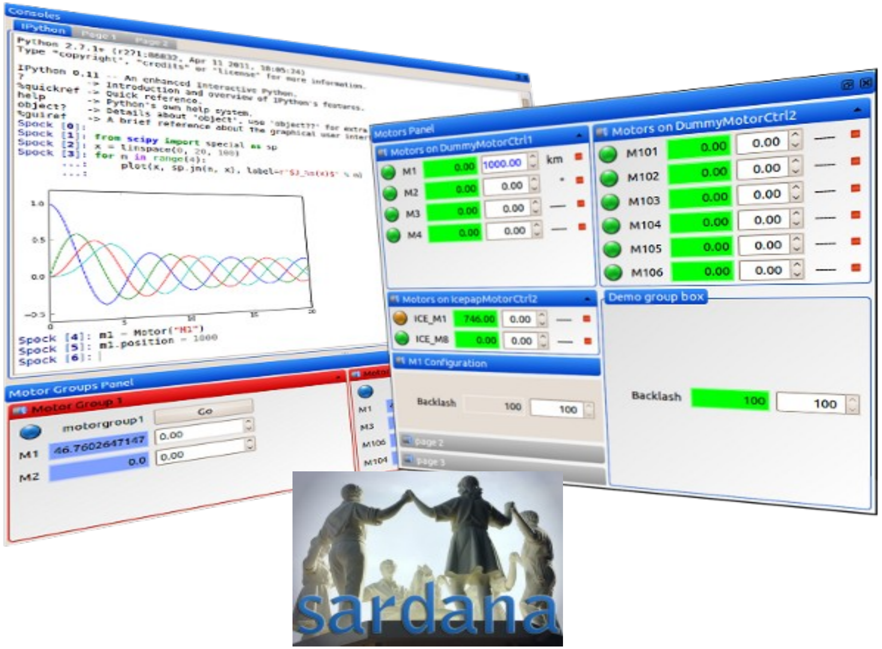


```
Python 2.7.1+ (r271:86832, Apr 11 2011, 18:05:24)
Type "copyright", "credits" or "license" for more information.

IPython 0.11 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra
%guiref   -> A brief reference about the graphical user inter
Spock [0]:
Spock [1]: from scipy import special as sp
Spock [2]: x = linspace(0, 20, 100)
Spock [3]: for n in range(4):
      ...:     plot(x, sp.jn(n, x), label=r'$J_%n(x)$' % n)
      ...:
Spock [4]: m1 = Motor("M1")
Spock [5]: m1.position = 1000
Spock [6]:
```

Introduction to Sardana

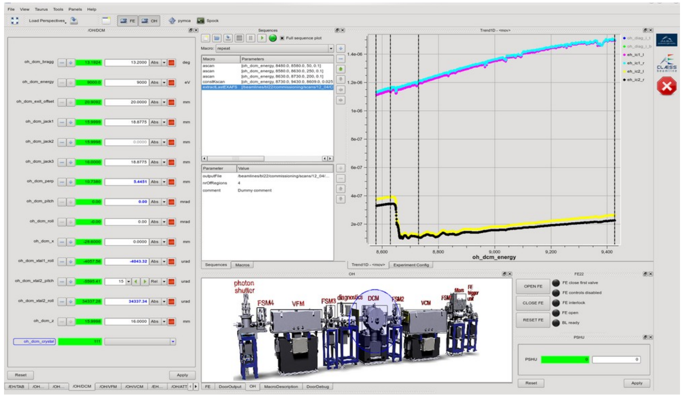
by **Zbigniew Reszela** (ALBA) on behalf of the **Sardana Community**

Tango Workshop @ ICALEPCS2021, 14.10.2021

Prerequisites

- Tango DB and TangoTest DS running
- https://github.com/sardana-org/sardana-training.git repository cloned (there is also step-by-step guide on how to reproduce the demos)

What is Sardana?



Taurus based GUIs

Scientific SCADA Suite

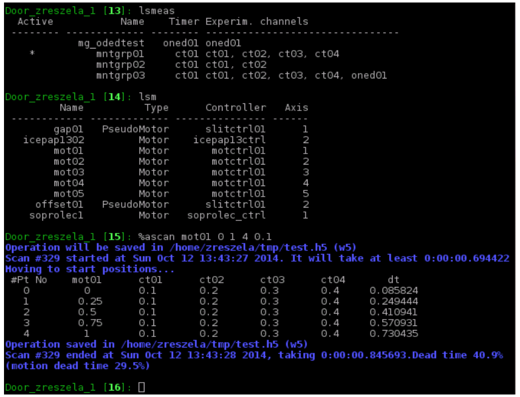
Suite = Sardana & Taurus projects

100 % Python

Built on top of Tango CS

Extendable with plugins

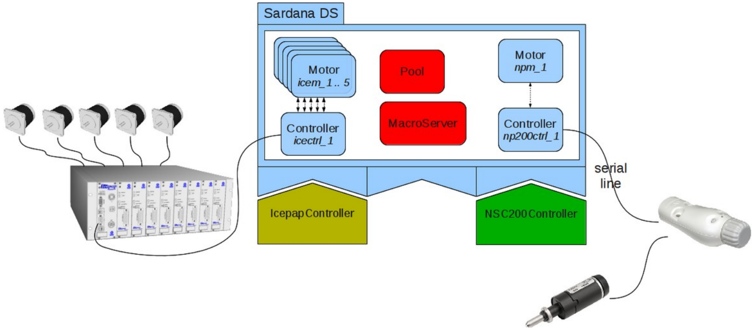
Configure, don’t program!



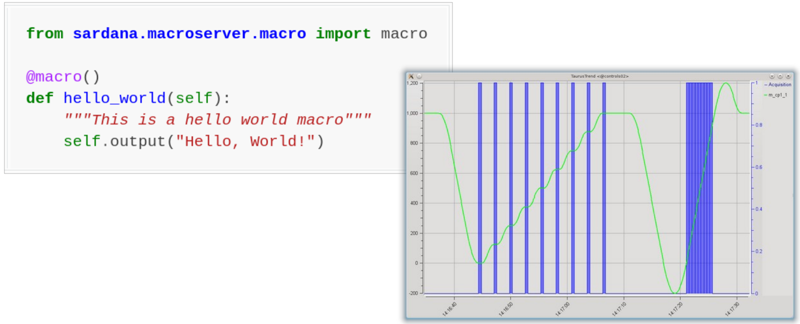
```
Door_zreszela_1 [13]: lsmeas
  Active            Name    Timer Experim. channels
 -------- ------------ -------- -------------------------------
             mg_odedtest   oned01 oned01
    *           mntgrp01     ct01 ct01, ct02, ct03, ct04
                mntgrp02     ct01 ct01, ct02
                mntgrp03     ct01 ct01, ct02, ct03, ct04, oned01

Door_zreszela_1 [14]: lsm
        Name          Type       Controller    Axis
 ----------- ------------ --------------- ------
       gap01   PseudoMotor       slitctrl01      1
   icepap1302        Motor    icepap13ctrl      2
       mot01         Motor        motctrl01      1
       mot02         Motor        motctrl01      2
       mot03         Motor        motctrl01      3
       mot04         Motor        motctrl01      4
       mot05         Motor        motctrl01      5
    offset01   PseudoMotor       slitctrl01      2
    soprolec1        Motor    soprolec_ctrl      1

Door_zreszela_1 [15]: %ascan mot01 0 1 4 0.1
Operation will be saved in /home/zreszela/tmp/test.h5 (w5)
Scan #329 started at Sun Oct 12 13:43:27 2014. It will take at least 0:00:00.694422
Moving to start positions...
 #Pt No    mot01      ct01      ct02      ct03      ct04       dt
   0         0        0.1       0.2       0.3       0.4     0.085824
   1        0.25      0.1       0.2       0.3       0.4     0.249444
   2        0.5       0.1       0.2       0.3       0.4     0.410941
   3        0.75      0.1       0.2       0.3       0.4     0.570931
   4         1        0.1       0.2       0.3       0.4     0.730435
Operation saved in /home/zreszela/tmp/test.h5 (w5)
Scan #329 ended at Sun Oct 12 13:43:28 2014, taking 0:00:00.845693.Dead time 40.9%
(motion dead time 29.5%)

Door_zreszela_1 [16]: 
```

Spock – IPython based CLI

Device Pool – HW access + low level control



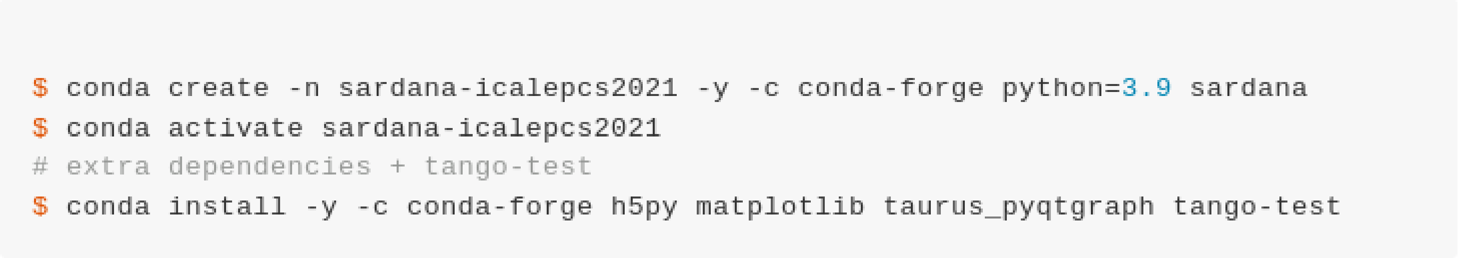
MacroServer – powerful sequencer



```python
from sardana.macroserver.macro import macro

@macro()
def hello_world(self):
    """This is a hello world macro"""
    self.output("Hello, World!")
```

Installation

- Debian packages (apt install python3-sardana)
- PyPI (pip install sardana)
- Conda:

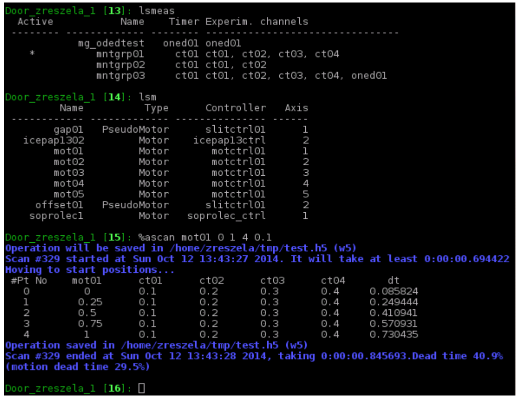


```
$ conda create -n sardana-icalepcs2021 -y -c conda-forge python=3.9 sardana
$ conda activate sardana-icalepcs2021
# extra dependencies + tango-test
$ conda install -y -c conda-forge h5py matplotlib taurus_pyqtgraph tango-test
```

Creating sar_demo environment

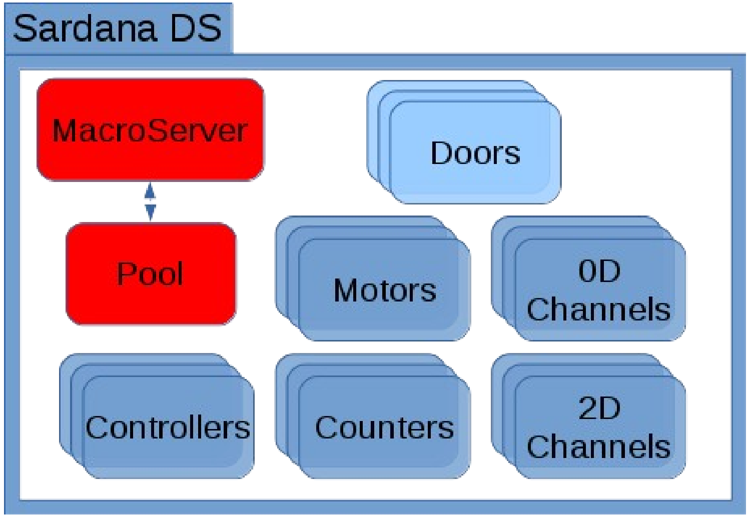
Creating sar_demo environment

- Start Sardana server
- Start Spock client
- List macros and elements
  - Built-in macros try to follow the SPEC syntax
- Run sar_demo macro



```
Door_zreszela_1 [13]: lsmeas
  Active            Name    Timer Experim. channels
 -------- ------------ -------- -------------------------------
          mg_odedtest   oned01 oned01
    *         mntgrp01     ct01 ct01, ct02, ct03, ct04
              mntgrp02     ct01 ct01, ct02
              mntgrp03     ct01 ct01, ct02, ct03, ct04, oned01

Door_zreszela_1 [14]: lsm
       Name          Type      Controller   Axis
 ------------ ------------- -------------- ------
       gap01   PseudoMotor      slitctrl01      1
   icepap1302         Motor    icepap13ctrl      2
        mot01         Motor       motctrl01      1
        mot02         Motor       motctrl01      2
        mot03         Motor       motctrl01      3
        mot04         Motor       motctrl01      4
        mot05         Motor       motctrl01      5
     offset01   PseudoMotor      slitctrl01      2
    soprolec1         Motor   soprolec_ctrl      1

Door_zreszela_1 [15]: %ascan mot01 0 1 4 0.1
Operation will be saved in /home/zreszela/tmp/test.h5 (w5)
Scan #329 started at Sun Oct 12 13:43:27 2014. It will take at least 0:00:00.694422
Moving to start positions...
 #Pt No    mot01      ct01      ct02      ct03      ct04       dt
   0         0        0.1       0.2       0.3       0.4     0.085824
   1        0.25      0.1       0.2       0.3       0.4     0.249444
   2        0.5       0.1       0.2       0.3       0.4     0.410941
   3        0.75      0.1       0.2       0.3       0.4     0.570931
   4         1        0.1       0.2       0.3       0.4     0.730435
Operation saved in /home/zreszela/tmp/test.h5 (w5)
Scan #329 ended at Sun Oct 12 13:43:28 2014, taking 0:00:00.845693.Dead time 40.9%
(motion dead time 29.5%)

Door_zreszela_1 [16]: 
```

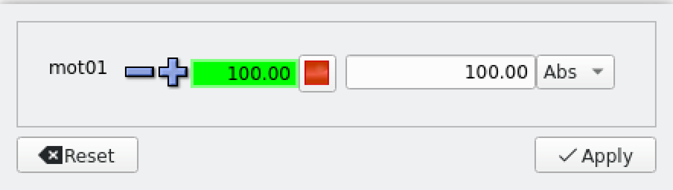
Macros execution in Spock CLI



Sardana Device Server with sar_demo elements

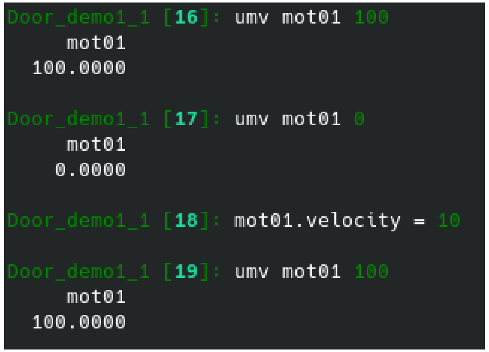
Motion control (simulators)

- Motor widget & motion macros
- Change motor's velocity



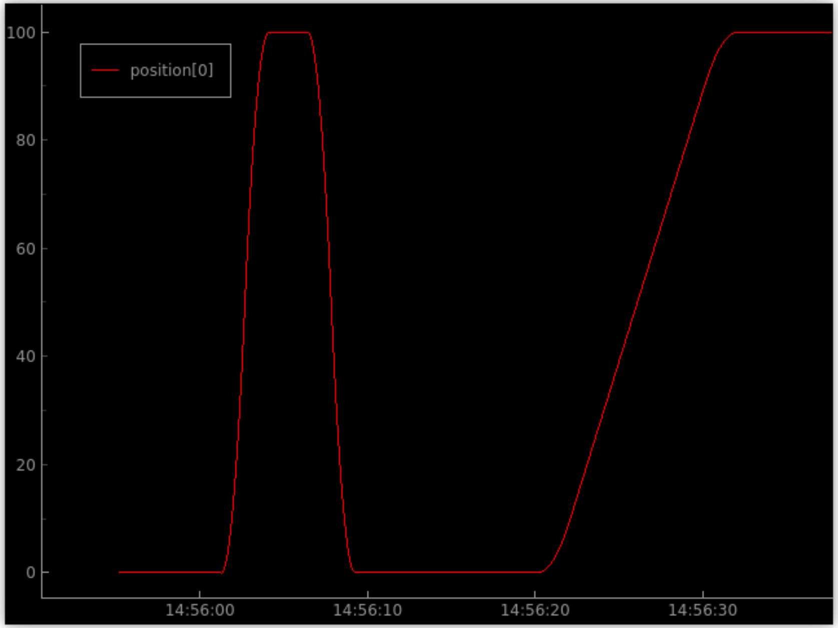
mot01 100.00 100.00 Abs

Taurus form with Motor widget



```
Door_demo1_1 [16]: umv mot01 100
     mot01
  100.0000

Door_demo1_1 [17]: umv mot01 0
     mot01
    0.0000

Door_demo1_1 [18]: mot01.velocity = 10

Door_demo1_1 [19]: umv mot01 100
     mot01
  100.0000
```

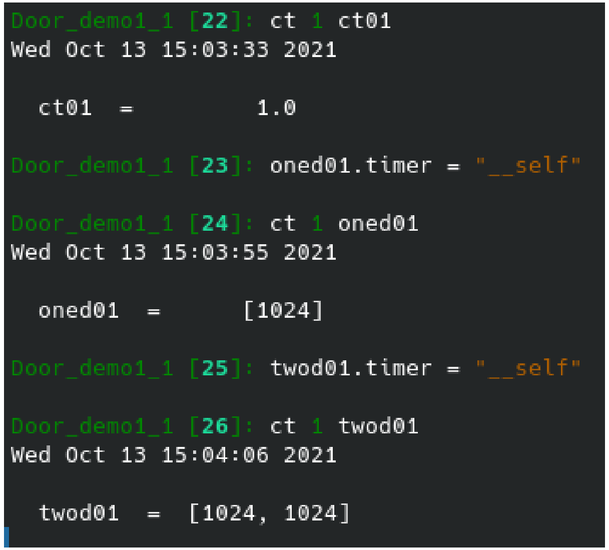
Execution of motion macros in Spock



Taurus trend tracking motor's position attribute

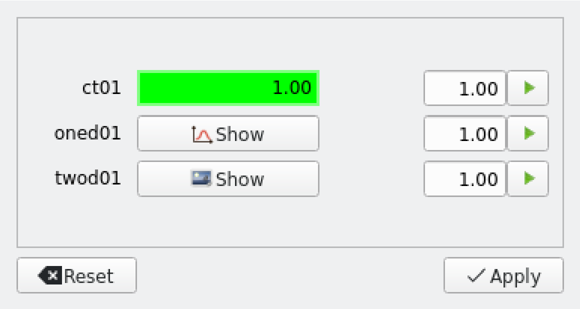
DAQ control (simulators)

- Channel widget and DAQ macros
- Change channel’s timer and integration time



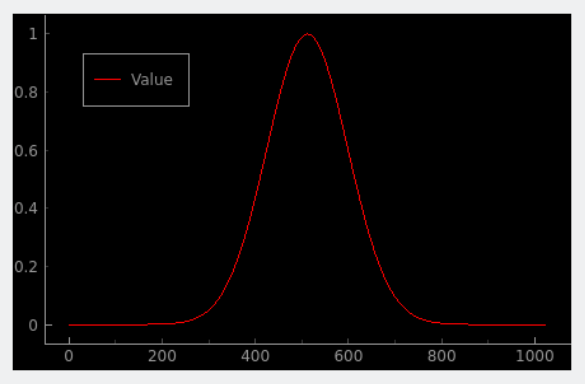
```
Door_demo1_1 [22]: ct 1 ct01
Wed Oct 13 15:03:33 2021

  ct01  =         1.0

Door_demo1_1 [23]: oned01.timer = "__self"

Door_demo1_1 [24]: ct 1 oned01
Wed Oct 13 15:03:55 2021

  oned01  =      [1024]

Door_demo1_1 [25]: twod01.timer = "__self"

Door_demo1_1 [26]: ct 1 twod01
Wed Oct 13 15:04:06 2021

  twod01  =  [1024, 1024]
```

Data acquisition macros execution in Spock

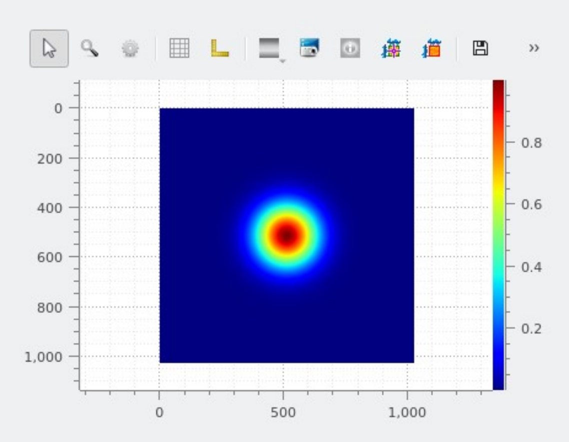


ct01	1.00	1.00
oned01	Show	1.00
twod01	Show	1.00

Taurus form with experimental channel widgets



Dummy 1D Value attribute



Dummy 2D Value attribute

Taurus GUI

Taurus GUI without programming a single line of code

- Execute command: taurus newgui
- Follow the wizard:
  - Choose the project directory (e.g: <your home>/demogui)
  - Choose GUI name (e.g: demogui)
  - Add synoptic (optional): sardana-training/short/res/demoBL.jdw
  - Enable Sardana communication (optional) select: MacroServer - MacroServer/demo1/1 and Door - Door/demo1/1
  - Generate panels from Sardana Pool (optional): choose yes
- We will skip some steps: custom logo, extra panels, Monitor list

Taurus GUI - interaction with *instruments*

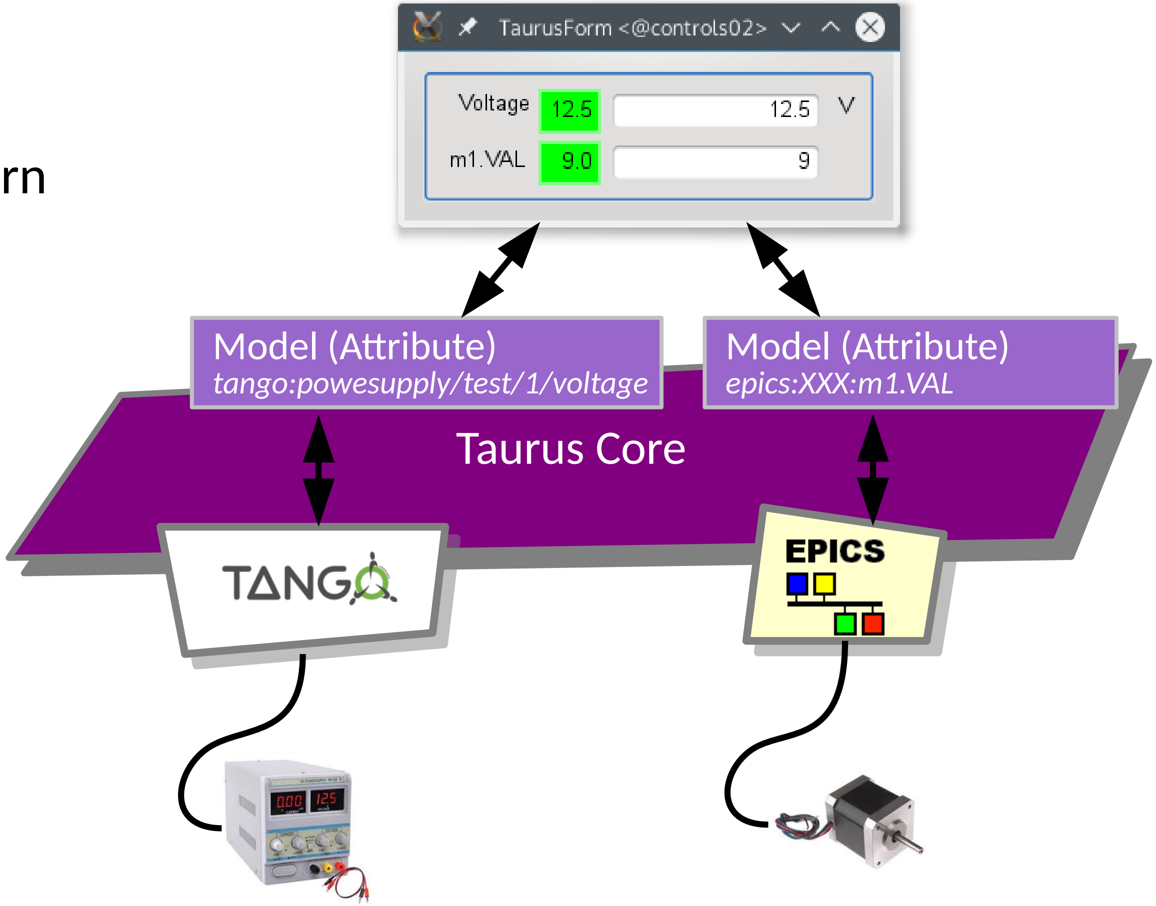
- Reorder the widgets for "interaction with instruments":
  - Go to Panels -> hide all panels
  - Click on the "demoBL" button in the toolbar to show the synoptic panel
  - Click on the "mirror" instrument in the synoptic (the area below "DCM"). This should show the "/mirror" panel
  - Move the "mirror" panel above the synoptic
  - Click on the slits in the synoptics (labeled "diagnostics" in the synoptic). This should show the "/slits" panel. Move it to a tab together with "/mirror"
  - Click on the monitor in the synoptics (labeled "FSM4" in the synoptic). This should show the "/monitor" panel. Move it to a tab together with "/mirror" and "/slits"
- Show the 2-ways communication between panels and synoptics
  - Click on the active areas of the synoptics and show that the corresponding panels are shown
  - Select the panels and see that the synoptic highlights the corresponding area
- Save as *Instruments* perspective

Taurus GUI - interaction with Tango CS

- Go to Panels -> hide all panels
- Create a Tango DB tree panel:
  - Use New Panel button
    - Select TaurusDBTreeWidget and use name db
    - Click on "Advanced Settings" and set tango://<your Tango DB host>:10000 as model and click on finish
- Create a Plot panel:
  - Use New Panel button
    - Select TaurusPlot and use name plot and click on finish
- Create a Form panel:
  - Use New Panel button
    - Select TaurusForm and use name form and click on finish
- Make sure that the "db" and "form" and "plot" panels are all simultaneously visible
- Add new elements to the "form" panel:
  - Navigate in the db panel to sys/tg_test/1/ampli, and drag and drop it into "form".
  - Navigate in the db panel to sys/tg_test/1/boolean_scalar, and drag and drop it into "form"
- Add a new element to the "plot" panel:
  - Navigate in the db panel to sys/tg_test/1/wave, and drag and drop it into "plot"
- Save as *Tango* perspective

Taurus models and schemes

- Taurus is **data source agnostic**
- Taurus uses **Model ↔ View** pattern
- Models are provided by ***schemes* plugins**



TaurusForm <@controls02>

Voltage 12.5 12.5 V

m1.VAL 9.0 9

Model (Attribute)
tango:powesupply/test/1/voltage

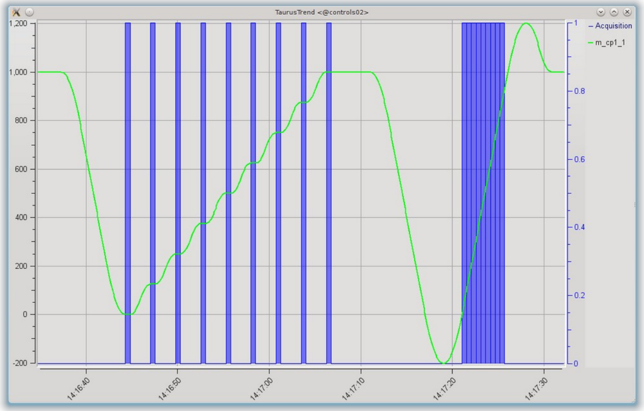
Model (Attribute)
epics:XXX:m1.VAL

Taurus Core

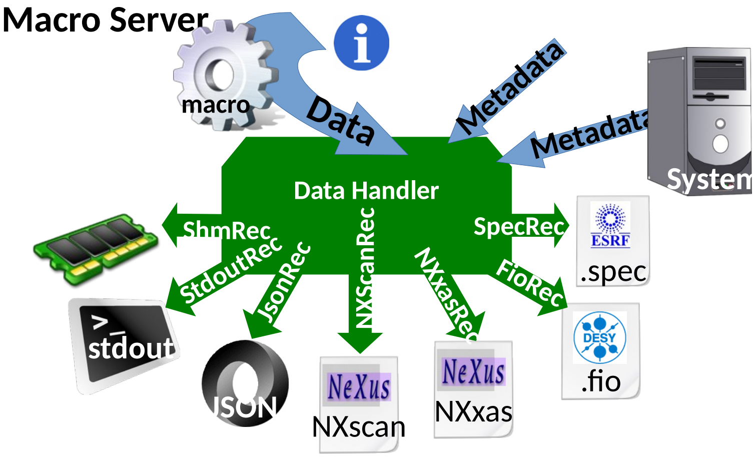
Scans

Generic Scan Framework

- Step and Continuous Scans
- Turn-key scan macros
- Framework for developing custom scans
- *Recorder* plugins for data storage



Taurus trend showing motion and acquisition during step and continuous scans



Data storage using Recorder plugins

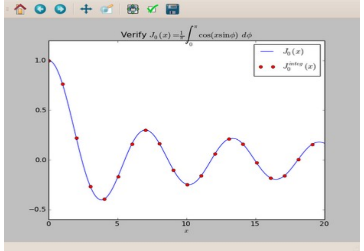
Developing macros and controllers

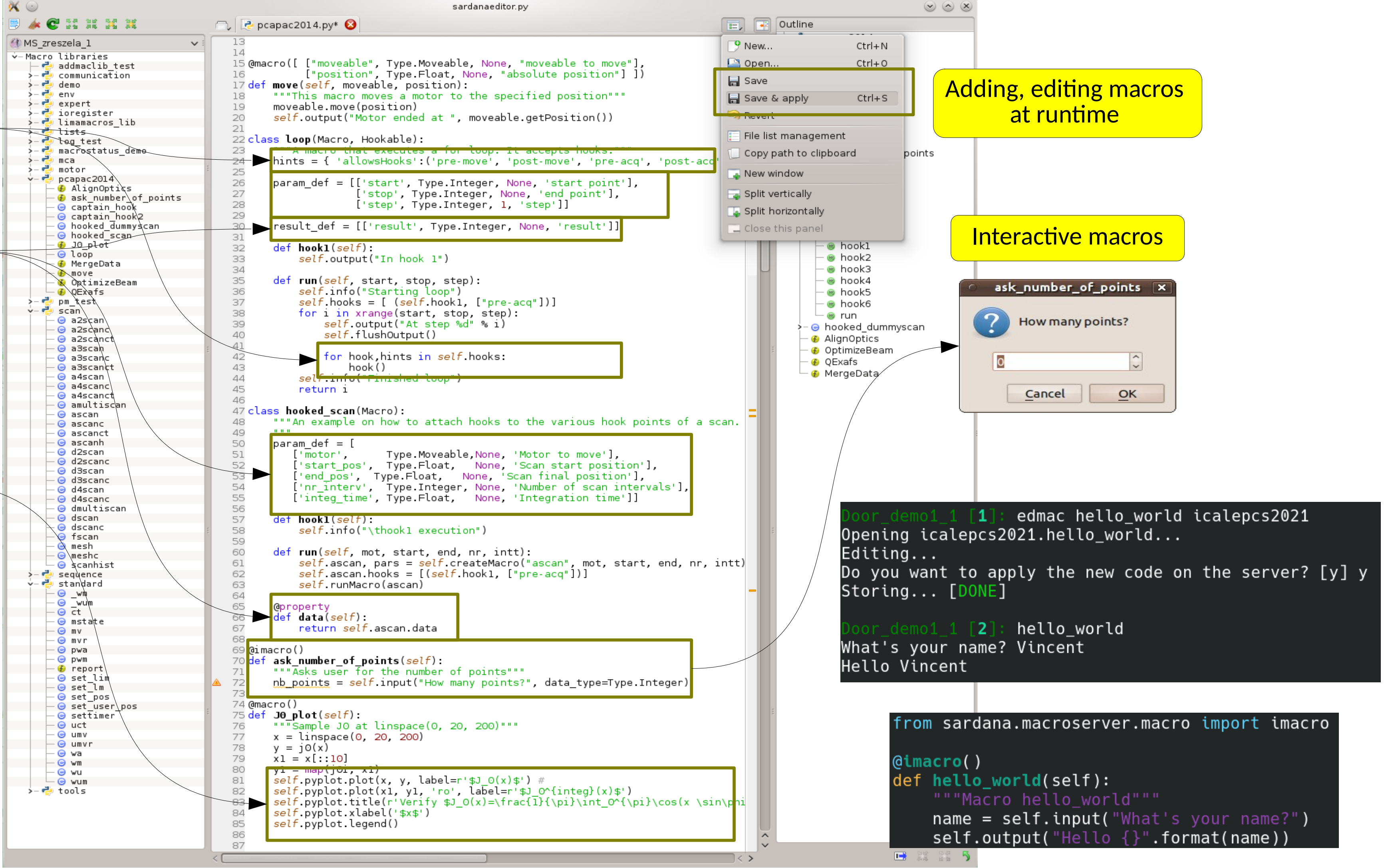
Developing custom macros

Hooks

Input parameters & results & data

Plotting





```python


@macro([ ["moveable", Type.Moveable, None, "moveable to move"],
         ["position", Type.Float, None, "absolute position"] ])
def move(self, moveable, position):
    """This macro moves a motor to the specified position"""
    moveable.move(position)
    self.output("Motor ended at ", moveable.getPosition())

class loop(Macro, Hookable):
    """A macro that executes a for loop. It accepts hooks."""
    hints = { 'allowsHooks':('pre-move', 'post-move', 'pre-acq', 'post-acq'

    param_def = [['start', Type.Integer, None, 'start point'],
                 ['stop', Type.Integer, None, 'end point'],
                 ['step', Type.Integer, 1, 'step']]

    result_def = [['result', Type.Integer, None, 'result']]

    def hook1(self):
        self.output("In hook 1")

    def run(self, start, stop, step):
        self.info("Starting loop")
        self.hooks = [ (self.hook1, ["pre-acq"])]
        for i in xrange(start, stop, step):
            self.output("At step %d" % i)
            self.flushOutput()

            for hook,hints in self.hooks:
                hook()
        self.info("Finished loop")
        return i

class hooked_scan(Macro):
    """An example on how to attach hooks to the various hook points of a scan.
    """
    param_def = [
       ['motor',      Type.Moveable,None, 'Motor to move'],
       ['start_pos',  Type.Float,   None, 'Scan start position'],
       ['end_pos',  Type.Float,   None, 'Scan final position'],
       ['nr_interv',  Type.Integer, None, 'Number of scan intervals'],
       ['integ_time', Type.Float,   None, 'Integration time']]

    def hook1(self):
        self.info("\thook1 execution")

    def run(self, mot, start, end, nr, intt):
        self.ascan, pars = self.createMacro("ascan", mot, start, end, nr, intt)
        self.ascan.hooks = [(self.hook1, ["pre-acq"])]
        self.runMacro(ascan)

    @property
    def data(self):
        return self.ascan.data

@imacro()
def ask_number_of_points(self):
    """Asks user for the number of points"""
    nb_points = self.input("How many points?", data_type=Type.Integer)

@macro()
def J0_plot(self):
    """Sample J0 at linspace(0, 20, 200)"""
    x = linspace(0, 20, 200)
    y = j0(x)
    x1 = x[::10]
    y1 = map(j0i, x1)
    self.pyplot.plot(x, y, label=r'$J_0(x)$') #
    self.pyplot.plot(x1, y1, 'ro', label=r'$J_0^{integ}(x)$')
    self.pyplot.title(r'Verify $J_0(x)=\frac{1}{\pi}\int_0^{\pi}\cos(x \sin\phi
    self.pyplot.xlabel('$x$')
    self.pyplot.legend()
```

Adding, editing macros at runtime

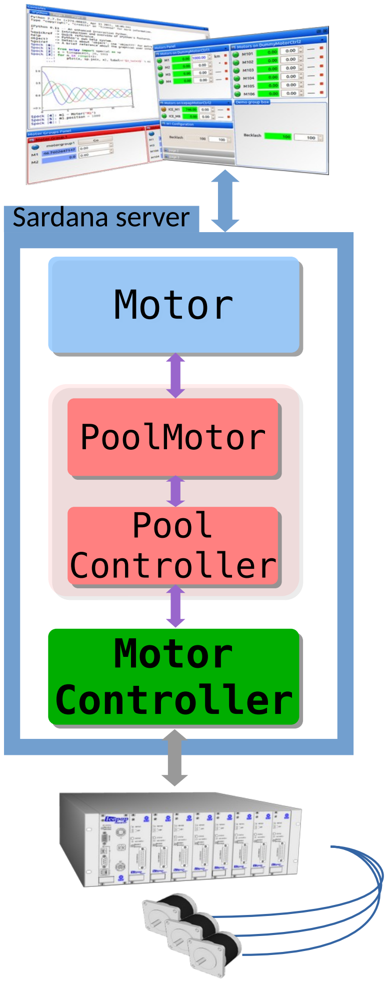
Interactive macros

```
Door_demo1_1 [1]: edmac hello_world icalepcs2021
Opening icalepcs2021.hello_world...
Editing...
Do you want to apply the new code on the server? [y] y
Storing... [DONE]

Door_demo1_1 [2]: hello_world
What's your name? Vincent
Hello Vincent
```

```python
from sardana.macroserver.macro import imacro

@imacro()
def hello_world(self):
    """Macro hello_world"""
    name = self.input("What's your name?")
    self.output("Hello {}".format(name))
```

Developing custom controllers

- Motor standard interface
- MotorController API
- *Blender Slits* as custom hardware
- Plugin discovery mechanism
- Define Sardana controllers and elements
- Scan *Blender Slits* offset



Motor

position: float
state: enum
offset: float
sign: int
steps_per_unit
…

Stop()
Abort()
…

Motor Tango device interface (attributes & commands)

```
class MyMotorCtrl(MotorController):

    def StateOne(self, axis):
        [...]

    def ReadOne(self, axis):
        [...]

    def StartOne(self, axis, pos):
        [...]

    def AbortOne(self, axis):
        [...]
```

MotorController minimal API

Sardana Community collaboration

Sardana applications

ALBA	8 Beamlines (BLs) in prod. (+5 BLs in constr.); Accelerator (ACC); 3 Labs
DESY	14 BLs in prod.
ESRF The European Synchrotron	ACC operation (MacroServer only)
IPANEMA	1 Lab
LPS ORSAY	1 Lab (diffractometer)
MAX IV LABORATORY	14 BLs
MBI Max Born Institute	7 Lab setups
SOLARIS NATIONAL SYNCHROTRON RADIATION CENTRE	4 BLs in prod.

~50 Sardana applications

Community Tools & Events

Docs: ***http://www.sardana-controls.org***
http://www.taurus-scada.org

Projects: *https://gitlab.com/taurus-org/taurus*
https://github.com/sardana-org/sardana *(soon on GitLab)*

SEP Sardana Enhancement Proposal **TEP** Taurus Enhancement Proposal

SEP index: *https://sardana-controls.org/sep/index.html*
TEP index: *https://taurus-scada.org/tep/index.html*

Follow-up: *https://meet.jit.si/sardana*
https://github.com/sardana-org/sardana-followup *(soon on GitLab)*

Sardana plugins catalog: *https://github.com/sardana-org/sardana-plugins*
(soon on GitLab)

Training and workshops:
https://github.com/sardana-org/sardana-training *(soon on GitLab)*
(satellite to Tango meetings or ICALEPCS conference)

To:
List

Announcements: *sardana-users@lists.sourceforge.net*
tauruslib-users@lists.sourceforge.net

Coordination: *sardana-devel@lists.sourceforge.net*
tauruslib-devel@lists.sourceforge.net

Thank you for your attention!