

- Graphical access to the MacroServer environment
- Improvement of the pseudo motor drift correction
- Graphical configuration tool for setting up Sardana

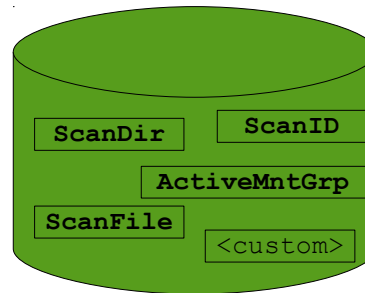


Access to the MS environment

2

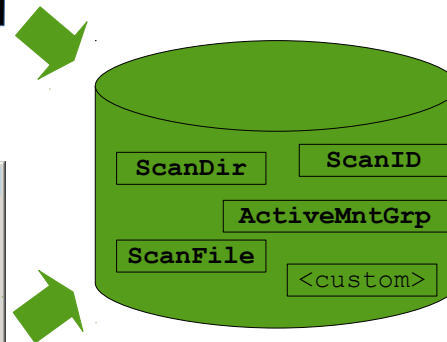
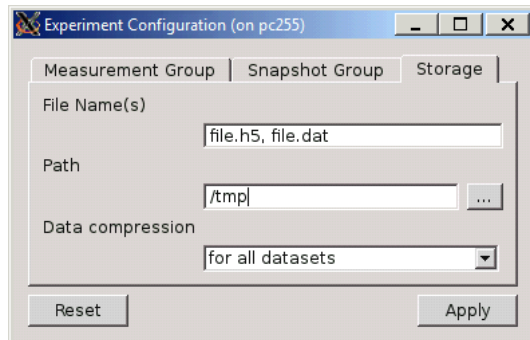
- **Motivation:** not user friendly access to the environment

- **Motivation:** not user friendly access to the environment
- **Current situation:** env. stored in a map file, powerful API is ready → env. manager, levels → global, door, macro, door.macro, Tango events with updates



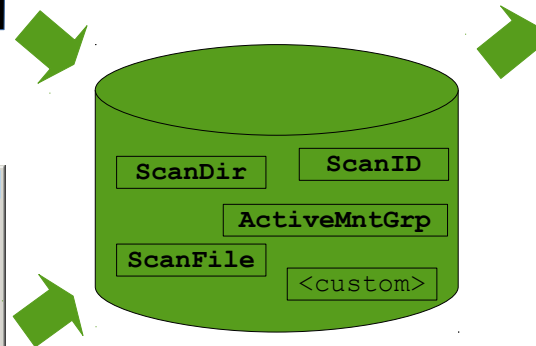
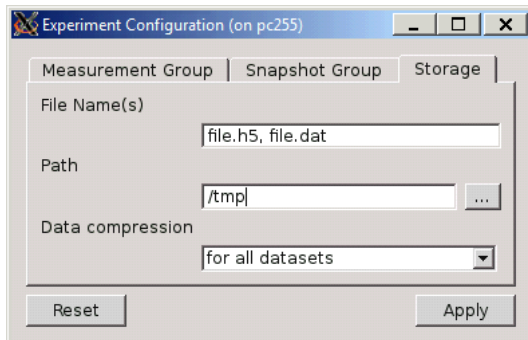
- **Motivation:** not user friendly access to the environment
- **Current situation:** env. stored in a map file, powerful API is ready → env. manager, levels → global, door, macro, door.macro, Tango events with updates
- **Current use:**

```
Door_1[1]: senv ActiveMntGrp mg01
ActiveMntGrp = mntgrp04
Door_1[2]: senv ScanFile '["file.h5", "file.dat"]'
ScanFile = ['file.h5', 'file.dat']
```



- **Motivation:** not user friendly access to the environment
- **Current situation:** env. stored in a map file, powerful API is ready → env. manager, levels → global, door, macro, door.macro, Tango events with updates
- **Current use:**

```
Door_1[1]: senv ActiveMntGrp mg01
ActiveMntGrp = mntgrp04
Door_1[2]: senv ScanFile '["file.h5", "file.dat"]'
ScanFile = ['file.h5', 'file.dat']
```



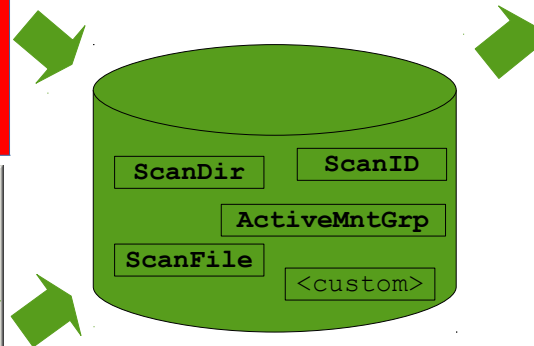
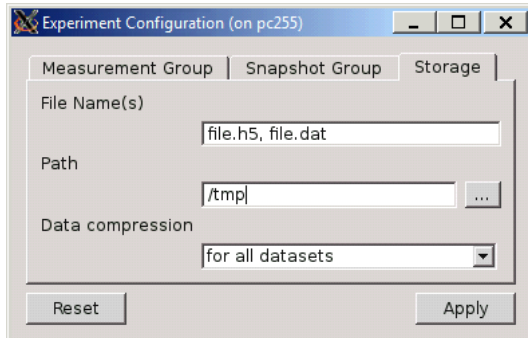
```
Door_1[2]: lsenv
```

Name	Value	Type
_____	_____	_____
_SAR_DEMO	{'cont [...]	dict
_ViewOptions	{'PosF [...]	dict
ActiveMntGrp	mntgrp03	str
DataCompressionRank	0	int
JsonRecorder	True	bool
PreScanSnapshot	[]	list
ScanDir	/tmp	str
ScanFile	['test.dat']	list
ScanHistory	['tit [...]	list
ScanID	649	int

- **Motivation:** not user friendly access to the environment
- **Current situation:** env. stored in a map file, powerful API is ready → env. manager, levels → global, door, macro, door.macro, Tango events with updates
- **Future use:**

```
Door_1[1]: senv ActiveMntGrp mg01
ActiveMntGrp = mntgrp04
Door_1[2]: senv ScanFile ['file.h5', 'file.dat']
ScanFile = ['file.h5', 'file.dat']
```

```
# Why not Spec-like macros
# e.g. newfile with ParamRepeat
Door_1[3]: newfile file.h5 file.dat
ScanFile = ['file.h5', 'file.dat']
```



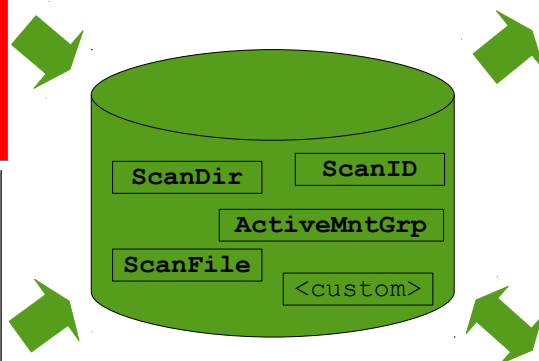
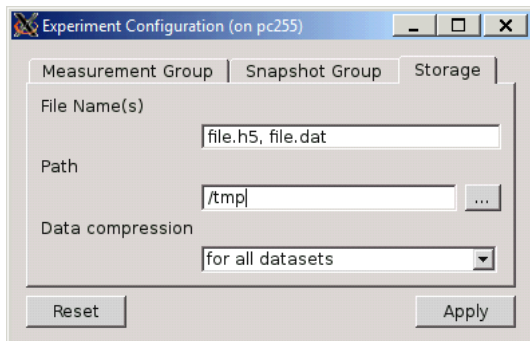
```
Door_1[2]: lsenv
```

Name	Value	Type
_SAR_DEMO	{'cont [...]	dict
_ViewOptions	{'PosF [...]	dict
ActiveMntGrp	mntgrp03	str
DataCompressionRank	0	int
JsonRecorder	True	bool
PreScanSnapshot	[]	list
ScanDir	/tmp	str
ScanFile	['test.dat']	list
ScanHistory	['tit [...]	list
ScanID	649	int

- **Motivation:** not user friendly access to the environment
- **Current situation:** env. stored in a map file, powerful API is ready → env. manager, levels → global, door, macro, door.macro, Tango events with updates
- **Future use:**

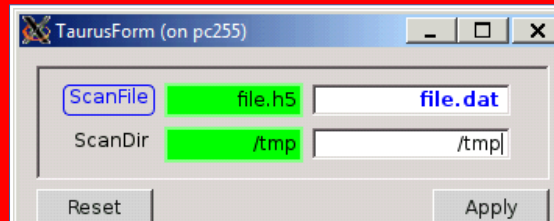
```
Door_1[1]: senv ActiveMntGrp mg01
ActiveMntGrp = mntgrp04
Door_1[2]: senv ScanFile ['file.h5', 'file.dat']
ScanFile = ['file.h5', 'file.dat']
```

```
# Why not Spec-like macros
# e.g. newfile with ParamRepeat
Door_1[3]: newfile file.h5 file.dat
ScanFile = ['file.h5', 'file.dat']
```



```
Door_1[2]: lsenv
```

Name	Value	Type
SAR_DEMO	{'cont [...]	dict
_ViewOptions	{'PosF [...]	dict
ActiveMntGrp	mntgrp03	str
DataCompressionRank	0	int
JsonRecorder	True	bool
PreScanSnapshot	[]	list
ScanDir	/tmp	str
ScanFile	['test.dat']	list
ScanHistory	['tit [...]	list
ScanID	649	int



- Simple scheme to read/write environment variables:
 - **Authority** → **Tango DB**; **Value** → **list of all MSs**.
Internally it should use proxy to Tango Database.
 - **Device** → **MS**; **Value** → **list of all variables**.
Internally it should use proxy to MS Environment.
Device should emit events when the env. variables are created, modified or deleted.
 - **Attribute** → **env. variable**; **Value** → **variable value**
Write the specific variable e.g. ascan.ScanFile.
Readout with fallback to the higher level e.g. ScanFile.
Attribute should emit events when the env. variable is created, modified or deleted.
- More details in **SEP14 by Carlos Falcon**
- Other alternatives: Tango attributes for the env. variables?

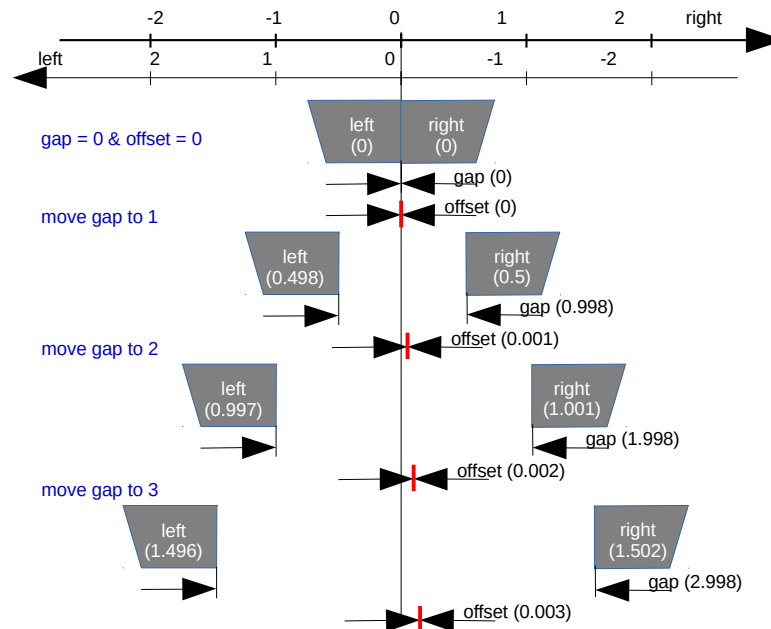
Questions? Comments?

Drift of the pseudo motors

Drift of the pseudo motors

- Who is affected? Pseudo motors which have siblings and are based on physical motors with an **inaccurate or a finite precision positioning** system.

- Who is affected? Pseudo motors which have siblings and are based on physical motors with an **inaccurate or a finite precision positioning** system.
- Why does it happen? Each move of a pseudo motor requires **calculation of the physical motors positions in accordance with the current positions of its siblings**. A consecutive movements of a pseudo motor can accumulate positioning & rounding errors and cause drift of its siblings.

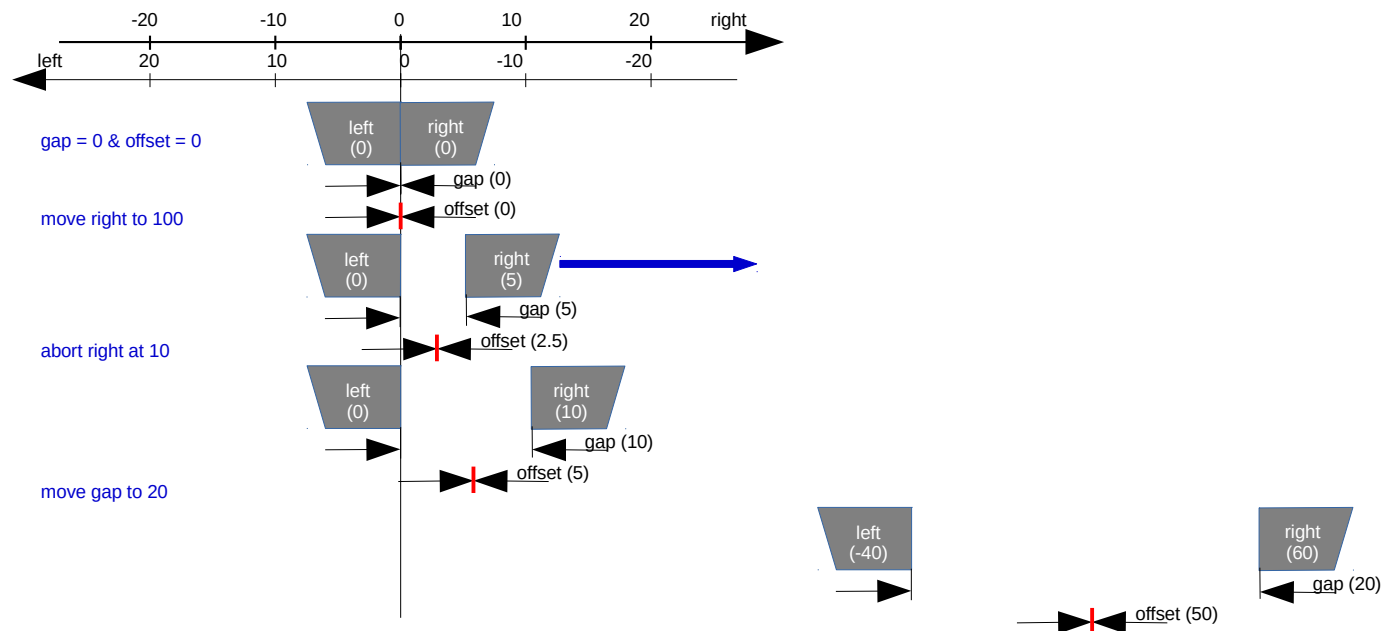


- Calculation of the new physical positions **use the write value, instead of the read value** of the siblings' positions, together with the new desired position of the pseudo motor being moved.

- Calculation of the new physical positions **use the write value, instead of the read value** of the siblings' positions, together with the new desired position of the pseudo motor being moved.
- The write value of the pseudo motor's position gets updated at each move of the pseudo motor or any of the underneath motors.

- Calculation of the new physical positions **use the write value, instead of the read value** of the siblings' positions, together with the new desired position of the pseudo motor being moved.
- The write value of the pseudo motor's position gets updated at each move of the pseudo motor or any of the underneath motors.
- Drift correction is optional but enabled by default.

- Movements being stopped unexpectedly: abort by the user, over-travel limit or any other exceptional condition may cause considerable discrepancy in the motor's write and read positions. In the subsequent pseudomotor's move, Sardana will also correct this difference by using the write instead of the read values.



- Add third option to configure the drift correction:
 - drift correction enabled but applied only in case of the successful most recent movement (use write values only when last motion succeeded)
 - drift correction enabled (strict use of the write values)
 - drift correction disabled
- The *PoolMotion* action should set the **stop code** attribute for each motor object, to indicate how ended the last move e.g.
 - success
 - user abort
 - hardware abort (e.g. limit)
- More details in **feature-request #370**

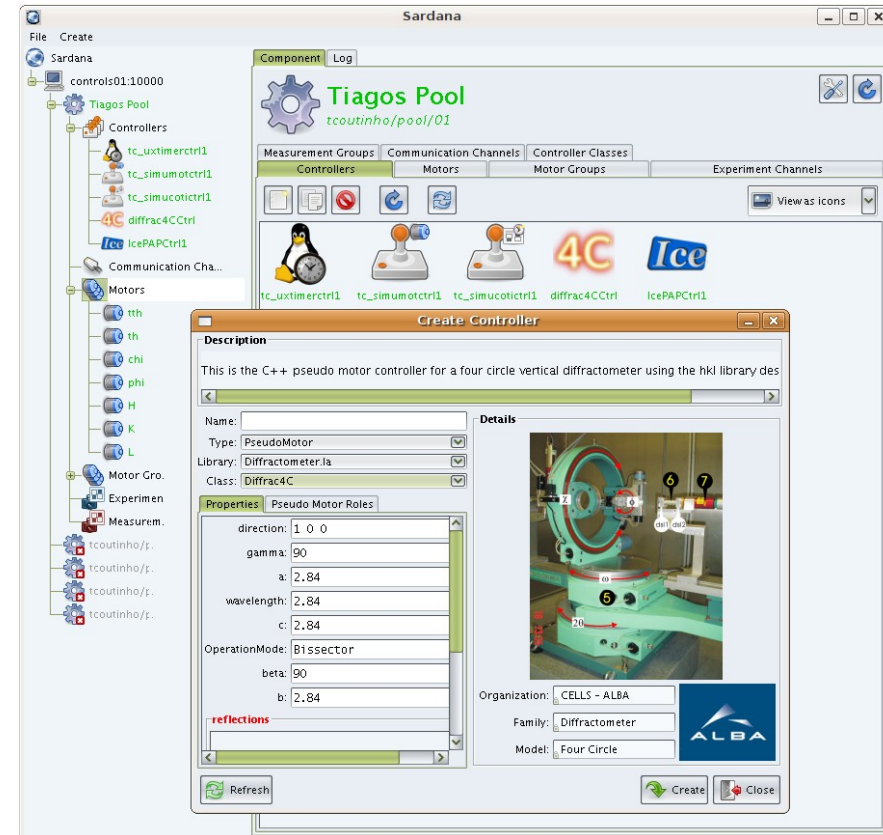
Questions? Comments?



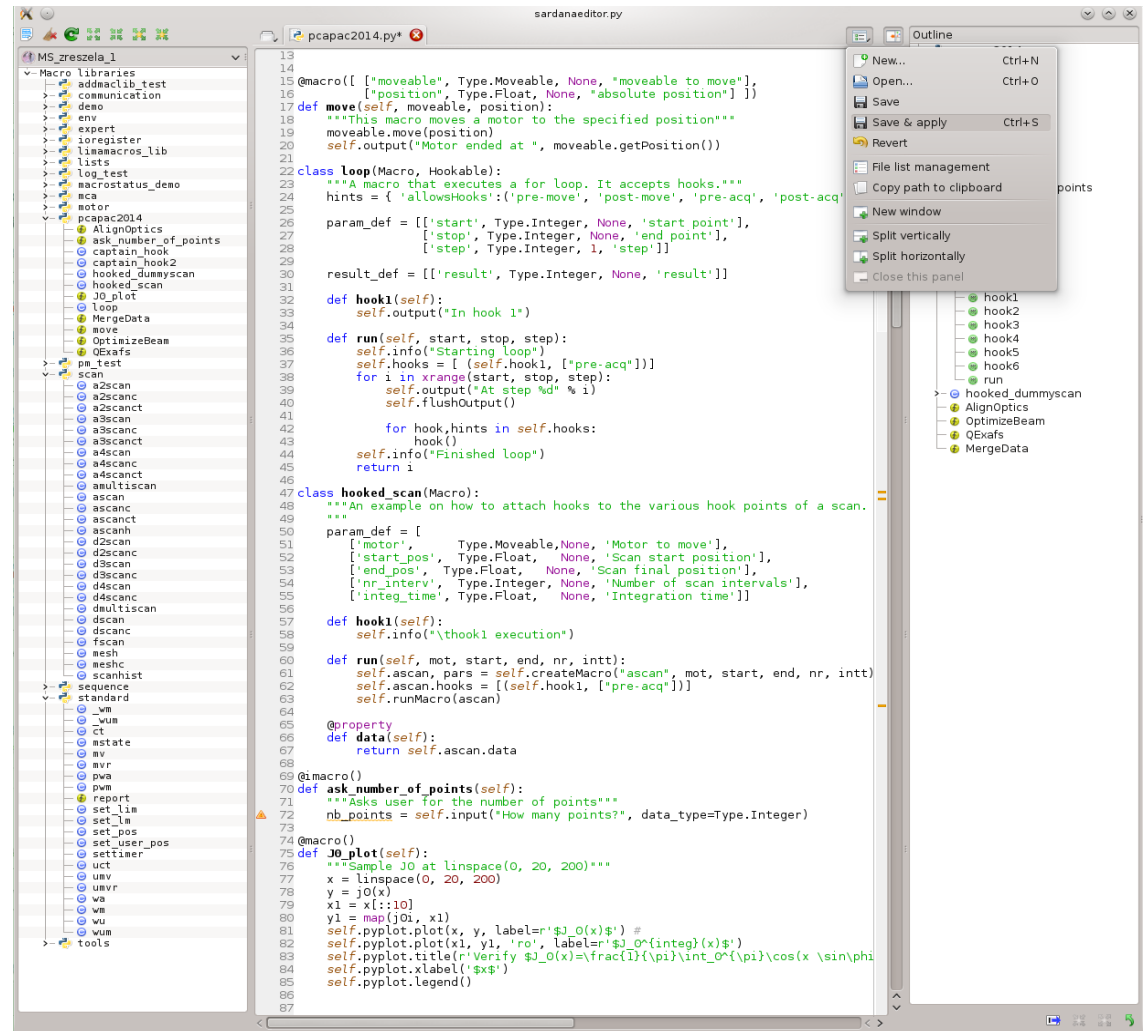
Sardana configuration GUI

20

- Initial Sardana IDE: written in Java, long time ago...
- OpenDocument Flat XML Spreadsheet: fods_to_sar, sar_to_fods
- Excel Spreadsheet: xls_to_sar
- Expert macros: defctrl, defelem, defm, defmeas, undefctrl, undefelem, undefmeas, sar_info



- Spyder based Sardana editor
- Expert macros:
 - addmaclib,
 - relmaclib
 - relmac,
 - edctrl,
 - edmac,
 - prdef
 - commit_ctrllib



```

13
14
15 @macro([ "moveable", Type.Moveable, None, "moveable to move"],
16         ["position", Type.Float, None, "absolute position"] ])
17 def move(self, moveable, position):
18     """This macro moves a motor to the specified position"""
19     moveable.move(position)
20     self.output("Motor ended at ", moveable.getPosition())
21
22 class Loop(Macro, Hookable):
23     """A macro that executes a for loop. It accepts hooks."""
24     hints = { 'allowsHooks': ('pre-move', 'post-move', 'pre-acq', 'post-acq')
25
26     param_def = [['start', Type.Integer, None, 'start point'],
27                  ['stop', Type.Integer, None, 'end point'],
28                  ['step', Type.Integer, 1, 'step']]
29
30     result_def = [['result', Type.Integer, None, 'result']]
31
32     def hook1(self):
33         self.output("In hook 1")
34
35     def run(self, start, stop, step):
36         self.info("Starting loop")
37         self.hooks = [ (self.hook1, ['pre-acq'])]
38         for i in xrange(start, stop, step):
39             self.output("At step %d" % i)
40             self.flushOutput()
41
42             for hook, hints in self.hooks:
43                 hook()
44             self.info("Finished loop")
45             return i
46
47 class hooked_scan(Macro):
48     """An example on how to attach hooks to the various hook points of a scan.
49     """
50     param_def = [
51         ['motor', Type.Moveable, None, 'Motor to move'],
52         ['start_pos', Type.Float, None, 'Scan start position'],
53         ['end_pos', Type.Float, None, 'Scan final position'],
54         ['nr_interv', Type.Integer, None, 'Number of scan intervals'],
55         ['integ_time', Type.Float, None, 'Integration time']]
56
57     def hook1(self):
58         self.info("\thook1 execution")
59
60     def run(self, mot, start, end, nr, intt):
61         self.ascan, pars = self.createMacro("ascan", mot, start, end, nr, intt)
62         self.ascan.hooks = [(self.hook1, ['pre-acq'])]
63         self.runMacro(self.ascan)
64
65     @property
66     def data(self):
67         return self.ascan.data
68
69 @imacro()
70 def ask_number_of_points(self):
71     """Asks user for the number of points"""
72     nb_points = self.input("How many points?", data_type=Type.Integer)
73
74 @macro()
75 def j0_plot(self):
76     """Sample J0 at linspace(0, 20, 200)"""
77     x = linspace(0, 20, 200)
78     y = j0(x)
79     x1 = x[:10]
80     y1 = map(j0, x1)
81     self.pyplot.plot(x, y, label='J_0(x)') #
82     self.pyplot.plot(x1, y1, 'ro', label='J_0(integ)(x)')
83     self.pyplot.title(r'Verify J_0(x)=\frac{1}{\pi}\int_0^{\pi}\cos(x \sin\phi)')
84     self.pyplot.xlabel('$x$')
85     self.pyplot.legend()
86
87

```

- Known problems:
 - Pool accumulates motor groups – listeners of the position updates
 - Pool do not warn when deleting elements used by another elements (groups)
e.g. a motor can be deleted even if a pseudo motor depends on it;
an experimental channel can be deleted even if a measurement group uses it
 - measurement group can not be recreated from scratch (reusing the same name)
 - measurement group breaks when eliminating one of its initial experimental channels

- Known problems:
 - Pool accumulates motor groups – listeners of the position updates
 - Pool do not warn when deleting elements used by another elements (groups)
e.g. a motor can be deleted even if a pseudo motor depends on it;
an experimental channel can be deleted even if a measurement group uses it
 - measurement group can not be recreated from scratch (reusing the same name)
 - measurement group breaks when eliminating one of its initial experimental channels
- How do other institutes manage Sardana configuration?

Sardana configuration

- Known problems:
 - Pool accumulates motor groups – listeners of the position updates
 - Pool do not warn when deleting elements used by another elements (groups)
e.g. a motor can be deleted even if a pseudo motor depends on it;
an experimental channel can be deleted even if a measurement group uses it
 - measurement group can not be recreated from scratch (reusing the same name)
 - measurement group breaks when eliminating one of its initial experimental channels
- How do other institutes manage Sardana configuration?
- Do we need a Sardana configuration GUI before fixing the known configuration problems?

Questions? Comments?