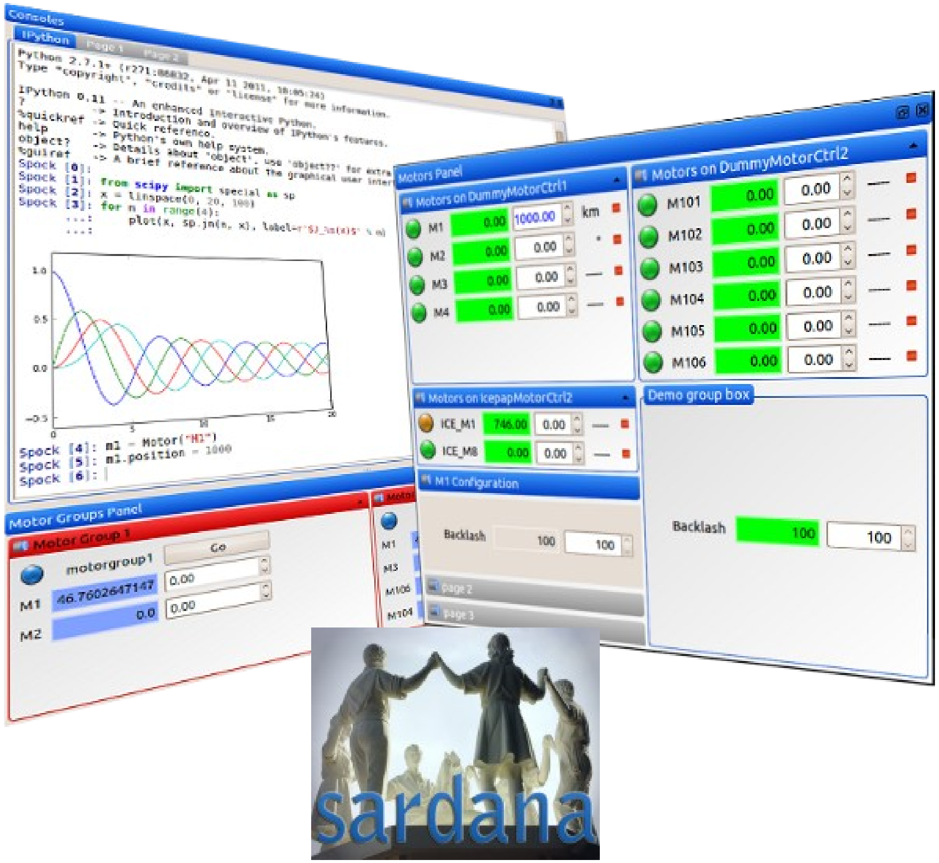
Sardana Suite Overview



```
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]:
```

Sardana – Scientific SCADA

- Sardana is an **open source**, **Python based**, **scientific SCADA** suite applicable in large spectrum of installations such as particle accelerators, experimental stations or small labs

www.sardana-controls.org

www.taurus-scada.org

- Sardana was initially an internal Alba project

  ...but after its successful use in other synchrotrons it became a **community** driven project

www.albasynchrotron.es

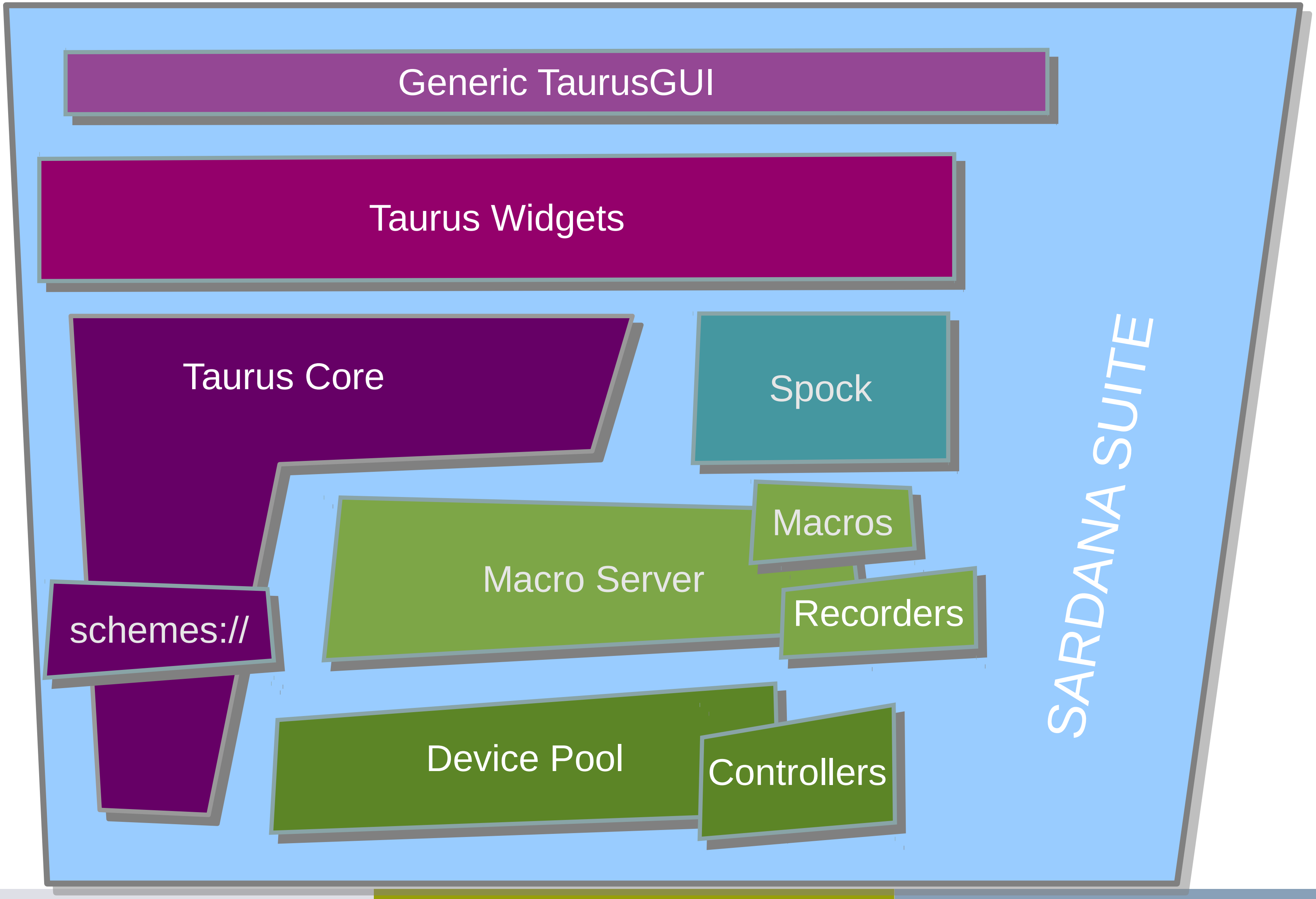
ALBA DESY MAX IV LABORATORY SOLARIS NATIONAL SYNCHROTRON RADIATION CENTRE

Sardana – Scientific SCADA

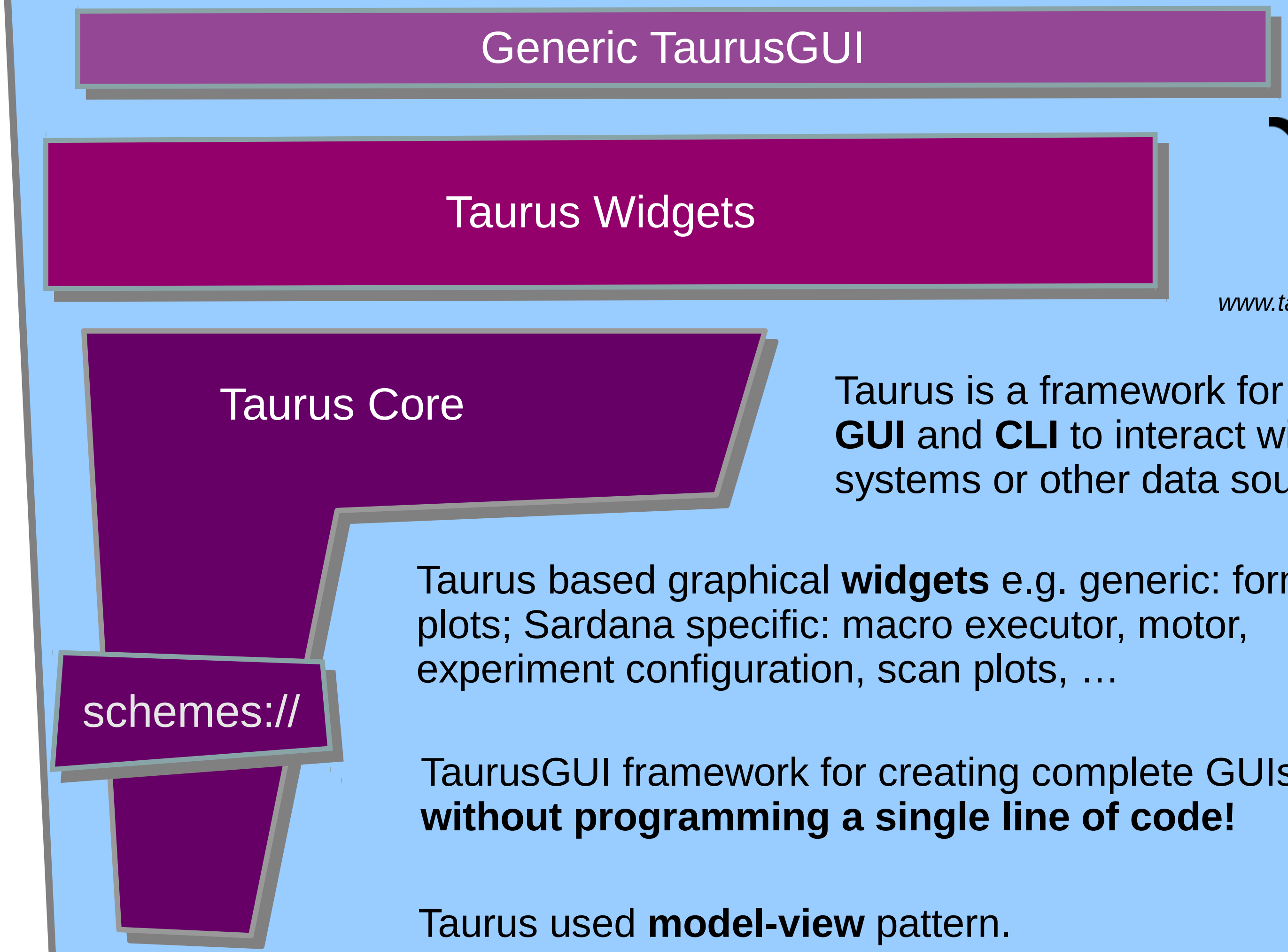
- Sardana is highly modular
- Kernel/Core is software library written in **Python**
- It is heavily based on the concept of **plugins** what allows easy customization of the system
- Its architecture is based on the **client-server** model
- Allows to build **distributed control system** depending on the selected server architecture
- Currently only **Tango** communication protocol exists but others could be implemented as plugins

www.tango-controls.org

Sardana Suite



Taurus library – client



www.taurus-scada.org

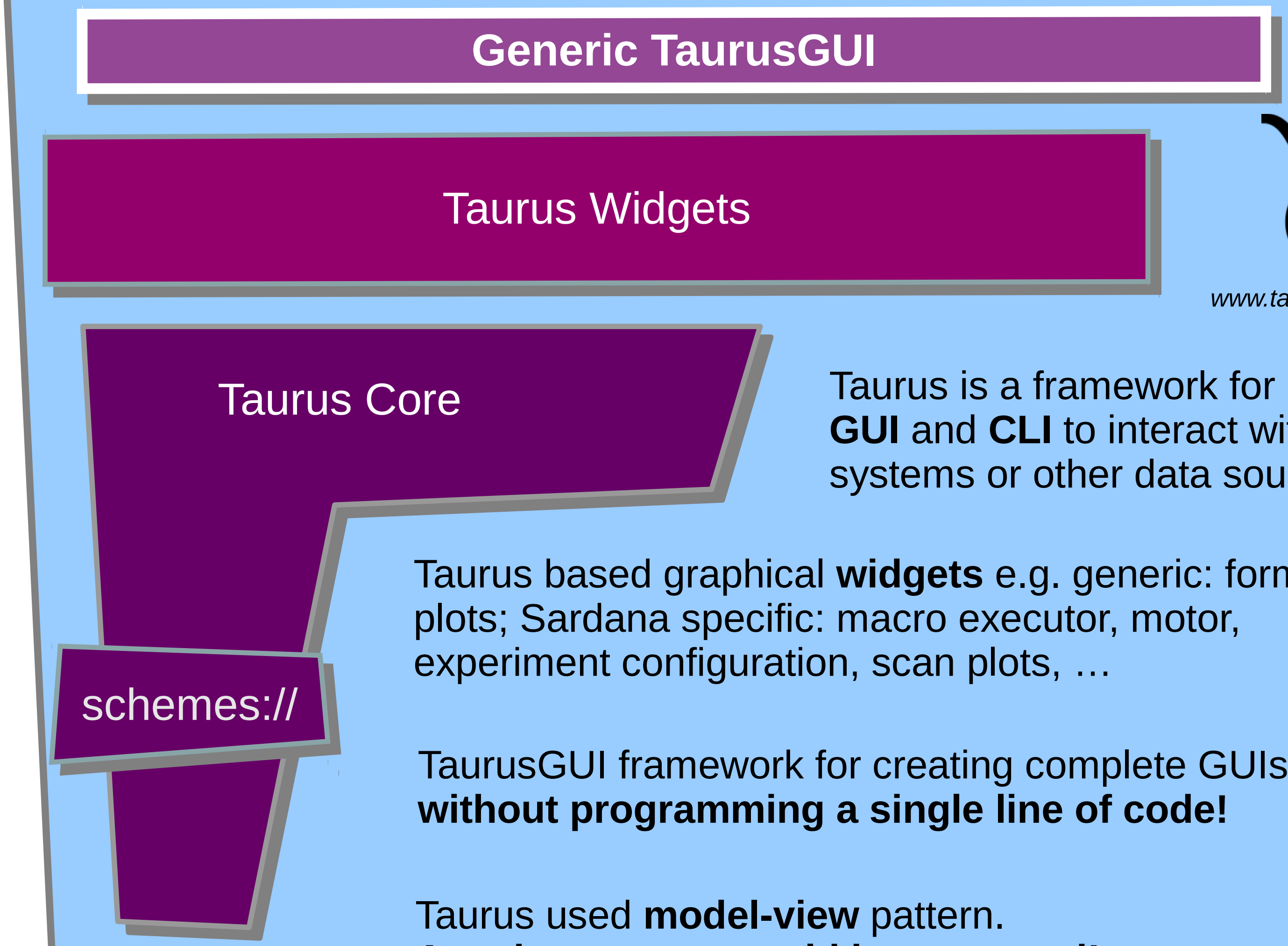
Taurus is a framework for creating **GUI** and **CLI** to interact with control systems or other data sources

Taurus based graphical **widgets** e.g. generic: forms, plots; Sardana specific: macro executor, motor, experiment configuration, scan plots, …

TaurusGUI framework for creating complete GUIs **without programming a single line of code!**

Taurus used **model-view** pattern.
Any data source could be accessed!

Taurus library – client



♉

www.taurus-scada.org

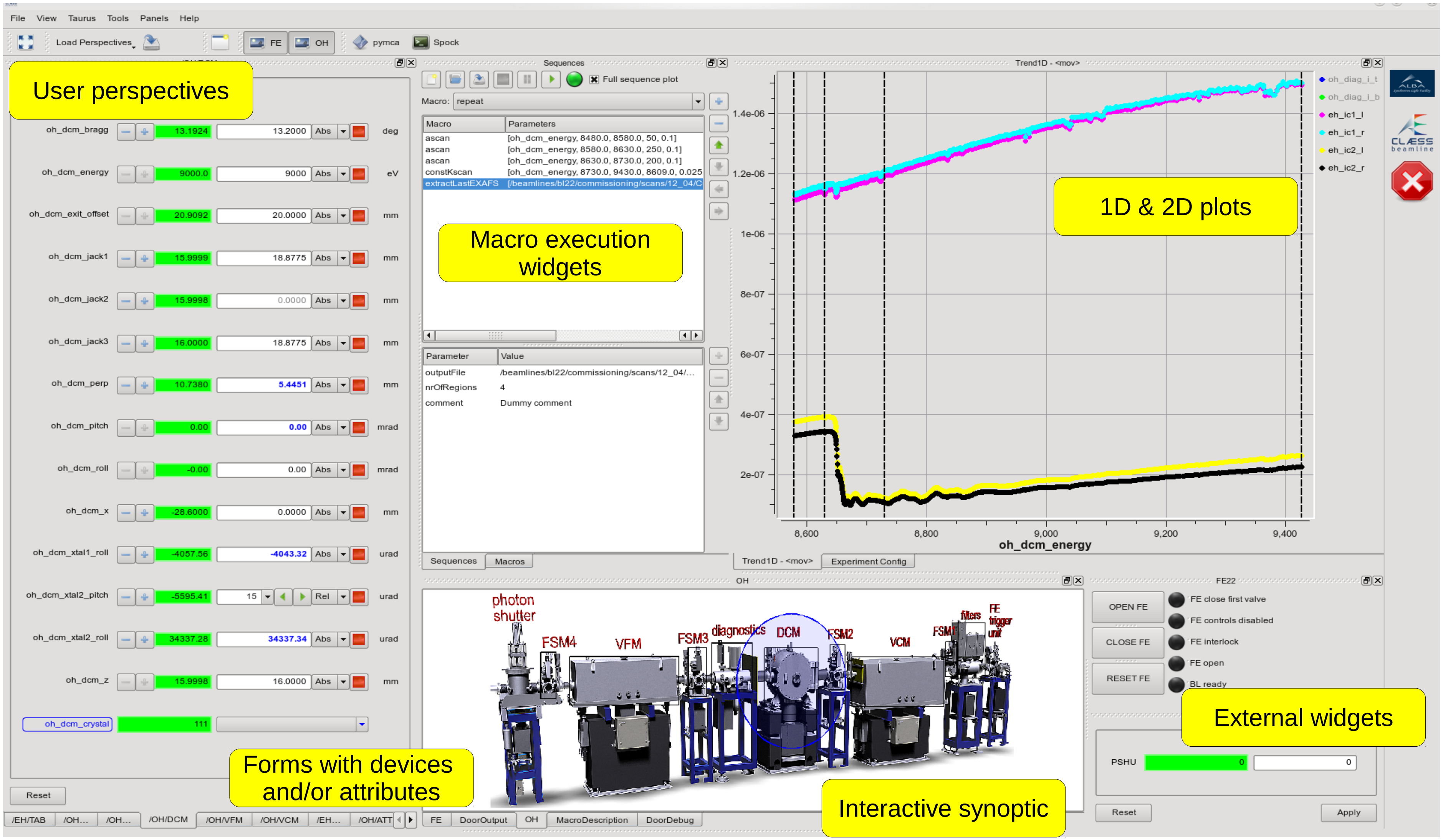
Taurus is a framework for creating **GUI** and **CLI** to interact with control systems or other data sources

Taurus based graphical **widgets** e.g. generic: forms, plots; Sardana specific: macro executor, motor, experiment configuration, scan plots, …

TaurusGUI framework for creating complete GUIs **without programming a single line of code!**

Taurus used **model-view** pattern.
Any data source could be accessed!

SardanaGUI w/o programming



User perspectives

oh_dcm_bragg	13.1924	13.2000	Abs	deg
oh_dcm_energy	9000.0	9000	Abs	eV
oh_dcm_exit_offset	20.9092	20.0000	Abs	mm
oh_dcm_jack1	15.9999	18.8775	Abs	mm
oh_dcm_jack2	15.9998	0.0000	Abs	mm
oh_dcm_jack3	16.0000	18.8775	Abs	mm
oh_dcm_perp	10.7380	5.4451	Abs	mm
oh_dcm_pitch	0.00	0.00	Abs	mrad
oh_dcm_roll	-0.00	0.00	Abs	mrad
oh_dcm_x	-28.6000	0.0000	Abs	mm
oh_dcm_xtal1_roll	-4057.56	-4043.32	Abs	urad
oh_dcm_xtal2_pitch	-5595.41	15	Rel	urad
oh_dcm_xtal2_roll	34337.28	34337.34	Abs	urad
oh_dcm_z	15.9998	16.0000	Abs	mm
oh_dcm_crystal	111			

☒ Full sequence plot

Macro: repeat

Macro	Parameters
ascan	[oh_dcm_energy, 8480.0, 8580.0, 50, 0.1]
ascan	[oh_dcm_energy, 8580.0, 8630.0, 250, 0.1]
ascan	[oh_dcm_energy, 8630.0, 8730.0, 200, 0.1]
constKscan	[oh_dcm_energy, 8730.0, 9430.0, 8609.0, 0.025
extractLastEXAFS	[/beamlines/bl22/commissioning/scans/12_04/C

Parameter	Value
outputFile	/beamlines/bl22/commissioning/scans/12_04/...
nrOfRegions	4
comment	Dummy comment

Macro execution widgets

1D & 2D plots

FE22

● FE close first valve

● FE controls disabled

● FE interlock

● FE open

● BL ready

PSHU 0 0

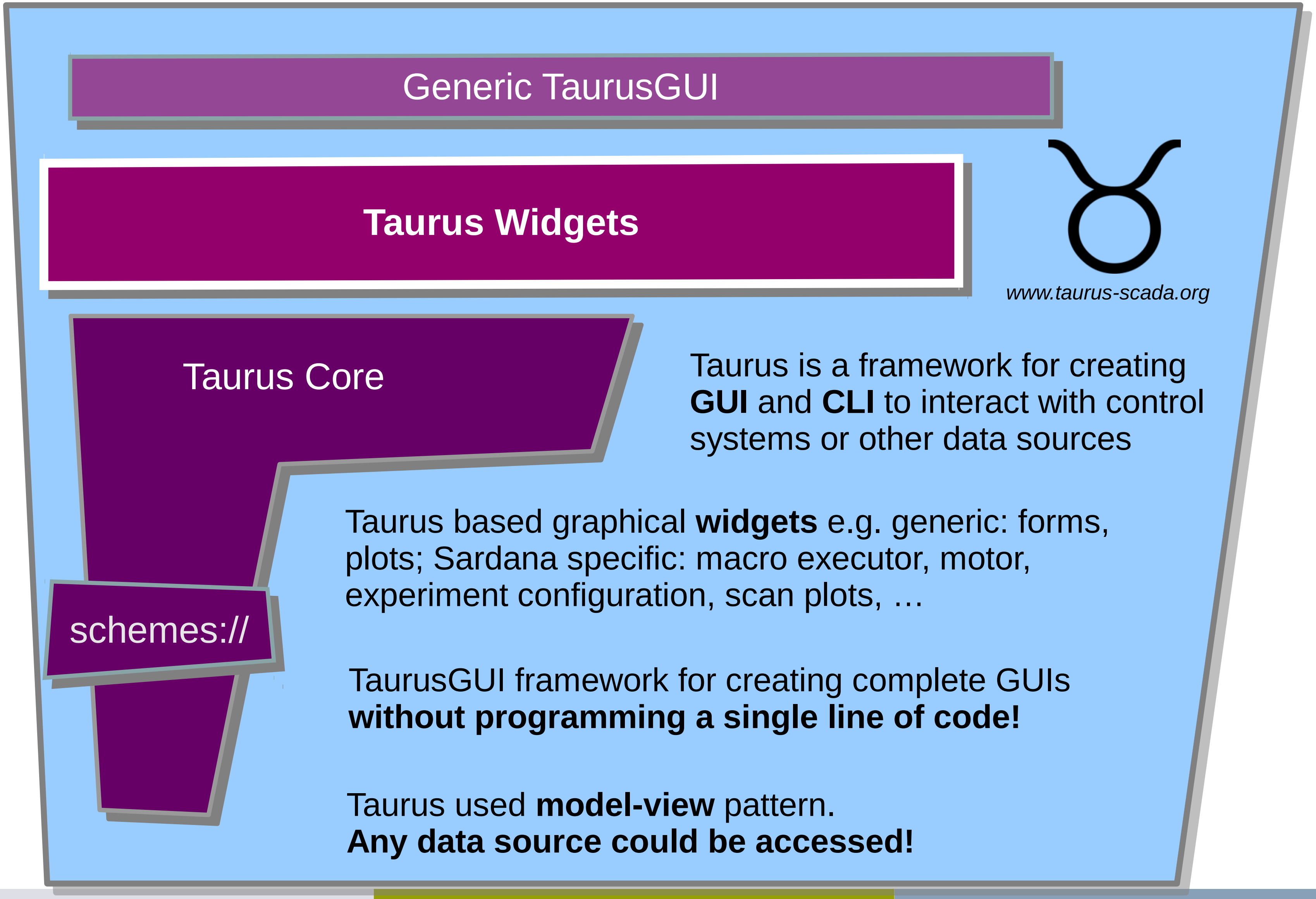
External widgets

Forms with devices and/or attributes

Interactive synoptic

BL22 (ALBA) GUI created with the TaurusGUI framework

Taurus library – client



www.taurus-scada.org

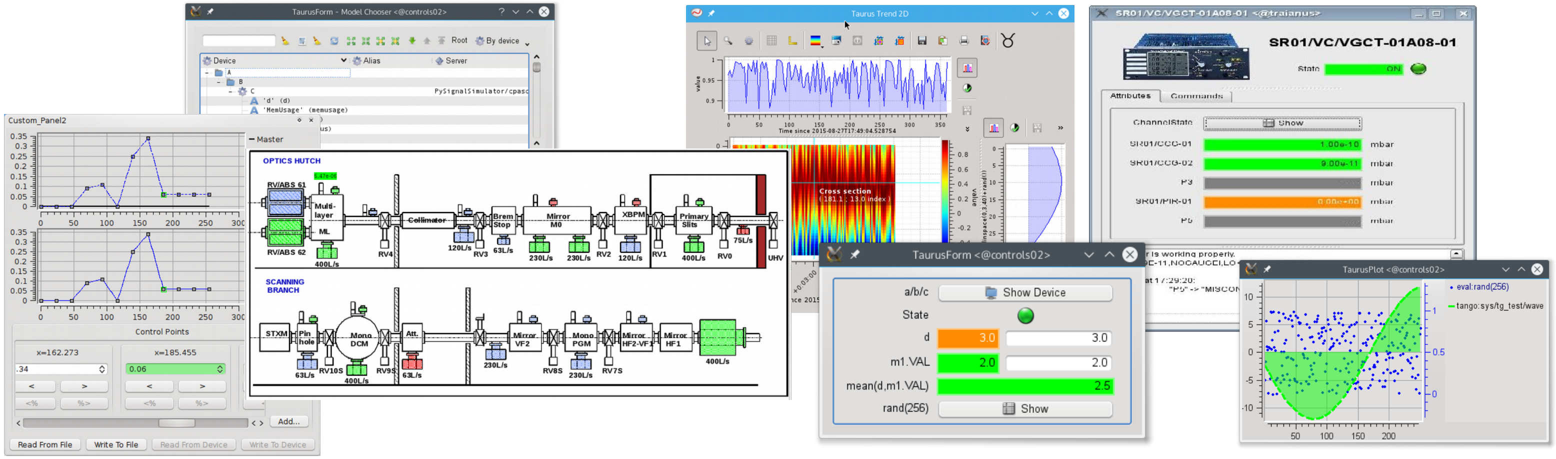
Taurus is a framework for creating **GUI** and **CLI** to interact with control systems or other data sources

Taurus based graphical **widgets** e.g. generic: forms, plots; Sardana specific: macro executor, motor, experiment configuration, scan plots, …

TaurusGUI framework for creating complete GUIs **without programming a single line of code!**

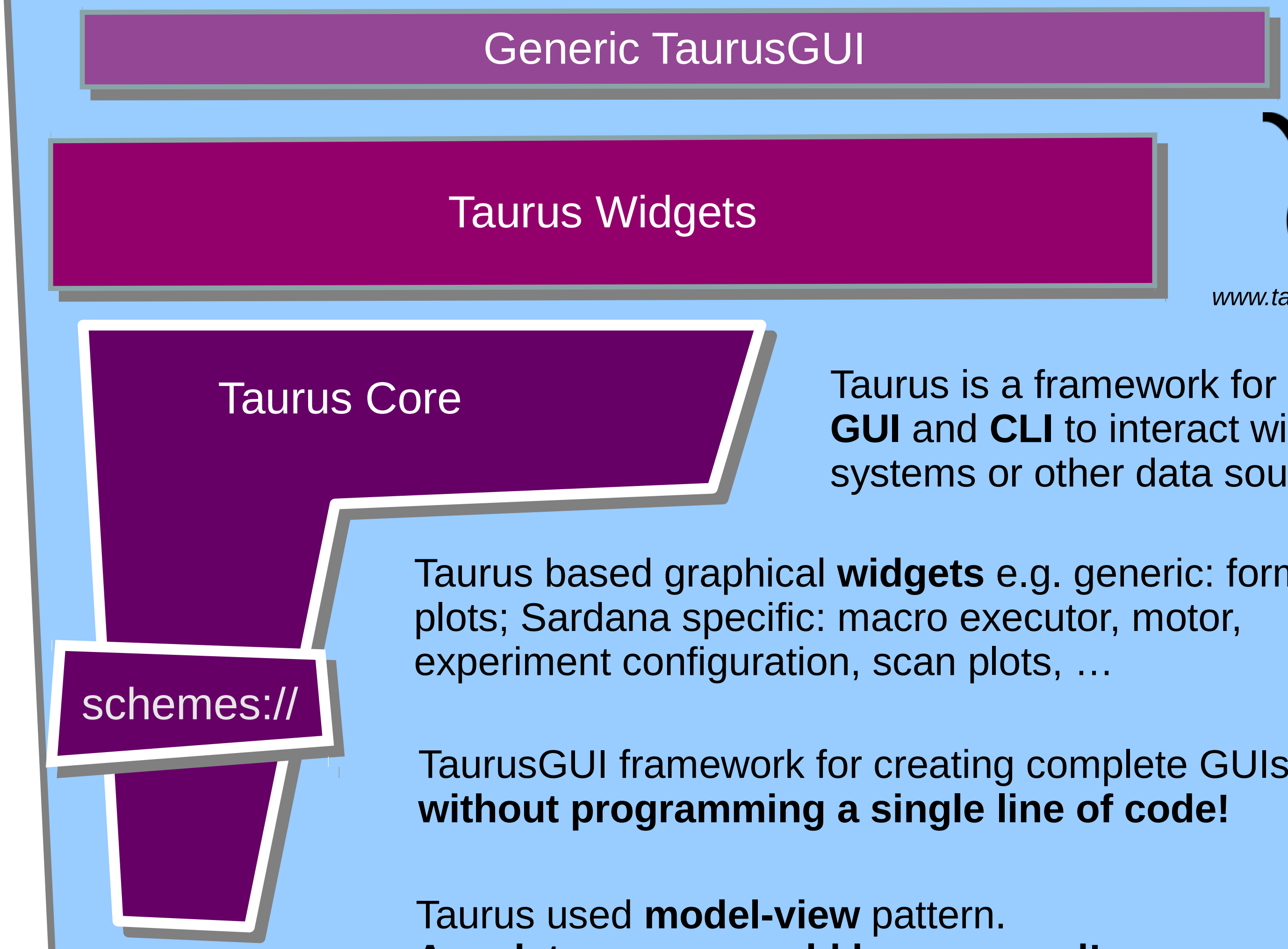
Taurus used **model-view** pattern.
Any data source could be accessed!

Taurus Widgets



- Taurus widgets are implemented by subclassing from the Qt widgets
- Taurus generic widgets: forms, plots, trends, grids…
- Sardana specifc widgets: motor, channel, macroexecutor,...

Taurus library – client



Generic TaurusGUI

Taurus Widgets

Taurus Core

schemes://

www.taurus-scada.org

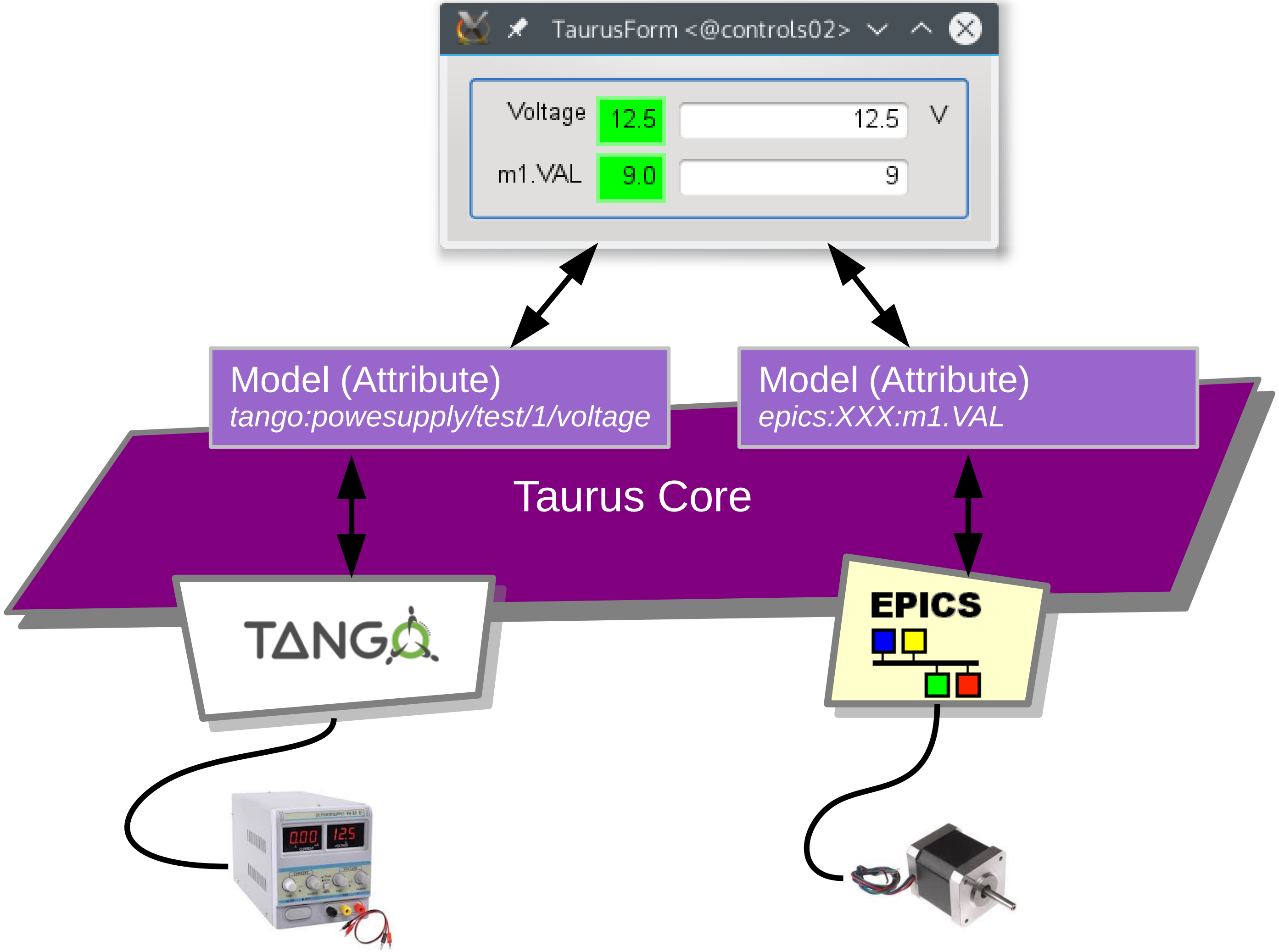
Taurus is a framework for creating **GUI** and **CLI** to interact with control systems or other data sources

Taurus based graphical **widgets** e.g. generic: forms, plots; Sardana specific: macro executor, motor, experiment configuration, scan plots, …

TaurusGUI framework for creating complete GUIs **without programming a single line of code!**

Taurus used **model-view** pattern.
Any data source could be accessed!

Taurus Core



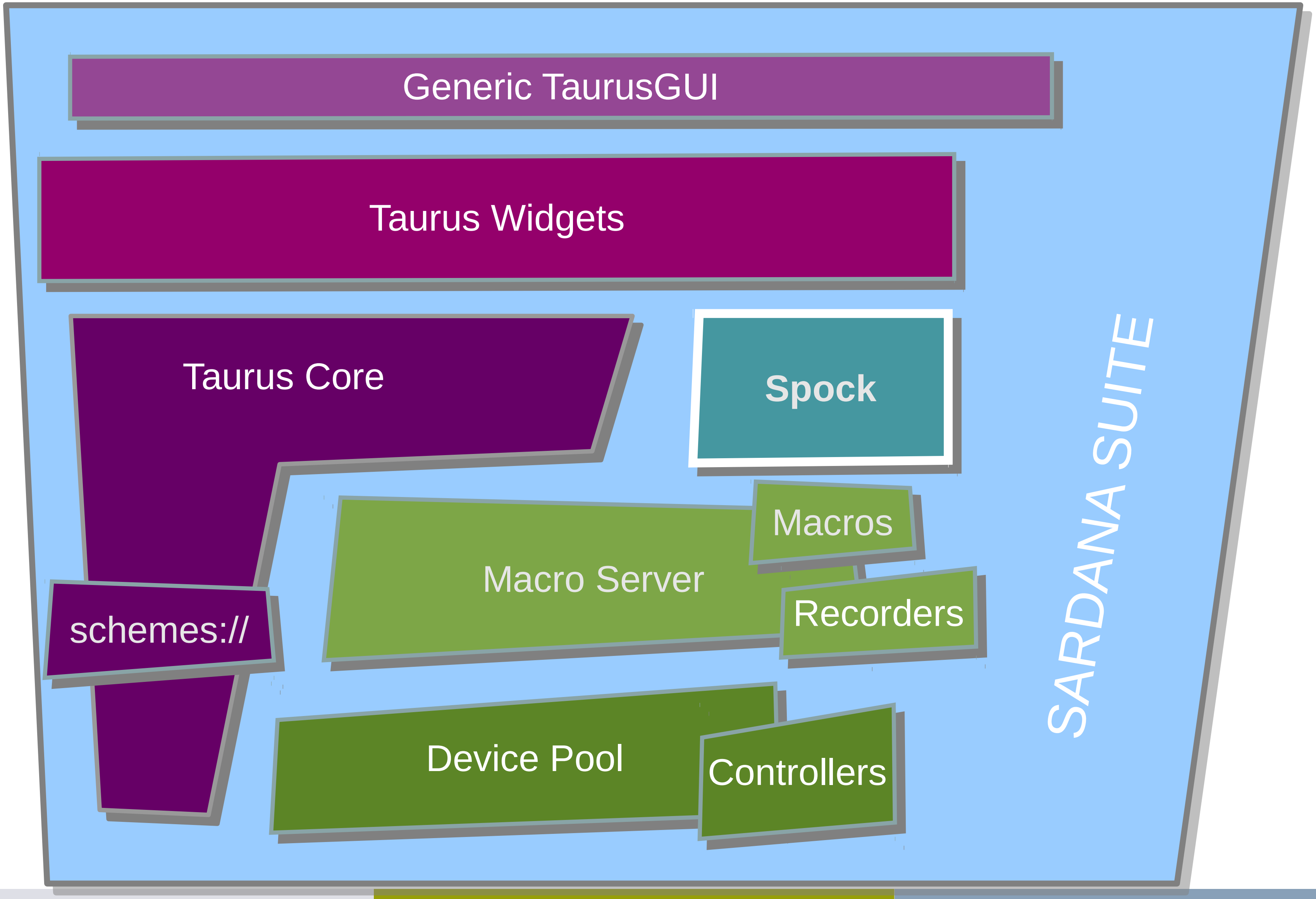
Voltage 12.5 12.5 V

m1.VAL 9.0 9

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

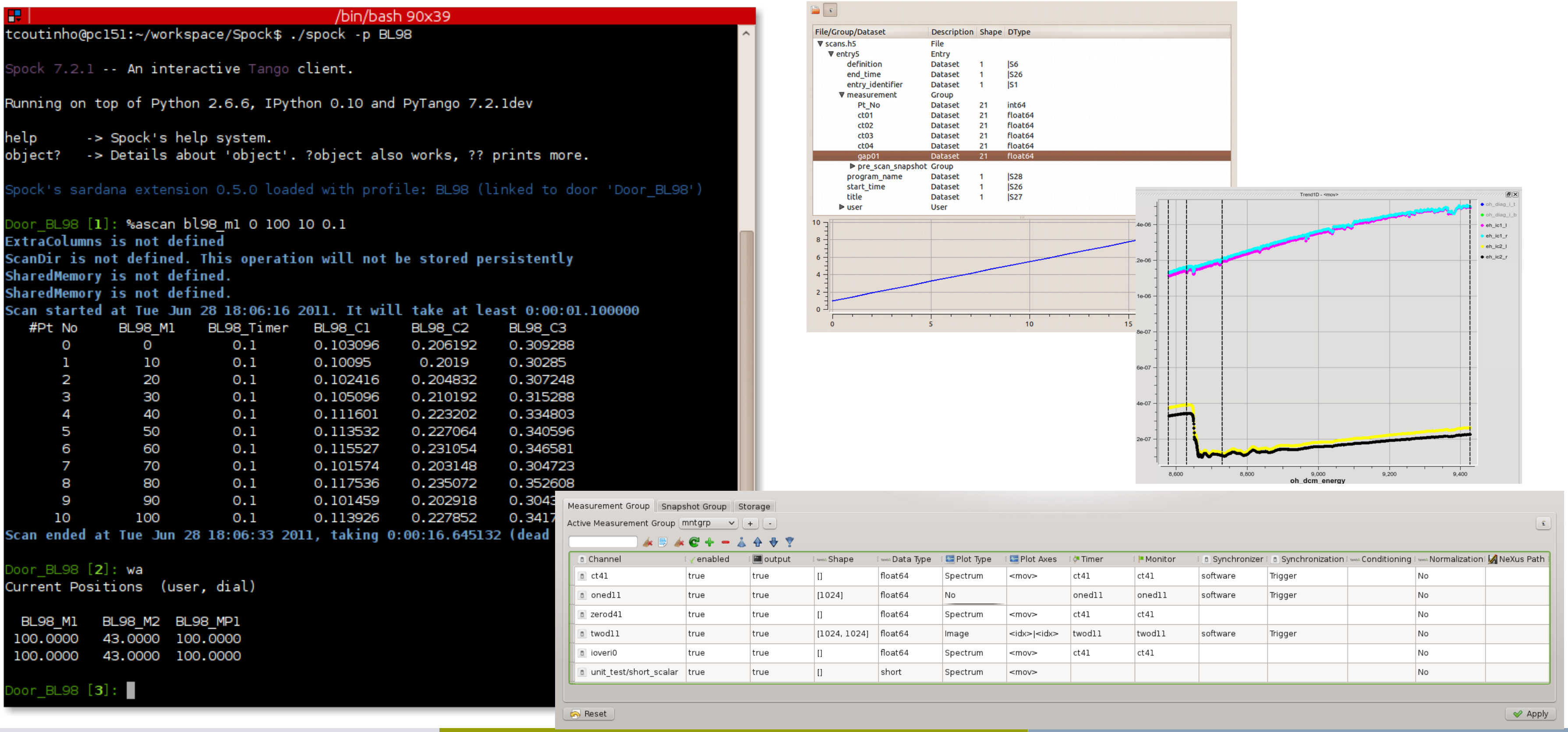
Model (Attribute)
epics:XXX:m1.VAL

Spock – client



Spock – client

Spock – **IPython** based Sardana CLI which syntax mimics **SPEC** commands, provides total control over the system: executes procedures, interacts with the elements, …



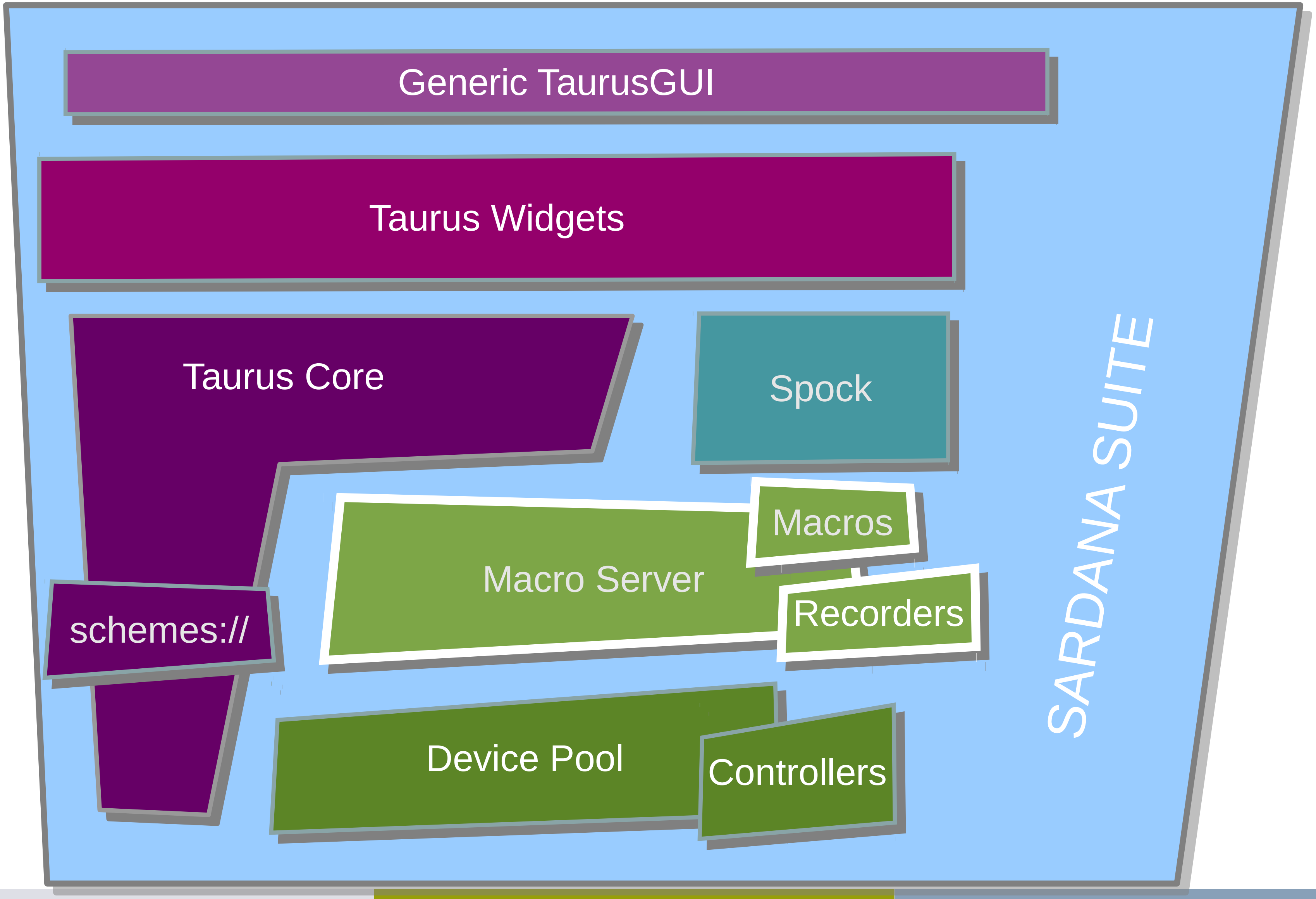
```
tcoutinho@pc151:~/workspace/Spock$ ./spock -p BL98

Spock 7.2.1 -- An interactive Tango client.

Running on top of Python 2.6.6, IPython 0.10 and PyTango 7.2.1dev

help      -> Spock's help system.
object?   -> Details about 'object'. ?object also works, ?? prints more.

Spock's sardana extension 0.5.0 loaded with profile: BL98 (linked to door 'Door_BL98')

Door_BL98 [1]: %ascan bl98_m1 0 100 10 0.1
ExtraColumns is not defined
ScanDir is not defined. This operation will not be stored persistently
SharedMemory is not defined.
SharedMemory is not defined.
Scan started at Tue Jun 28 18:06:16 2011. It will take at least 0:00:01.100000
   #Pt No     BL98_M1    BL98_Timer     BL98_C1      BL98_C2      BL98_C3
       0          0          0.1      0.103096     0.206192     0.309288
       1         10          0.1      0.10095       0.2019      0.30285
       2         20          0.1      0.102416     0.204832     0.307248
       3         30          0.1      0.105096     0.210192     0.315288
       4         40          0.1      0.111601     0.223202     0.334803
       5         50          0.1      0.113532     0.227064     0.340596
       6         60          0.1      0.115527     0.231054     0.346581
       7         70          0.1      0.101574     0.203148     0.304723
       8         80          0.1      0.117536     0.235072     0.352608
       9         90          0.1      0.101459     0.202918     0.3043
      10        100          0.1      0.113926     0.227852     0.3417
Scan ended at Tue Jun 28 18:06:33 2011, taking 0:00:16.645132 (dead

Door_BL98 [2]: wa
Current Positions  (user, dial)

  BL98_M1    BL98_M2   BL98_MP1
 100.0000   43.0000  100.0000
 100.0000   43.0000  100.0000

Door_BL98 [3]: 
```

File/Group/Dataset	Description	Shape	DType
▼ scans.h5	File		
▼ entry5	Entry		
definition	Dataset	1	|S6
end_time	Dataset	1	|S26
entry_identifier	Dataset	1	|S1
▼ measurement	Group		
Pt_No	Dataset	21	int64
ct01	Dataset	21	float64
ct02	Dataset	21	float64
ct03	Dataset	21	float64
ct04	Dataset	21	float64
gap01	Dataset	21	float64
▶ pre_scan_snapshot	Group		
program_name	Dataset	1	|S28
start_time	Dataset	1	|S26
title	Dataset	1	|S27
▶ user	User		

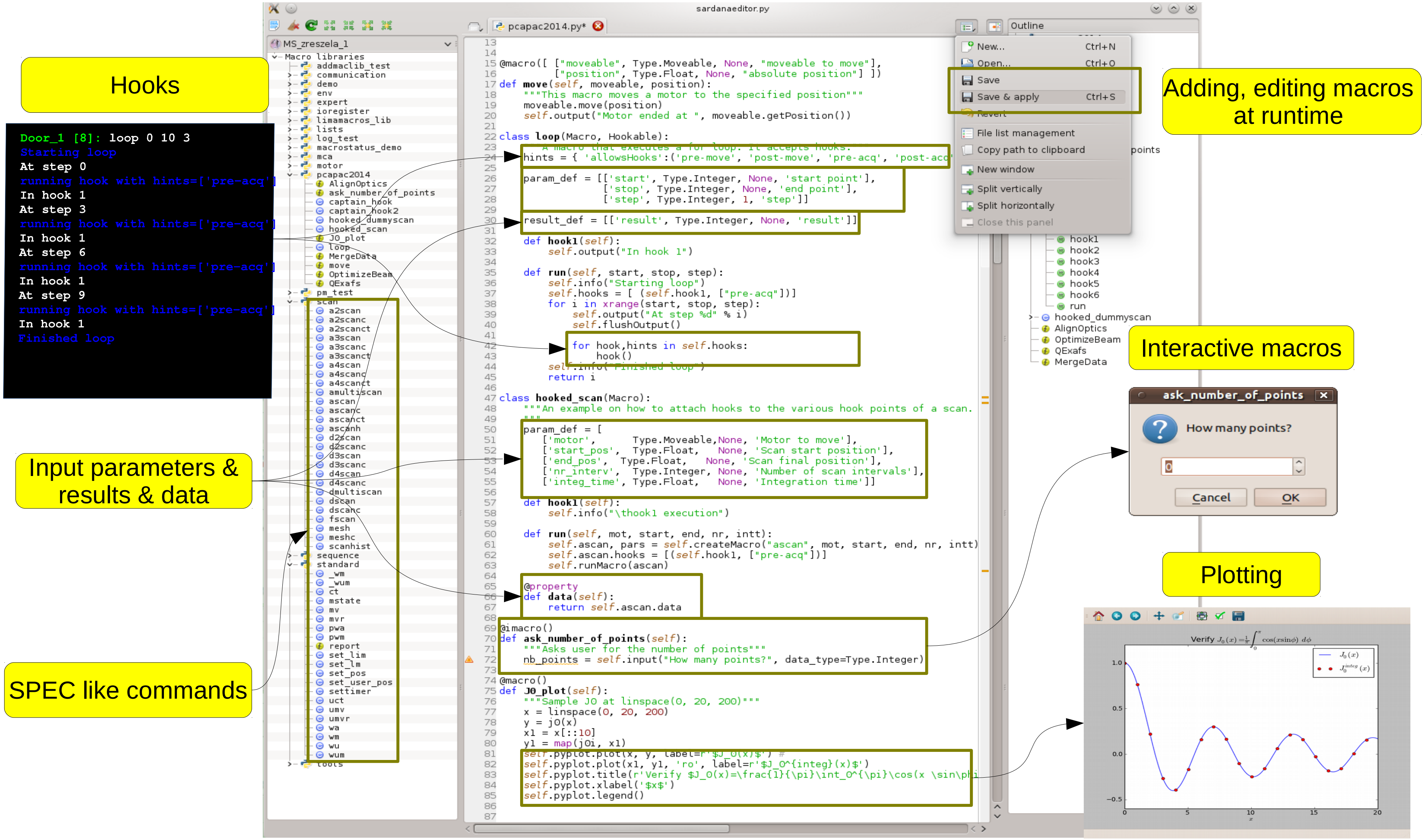
Active Measurement Group mntgrp

Channel	enabled	output	Shape	Data Type	Plot Type	Plot Axes	Timer	Monitor	Synchronizer	Synchronization	Conditioning	Normalization	NeXus Path
ct41	true	true	[]	float64	Spectrum	<mov>	ct41	ct41	software	Trigger		No	
oned11	true	true	[1024]	float64	No		oned11	oned11	software	Trigger		No	
zerod41	true	true	[]	float64	Spectrum	<mov>	ct41	ct41				No	
twod11	true	true	[1024, 1024]	float64	Image	<idx>|<idx>	twod11	twod11	software	Trigger		No	
ioveri0	true	true	[]	float64	Spectrum	<mov>	ct41	ct41				No	
unit_test/short_scalar	true	true	[]	short	Spectrum	<mov>						No	

MacroServer



MacroServer & User Procedures



Hooks

```
Door_1 [8]: loop 0 10 3
Starting loop
At step 0
running hook with hints=['pre-acq']
In hook 1
At step 3
running hook with hints=['pre-acq']
In hook 1
At step 6
running hook with hints=['pre-acq']
In hook 1
At step 9
running hook with hints=['pre-acq']
In hook 1
Finished loop
```

Input parameters & results & data

SPEC like commands

```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

ask_number_of_points

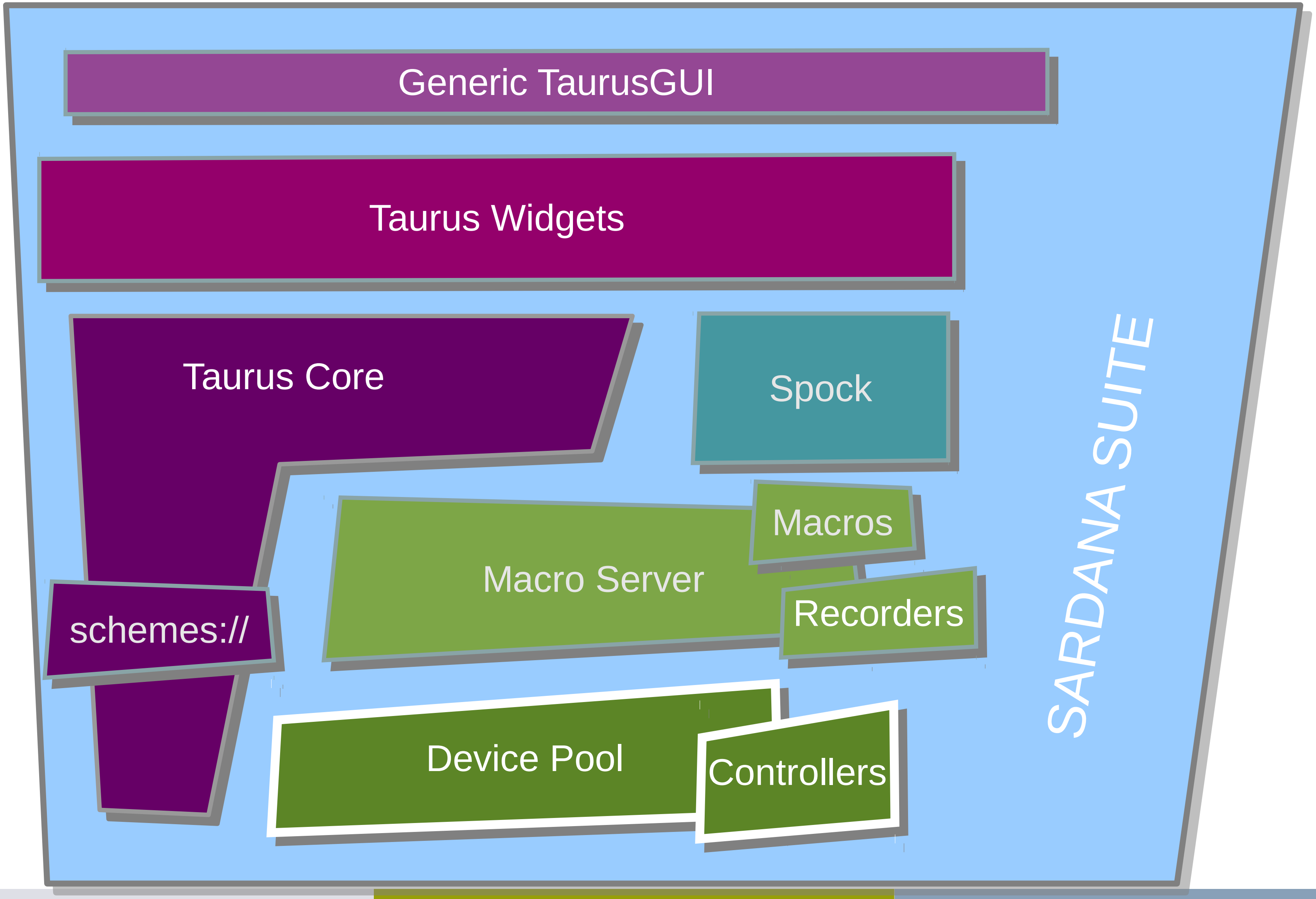
How many points?

0

Plotting

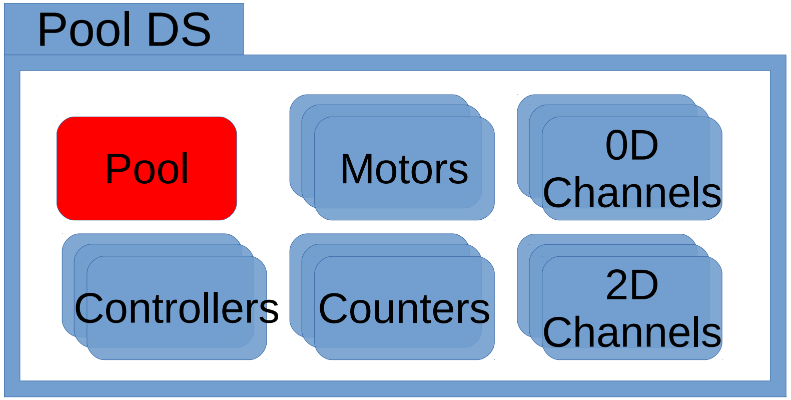
Macro editor with exemplary macros demonstrating advanced macro programming features.

Pool


Generic TaurusGUI
Taurus Widgets
Taurus Core
Spock
Macros
Macro Server
Recorders
schemes://
Device Pool
Controllers
SARDANA SUITE

Pool - Unifies Hardware Access

- All the equipments are interfaced via Pool and its **plug-in** controller classes (**Python**)
- Generic elements' interfaces allow building high level layers on top of them e.g. MeasurementGroup, pseudo elements, generic widgets, etc.

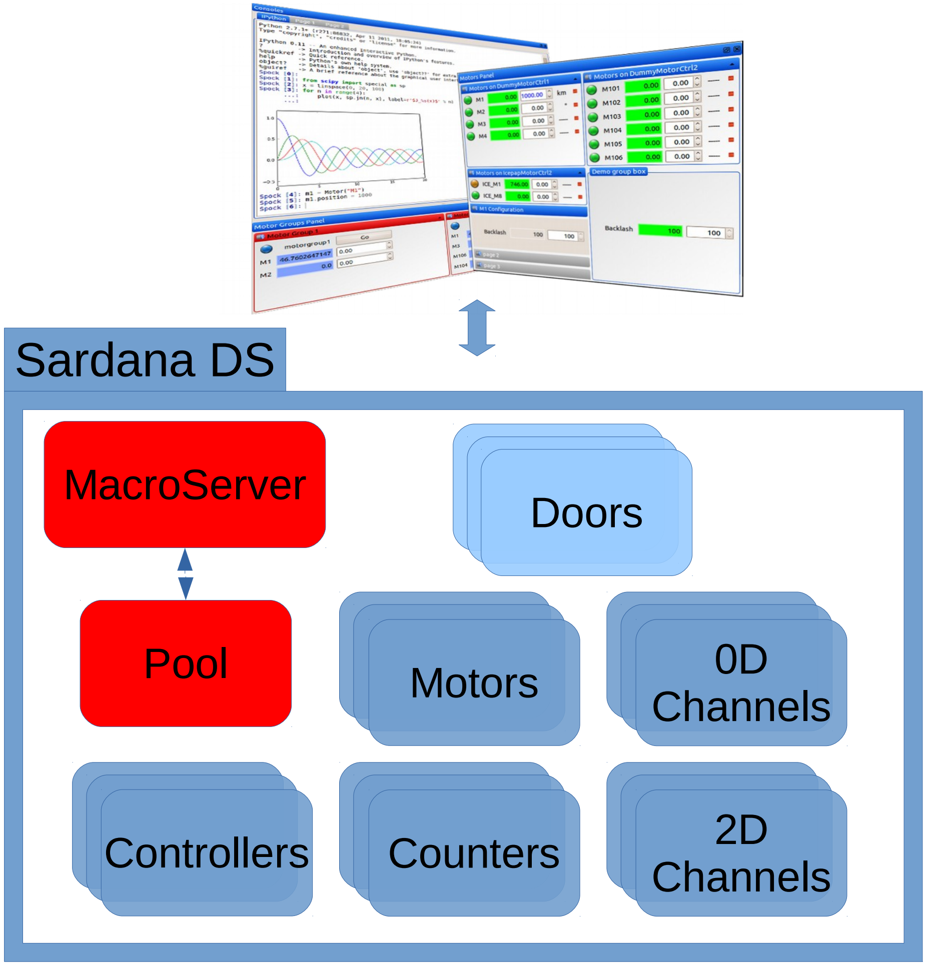


Pool Device Server and its elements

Element Type	*Example of application*
Motor	stepper, servo or piezo actuator
PseudoMotor	energy, HKL of a diffractometer, slit's gap or offset
CounterTimer	event counter, position measurement
PseudoCounter	vertical beam position in the X-ray beam position monitor (XBPM)
0DExpChannel	analog to digital converter (ADC), low current electrometer
1DExpChannel	position sensitive detector (PSD), multichannel analyzer (MCA)
2DExpChannel	CCD camera, 2D X-ray detector

Sardana element types and its examples

Server architectures



Consoles

IPython | Page 1 | Page 2

```
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'$J_%s(x)$' % n)
       ...:
Spock [4]: m1 = Motor("M1")
Spock [5]: m1.position = 1000
Spock [6]:
```

Motors Panel

Motors on DummyMotorCtrl1

M1	0.00	1000.00	km
M2	0.00	0.00	*
M3	0.00	0.00	—
M4	0.00	0.00	—

Motors on IcepapMotorCtrl2

ICE_M1	746.00	0.00	—
ICE_M8	0.00	0.00	—

M1 Configuration

Backlash 100 100

Motors on DummyMotorCtrl2

M101	0.00	0.00	—
M102	0.00	0.00	—
M103	0.00	0.00	—
M104	0.00	0.00	—
M105	0.00	0.00	—
M106	0.00	0.00	—

Demo group box

Backlash 100 100

Motor Groups Panel

Motor Group 1

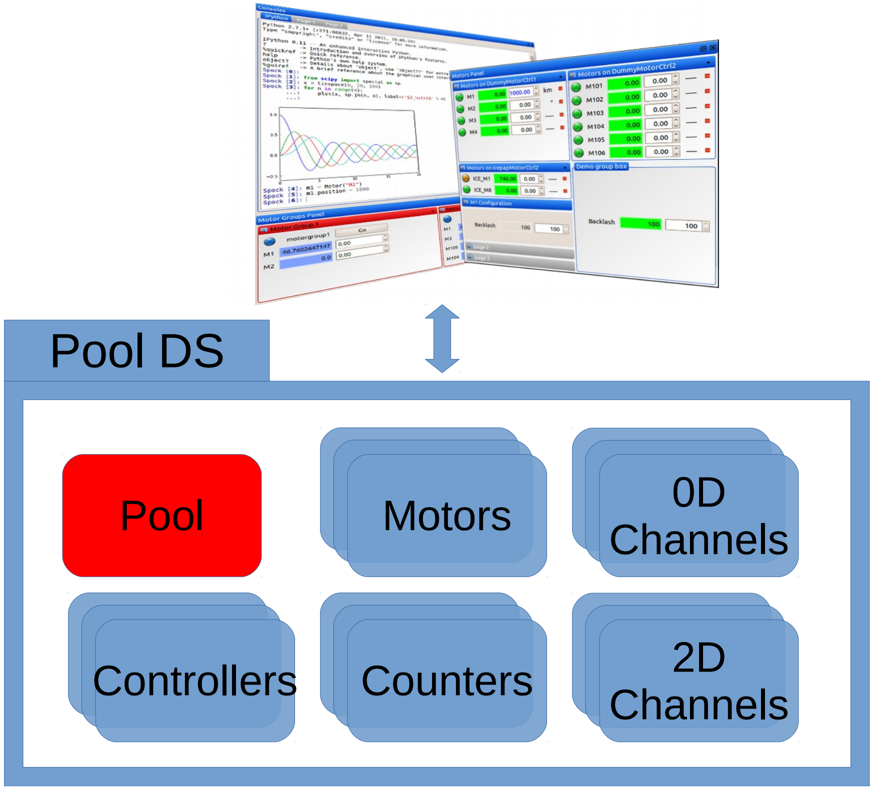
motorgroup1 Go

M1 46.7602647147 0.00

M2 0.0 0.00

A diagram representing a Sardana server and its objects

Server architectures



Pool DS

Pool

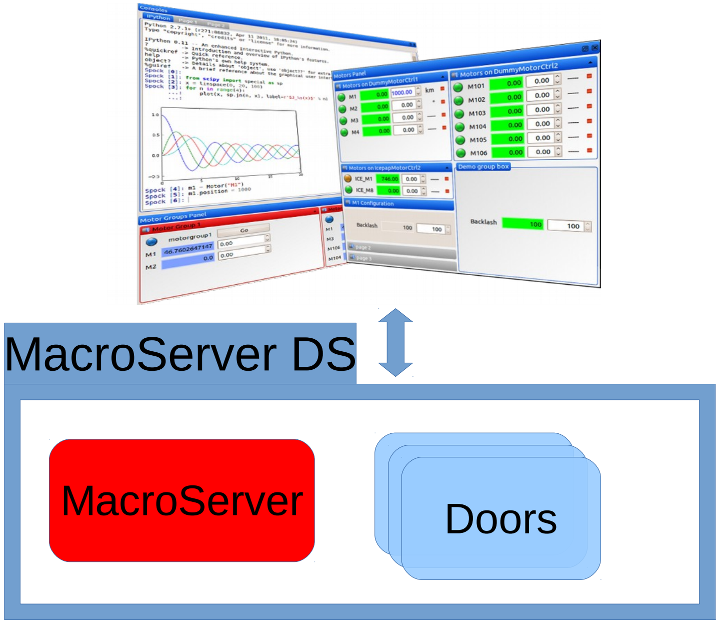
Motors

0D Channels

Controllers

Counters

2D Channels



MacroServer DS

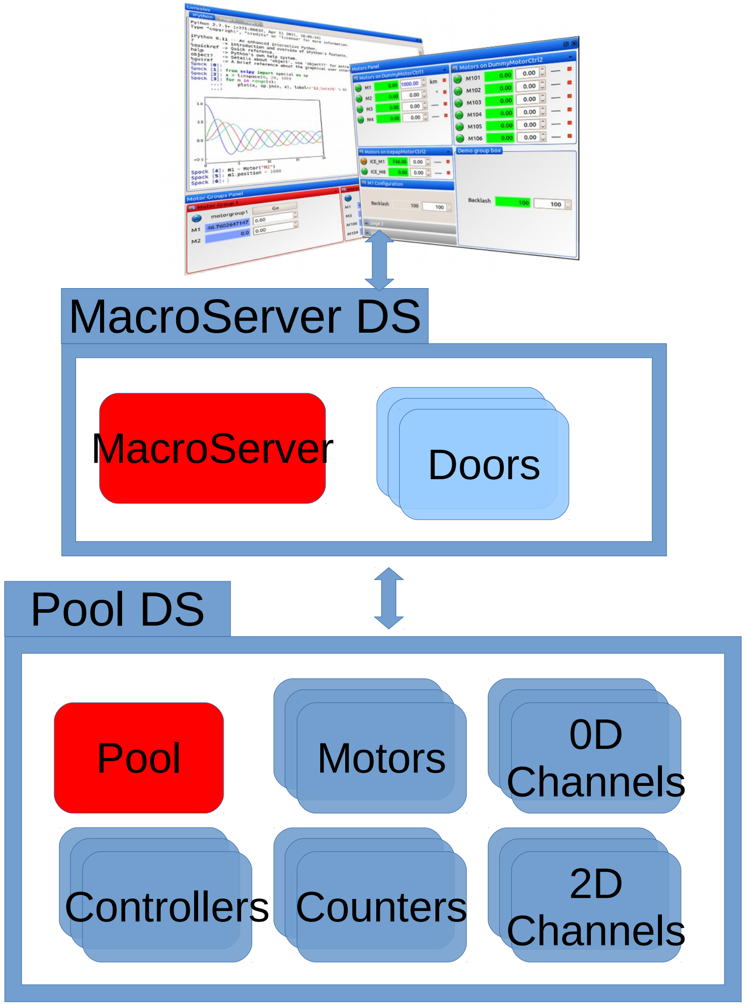
MacroServer

Doors

Sardana configured to be a single MacroServer DS (no Pool present)

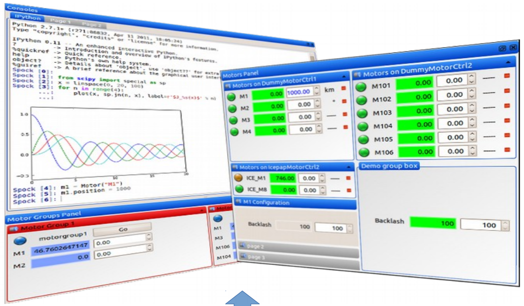
Sardana configured to be a single Pool DS (no MacroServer present)

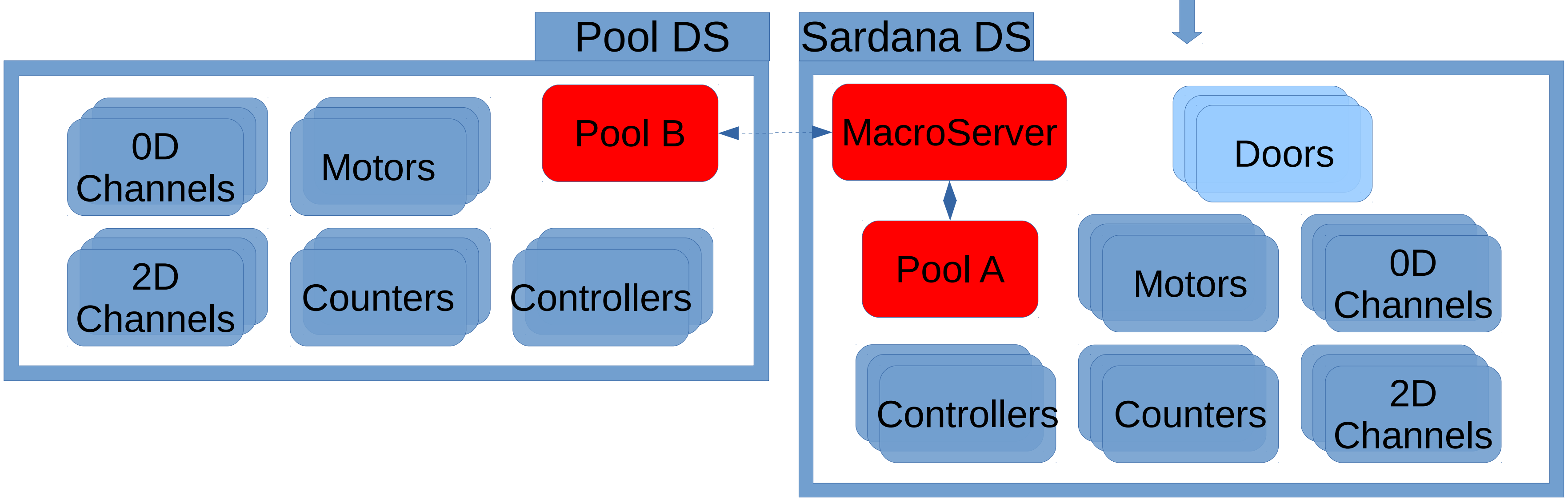
Server architectures



Sardana configured with a MacroServer DS connecting to an underlying Pool DS

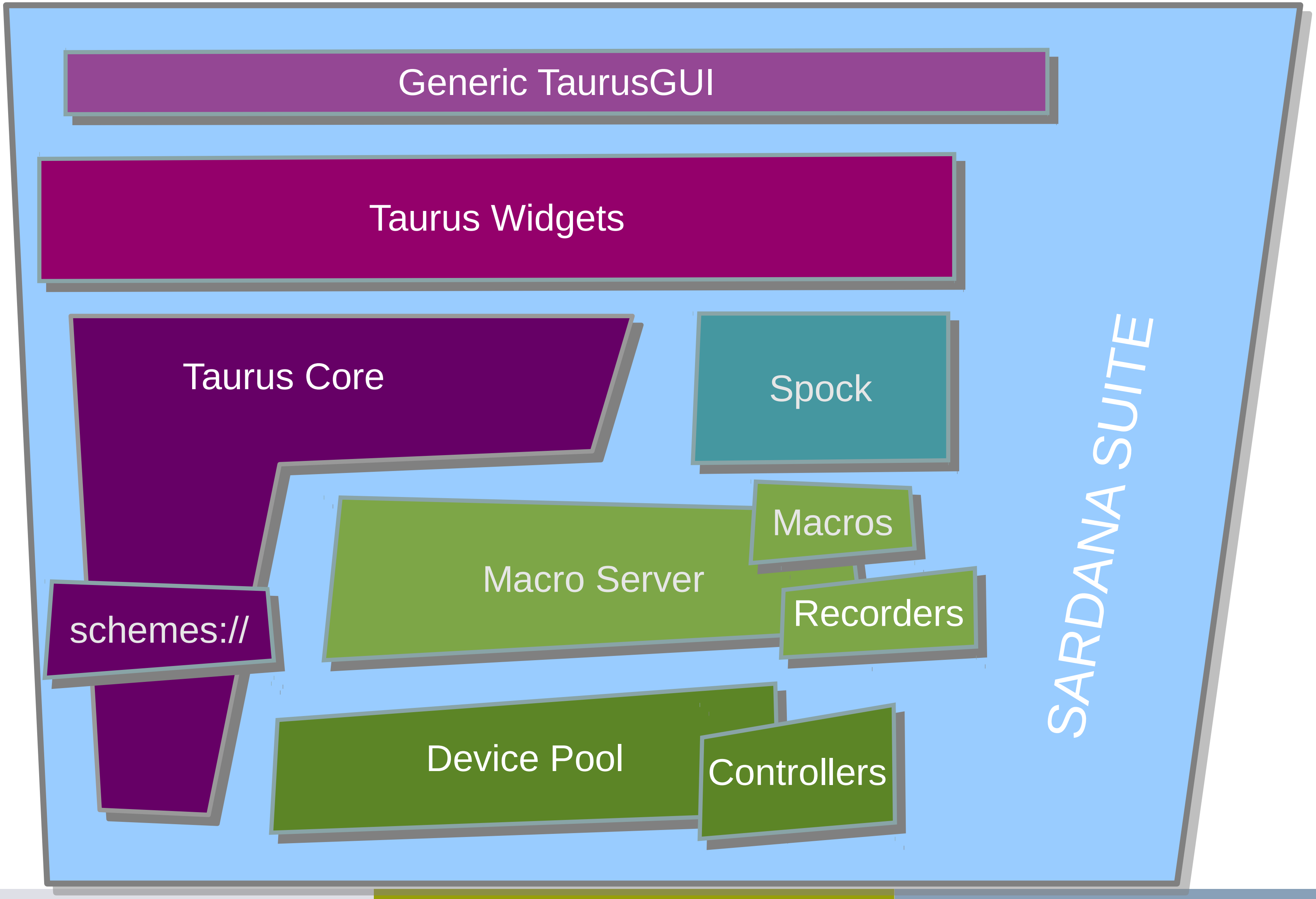
Server architectures





Sardana configured with a Sardna DS connecting to another underlying Pool DS

Sardana Suite – Summary



Thank you for your attention!