



# Sardana: SCADA with Tango Controls

Jordi Aguilar  
Fulvio Becheri  
Roberto Homs  
Zbigniew Reszela  
Oriol Vallcorba

*Tango Controls Workshop 2025*  
*September 20, 2025*  
*ICALEPCS 2025 Chicago*

# Index

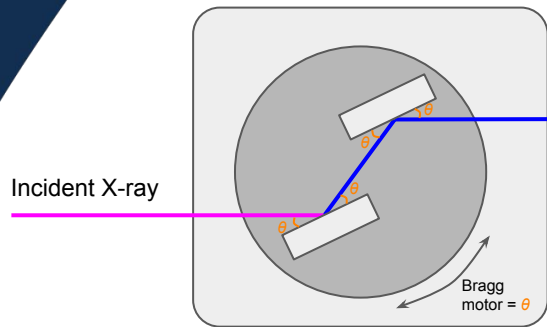


- Motivation
- Introduction to Sardana
- Sardana installation and simulated environment (sar\_demo)
- Measurement group and scans
- Tango controllers
- Extras

# Motivation

# Beamline User Case

## Double Crystal Monochromator



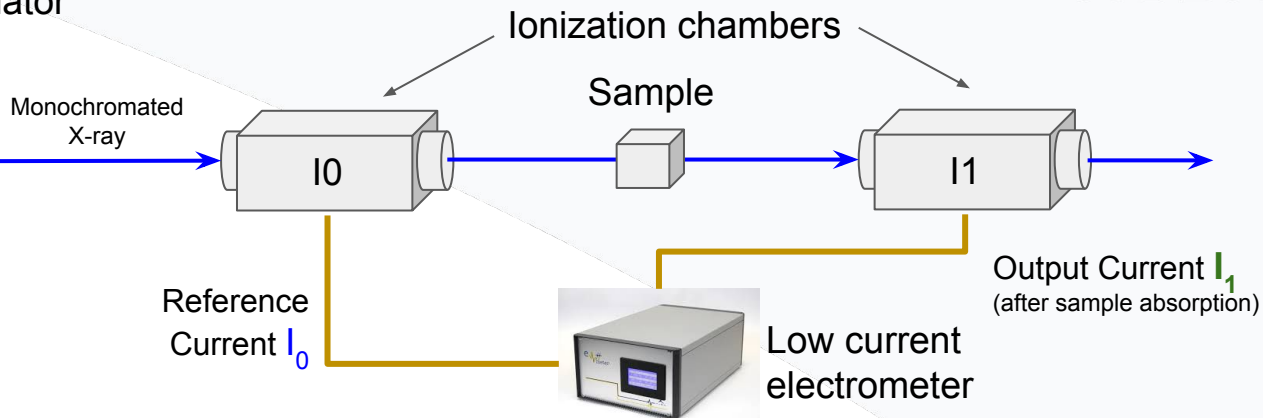
Physical motor = Bragg =  $\theta$

$\Downarrow$   
 Bragg law:  $n\lambda = 2d \sin\theta$   
 Plank eq:  $E = h \times c / \lambda$

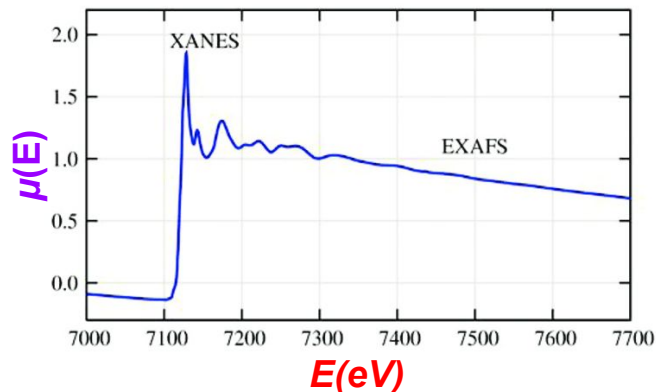
Pseudo motor = Energy =  $E$  (eV)

### Goal:

We need to scan the **Energy** and read two **currents** in an orchestrated way. This involves moving the motor related non-linearly with the energy and operate with both read currents to calculate the absorption coefficient and plot the Absorption Spectrum.

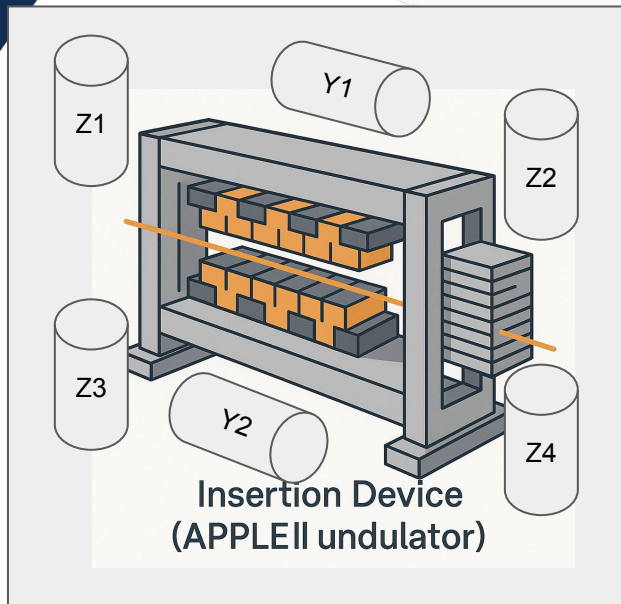


$$\mu(E) = \log(I_0/I_1) \text{ (absorption coefficient)}$$



# Accelerator User Case

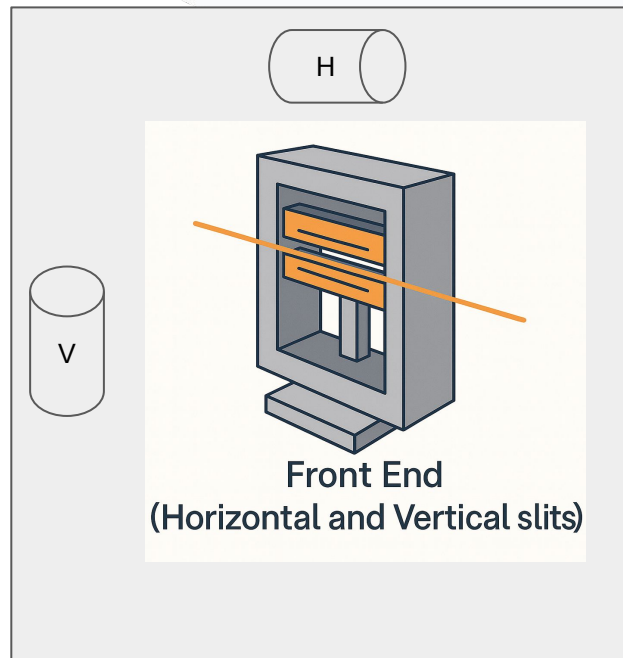
For different (Energy, Polarization) pairs:  
Mesh Scan of FE\_H and FE\_V



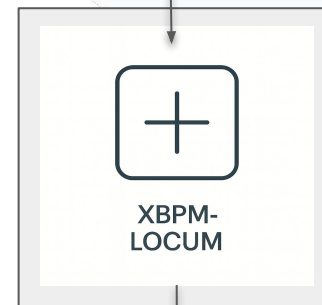
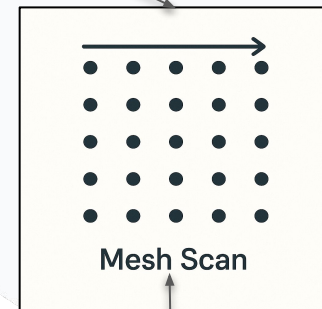
PseudoMotors	(Gap, Phase) => Energy (Gap, Phase) => Polarization
--------------	--

PseudoMotors	Gap = $(Z1+Z2)/2 - (Z3+Z4)/2$ Phase = $Y1 - Y2$
--------------	--

Real Motors	Z1, Z2,Z3,Z4,Y1,Y2
-------------	--------------------



Real Motors	FE_H, FE_V
-------------	------------

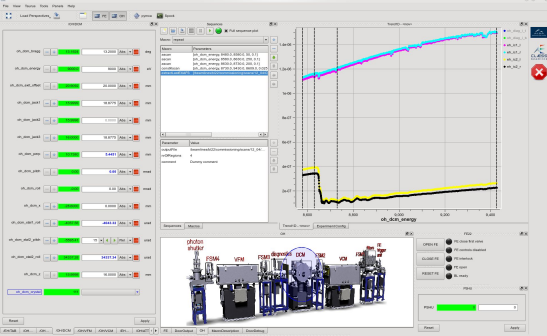


**Beam Positioning**

# Introduction to Sardana

**Sardana - Scientific SCADA Suite**  
Built on top of Tango Control System





Taurus based GUIs

```

User_zrzsze1 | 13 | lsneas
Active      Name      Timer Experi channels
-----
ng_odedtest oned01 oned01
mntgrp01   ct01 ct01, ct02, ct03, ct04
mntgrp02   ct01 ct01, ct02
mntgrp03   ct01 ct01, ct02, ct03, ct04, oned01

User_zrzsze1 | 14 | lsm
Name      Type      Controller  Axis
-----
gsp01     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

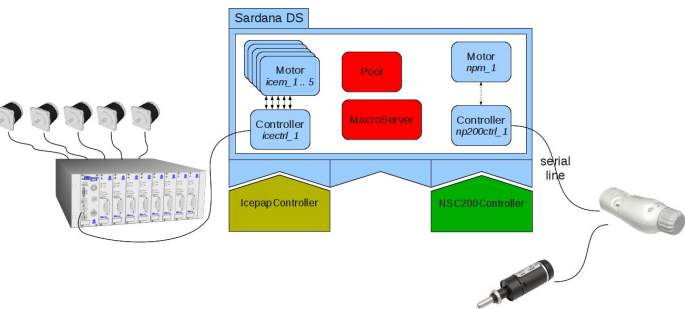
User_zrzsze1 | 15 | nscan mot01 0 1 4 0.1
Operation will be saved in /home/zrzsze1/tmp/test_h5 (w5)
Scan #329 started at Sun Oct 12 13:43:27 2014. It will take at least 0:00:00.694422
Moving to start positions...
#Fit No  mot01    ct01    ct02    ct03    ct04    dt
0      0      0.1    0.2    0.3    0.4    0.085824
1      0.25  0.1    0.2    0.3    0.4    0.248444
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/zrzsze1/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%)

```

Spock - IPython based CLI

**Sardana - Scientific SCADA Suite**  
 Built on top of Tango Control System  
 100% Python  
 Four pillars extendable with plugins  
 Suite = Sardana & Taurus projects

Device Pool - access to the hardware



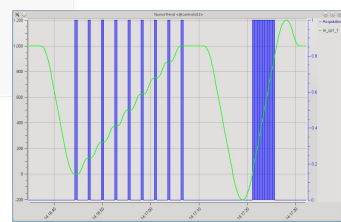
MacroServer - powerful sequencer

```

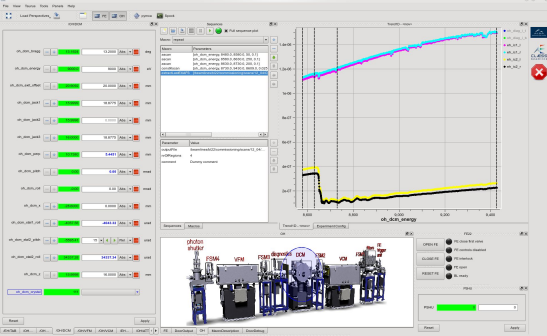
from sardana.macroserver.macro import macro

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

```







Taurus based GUIs



```

Active Name Timer Experi. channels
ng_odedtest oned01 oned01
mntgrp01 ct01 ct01, ct02, ct03, ct04
mntgrp02 ct01 ct01, ct02
mntgrp03 ct01 ct01, ct02, ct03, ct04, oned01

User_zrszszala | 144 | 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
soprotec1 Motor soprotec_ctrl 1

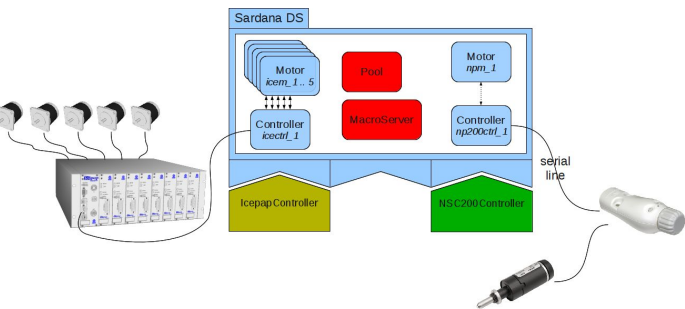
User_zrszszala | 145 | nscan mot01 0 1 4 0.1
Operation will be saved in /home/zrszszala/tmp/test_h5 (w5)
Scan #329 started at Sun Oct 12 13:43:27 2014. It will take at least 0:00:00.694422
Moving to start positions...
#Fit No mot01 ct01 ct02 ct03 ct04 dt
0 0 0.1 0.2 0.3 0.4 0.085824
1 0.25 0.1 0.2 0.3 0.4 0.248444
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/zrszszala/tmp/test_h5 (w5)
Scan #329 ended at Sun Oct 12 13:43:28 2014, taking 0:00:00.845693,Dead time 40.9%
(tauton dead time 29.5%)
  
```

Spock - IPython based CLI

**Sardana - Scientific SCADA Suite**  
 Built on top of Tango Control System  
 100% Python  
 Four pillars extendable with plugins  
 Suite = Sardana & Taurus projects  
 Community of users and developers

Device Pool - access to the hardware

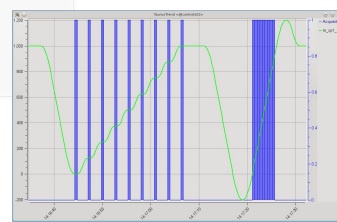


MacroServer - powerful sequencer

```

from sardana.macroserver.macro import macro

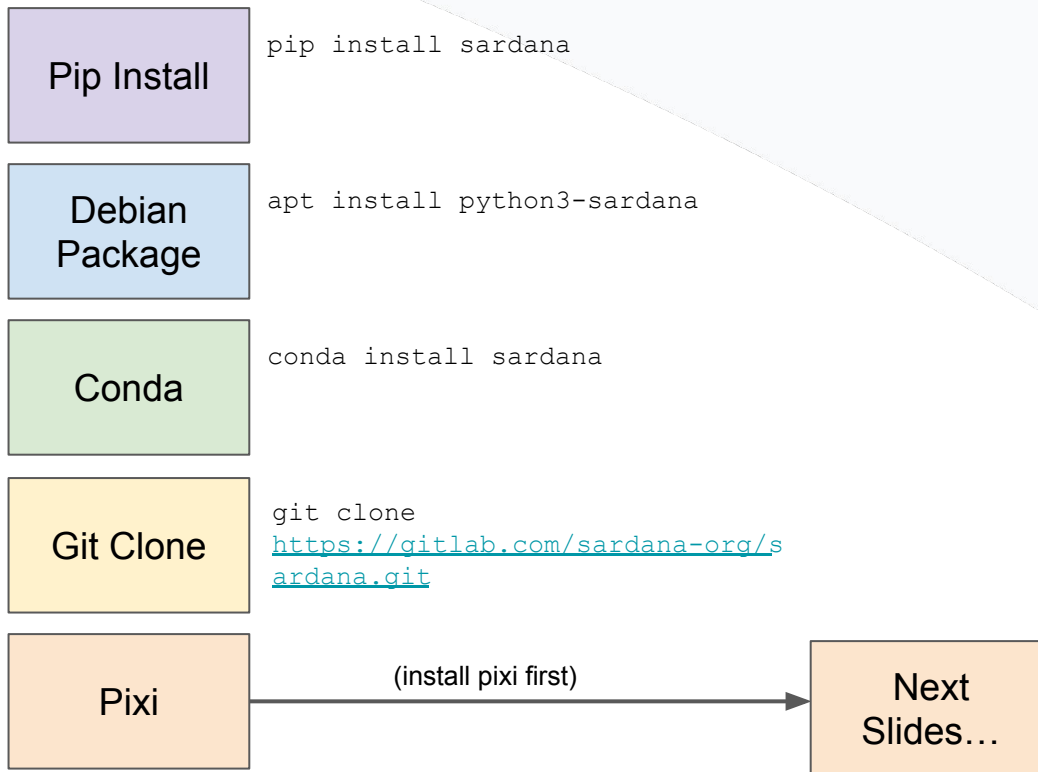
@macro()
def hello_world(self):
    """This is a hello world macro"""
    self.output("Hello, World!")
  
```



# Sardana installation

# Installation

(from [https://www.sardana-controls.org/users/getting\\_started/installing.html](https://www.sardana-controls.org/users/getting_started/installing.html))



# Prerequisites: Tango + TangoTest

1) Clone the repo with `pixi.toml` configuration for the demo:

```
$ git clone https://gitlab.com/alba-synchrotron/controls-section/icalepcs2025-workshop.git
```

2) Create a `sqlite` `TangoDB` and database `DS` (python version)

```
$ pixi run pydb
```

```
✨ Pixi task (pydb in pydb): PyDatabases 2  
Ready to accept request
```

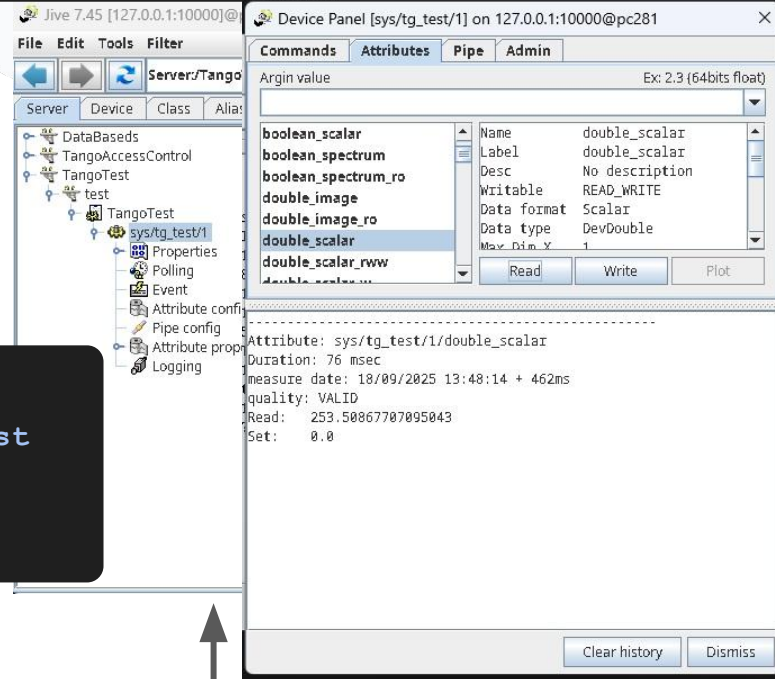
3) Run `TangoTest` device server test instance

```
$ pixi run start_tangotest
```

```
✨ Pixi task (start_tangotest in default): TangoTest test  
...  
Ready to accept request
```

4) Run `jive` to check `DB`

```
$ pixi run jive
```



The screenshot shows a Jive 7.45 terminal window on the left and a Device Panel window on the right. The Device Panel window displays a list of attributes for the device `sys/tg_test/1`. The attributes listed are:

Attribute Name	Value
<code>boolean_scalar</code>	<code>double_scalar</code>
<code>boolean_spectrum</code>	<code>double_scalar</code>
<code>boolean_spectrum_ro</code>	No description
<code>double_image</code>	Writable READ_WRITE
<code>double_image_ro</code>	Data format Scalar
<code>double_scalar</code>	Data type DevDouble
<code>double_scalar_rww</code>	Max Dim Y 1

Below the attribute list, the terminal output shows the following information for the `double_scalar` attribute:

```
Attribute: sys/tg_test/1/double_scalar  
Duration: 76 msec  
measure date: 18/09/2025 13:48:14 + 462ms  
quality: VALID  
Read: 253.50867707095043  
Set: 0.0
```

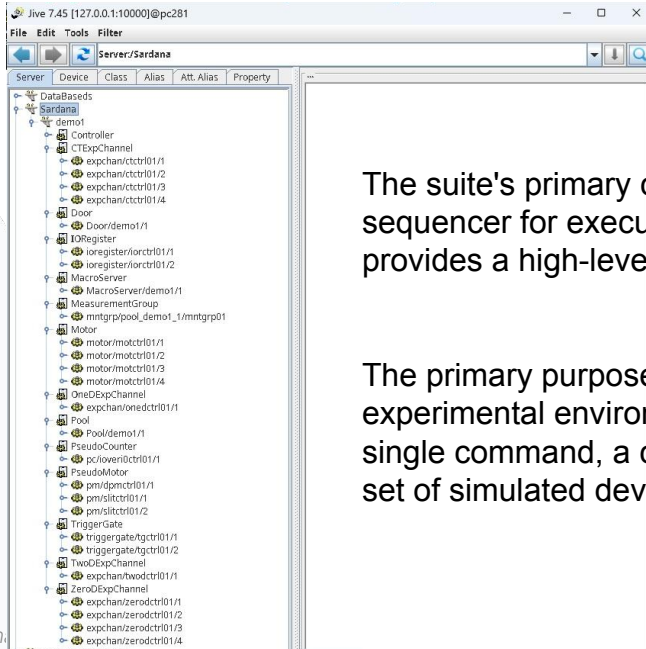
# Simulated environment: sar\_demo elements

```
$ pixi run create_sar_demo
```

```
🌟 Pixi task (create_sar_demo in default): sardanactl config load --write sar_demo.yaml
```

```
...
```

*This command populates the tangodb with a Sardana instance and elements for demo purposes (called sar\_demo)*



The suite's primary components include the **MacroServer**, a sophisticated sequencer for executing complex experimental procedures; the **Device Pool**, which provides a high-level interface for accessing and controlling hardware

The primary purpose of the **sar\_demo macro** is to instantiate a simulated experimental environment for testing and demonstration purposes. By executing this single command, a complete virtual laboratory is created, complete with a diverse set of simulated devices, without the need for any physical hardware.

# Start Sardana server

```
$ pixi run start_sardana
✨ Pixi task (start_sardana in default): Sardana demo1
...
```

- *This command starts the Sardana server, containing Pool and MacroServer (they can also be executed as 2 different processes)*
- *If demo1 instance does not exist in the Tango database, the script will ask for its creation (but since we populated it before with sar\_demo it is already there)*

# **User interaction with Sardana**

## Sardana clients (GUI, CLI)

# Spock: Command Line Interface



```
$ pixi run spock
Profile 'spockdoor' does not exist. Do you want to create one now ([y]/n)?
y
Available Door devices from 127.0.0.1:10000 :
Door_demo1 1 (a.k.a. Door/demo1/1) (running)
Door name from the list? Door_demo1_1
..
Spock 3.6.0 -- An interactive laboratory application.

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

IPython profile: spockdoor

Connected to Door_demo1_1

Door_demo1_1 [1]: lsm
-----
      Name      Type      Controller  Axis
-----
discretepm01   PseudoMotor  dpmctrl01    1
gap01          PseudoMotor  slitctrl01   1
mot01          Motor        motctrl01    1
mot02          Motor        motctrl01    2
mot03          Motor        motctrl01    3
mot04          Motor        motctrl01    4
offset01       PseudoMotor  slitctrl01   2

Door_demo1_1 [2]:
```

IPython based CLI.

Uses ipython profiles. Creates one if none exists.

Connects to the MacroServer through **Door** devices.

Can execute **macros** (e.g. lsm is a build-in macro to list motors)



# Taurus for GUIs

*Start a form with a motor widget*

```
$ pixi run taurus form mot01
```



*Plot and monitor the position of a motor*

```
$ pixi run taurus trend mot01/position
```



# Measurement Group and Scans

# Orchestrating the Experiment



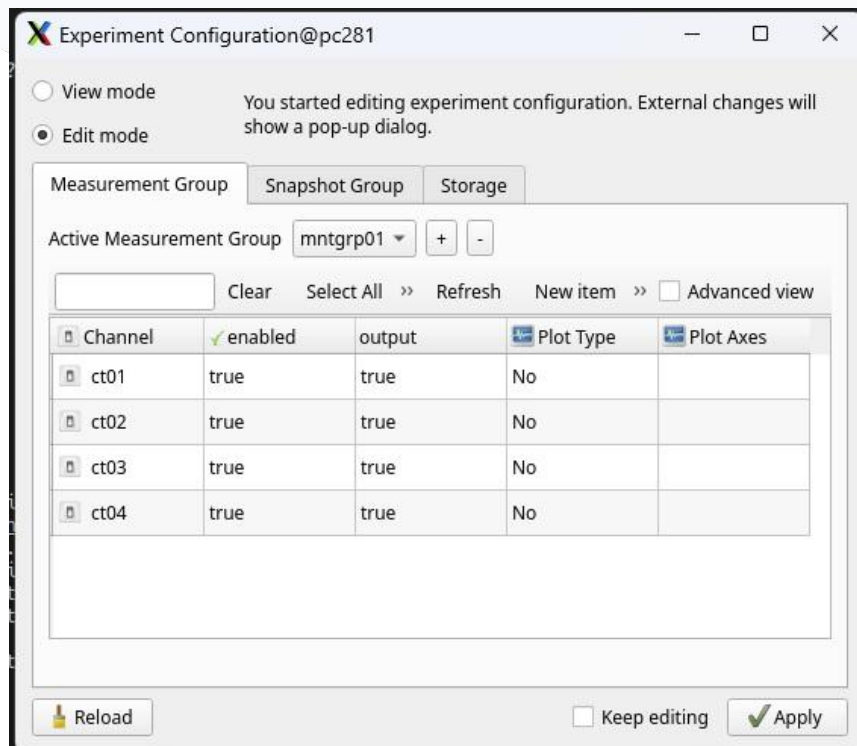
With Sardana elements defined and linked to the Tango device attributes, we can now orchestrate a full experiment. The Sardana MacroServer is a powerful sequencer for running complex operations and custom macros, including step and continuous scans.

- 1. Define a Measurement Group:** Configure the channels that will be used for data acquisition during the experiment.
- 2. Execute a Scan:** Use built-in macros, such as `ascan`, to automatically coordinate the motion of a motor with data acquisition.
- 3. Visualize and Store Data:** The data can be displayed in real time using tools like `showscan` and `taurustrend`, and permanently stored in files (e.g., HDF5 format).

# Measurement groups

Open experiment configuration GUI from spock

```
$ pixi run spock
...
Door_demo1_1 [1]: expconf
```



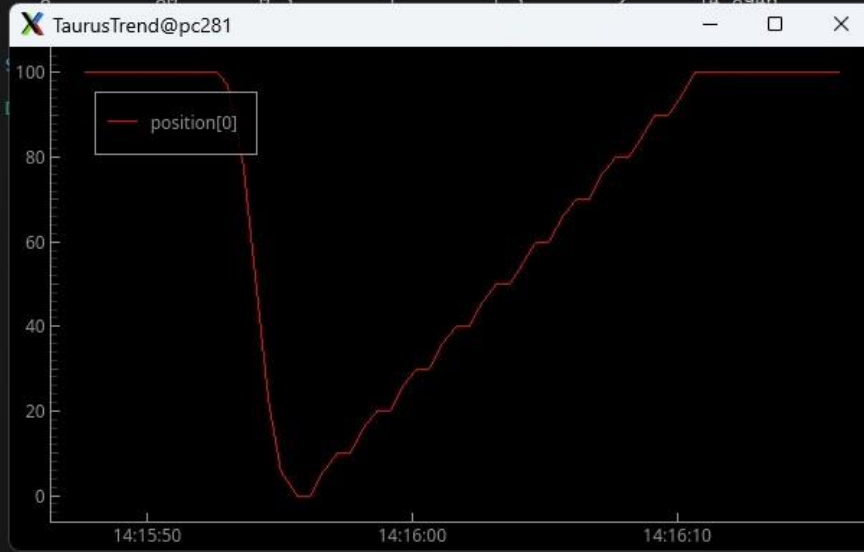
The screenshot shows the 'Experiment Configuration@pc281' window. It has two modes: 'View mode' (unselected) and 'Edit mode' (selected). A message states: 'You started editing experiment configuration. External changes will show a pop-up dialog.' Below this are three tabs: 'Measurement Group', 'Snapshot Group', and 'Storage'. The 'Active Measurement Group' is set to 'mntgrp01'. There are '+', '-', and 'Clear' buttons. A toolbar includes 'Select All', 'Refresh', 'New item', and 'Advanced view' (unchecked). A table lists channels with columns for 'Channel', 'enabled', 'output', 'Plot Type', and 'Plot Axes'. The table contains four rows for channels ct01 through ct04. At the bottom, there are 'Reload' and 'Apply' buttons, and a 'Keep editing' checkbox.

Channel	enabled	output	Plot Type	Plot Axes
ct01	true	true	No	
ct02	true	true	No	
ct03	true	true	No	
ct04	true	true	No	

# Execute a Step Scan

```
Door_demo1_1 [6]: ascan mot01 0 100 10 0.5
This operation will not be stored persistently. Use "exconf" or "newfile" to configure data storage (or eventually "senv ScanDir
ir <abs directory>" "senv ScanFile <file name(s)>")
Scan #2 started at Thu Sep 18 14:15:52 2025. It will take at least 0:00:17.272699
```

#Pt	No	mot01	ct01	ct02	ct03	ct04	dt
0	0	0	0.5	1	1.5	2	2.87304
1	10	0.5	1	1.5	2	4.37032	
2	20	0.5	1	1.5	2	5.87271	
3	30	0.5	1	1.5	2	7.37205	
4	40	0.5	1	1.5	2	8.88519	
5	50	0.5	1	1.5	2	10.388	
6	60	0.5	1	1.5	2	11.8931	
7	70	0.5	1	1.5	2	13.4	
8	80	0.5	1	1.5	2	14.8946	



TaurusForm@pc281

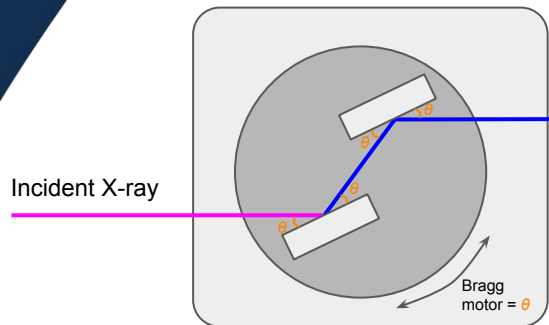
mot01  Abs

Reset Apply

# Tango Controllers

# Beamline User Case

## Double Crystal Monochromator



Physical motor = Bragg =  $\theta$

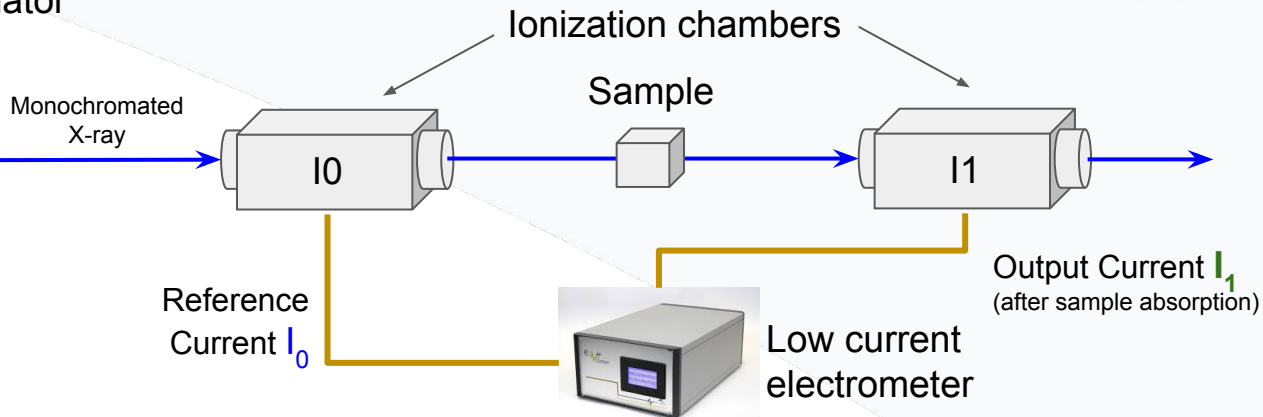


Bragg law:  $n\lambda = 2d \sin\theta$   
Plank eq:  $E = h \times c / \lambda$

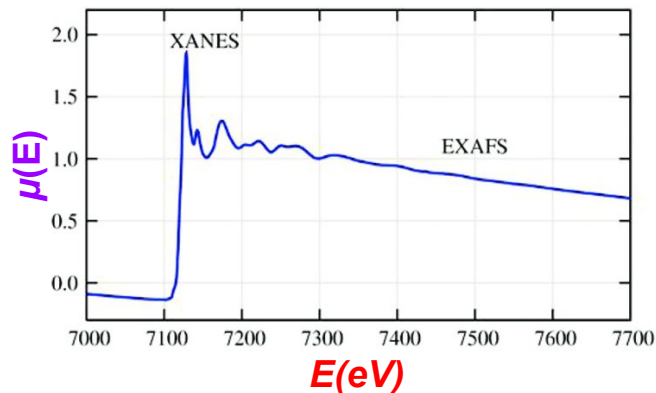
Pseudo motor = Energy =  $E$  (eV)

### Goal:

We need to scan the **Energy** and read two **currents** in an orchestrated way. This involves moving the motor related non-linearly with the energy and operate with both read currents to calculate the absorption coefficient and plot the Absorption Spectrum.



$$\mu(E) = \log(I_0/I_1) \text{ (absorption coefficient)}$$



# Beamline Simulation: TangoTest + Sardana



## Simulating an Experiment with Tango and Sardana

Sardana provides a crucial high-level layer on top of Tango, enabling scientists to define and execute complex experiments with ease while leveraging existing Tango infrastructure. We can demonstrate this by using the ubiquitous `sys/tg_test/1` Tango device to simulate a realistic experiment. The device's attributes can be configured to represent real-world physical quantities in Sardana.

## Mapping Tango Attributes to Sardana Elements via TangoAttrXXXControllers

The following table illustrates how Tango device attributes are mapped to Sardana elements using dedicated controllers. These elements appear in the Sardana Device Pool and can be accessed via the Spock command line interface (CLI) or Jive.

Tango Attribute	Purpose	Sardana Controller and Element
<code>sys/tg_test/1/ampli</code>	Represents a motor position (e.g., a monochromator).	TangoAttrMotorController (e.g., Energy)
<code>sys/tg_test/1/double_scalar</code>	Represents an analog reading (e.g., I0 Current).	TangoAttrZeroDController (e.g., I0_Current)
<code>sys/tg_test/1/double_scalar</code>	Represents an analog reading (e.g., I1 Current).	TangoAttrZeroDController (e.g., I1_Current)
<code>sys/tg_test/1/wave</code>	Represents a 1D array attribute (e.g., an absorption curve).	TangoAttrOneDController (e.g., Spectrum)
<code>sys/tg_test/1/boolean_scalar</code>	Represents a binary state (e.g., a shutter open/close).	TangoAttrIORController (e.g., BeamShutter)

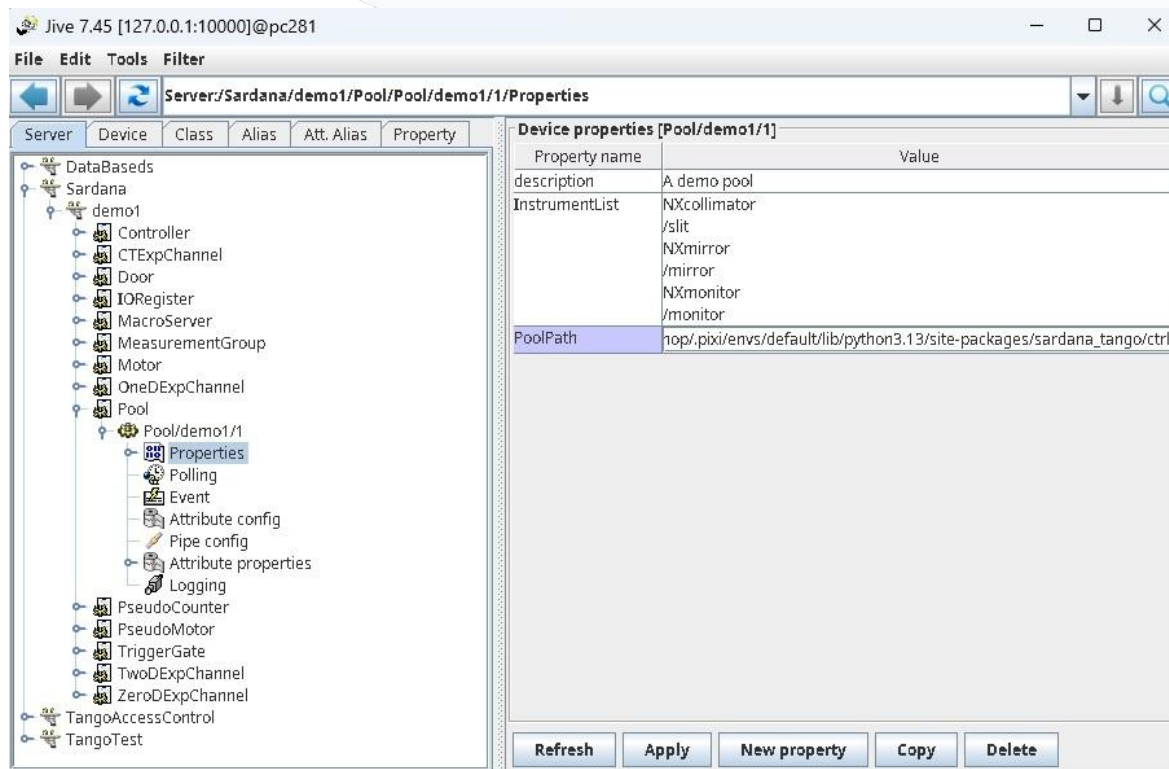


# tango\_attr\_xxxx\_controllers

# Add plugins to Sardana

```
cd .pixi/envs/default/lib/python3.13/site-packages/sardana_tango/ctrl/
```

```
pwd → output should go to PoolPath Pool Tango property
```



The screenshot shows the Jive 7.45 interface with the following components:

- Window title: Jive 7.45 [127.0.0.1:10000]@pc281
- Menu bar: File Edit Tools Filter
- Navigation buttons: Back, Forward, Refresh
- Address bar: Server/Sardana/demo1/Pool/Pool/demo1/1/Properties
- Tab bar: Server Device Class Alias Att. Alias Property
- Tree view (left):
  - DataBases
  - Sardana
    - demo1
      - Controller
      - CTExpChannel
      - Door
      - IORegister
      - MacroServer
      - MeasurementGroup
      - Motor
      - OneDExpChannel
      - Pool
        - Pool/demo1/1
          - Properties (selected)
          - Polling
          - Event
          - Attribute config
          - Pipe config
          - Attribute properties
          - Logging
        - PseudoCounter
        - PseudoMotor
        - TriggerGate
        - TwoDExpChannel
        - ZeroDExpChannel
    - TangoAccessControl
    - TangoTest

- Table (right): Device properties [Pool/demo1/1]

Property name	Value
description	A demo pool
InstrumentList	NXcollimator /slit NXmirror /mirror NXmonitor /monitor
PoolPath	hop/.pixi/envs/default/lib/python3.13/site-packages/sardana_tango/ctrl/
- Buttons (bottom): Refresh Apply New property Copy Delete

# Creation of a TangoAttrZeroDController



```
Door_demo1_1 [27]: ascancvt mot01 0 1000 10 0.5 0.15
This operation will not be stored persistently. Use "expconf" or "newfile" to configure data storage (or eventually "senv Scan0
lr <abs directory>" "senv ScanFile <file name(s)>")
Scan #9 started at Thu Sep 18 16:09:10 2025. It will take at least 8:00:00
Motor positions and relative timestamp (dt) columns contains theoretical values
```

Motor	Velocity[u/s]	Acceleration[s]	Deceleration[s]	Start[s]	End[s]
mot01	153.846	2	2	-153.846	1259.85

#Pt No	mot01	ct01	ct02	ct03	ct04	I0_Current	I1_Current	dt
0	0	1	1.5	2	-233.868	102	15.9962	
1	100	0.5	1	1.5	2	-233.559	122.167	16.5562
2	200	0.5	1	1.5	2	-232.015	223	17.2062
3	300	0.5	1	1.5	2	-232.015	223	17.8562
4	400	0.5	1	1.5	2	-232.015	223	18.5062
5	500	0.5	1	1.5	2	-230.892	98	19.1562
6	600	0.5	1	1.5	2	-230.892	98	19.8062
7	700	0.5	1	1.5	2	-230.892	98	20.4562
8	800	0.5	1	1.5	2	-228.098	121	21.1062
9	900	0.5	1	1.5	2	-228.098	121	21.7562

Experiment Configuration@pc281

View mode  
 Edit mode

You started editing experiment configuration. External changes will show a pop-up dialog.

Measurement Group Snapshot Group Storage

Active Measurement Group mntgrp01 + -

Clear Select All Clear selection Refresh New item >>  Advanced view

Channel	enabled	output	Plot Type	Plot Axes
ct01	true	true	No	
ct02	true	true	No	
ct03	true	true	No	
ct04	true	true	No	
I0_Current	true	true	No	

Keep editing  Apply

```
Door_demo1_1 [10]: %defctrl TangoAttrZeroDController CurrentCtrl
Created CurrentCtrl in Pool_demo1_1

Door_demo1_1 [11]: %defelem I0_Current CurrentCtrl 1
Created I0_Current in Pool_demo1_1

Door_demo1_1 [12]: %defelem I1_Current CurrentCtrl 2
Created I1_Current in Pool_demo1_1

Door_demo1_1 [13]: I0_Current.TangoAttribute = "sys/tg_test/1/double_scalar"

Door_demo1_1 [14]: I1_Current.TangoAttribute = "sys/tg_test/1/float_scalar"
```

# Custom hardware and macro integration

# Custom hardware integration

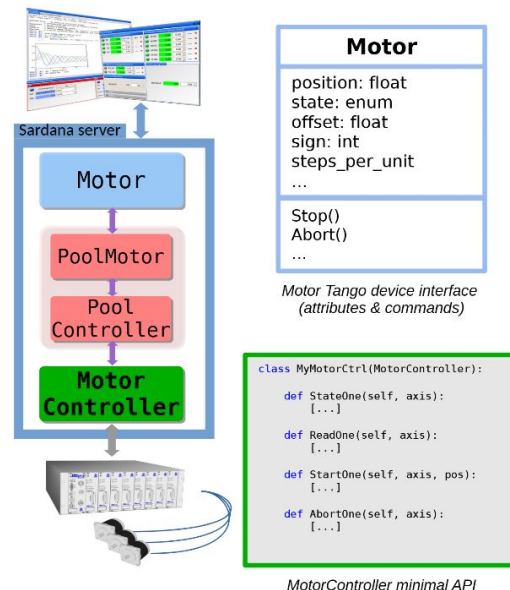
1. **Tango Device** → **Configure the sardana Tango controller (as you have seen it today).**
2. **Tango Device** → **Develop a custom sardana controller using DeviceProxy.**
3. **No Tango Device** → **Develop a custom sardana controller e.g. using a socket.**

## Sardana plugins for specific hardware

Below you will find a table with Sardana plugins for specific hardware like for example motion controllers, detectors, etc.

Name	Description	Link(s) to project
AdLink	AdLink DAQ cards e.g. 2005	<a href="#">sardana-adlink</a>
AglisAGAP	Aglis Conex AGAP mirror mount	<a href="#">AglisAGAPMotorController</a>
AglisAGP	Aglis Conex AGP rotational mount	<a href="#">AglisAGPMotorController</a>
ALBA Em Electrometer	Low current electrometer	<a href="#">sardana-albaem</a>
AmptekOneD	AmptekPX5 Multi-channel analyzer as oned	<a href="#">AmptekOneDCtrl</a>
AmptekPX5	AmptekPX5 Multi-channel analyzer	<a href="#">AmptekPX5</a>
CaenFastPS	Caen FastPS power supply	<a href="#">CaenFastPSMotorController</a>
DGO2	DGO2 timer	<a href="#">DGO2Ctrl</a>
EigerDectris	Eiger Dectris	<a href="#">EigerDectris</a>
EigerPSI	Eiger PSI	<a href="#">EigerPSI</a>
EpicsMotor	Epics Motor	<a href="#">EpicsMotorController</a>
EpicsZeroD	Epics ZeroD	<a href="#">EpicsZeroDControllerr</a>

Sardana extra projects catalogue:  
<https://gitlab.com/sardana-org/sardana-extra>



MotorController minimal API

# Custom macro development



```
$ pixi run spock
Profile 'spockdoor' does not exist. Do you want to create one now ([y]/n)?
y
Available Door devices from 127.0.0.1:10000 :
Door demo1 1 (a.k.a. Door/demo1/1) (running)
Door name from the list? Do or_demo1_1
..
Spock 3.6.0 -- An interactive laboratory application.

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

IPython profile: spockdoor
Connected to Door_demo1_1

Door_demo1_1 [1]: edmac myexperiment mymodule
```

## Macro development features:

- parameters
- result
- data
- environment
- hooks

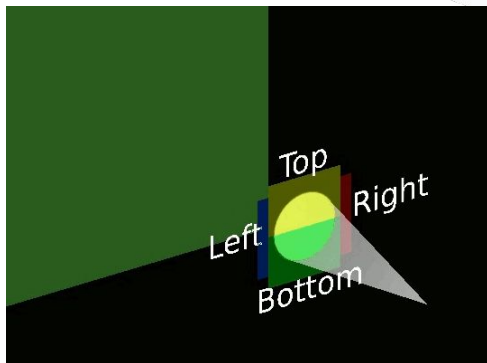
```
import tango
from sardana.macroserver.macro import Macro, macro, Type

@macro()
def mymacro(self):
    """Macro mymacro"""
    self.output("Running mymacro...")
    shutter = tango.AttributeProxy(
        "sys/tg_test/1/boolean_scalar")
    shutter.write(True)
    self.output(shutter.read().value)
    self.ascan("mot01", 0, 10, 10, 0.1)
    shutter.write(False)
    self.output(shutter.read().value)
```

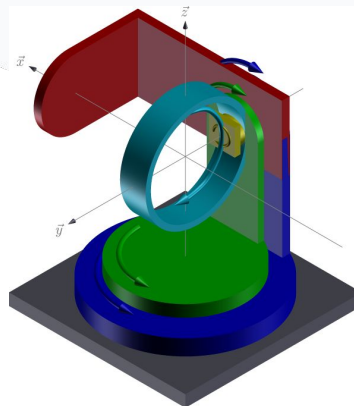
- interactive macros
- macro progress
- plotting
- ...

# Extras

# More Sardana capabilities



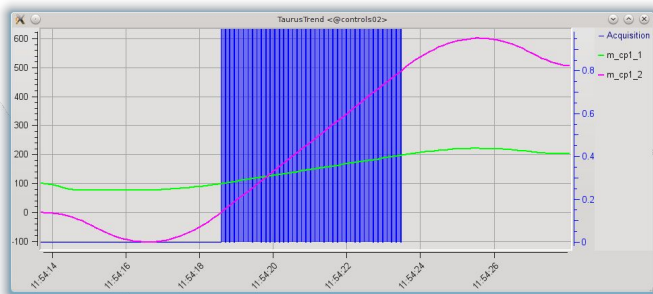
Pseudo motors



Diffractometer control (using hkl lib by F. Picca)

```
15 pools:
16   demo1:
17     description: "A demo pool"
18     instruments:
19       /slit:
20         class: NXcollimator
21       /mirror:
22         class: NXmirror
23       /monitor:
24         class: NXmonitor
25
26   controllers:
27     motctrl01:
28       description: "A motor controller"
29       type: Motor
30       python_module: DummyMotorController.py
31       python_class: DummyMotorController
32     elements:
33       mot01:
34         axis: 1
35         instrument: /slit
36       mot02:
37         axis: 2
38         instrument: /slit
39       mot03:
40         axis: 3
41         instrument: /mirror
42       mot04:
43         axis: 4
44         instrument: /mirror
45     ...
```

YAML-based config format and tools



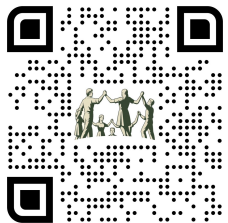
Continuous Scans  
(hardware & software synchronized)



HDF5 data recording, also SPEC,  
FIO, blissdata by the ESRF  
(Redis)

...and more,  
and more to come...

# Sardana Project and Community



<https://sardana-controls.org>  
<https://gitlab.com/sardana-org>



- Monthly follow-ups organized by ALBA, DESY, MAX-IV, SOLARIS.  
<https://gitlab.com/sardana-org/sardana-followup>
- Community Workshops (~yearly)
  - 2023 Continuous Scans Workshop in SOLARIS <https://indico.solaris.edu.pl/event/5/>
  - 2024 Sardana workshop in Tango Meeting at SOLEIL  
<https://gitlab.com/sardana-org/sardana-followup/-/blob/main/20240530-SOLEIL/AGENDA.md>
  - 2025 Sardana Workshop in MAX IV <https://indico.maxiv.lu.se/event/5634/>
  - 2026? Tango Meeting is in BCN...



**Thank you for your attention!**  
**Any questions?**

**Sardana & Taurus Status**  
**WEPD026 Poster & Mini-oral**  
**24 Sept 2025, 15:45 16:30**

# Beamline User Case

## Beamline Context

This slide describes a typical experimental procedure on a Beamline where Sardana is used for orchestration.

- **Goal:** To obtain an absorption spectrum of a sample.
- **Process Orchestration (Sardana):**
  - **Step 1: Mono-Energy Scan:** The Sardana MacroServer orchestrates a scan of the mono-energy. This involves controlling the monochromator motors to step through a defined range of energies.
  - **Step 2: Electrometer Acquisition:** At each energy step, Sardana triggers the acquisition of data from an electrometer to measure the current passing through the sample.
  - **Step 3: Data Analysis:** After the acquisition at each point, Sardana can perform real-time data analysis to calculate the absorption value.
- **Result:** The coordinated scan and acquisition results in the generation of an absorption spectrum, which is then presented to the user.

# Accelerator User Case

## Service Area Context

This slide describes a control and measurement scenario within a Service Area, using Sardana to manage various devices.

- **Goal:** To verify the stability and position of the X-ray beam using Insertion Devices (IDs) and Front Ends (FEs). → `fescan_lorea.py`
- **Process Orchestration (Sardana):**
  - **Step 1: Device Scans:** The Sardana MacroServer orchestrates a scan of the Insertion Device (ID) and Front End (FE) motors. This can be used to tune or characterize the beamline components.
  - **Step 2: XBPM-Locum Reading:** Throughout the scan, Sardana reads the values from the X-ray Beam Position Monitor (XBPM-Locum) to continuously monitor the beam's position and stability.
- **Result:** The synchronized scanning and reading provides critical diagnostic information, ensuring the beam is correctly positioned and stable for user experiments.