Karlsruhe Institute of Technology

# Beamline Control System


## Tomáš Faragó

Institute For Synchrotron Radiation

# Goals

- Abstract concept which can be implemented for many beamlines
- Scalability for fitting various data flow speeds and types
- Control of all components needed for acquisition with their current state monitoring option
- Specifying processing steps for datasets prior to reconstruction
- Availability to store acquired datasets in standardized file formats
- User friendly GUI(s) for setting experiment parameters, monitoring current status and specifying processing steps
- CLI for users who prefer command line based applications

# TANGO

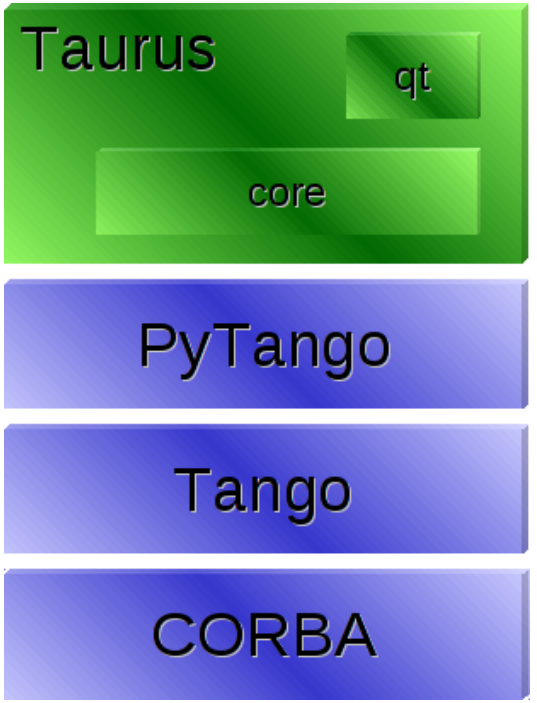
- Could be “glue” for handling communication for “slow” control
- Object oriented distributed control system
- Provides hardware abstraction
- All objects are representations of devices which can be on the same computer or distributed over a number of computers interconnected by a network
- Network communication is done using CORBA (synchronous, asynchronous or event driven)
- Configuration data is stored in a database
- Programming support is provided for C++, Java and Python
- Time scale for controlling devices is tens of milliseconds

# Sardana

- TANGO based control system designed for instrument control and data acquisition
- Designed to hide the complexity and heterogeneity of the control system
- Composed of
  - **Device Pool Server** - provides access to groups or pools of generic hardware devices such as motors or counters, event based
  - **MacroServer** - device server that asks the Device Pool Server to perform actions on pool devices, provides a group of macros (functions) to interact with the hardware through the Pool.
  - **Doors** - allow users to connect to the MacroServer, allow the client to invoke actions on the pool devices
  - **Spock** - offers a way to connect to a MacroServer using a door, acting as a Command Line Interface MacroServer client
  - **SardanaGUI** - tool used for the creation and configuration of the Device Pool Server

# TAURUS (TAngo User interface ‘R’ US )

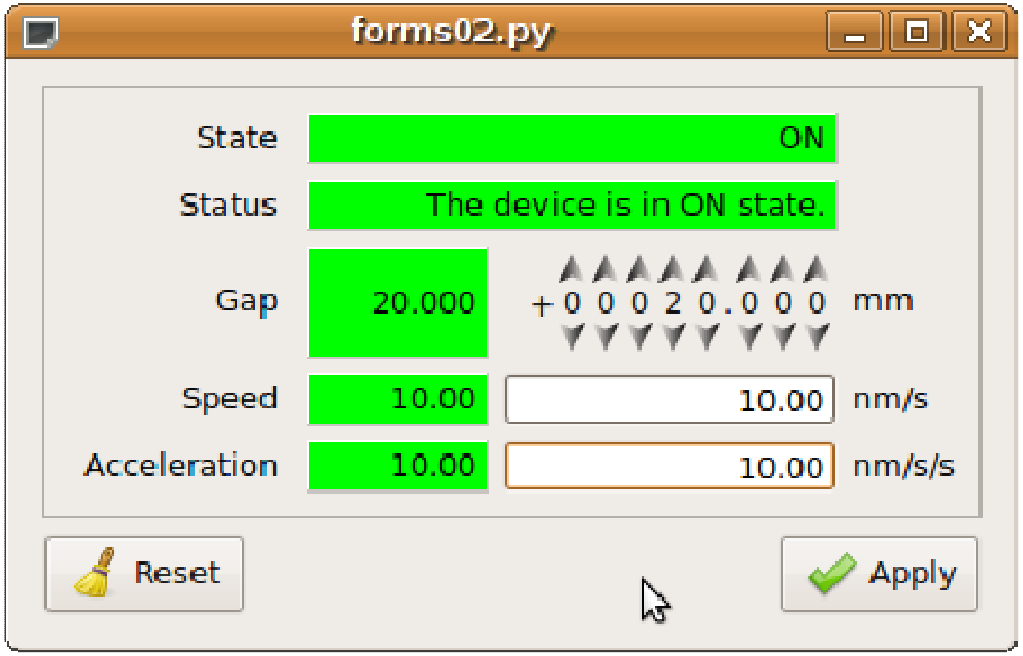
- Python framework for both CLI and GUI Tango applications. It is build on top of PyTango and PyQt
- Goals
  - provide a simple Tango API to the end-user application
  - speed up development of Tango based applications
  - provide a standardized look-and-feel
- Two main parts of framework
  - core module which handles all interaction with PyTango
  - widget module which provides a collection of widgets that can be used inside any PyQt based GUI



# TAURUS Example

```
from taurus.qt.qtgui.panel import TaurusForm
from taurus.qt.qtgui.display import TaurusValueLabel
from taurus.qt.qtgui.input import TaurusWheelEdit

panel = TaurusForm()
props = [ 'state', 'status', 'position', 'velocity', 'acceleration' ]
model = [ 'sys/taurustest/1/%s' % p for p in props ]
panel.setModel(model)
panel.getItemByIndex(0).setReadWidgetClass(TaurusValueLabel)
panel.getItemByIndex(2).setWriteWidgetClass(TaurusWheelEdit)
```



# Conclusion

- Lot of work is already done by others
- Beamline Control System could be based on Sardana and Taurus
- Main task is to integrate existing components into a new system which suits our needs
- Modularity preservation is crucial for maintainability and easy adjustment to other beamlines