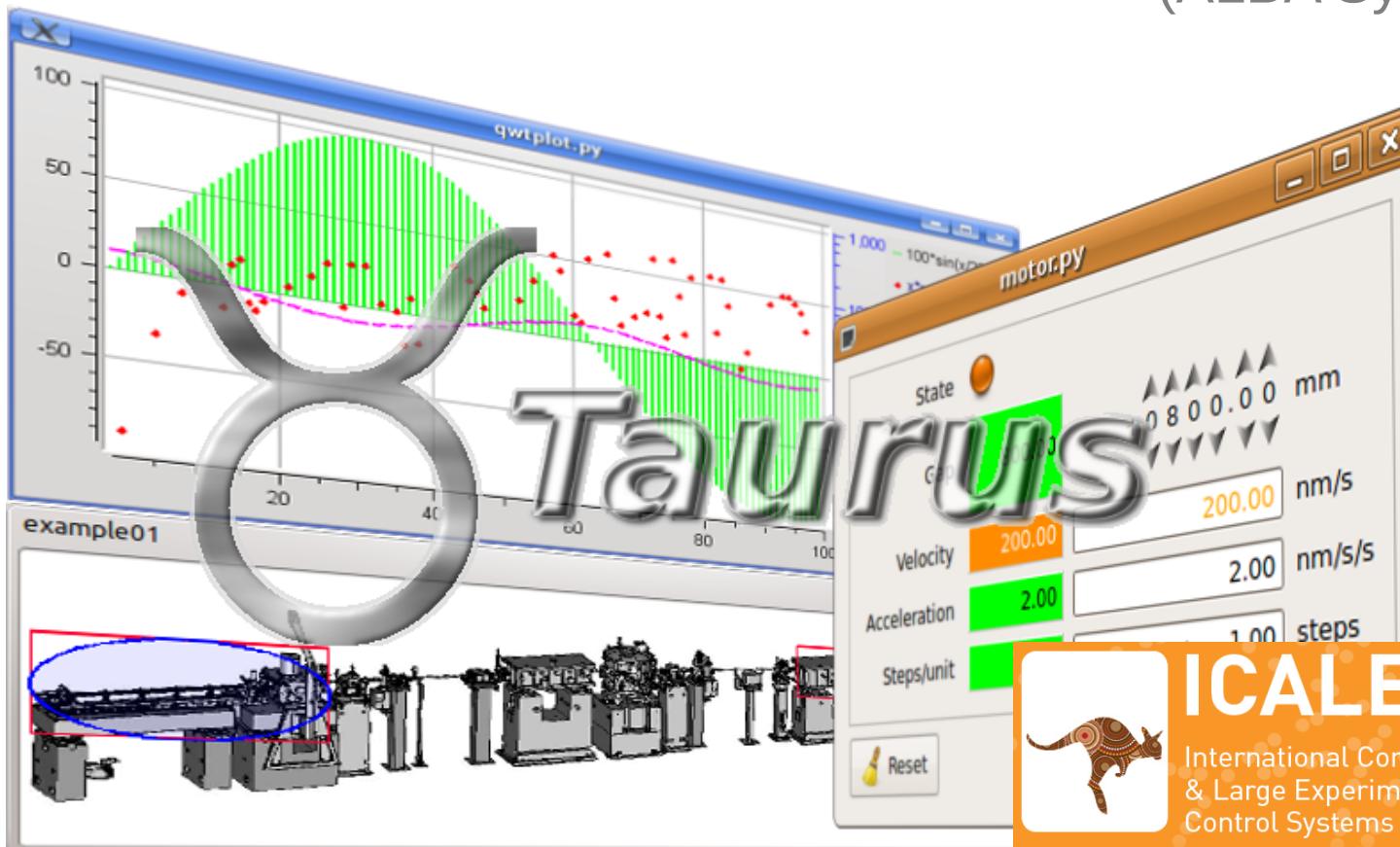


Effortless creation of GUIs with Taurus

by Carlos Pascual-Izarra
(ALBA Synchrotron, Spain)



The image displays three distinct applications of the Taurus framework:

- Plot Window:** A graph titled "qwtplot.py" showing a green shaded area representing a signal envelope and red dots representing discrete data points. The x-axis ranges from 0 to 100, and the y-axis ranges from -100 to 100.
- Motor Control Panel:** A window titled "motor.py" with the Taurus logo. It features several sliders and input fields for controlling a motor's state, velocity (200.00 nm/s), acceleration (2.00 nm/s/s), and steps/unit (1.00 steps).
- 3D CAD Model:** A detailed 3D rendering of a synchrotron storage ring. Specific components are highlighted with colored boxes: a blue box around a central dipole magnet, a red box around a quadrupole magnet, and a green box around a sextupole magnet.

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

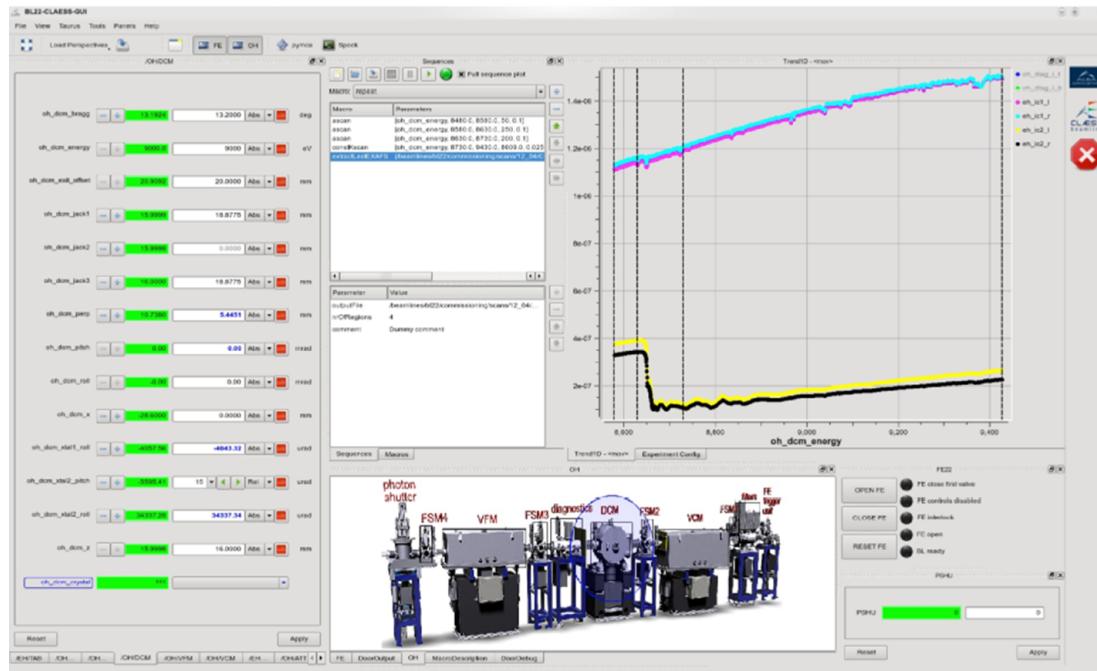


Contents

- **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



TaurusGUIs

Structure of Taurus

TaurusGUIs

**External
Hardware and
data sources**



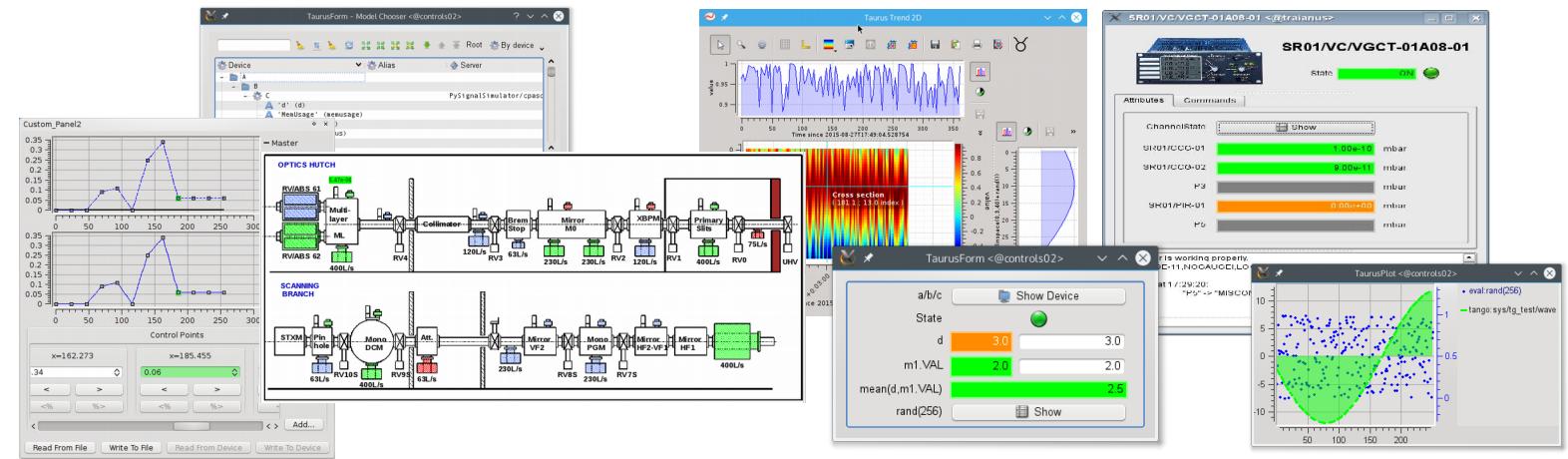


Structure of Taurus

TaurusGUIs

TaurusGUIs

Taurus Qt Widgets



External
Hardware and
data sources



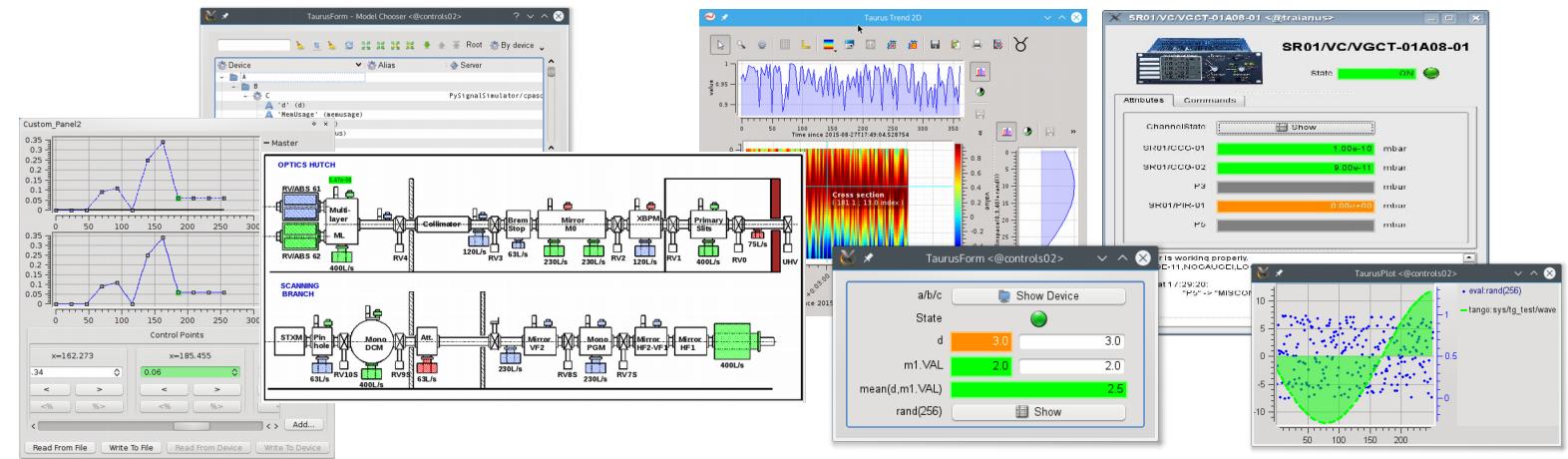


Structure of Taurus

TaurusGUIs

TaurusGUIs

Taurus Qt Widgets



Taurus Core

Taurus Core

External
Hardware and
data sources



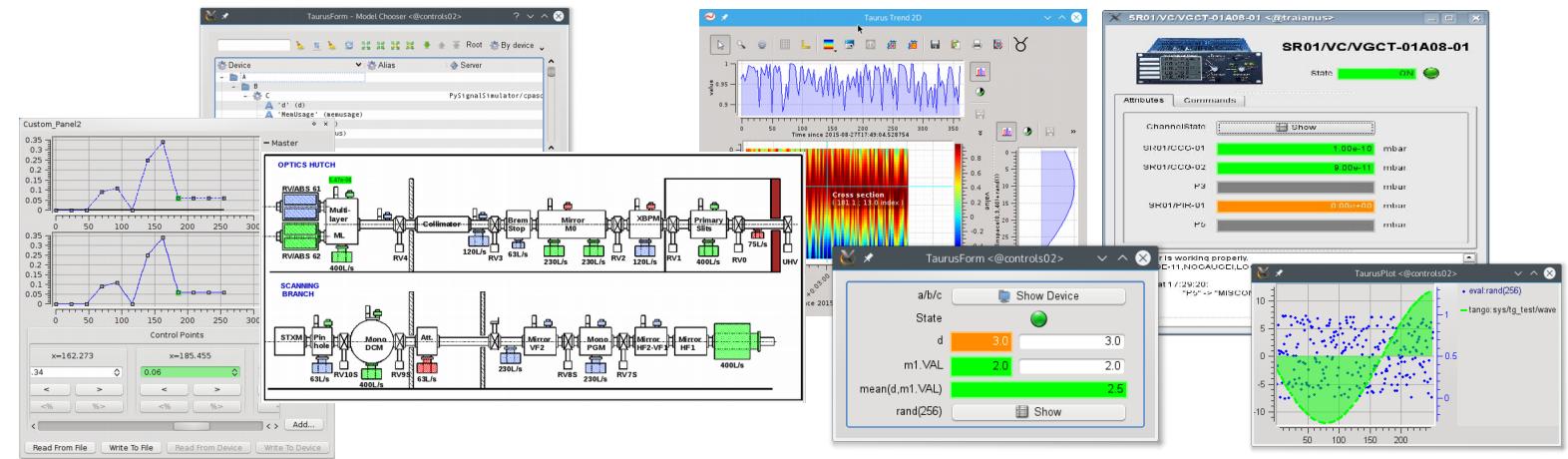


Structure of Taurus

TaurusGUIs

TaurusGUIs

Taurus Qt Widgets



Model Objects

model

model

model

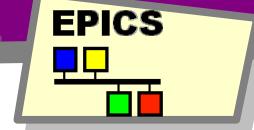
model

model

model

Taurus Core

Schemes



Taurus Core

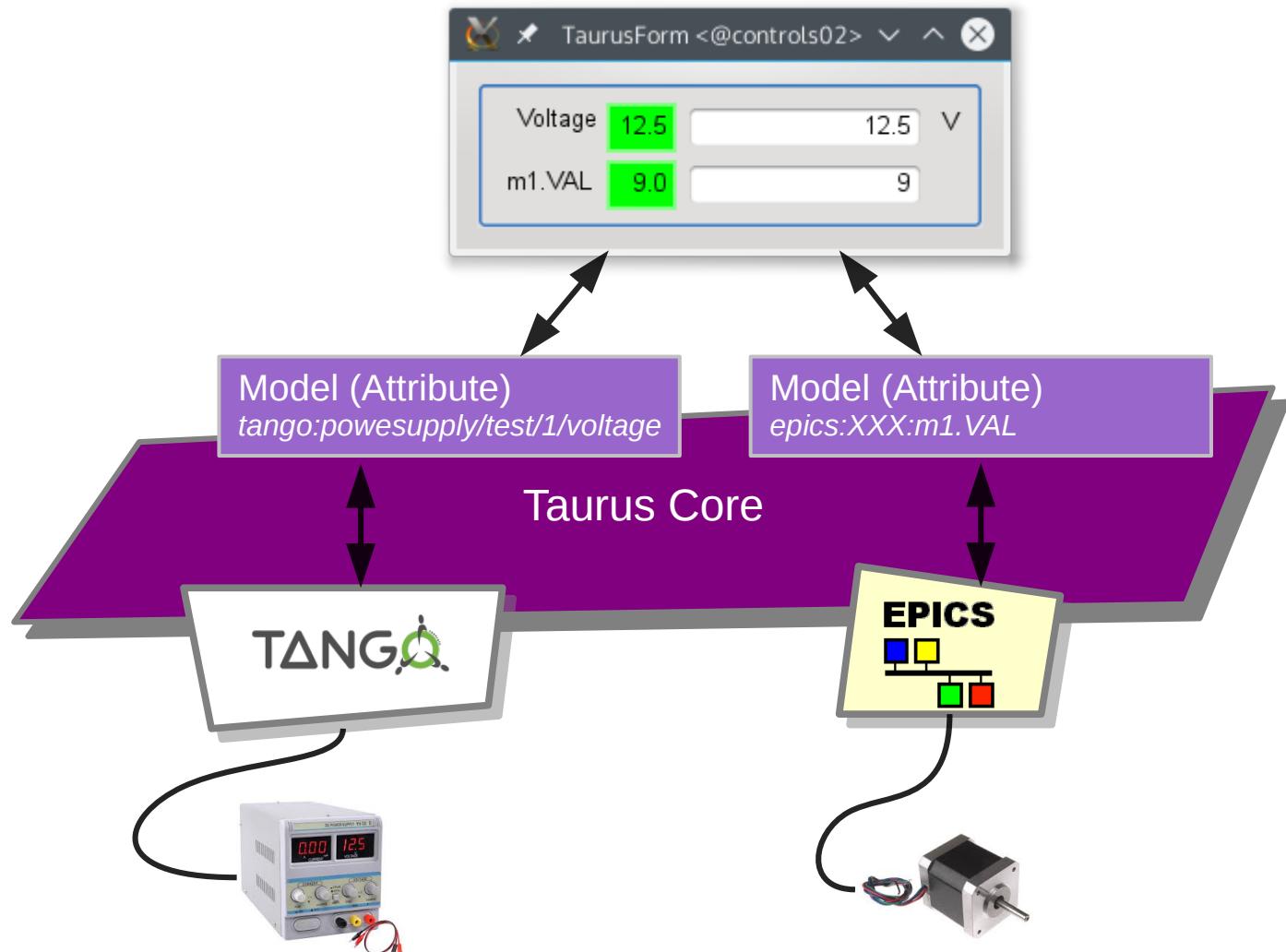
External Hardware and data sources





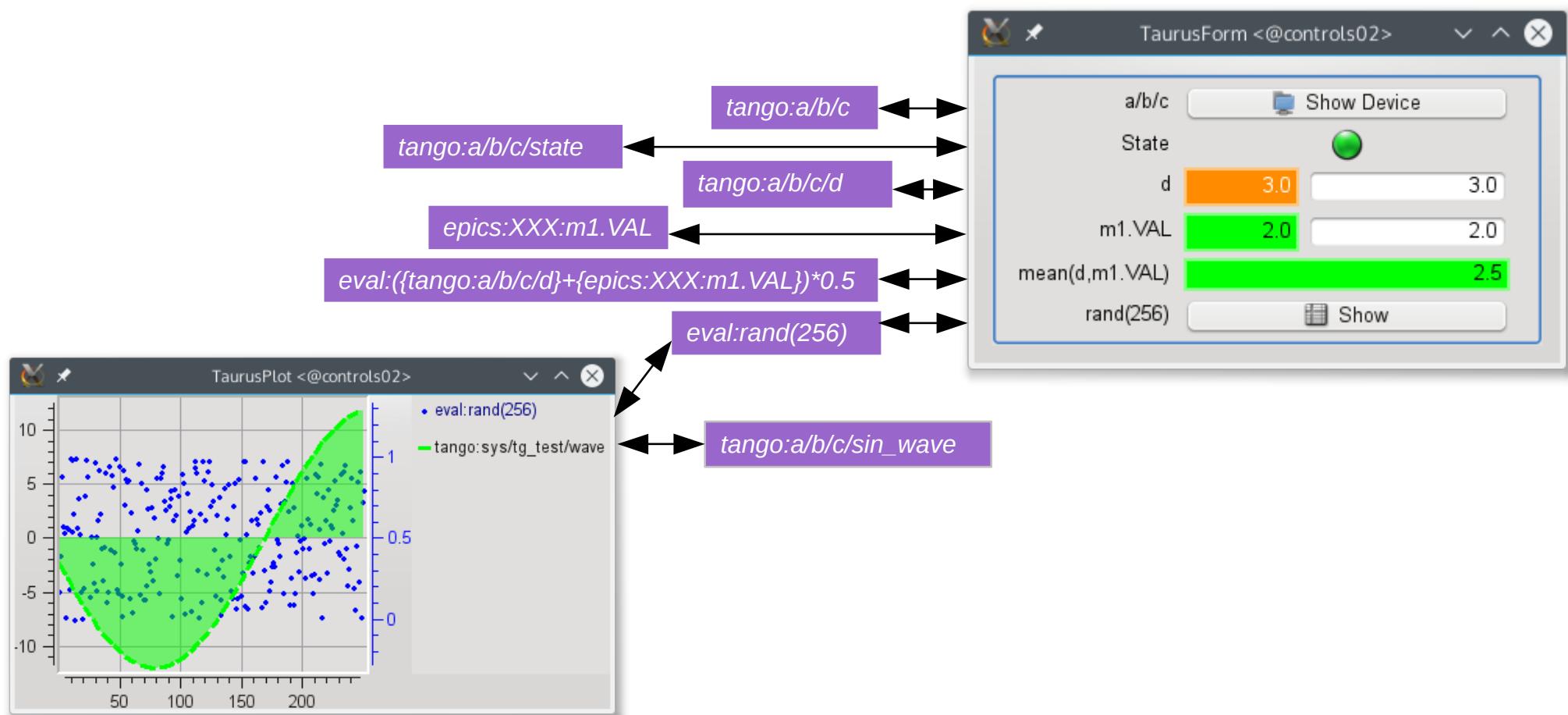
Model-View-controller

**Taurus Qt Widgets
(View & Controller)**

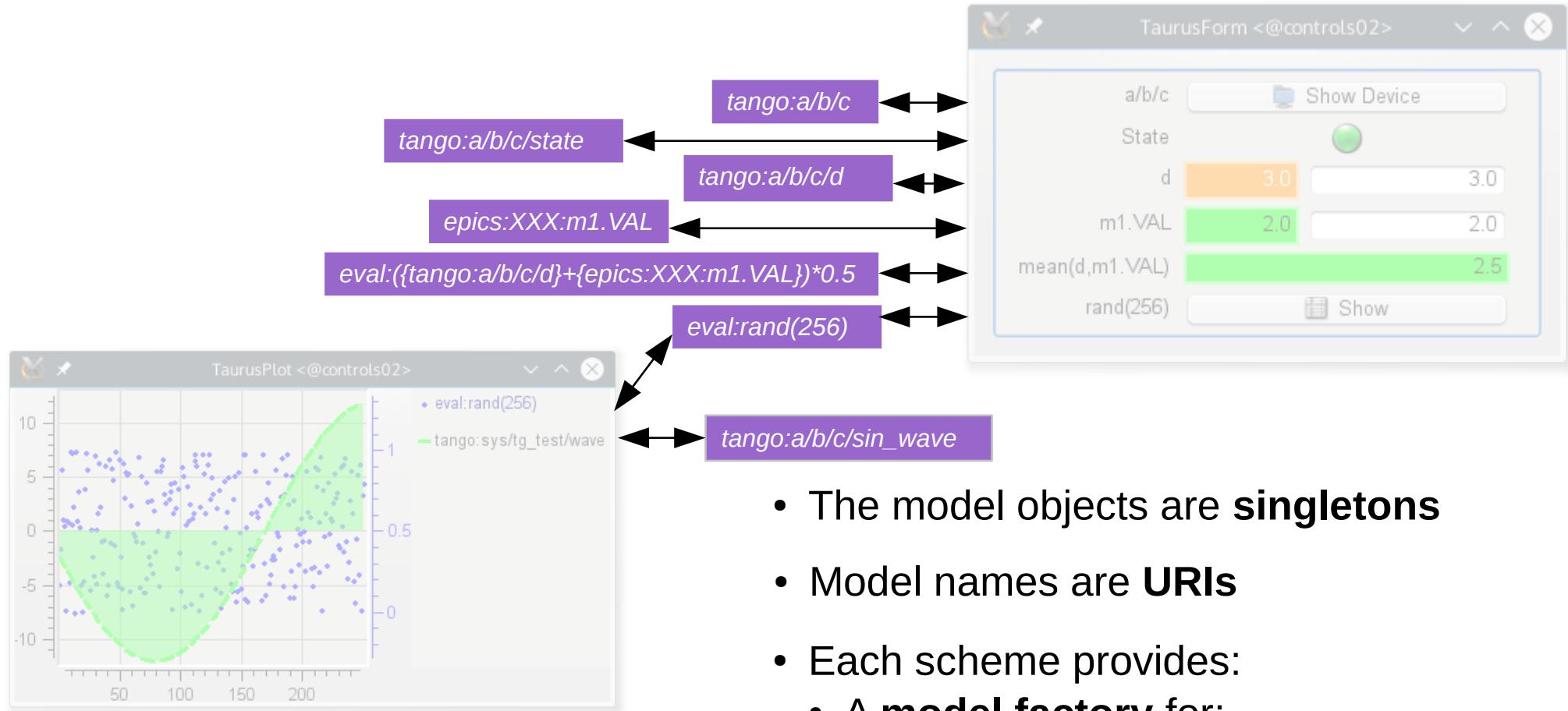




Model-View-controller

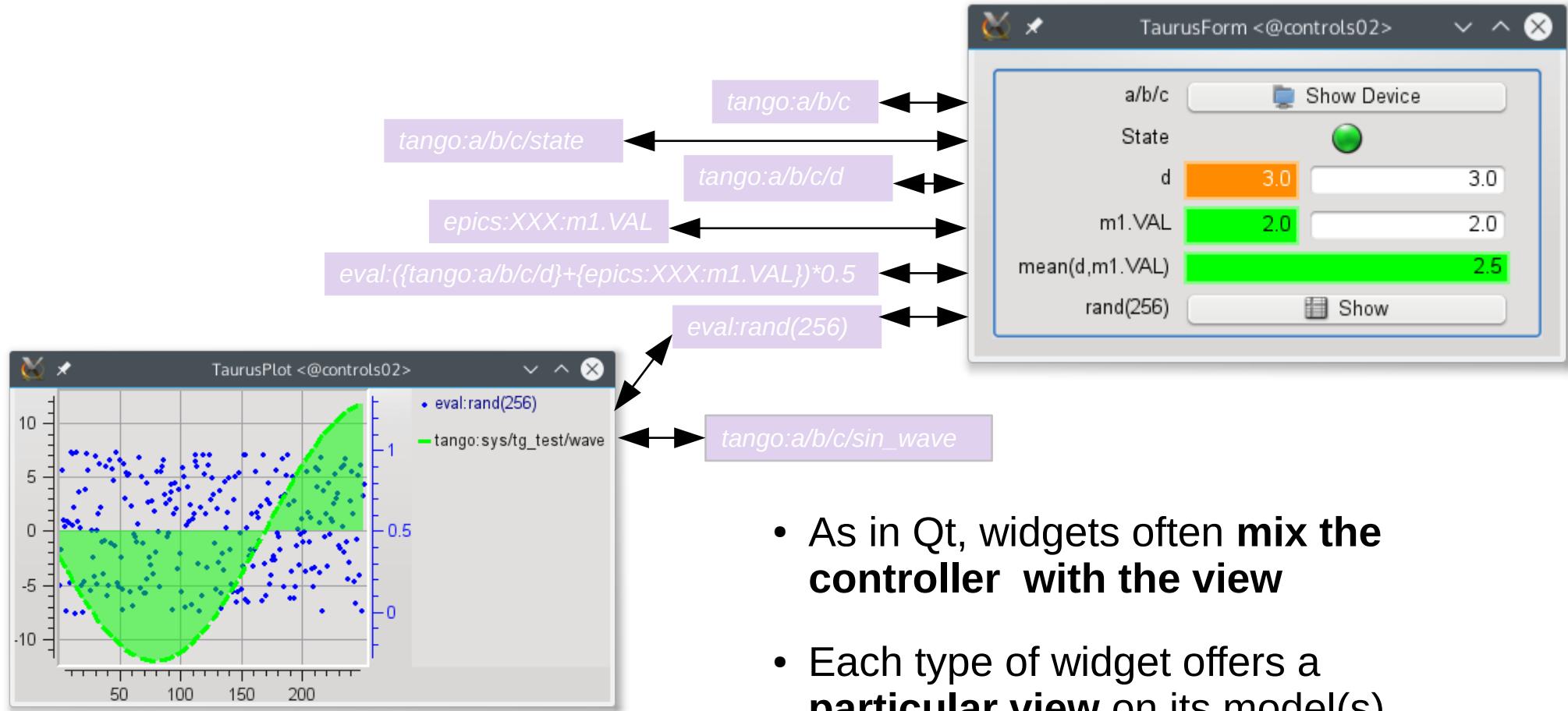


Model-View-controller



- 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)

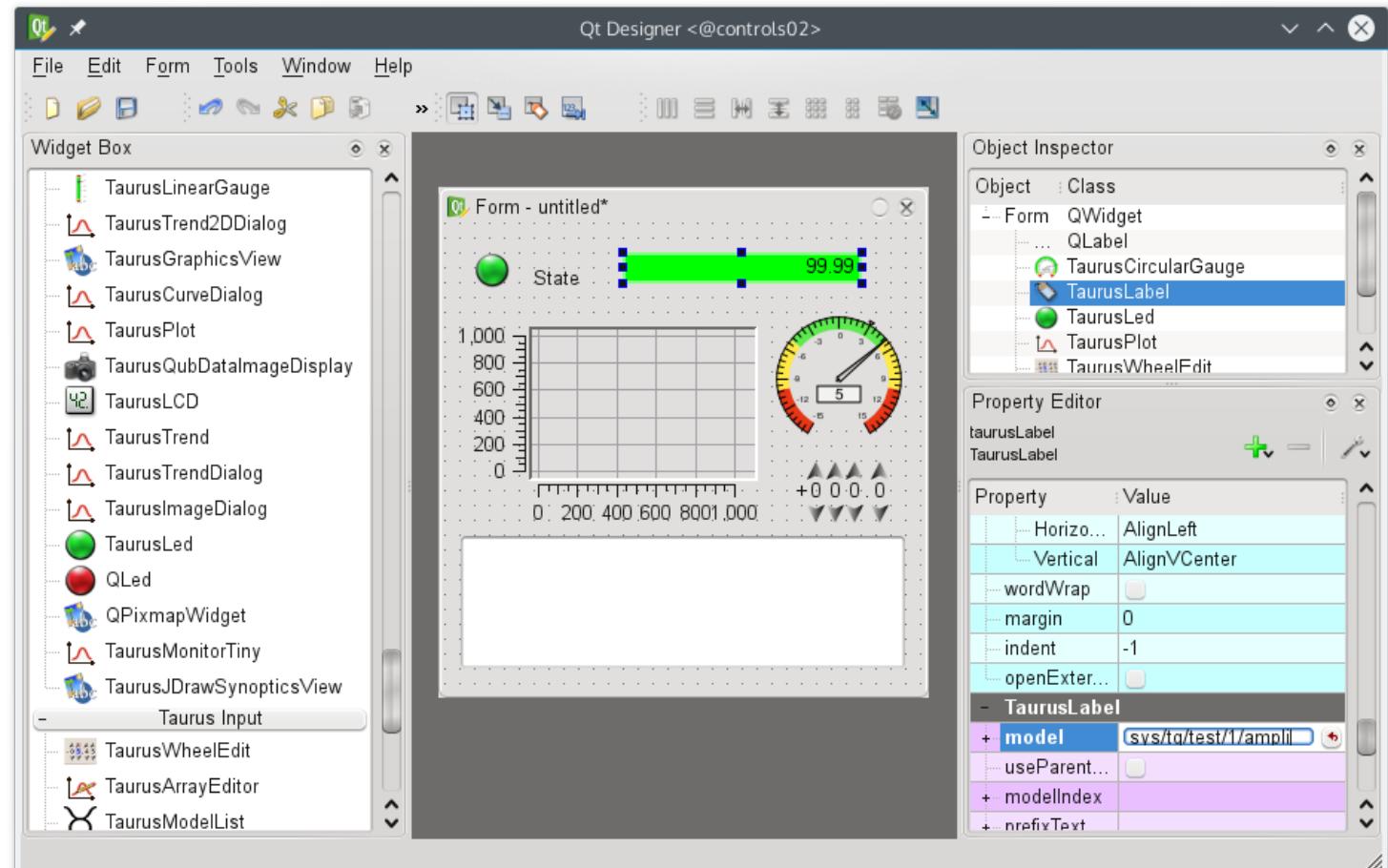


TaurusDesigner (Qt Designer)

GUIs can be created using the Qt Designer (*)

The Taurus widgets are available in the catalogue

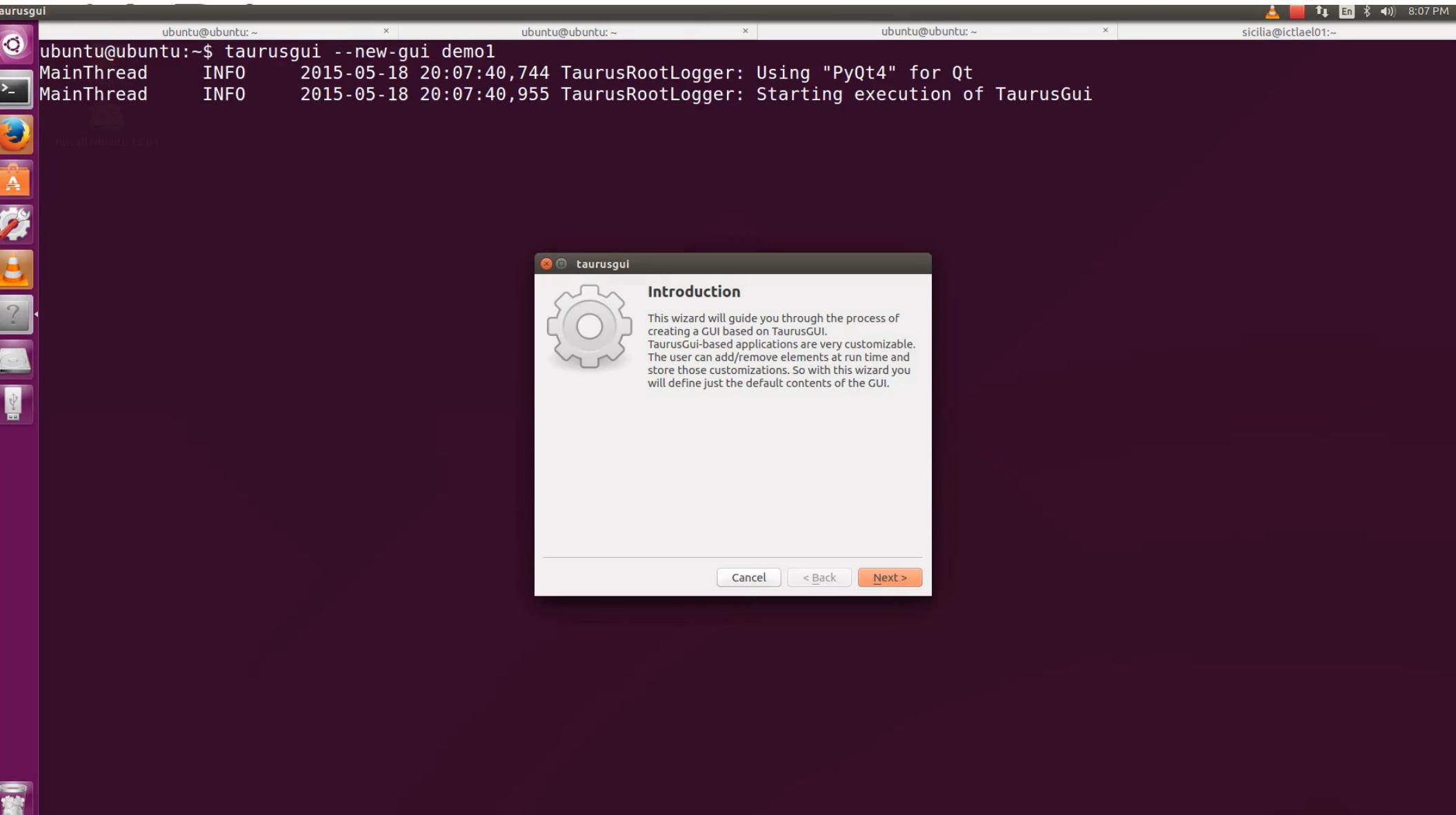
The model name can be set as a Qt property



(*) 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,...)

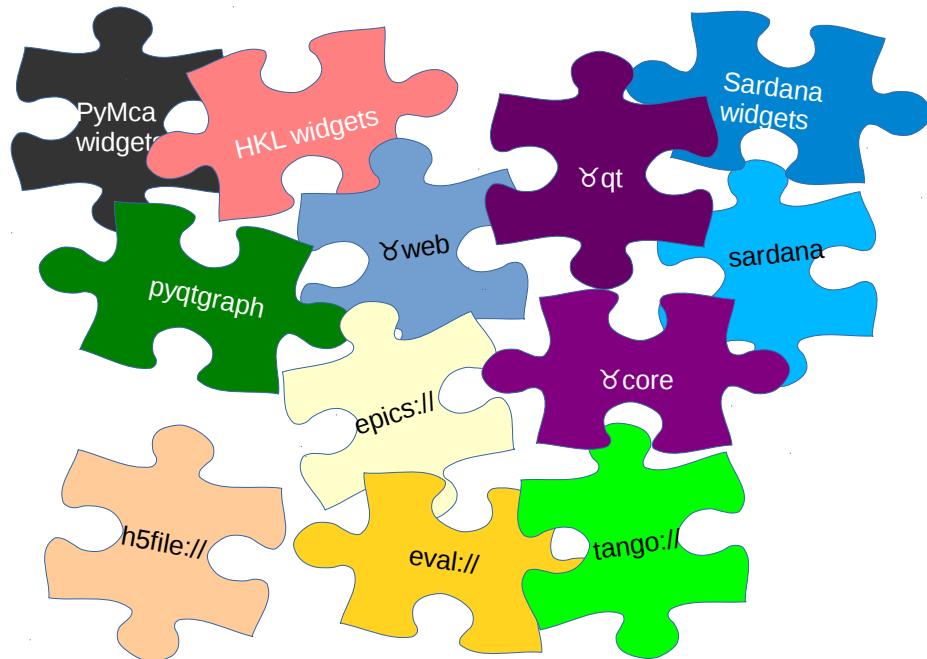


TaurusGUI (summary)

- 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)

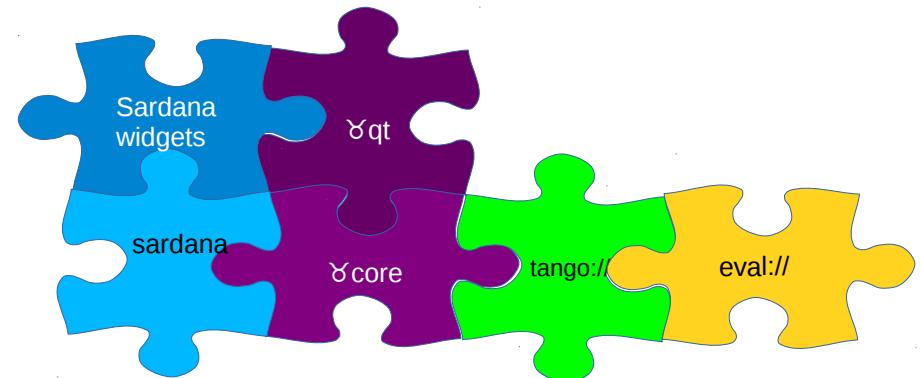
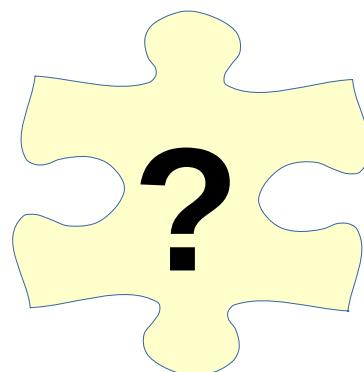


Is Taurus for you?

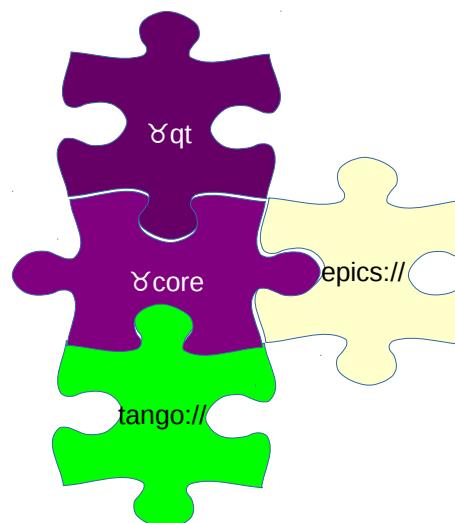


Suggested schemes:
Spec, LIMA, MADOCAP2,
Archiving, SQL, Icat,
CDMA, ASCII tables

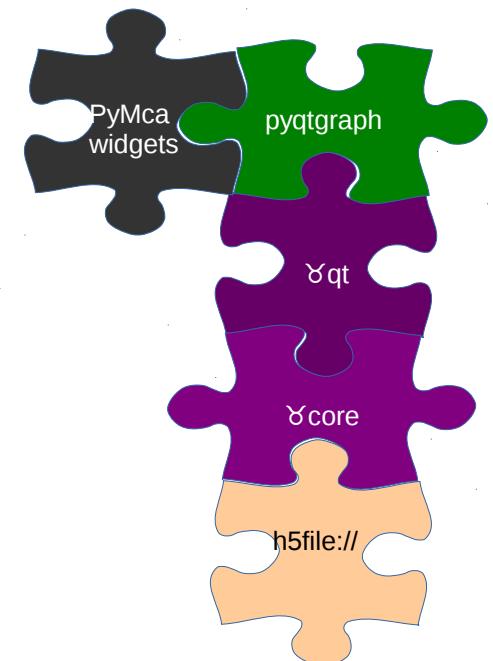
Suggested Views:
web, gtk, PyQtgraph,
PyMca



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



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
- ...

A screenshot of a web browser displaying the Taurus documentation homepage. The page has a blue header with the Taurus logo and a search bar. The main content area features a large heading "Welcome to Taurus's Home Page!" and a plot with green and red data points. To the right is a control interface for a motor, showing parameters like Velocity, Acceleration, and Step/unit, with a "Taurus" watermark overlaid. Below the plot is a descriptive text block.

Welcome to Taurus's Home Page!

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.