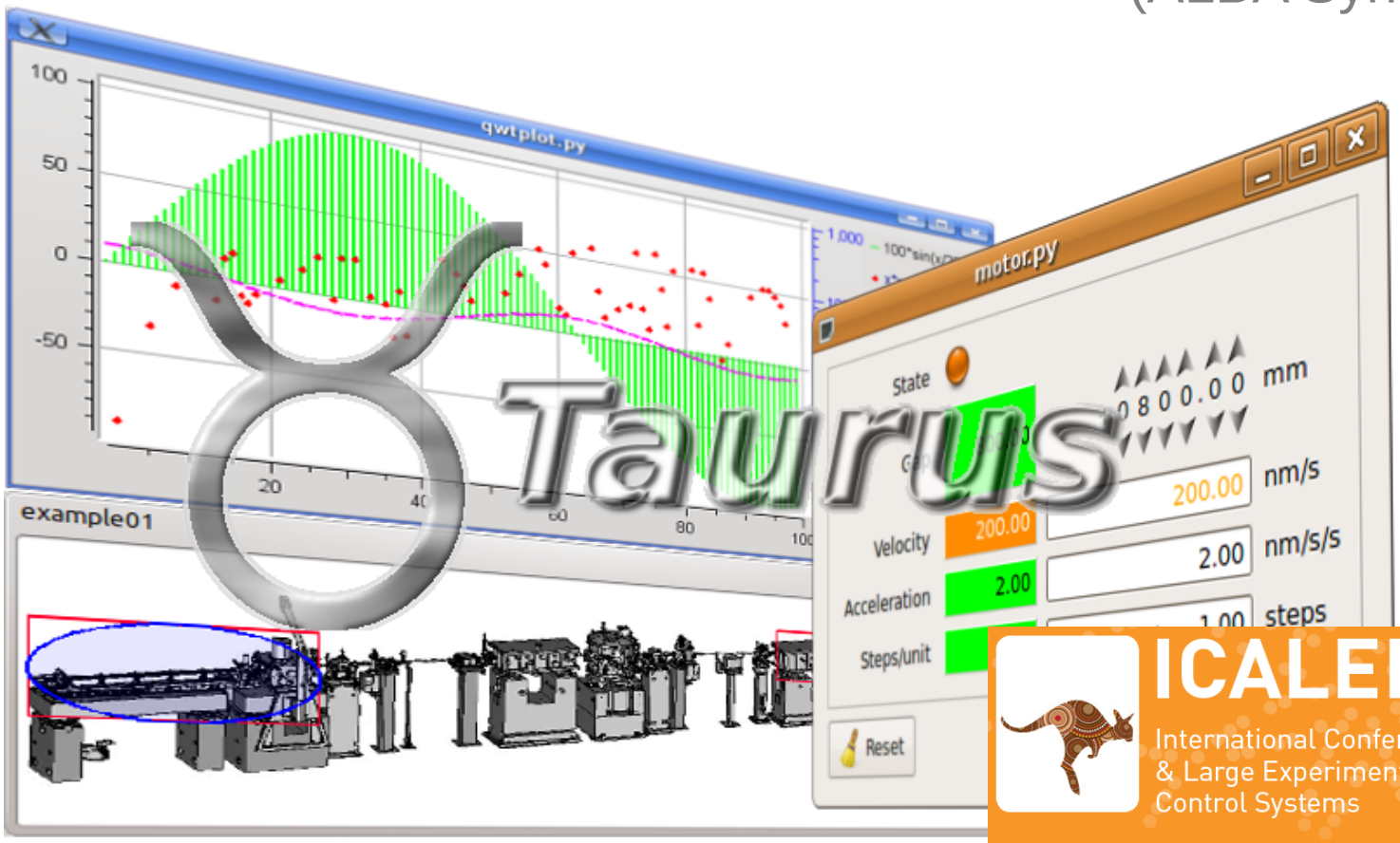
Effortless creation of GUIs with Taurus

by Carlos Pascual-Izarra
(ALBA Synchrotron, Spain)



Also by:

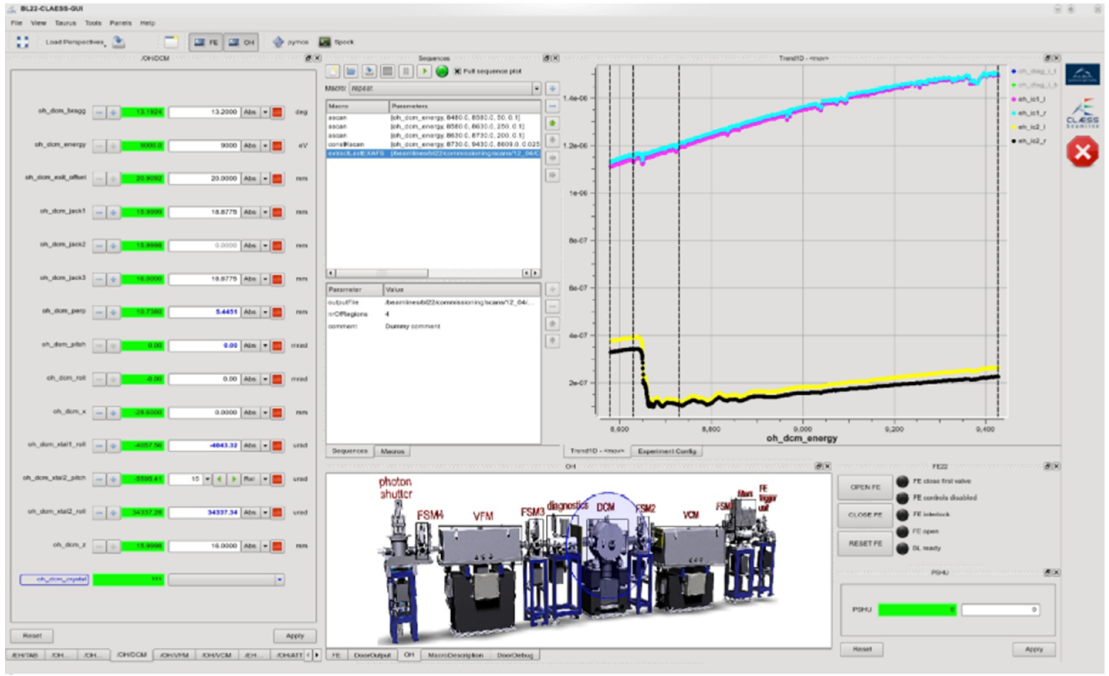
- Guifré Cuní
- Carlos Falcón-Torres
- David Fernández-Carreiras
- Zbigniew Reszela
- Marc Rosanes
- Tiago Coutinho (ESRF)

ICALEPCS 2015
International Conference on Accelerator & Large Experimental Physics Control Systems
17-23 OCTOBER 2015 MCEC MELBOURNE

Contents

- **Introduction**
  - What is Taurus
  - Structure of Taurus
- **Fast GUI creation**
  - Model-View-Controller approach
  - TaurusGUI vs Qt Designer
  - TaurusGUI demo (videos)
- **Is Taurus for You?**

Taurus is...



*"**Taurus** is a **python** framework for control and data acquisition **CLIs** and **GUIs** in scientific/industrial environments. It supports multiple control systems or data sources: **Tango**, **EPICS**, **Spec**... New control system libraries can be integrated through plugins."*

- Widely used
- Production-ready
- Well supported
- Actively developed
- Free/Open Source
- Community-driven
- Modular
- Multi-platform
- Based on Python and Qt
- Easy to install

Structure of Taurus

TaurusGUIs

External Hardware and data sources





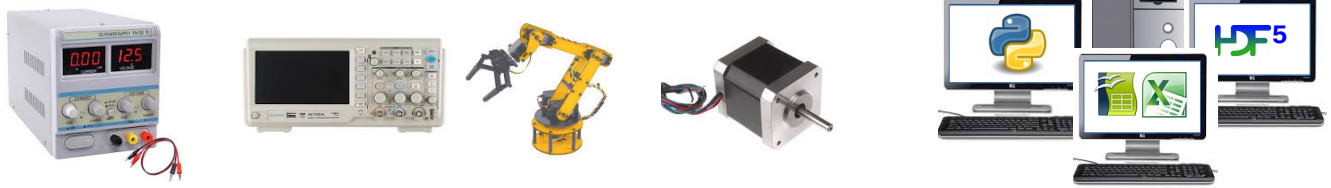
Structure of Taurus

TaurusGUIs

TaurusGUIs

Taurus Qt Widgets

External Hardware and data sources



Structure of Taurus

TaurusGUIs

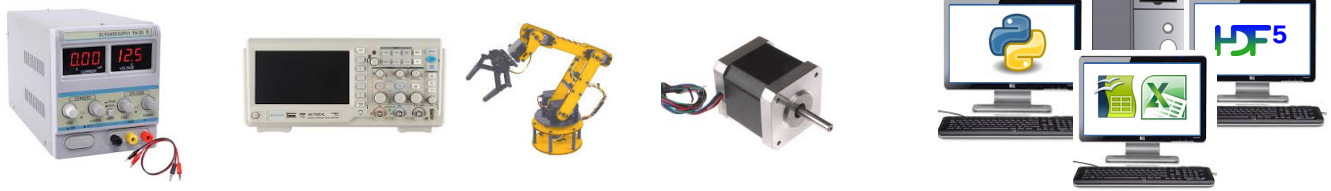
TaurusGUIs

Taurus Qt Widgets

Taurus Core

Taurus Core

External Hardware and data sources



Structure of Taurus

TaurusGUIs

TaurusGUIs

Taurus Qt Widgets

Taurus Core

Model Objects

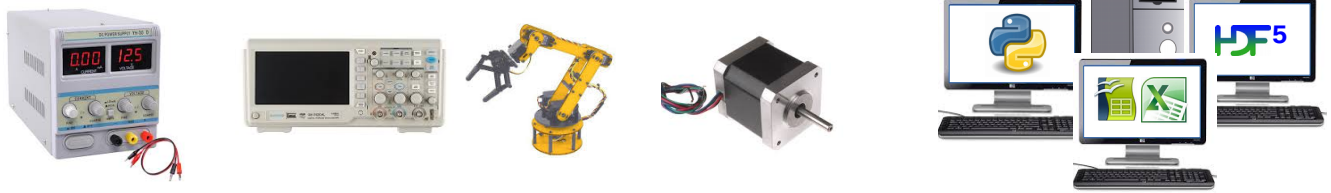
model model model model model model model

Taurus Core

Schemes

TANGO eval EPICS ssheet h5file

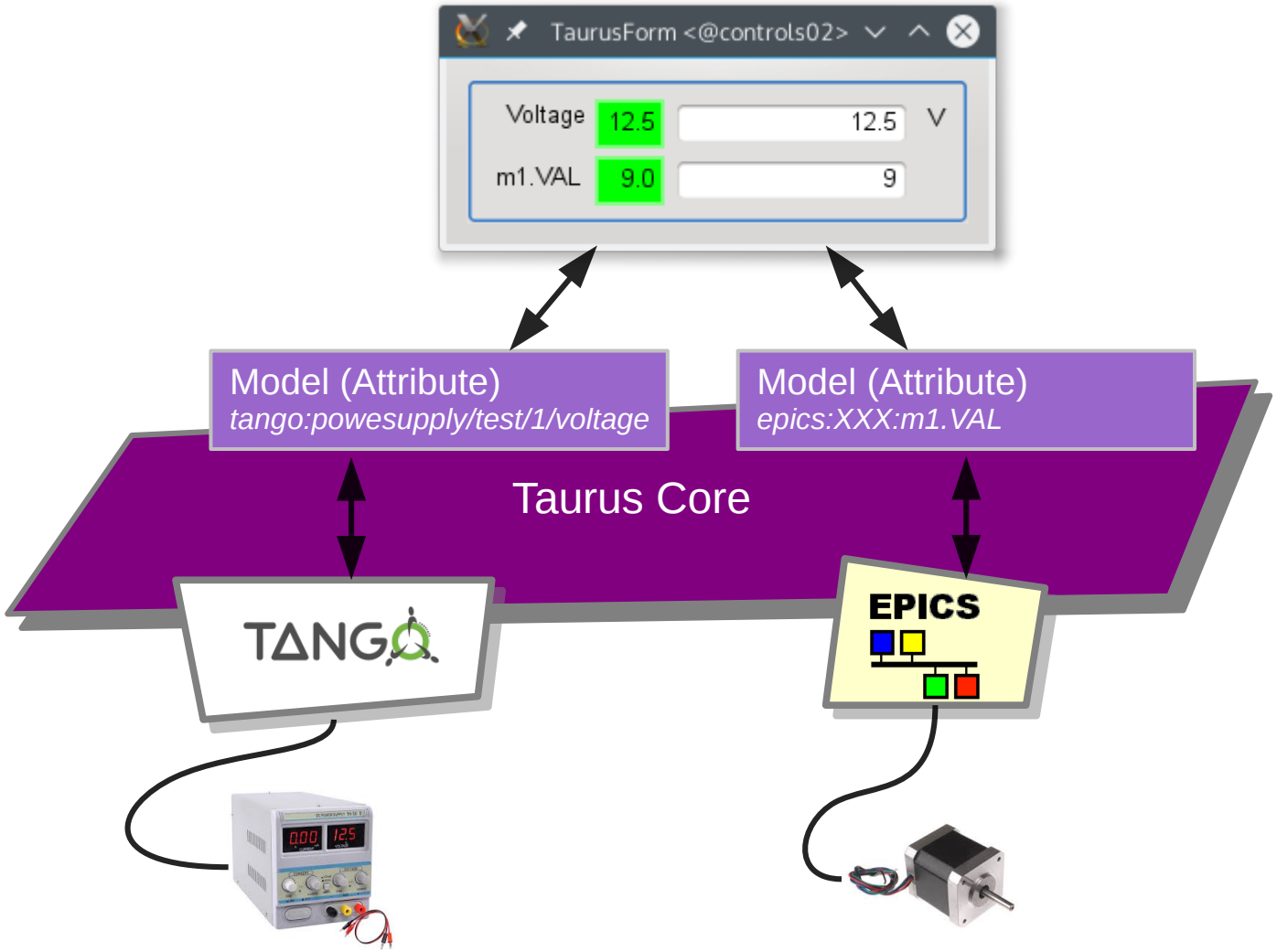
External Hardware and data sources



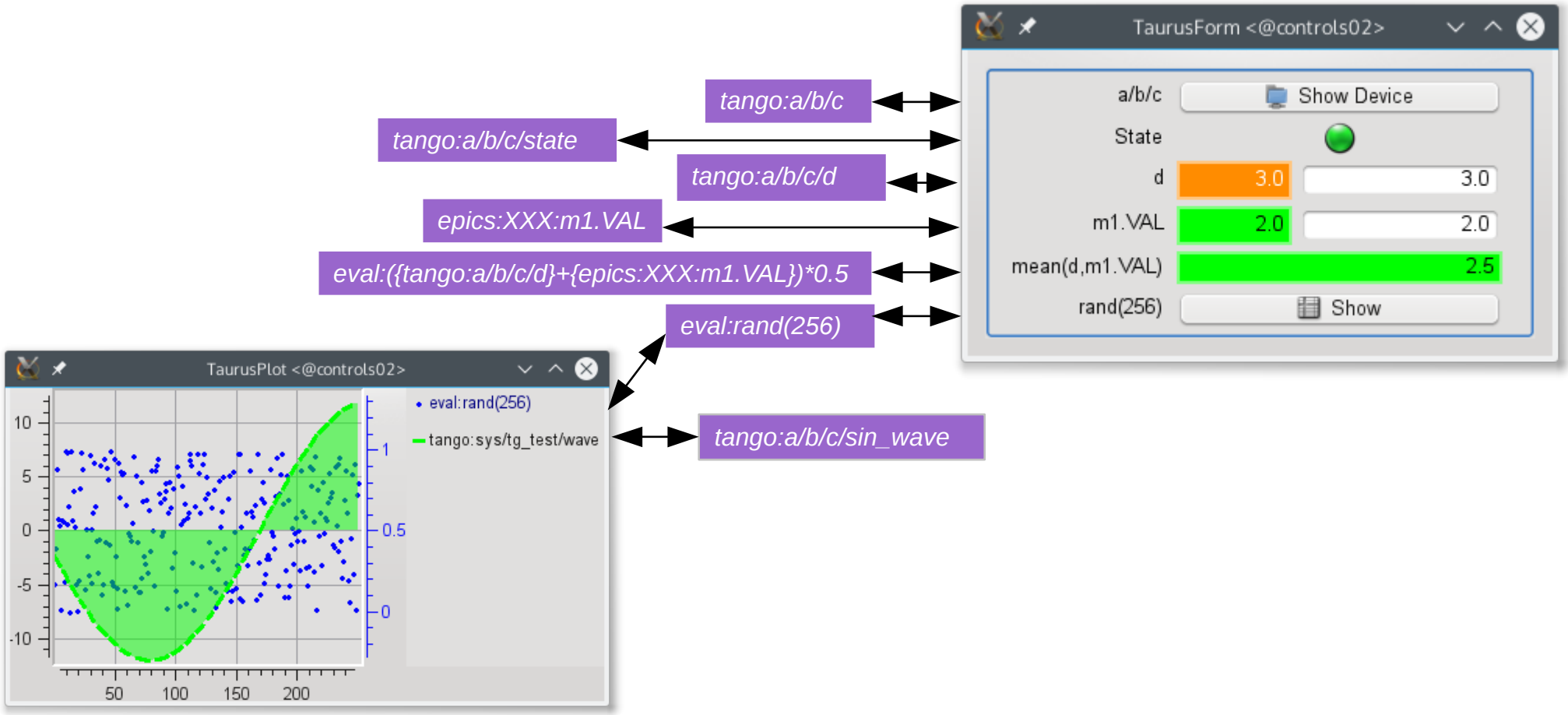
Model-View-controller

Taurus Qt Widgets (View & Controller)

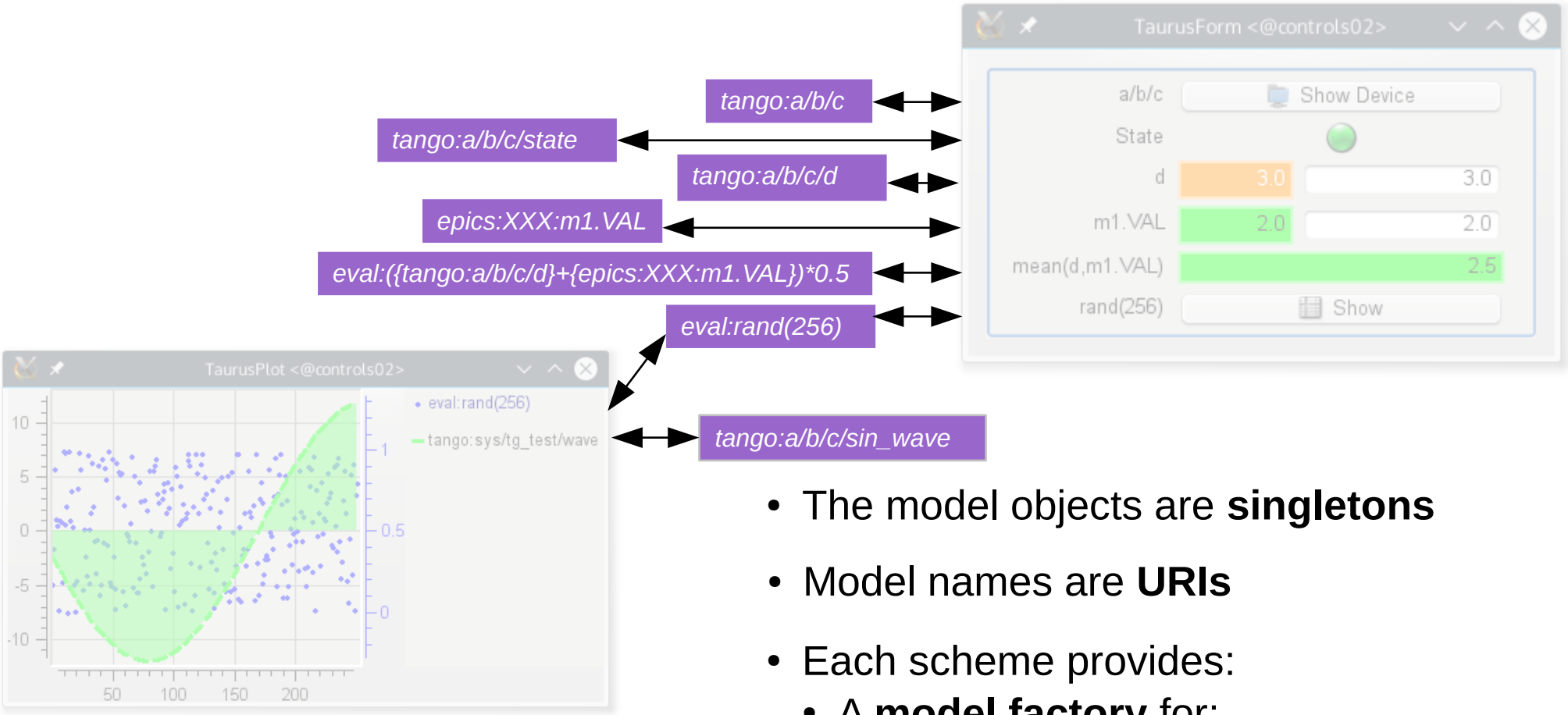
Taurus Core (Model)



Model-View-controller

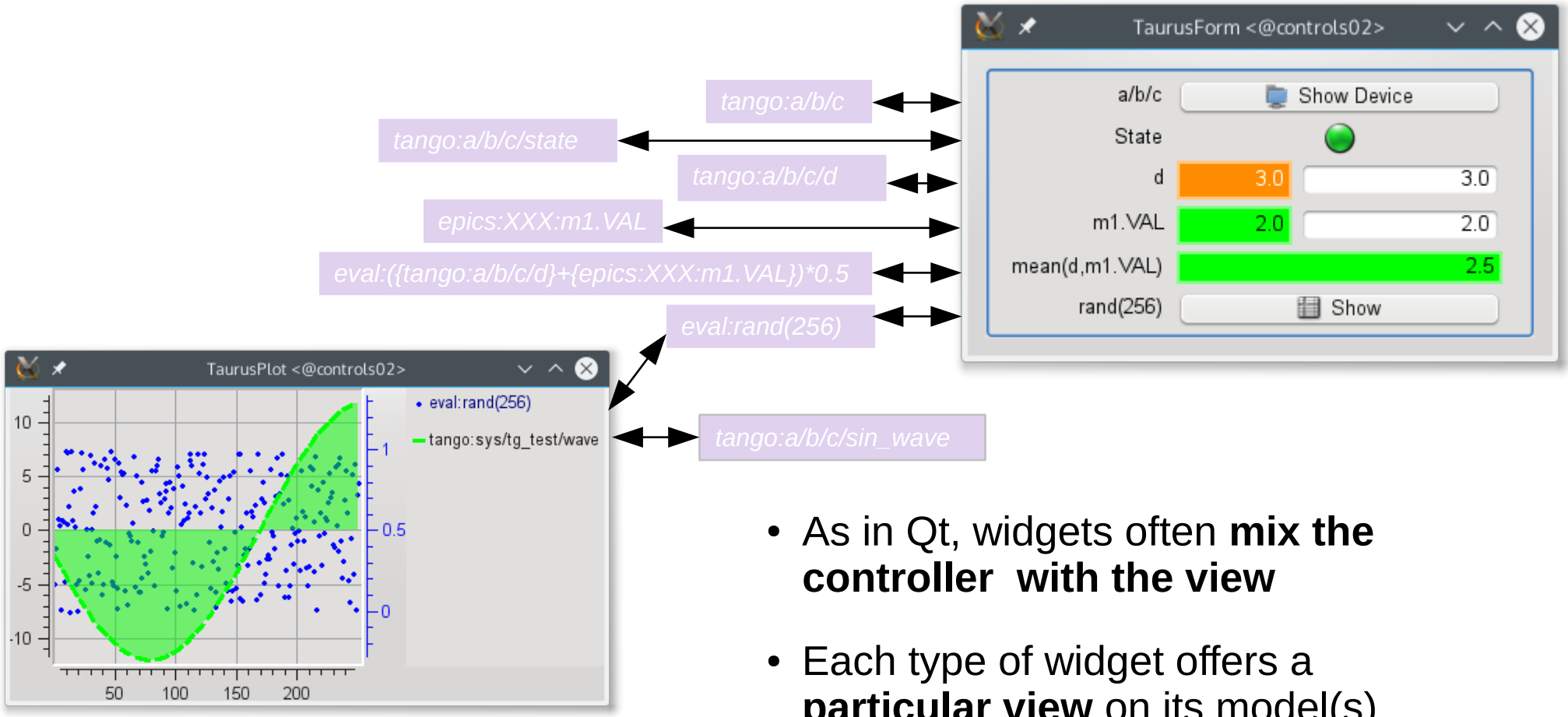


Model-View-controller



- The model objects are **singletons**
- Model names are **URIs**
- Each scheme provides:
  - A **model factory** for:
    - Authority
    - Device
    - Attribute
  - **Model name validators**

Model-View-controller



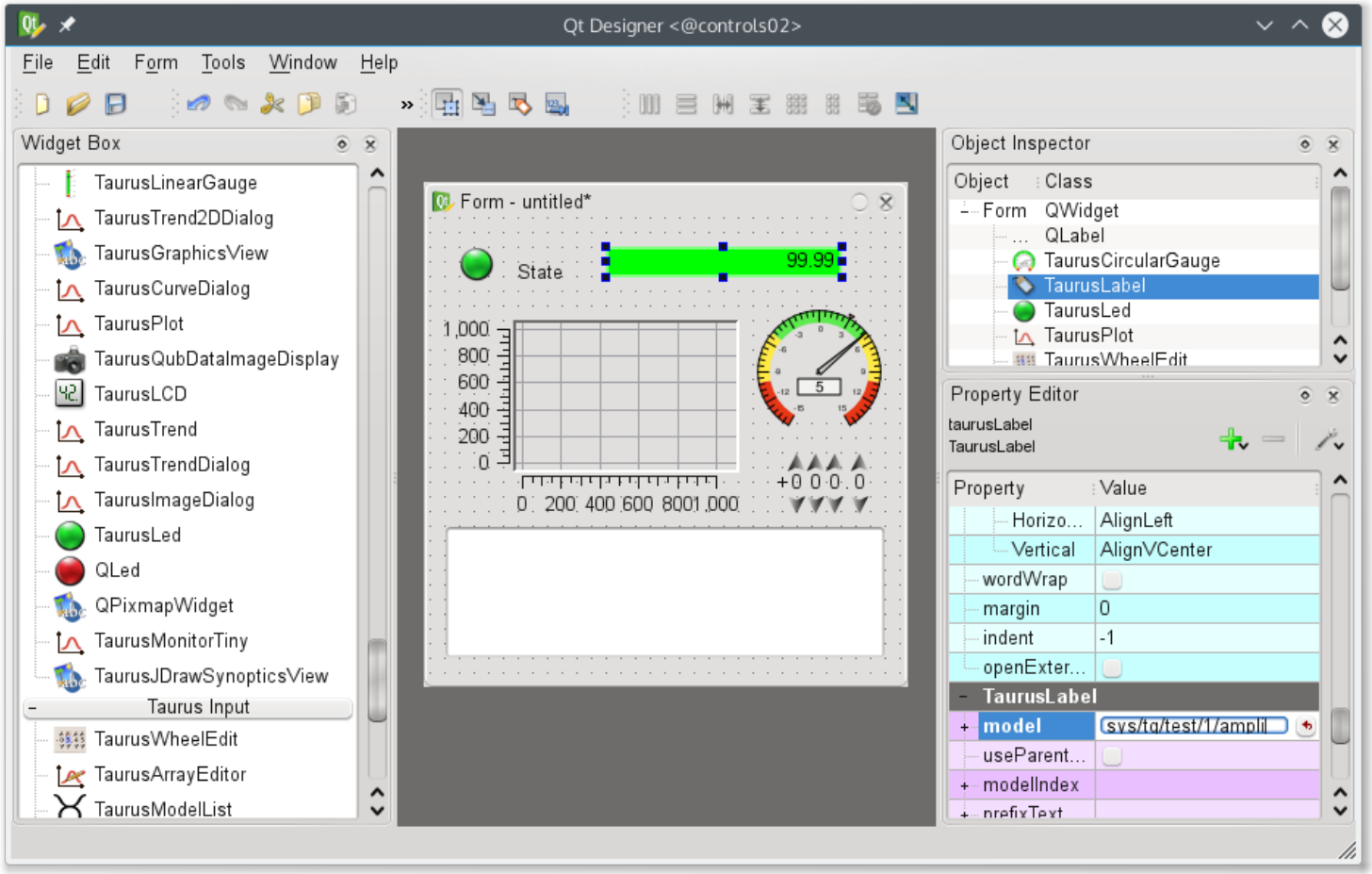
- As in Qt, widgets often **mix the controller with the view**
- Each type of widget offers a **particular view** on its model(s)
- All functionality is enabled by just **attaching** the widget to a model (i.e. providing its URI)

TaurusDesigner (Qt Designer)

GUIs can be created using the Qt Designer (*)

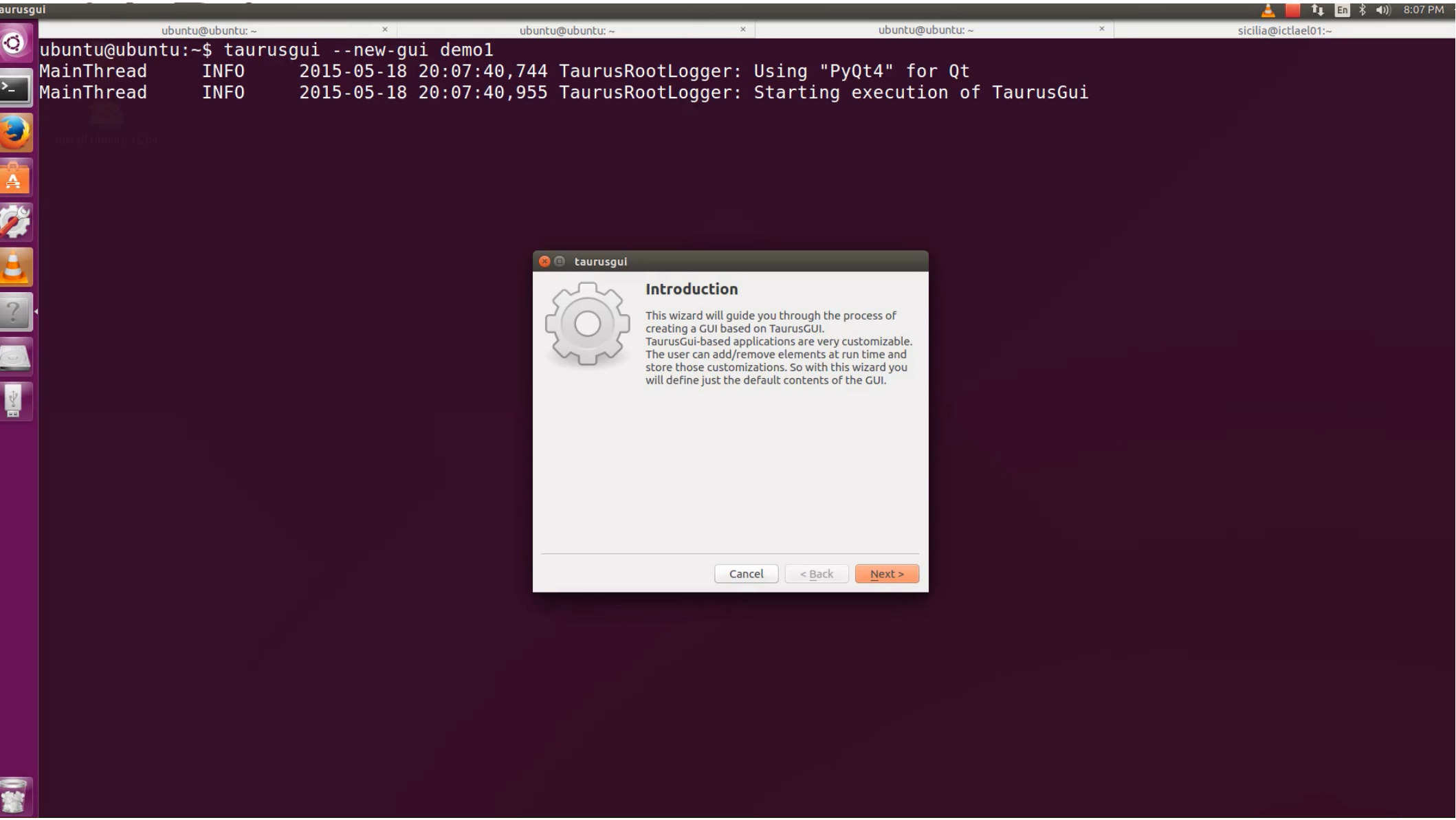
The Taurus widgets are available in the catalogue

The model name can be set as a Qt property



(*) But for **really fast** GUI development, we use the **TaurusGUI** framework

TaurusGUI (new-gui.mp4)



Creating a TaurusGUI from scratch using *taurusgui --new-gui*

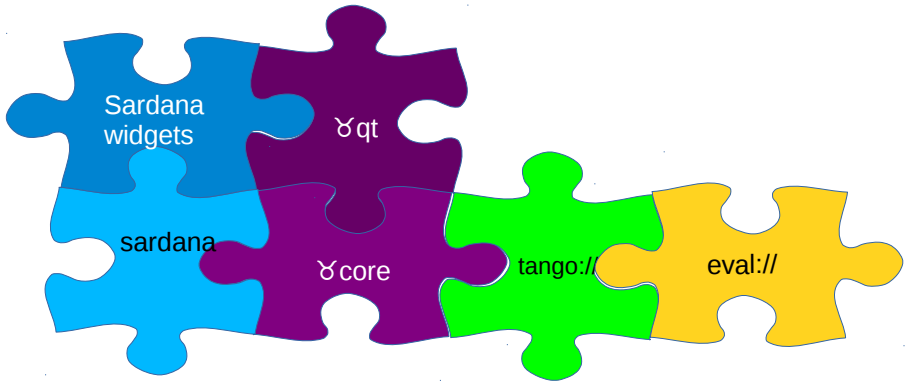
TaurusGUI (demo_taurus.mp4)

Demonstrating features of a TaurusGUI (extending, configuring, drag&drop, perspectives,...)

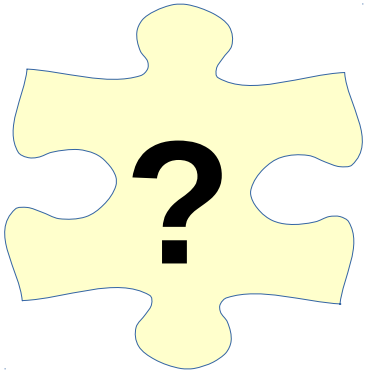
TaurusGUI (summary)

- An empty GUI can be created with a wizard (<1min) and then populated with panels (<1h) **without writing code**
- TaurusGUIs are fully modifiable and customizable **at run time**
- Specialised widgets can be created with standard Qt Designer (but most applications only require standard widgets)

Is Taurus for you?

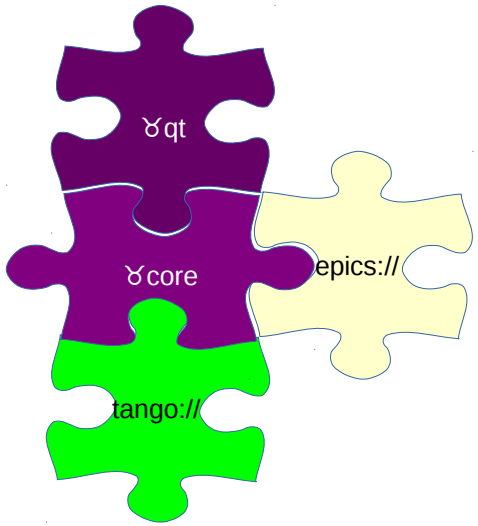


***Example**: Taurus+Sardana as we use it now in ALBA*

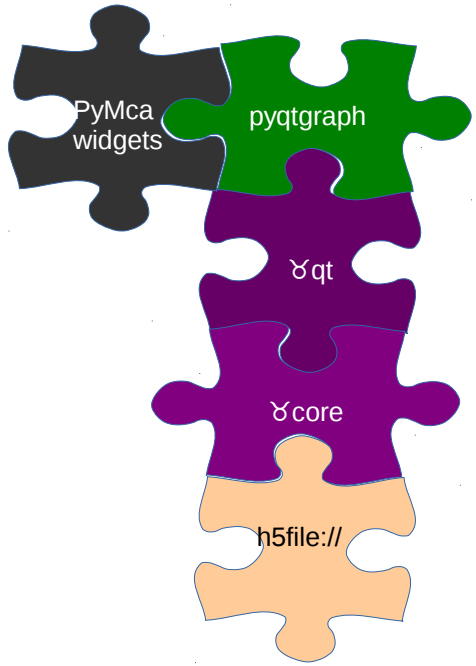


Suggested schemes:
Spec, LIMA, MADOCA2, Archiving, SQL, Icat, CDMA, ASCII tables

Suggested Views:
web, gtk, PyQtgraph, PyMca



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



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

Home Page

http://www.taurus-scada.org

Access to:

- Documentation
- Releases
- Git repository
- Mailing lists
- Bugs & Requests tracker
- Enhancement Proposals
- ...