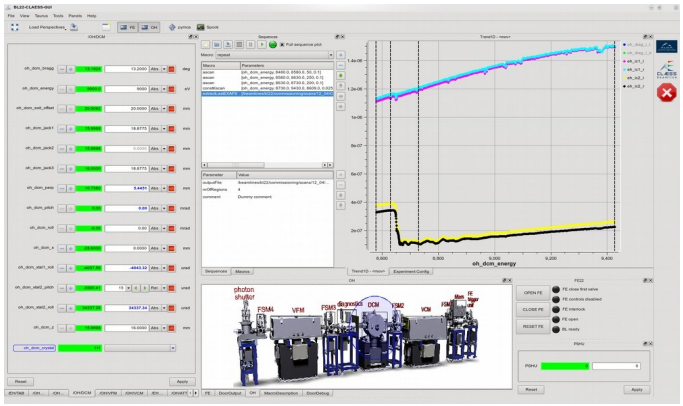
Sardana Macros

by Zbigniew Reszela
(ALBA Synchrotron, Spain)
on behalf of Sardana Community

Contents

- **Introduction**
  - What is Sardana and MacroServer
  - Usage of macros
  - Installation
- **Macro development**
  - Create simulated environment
  - Macro execution clients
  - Hello world example
  - Macro features
- **Scan with Tango attributes**

Sardana is...



Taurus based GUIs

Scientific SCADA Software Suite

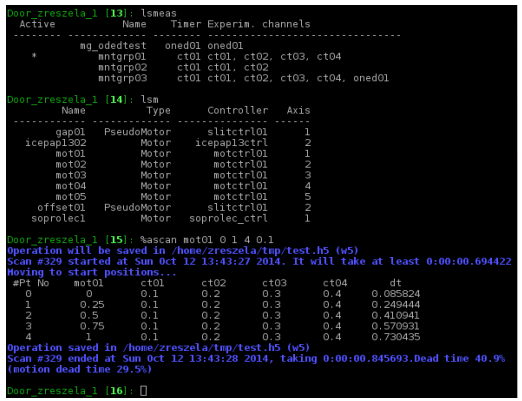
Suite = Sardana & Taurus projects

100 % Python

Built on top of TANGO

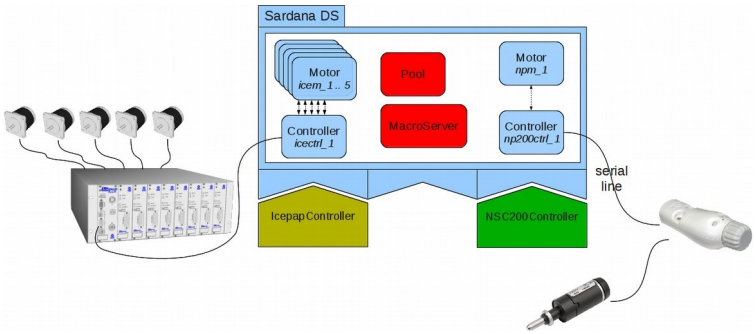
Extendable with plugins

Community driven:



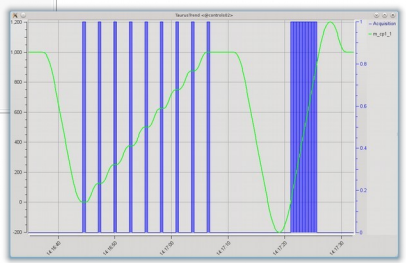
Spock – IPython based CLI

Device Pool – HW access + low level control



MacroServer – powerful sequencer

```
from sardana.macroserver.macro import macro

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



Macros – Python function or classes

Hooks

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

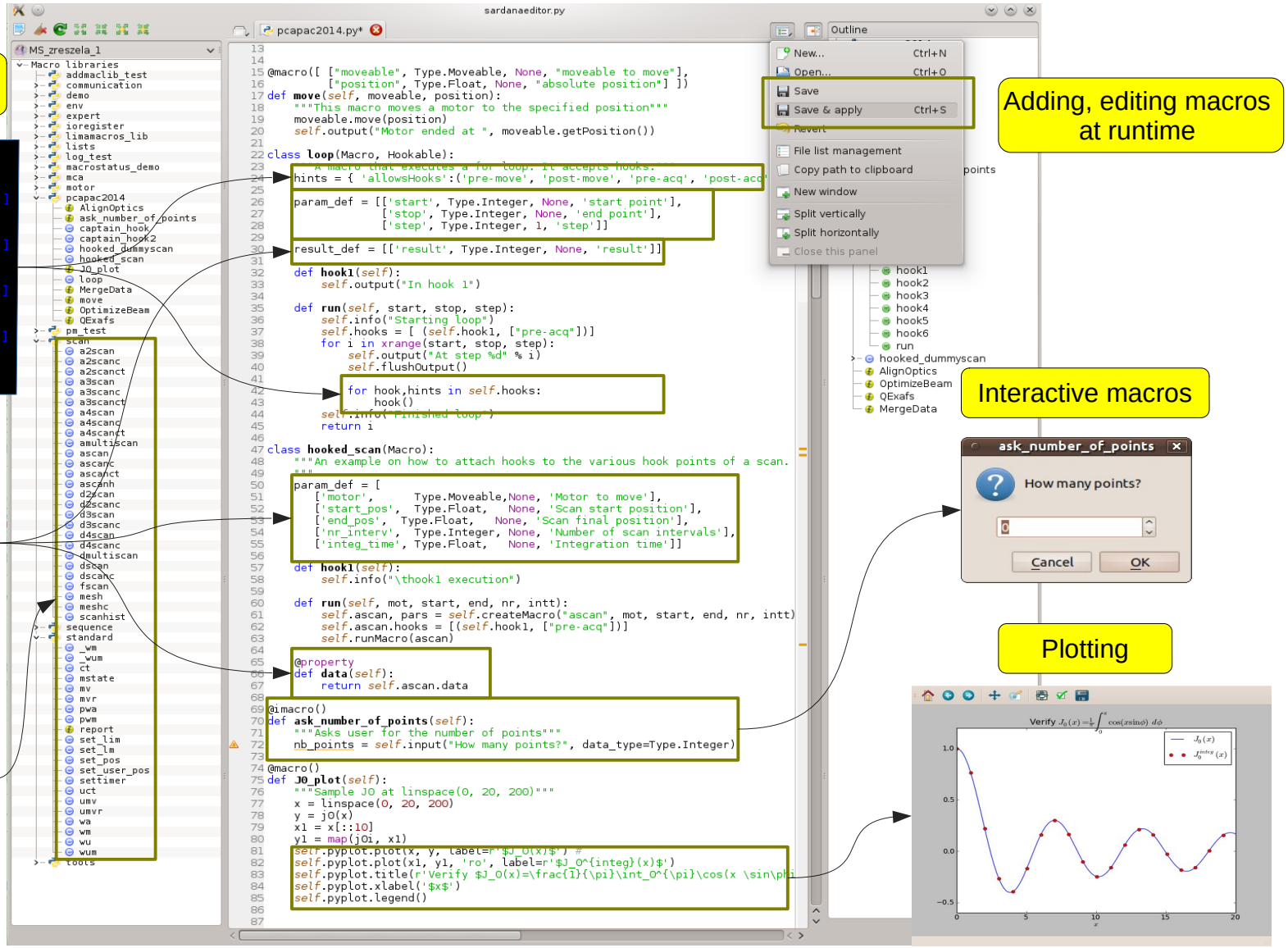
Input parameters & results & data

SPEC like commands

Adding, editing macros at runtime

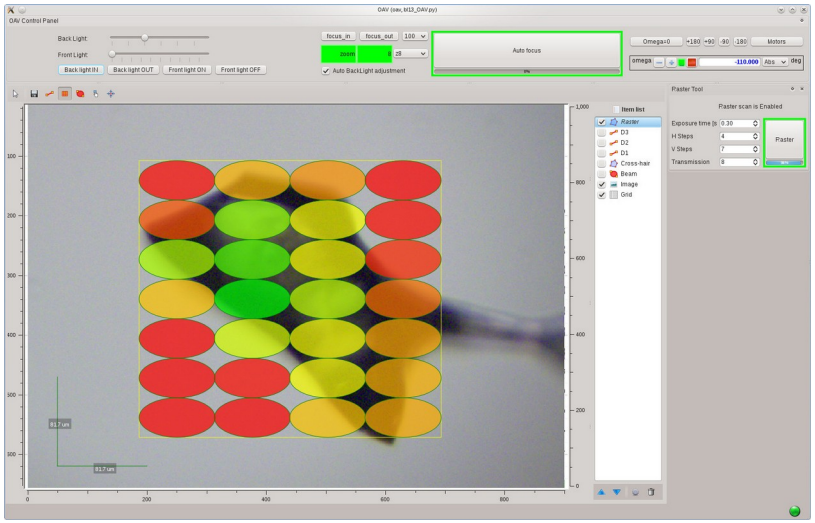
Interactive macros

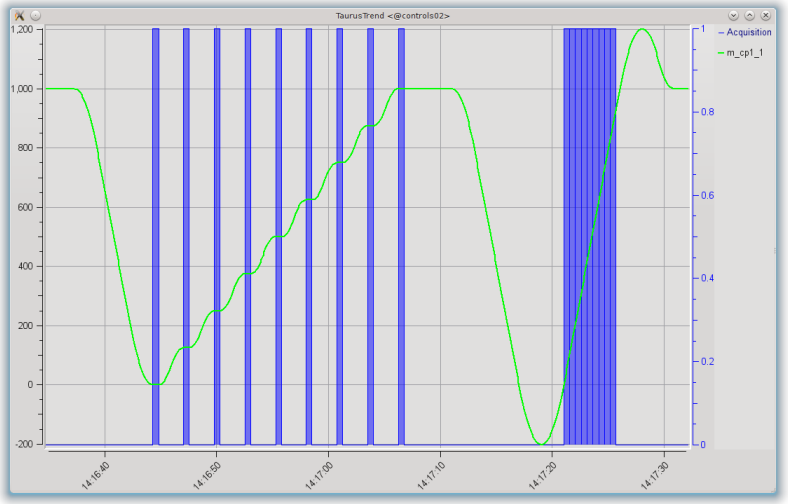
Plotting

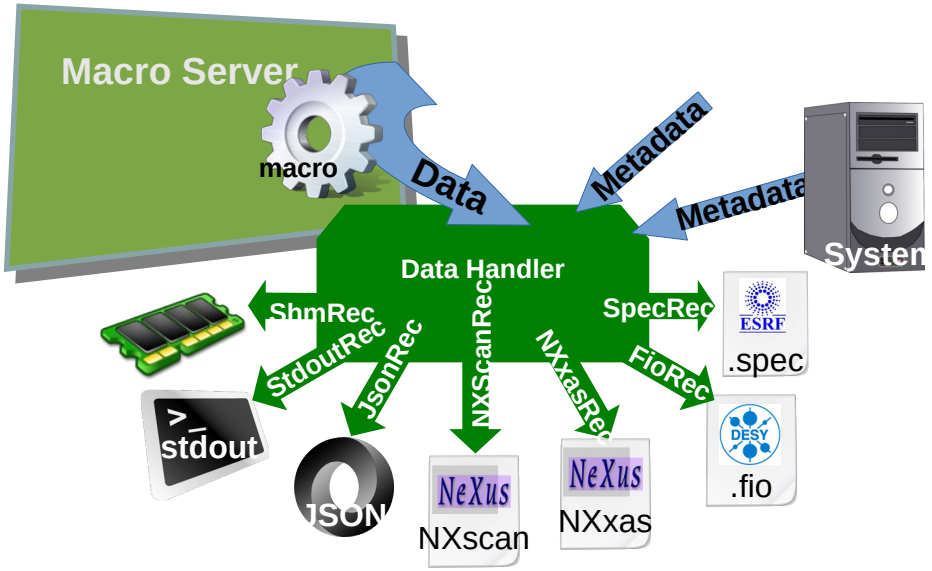


Usage in Experiment Control

- Macros for experiment control:
  - Scans: step & continuous
  - Optical elements alignments
  - Experiment sequences
  - Sample environment preparation

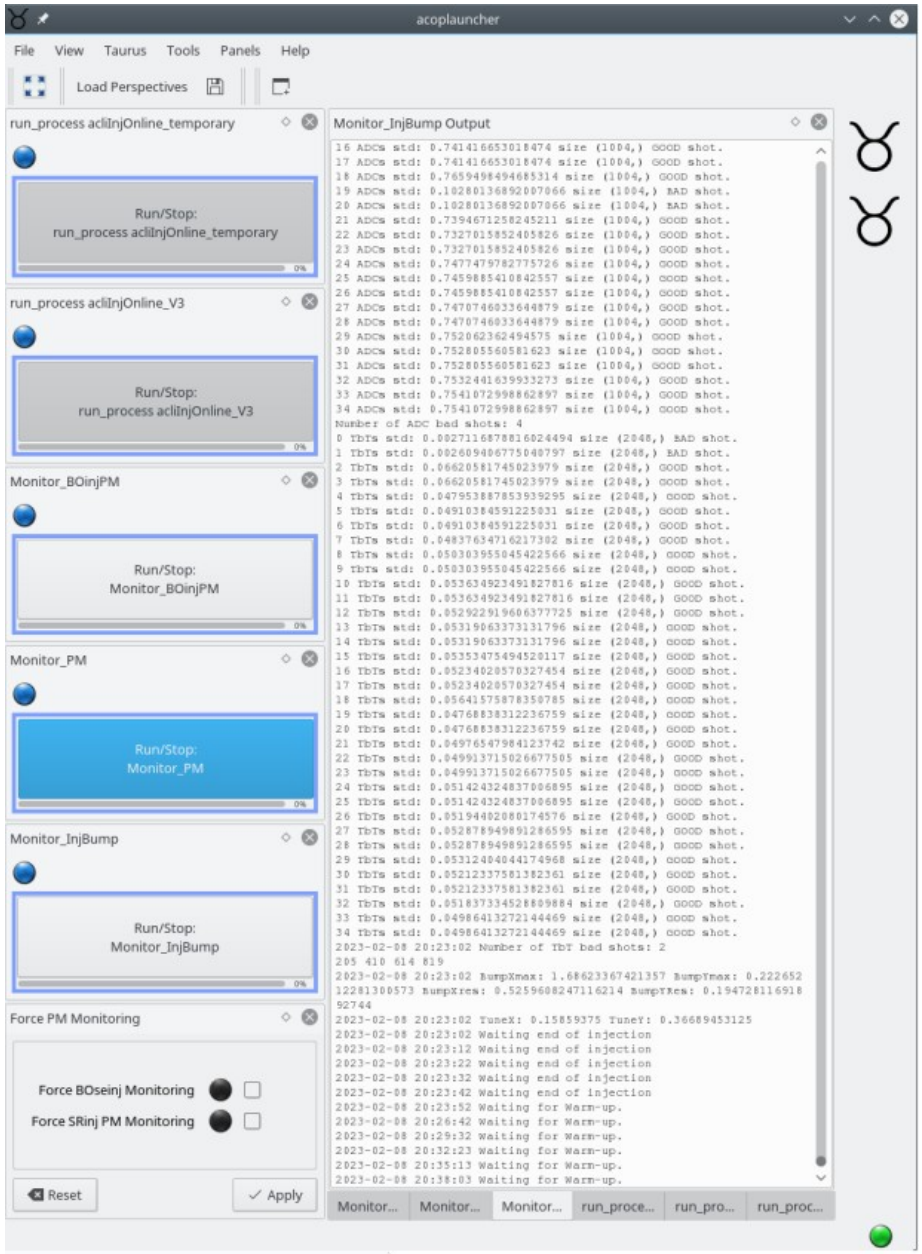






Usage in Accelerator Control

- Macros to support accelerators operation and monitor:
  - LINAC injection parameters
  - Injection to Booster
  - Injection to SR
  - Injection bump



Installation

- PyPI
- Debian Linux
- **conda**

```
conda create -c conda-forge -y -n macros_tutorial \
    sardana \
    matplotlib \  # macro plotting
    h5py \        # HDF5 file recorder
    qtconsole \   # QtSpock
    jupyter_client=6.1.11 \  # pinned version for QtSpock
    pyqtgraph \   # scan plotting
    silx \        # opening scan file
   tango-test     # for demo purposes
conda activate macros_tutorial
pip install git+https://github.com/ALBA-Synchrotron/sardana-
tango.git
```

More on: https://sardana-controls.org/users/getting_started.html

Contents

- **Introduction**
  - What is Sardana and MacroServer
  - Usage of macros
  - Installation

- **Macro development**
  - Create simulated environment
  - Macro execution clients
  - Hello world example
  - Macro features

- **Scan with Tango attributes**

Server, Spock and sar_demo...

```
# Start the Sardana server
Sardana demo1
 # Confirm with [Y]
# Start the Spock client – SPOC, based on IPython
Spock
 # Confirm with [y]
 # select the door Door_demo1_1
# In Spock create simulated elements:
sar_demo
# List created elements
lsa
# Execute a step scan
ascan mot01 0 5 5 0.1
```

```
Door_demo1_1 [7]: lsa
      Name              Type          Controller    Axis
 ---------- ----------------- --------------- ------
       ct01      CTExpChannel        ctctrl01      1
       ct02      CTExpChannel        ctctrl01      2
       ct03      CTExpChannel        ctctrl01      3
       ct04      CTExpChannel        ctctrl01      4
      gap01       PseudoMotor      slitctrl01      1
      ior01        IORegister       iorctrl01      1
      ior02        IORegister       iorctrl01      2
   ioveri001     PseudoCounter   ioveri0ctrl01      1
      mot01             Motor       motctrl01      1
      mot02             Motor       motctrl01      2
      mot03             Motor       motctrl01      3
      mot04             Motor       motctrl01      4
    offset01       PseudoMotor      slitctrl01      2
     oned01    OneDExpChannel       onedctrl01      1
       tg01       TriggerGate         tgctrl01      1
     twod01    TwoDExpChannel       twodctrl01      1
    zerod01   ZeroDExpChannel      zerodctrl01      1
    zerod02   ZeroDExpChannel      zerodctrl01      2
    zerod23   ZeroDExpChannel      zerodctrl01      3
    zerod24   ZeroDExpChannel      zerodctrl01      4
```

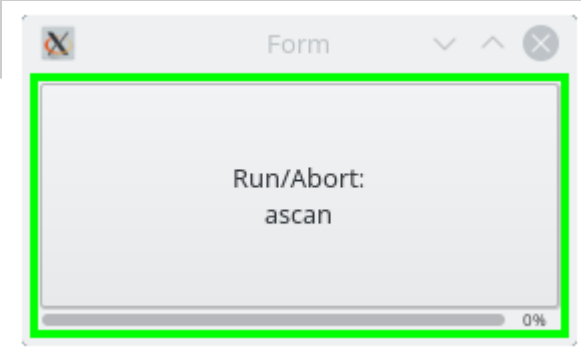
How does it work?

- Check macro code with:
  prdef sar_demo
- Macro is available in Spock as a magic command.
- Macros code is executed on the server side.

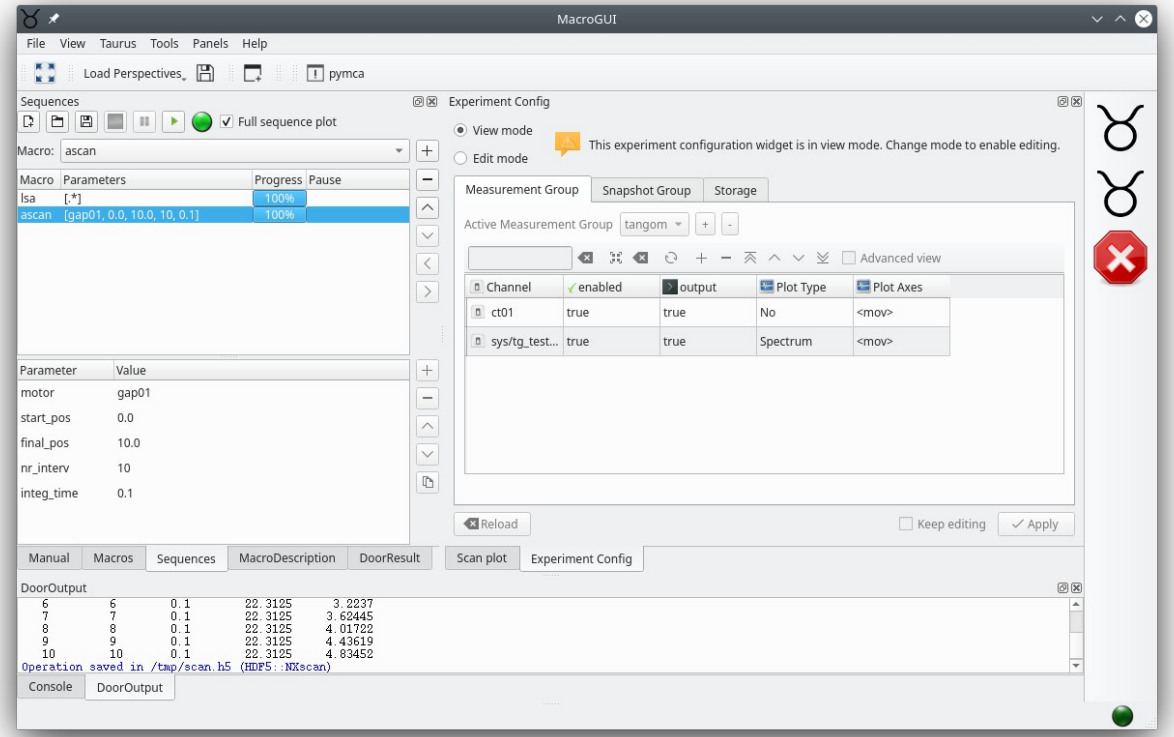
```
Door_demo1_1 [9]: prdef sar_demo
@macro([
    ["elements",[
        ["elem_type", Type.String, None, "Element type"],
        ["elem_quant", Type.Integer, 0, "Element quantity"],
        {'min': 0, 'max': None}
    ], None, "Number of elements to be created per type"]
])
def sar_demo(self, elements):
    """Sets up a demo environment. It creates many elements for testing"""

    try:
        SAR_DEMO = self.getEnv(_ENV)
        self.error("A demo has already been prepared on this sardana")
        return
    except:
        pass

    db = PyTango.Database()

    elements_quant = default_elements_quant.copy()
    for elem_type, elem_quant in elements:
        elem_type_lower = elem_type.lower()
        if elem_type_lower not in elements_quant:
            raise ValueError(
                "element type '{}' is not recognised (allowed types: {})".format(
                    elem_type, list(elements_quant.keys())))
        # Replace the default quantity with the configured one
        elements_quant[elem_type_lower] = elem_quant

    mot_ctrl_name = get_free_names(db, "motctrl", 1)[0]
    ct_ctrl_name = get_free_names(db, "ctctrl", 1)[0]
```

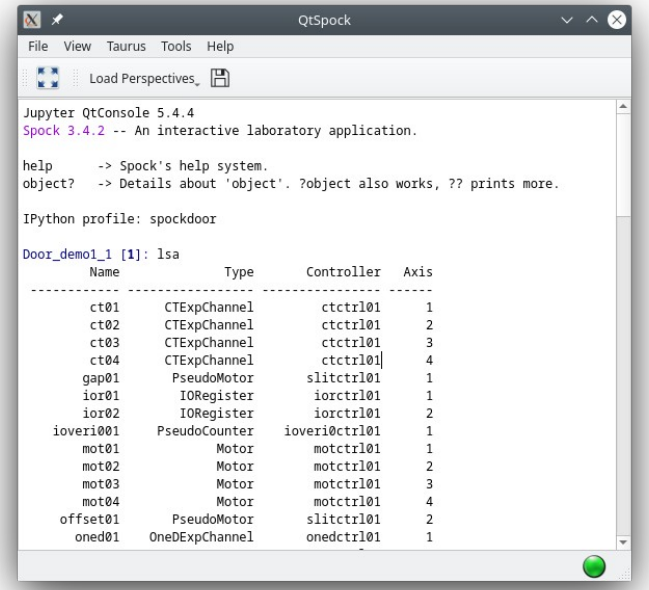
Other macro execution clients

```
# Start the Macro execution GUI
taurus gui macrogui
# Connect to the MacroServer and Door: Menu → Taurus → Macro execution configuration…
# Compose and run a sequence
```



Macro Button





QtSpock

How to plug-in a macro?

- Macro discovery:
  - The plugin discovery system is based on directory scanning and python module inspection
  - Custom macros should be installed in one of the MacroPath directories

```
# Create a directory for you macros:
mkdir /tmp/macros
# In Spock set the MacroPath property:
_MACRO_SERVER.put_property({"MacroPath":["/homelocal/zreszela/miniconda3/envs/macros_tutorial/lib/python3.10/site-packages/sardana/macroserver/macros/examples/", "/tmp/macros"]})
# Restart the Sardana server and Spock
# List newly added macro libraries
lsmaclib
```

Hello world example

- Macro functions are decorated with `@macro()`
- Macros log messages are sent to the clients
- Different log levels can be used
- Macros may use input parameters and results
- >50 parameter types are available

```
# In Spock edit the macro:
edmac hello salute
# Apply the new code with [y]
# Run the macro:
hello
# Let’s add a macro parameter of type String called “name”
edmac hello
```

```
Door_demo1_1 [13]: prdef hello
@macro([["name", Type.String, "Cape Town", "Whom to greet?"]])
def hello(self, name):
    """Macro hello"""
    self.output(f"Hello {name}!")
```

Macro features

- Macros can **call other macros**: self.<macro_name>() e.g. self.hello(“Paul”)
- **Interactive macros** - macros can ask for input e.g. ask_peak.
- **Plotting** – macro can plot data e.g. mandelbrot 10 100
- **Interrupting** – macros can be stopped, aborted, paused and resumed. Run a scan macro and try <Ctrl+C>.
- Macros can access to a shared **environment**: lsenv, senv, usenv

```
# In Spock define MultiGreet environment variable
senv MultiGreet 3
# Edit the macro and repeat greeting MultiGreet times
edmac hello
# Apply the new code with [y]
# Run the macro:
hello
```

```
Door_demo1_1 [36]: prdef hello
@macro([["name", Type.String, "Cape Town", "Whom to greet?"]])
def hello(self, name):
    """Macro hello"""
    for _ in range(self.getEnv("MyEnvVar")):
        self.output(f"Hello {name}!")
```

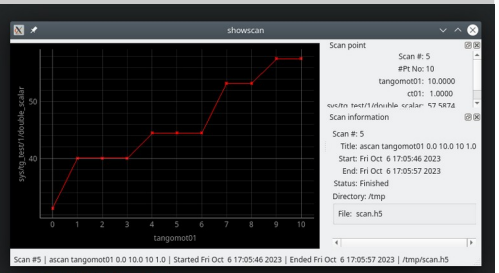
Contents

- **Introduction**
  - What is Sardana and MacroServer
  - Usage of macros
  - Installation
- **Macro development**
  - Create simulated environment
  - Macro execution clients
  - Hello world example
  - Macro features
- **Scan with Tango attributes**

Scan with Tango attributes

```
# In Spock set the PoolPath property and restart the server
Pool_demo1_1.put_property({"PoolPath":
"/homelocal/zreszela/miniconda3/envs/macros_tutorial/lib/python3.1
0/site-packages/sardana_tango/ctrl/"})
# Define the controller instance
defctrl TangoAttrMotorController tangomotctrl01
# Define the motor instance
defelem tangomot01 tangomotctrl01 1
# Configure the motor to point to a Tango attribute
tangomot01.TangoAttribute = "sys/tg_test/1/ampli"
# In expconf define a measurement group with ct01
sys/tg_test/1/double_scalar, configure plotting and storage
expconf
# Open online plotting widget
showscan
# Run a scan
ascan tangomot01 0 10 10 1
# Open the data file
silx view /tmp/scan.h5
```

```
Door_demo1_1 [21]: ascan tangomot01 0 10 10 1
Operation will be saved in /tmp/scan.h5 (HDF5::NXscan from NXscanH5_FileRecorder)
Scan #5 started at Fri Oct  6 17:05:46 2023. It will take at least 0:00:11
                                    sys
                                 tg_test
                                    1
 #Pt No   tangomot01    ct01    double_scalar     dt
   0          0          1         31.1985     0.0386212
   1          1          1         40.0472     1.10815
   2          2          1         40.0472     2.19201
   3          3          1         40.0472     3.27303
   4          4          1         44.4539     4.33942
   5          5          1         44.4539     5.40785
   6          6          1         44.4539     6.48691
   7          7          1         53.2253     7.54208
   8          8          1         53.2253     8.62113
   9          9          1         57.5874     9.70655
   10         10         1         57.5874     10.7654
Operation saved in /tmp/scan.h5 (HDF5::NXscan)
Scan #5 ended at Fri Oct  6 17:05:57 2023, taking 0:00:11.795911. Dead time 6.6% (setup time 0.1%, motion dead time 2.6%)
```



Home Page

http://www.sardana-controls.org

Access to:

- Documentation
- Releases
- Git repository
- Mailing lists
- Bugs & Requests tracker
- Enhancement Proposals
- ...