

NeXus Files from Tango and Sardana

PANDATA WP5 / HDRI

Jan Koteński, Eugen Wintersberger

Deutsches Elektronen-Synchrotron



May 7, 2014

- NeXus/HDF5 files
 - HDRI/Pandata projects
- NeXus Writer Configuration
 - Components with DataSources and Strategy
- User Interfaces
 - Sardana + GUI
 - NeXus Recorder
 - Sardana + Macros
 - Beamline specific macros
 - Python scripts

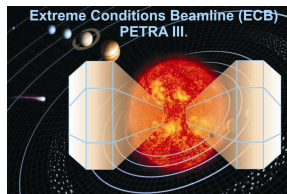
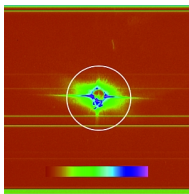
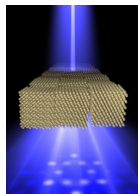
NeXus/HDF5 files

HDRI/Pandata projects

Why NeXus ?

- store all data in one container – NeXus file
- full description of experiment, metadata catalogs
searching via keywords, i.e. γ -Portal
- data provenance – sufficient details to allow reproducibility
managing, sharing, and reusing data
- application definitions for specific experiment types
common data structure in photon-science facilities

This way data can be managed efficiently



NeXus Files as a file system

They contain four types of **entity**: **groups**, **fields**, **attributes**, and **links**

- **group** – **folder** with a defined **NeXus type**; building block of the **NeXus structure**
- **field** – **data-set** with a name and a NeXus path related to its physical meaning
- **attribute** – **extra information** required to describe a particular group or field
- **link** – used to reference the plottable data from NXdata



NeXus Files as a file system

They contain four types of **entity**: **groups**, **fields**, **attributes**, and **links**

- **group** – **folder** with a defined **NeXus type**; building block of the **NeXus structure**
- **field** – **data-set** with a name and a NeXus path related to its **physical meaning**
- **attribute** – extra information required to describe a particular group or field
- **link** – used to reference the plottable data from NXdata



NeXus Files as a file system

They contain four types of **entity**: **groups**, **fields**, **attributes**, and **links**

- **group** – **folder** with a defined **NeXus type**; building block of the **NeXus structure**
- **field** – **data-set** with a name and a NeXus path related to its **physical meaning**
- **attribute** – **extra information** required to describe a particular **group** or **field**
- **link** – used to reference the plottable data from NXdata



NeXus Files as a file system

They contain four types of **entity**: **groups**, **fields**, **attributes**, and **links**

- **group** – **folder** with a defined **NeXus type**; building block of the **NeXus structure**
- **field** – **data-set** with a name and a NeXus path related to its **physical meaning**
- **attribute** – **extra information** required to describe a particular **group** or **field**
- **link** – used to **reference the plottable data** from NXdata



Simply creation of NeXus files in C++ (HDRI project)

- `libpnicore` provides
 - types of well defined size
 - templates for buffers and arrays
 - reader code to import data from proprietary formats
- `libpniio` classes
 - write NeXus files using HDF5 as its storage back-end
 - make the development independent of the NeXus API
- `python-pniio` Python bindings via the Python package

All libraries are actually in use to develop the software required to establish NeXus as a data format at DESY

<http://code.google.com/p/pni-libraries/>

<http://www.pni-hdri.de>

NeXus Writer Configuration

Components, DataSources and Strategy

Configuration Components

To store the data we need to know:

- Where? – NeXus path with physical meaning
- When and How? – writing strategy: INIT, STEP, FINAL, POSTRUN
- What? – DataSources: CLIENT, TANGO, DB, Python scripts

Configuration Components:

- XML Strings in the NXDL format extended by strategy and datasources tags
- correspond to one or more devices and like devices can be switch on/off
- set positions of stored data in the NeXus tree

Configuration Components

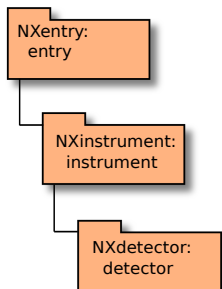
To store the data we need to know:

- Where? – NeXus path with physical meaning
- When and How? – writing strategy: INIT, STEP, FINAL, POSTRUN
- What? – DataSources: CLIENT, TANGO, DB, Python scripts

Configuration Components:

- XML Strings in the NXDL format extended by strategy and datasources tags
- correspond to one or more devices and like devices can be switch on/off
- set positions of stored data in the NeXus tree

Components and their merging

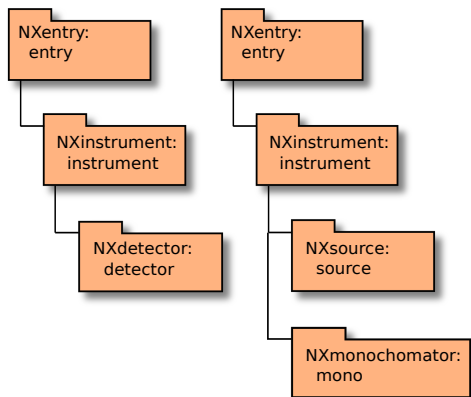


Detector
Component

Default
Component

Merged
Components

Components and their merging

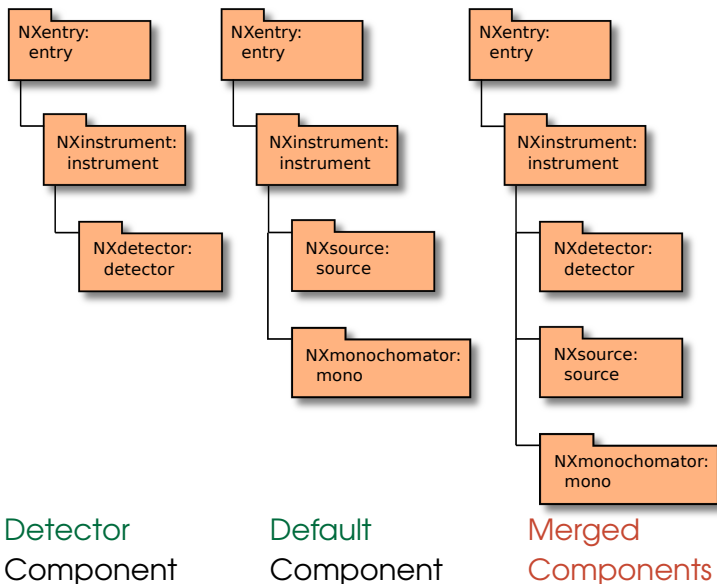


Detector
Component

Default
Component

Merged
Components

Components and their merging



Detector
Component

Default
Component

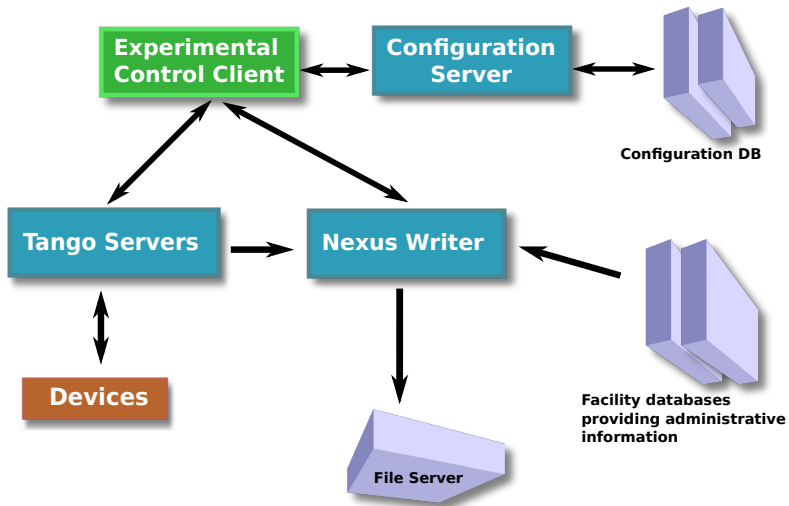
Merged
Components

Component Designer

The screenshot displays the NX5 Component Designer interface. The main window shows a tree view of components for 'scan34'. The 'DataSources' panel on the left lists 'exp_c02' through 'exp_c11'. A 'counter3' configuration dialog is open, showing 'Name: counter3', 'Units: m', and 'Value: \$datasources.exp_c03'. A second dialog for 'exp_c03 [DataSource]' is also open, showing 'Type: CLIENT', 'Name: exp_c03', and 'Record name: haso228k:10000/expchan/sis3820_exp/3'. The status bar at the bottom shows the current path and data source path.

The Configuration Client Tool allows to create configuration components as well as datasources (for FS-EC use)

NeXus Writer

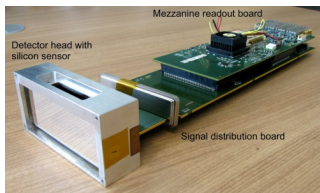


<http://code.google.com/p/nexdatas/>

Data Collector

For **fast detector data** is:

- **saved** locally by **Detector Tango Server**, e.g. lambda detector in NeXus files
- **merged with data** from NeXus Writer after performed experiment
- **merging** will be done by **Data Collector** (under discussion)



User Interfaces

GUI, Macros and Python

- Sardana + GUI
 - NeXus Component Selector
 - NeXus Recorder in Sardana
- Sardana + Macros
 - Beamline specific macros
 - NeXus Writer and Configuration Server used directly
 - examples:
 - Continuous scans with external trigger
 - Control of XA or MCS (PDM)
 - Control of SRS (PDM)
 - Control of SRS (PDM)
- Python scripts
 - ...

- Sardana + GUI
NeXus Component Selector
NeXus Recorder in Sardana
- Sardana + Macros
Beamline specific macros
 - NeXus Writer and Configuration Server used directly
 - examples:
 - Continuous scans with external trigger (Zebra & XIA & MCS) (P06)
 - Sweep Scans (P02)
- Python scripts
- ...

- Sardana + GUI
NeXus Component Selector
NeXus Recorder in Sardana
- Sardana + Macros
Beamline specific macros
 - NeXus Writer and Configuration Server used directly
 - examples:
 - Continuous scans with external trigger (Zebra & XIA & MCS) (P06)
 - Sweep Scans (P02)
- Python scripts
- ...

Client code example

A simple control client in Python:

```
from PyTango import DeviceProxy
dpx = DeviceProxy("p09/nxsdatawriter/01")
dpx.Init()
dpx.FileName = "/tmp/my_scan.nxs"
dpx.OpenFile()

ncs = DeviceProxy("p09/nxsconfigserver/01")
# create configuration from components
ncs.CreateConfiguration( ["ten_channel_detector",
                          "slits", "beamstop"])
# send XMLString for a new scan
dpx.XMLSettings = ncs.XMLString

dpx.JSONRecord = '{"data": {"parameterA":0.2}}'
dpx.OpenEntry()
# ...
```

Client code example

```
# experiment main loop
# (write data with strategy STEP)
for i in range(100):
    dpx.Record(
        '{"data": {"exp_c01":%s,"exp_c02":%s}}' % \
            (i*0.1, i*1.2))

# write final data (strategy FINAL)
# - close the entry - close the file
dpx.JSONRecord = '{"data": {"parameterB":0.3}}'
dpx.CloseEntry()
dpx.CloseFile()
```


First data on P02.1 viewed in Dawn

High Resolution Powder Diffraction Beamline: p02sweep.py script

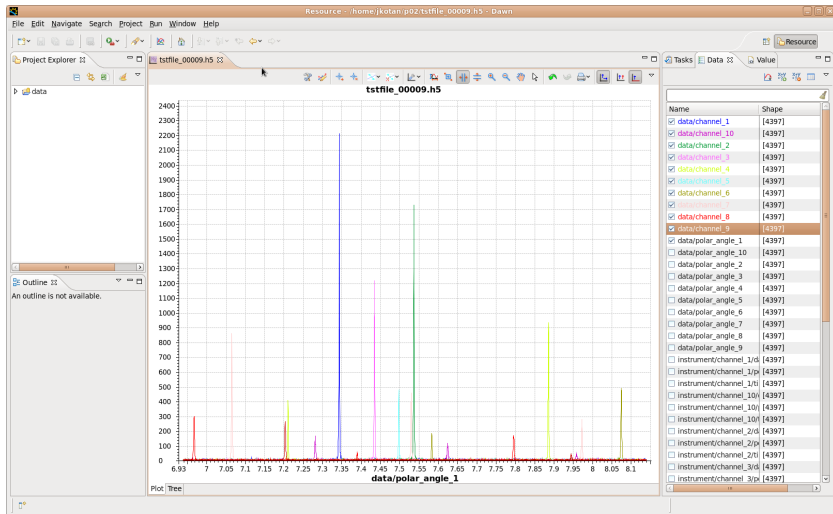
The screenshot displays the Dawn software interface. The main window shows a project tree on the left and a data table in the center. The project tree is expanded to show the 'data' folder, which contains a 'NexusConfigurationLogs' group and an 'entry' group. The 'entry' group contains a 'data' folder with 10 channels (channel_1 to channel_10) and 9 polar_angle entries (polar_angle_1 to polar_angle_9). The data table in the center lists the following data:

Name	Class	Dims	Description	Size
NexusConfigurationLogs	Group			
Nexus_entry_1_KML	Dataset	[1]	String, length = variat	
entry	Group			
data	Group			
channel_1	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_10	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_2	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_3	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_4	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_5	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_6	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_7	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_8	Dataset	[4397]	64-bit unsigned intege	34.35 KB
channel_9	Dataset	[4397]	64-bit unsigned intege	34.35 KB
polar_angle_1	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_10	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_2	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_3	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_4	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_5	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_6	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_7	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_8	Dataset	[4397]	64-bit floating-point	34.35 KB
polar_angle_9	Dataset	[4397]	64-bit floating-point	34.35 KB
end_time	Dataset	[1]	String, length = variat	
experiment_identifier	Dataset	[1]	String, length = variat	
instrument	Group			
monitor	Group			
sample	Group			
start_time	Dataset	[1]	String, length = variat	

The right-hand side of the interface shows a 'Value' pane with a list of data items and their shapes. The list includes 'data/channel_1' through 'data/channel_10', 'data/polar_angle_1' through 'data/polar_angle_10', and 'instrument/channel_1' through 'instrument/channel_3'. The 'data/channel_9' item is currently selected and highlighted in blue.

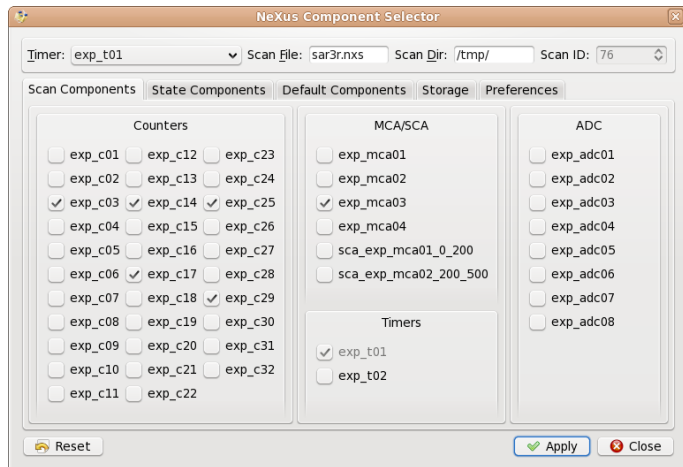
First data on P02.1 viewed in Dawn

High Resolution Powder Diffraction Beamline: p02sweep.py script



NeXus Component Selector

Device Selection Editor/View



Now everybody can use **Sardana!**

Experimental Channels from Sardana Pool

User scan in spock with the exp_mot04 motor.

```

jkotan@haso228k:jkotan
File Edit View Search Terminal Help

Door_demo1_1 [64]: ascan exp_mot04 0 1 20 0.2
Operation will be saved in /tmp/sar3r_00078.nxs (nxs)
Scan #78 started at Fri May 2 15:05:35 2014. It will take at least 0:00:04.300006
Moving to start positions...
#Pt No   exp_mot04  exp_c25  exp_mca03  exp_c29  exp_t01  exp_c17  exp_c14  exp_c03  dt
0        0         45      (2048,)    0        0.2      0        6        38      1.06836
1        0.05      82      (2048,)    0        0.2      0        18       64      1.8351
2        0.1       120     (2048,)    2        0.2      0        39       85      2.59182
3        0.15      147     (2048,)    12       0.2      9        77       99      3.3311
4        0.2       178     (2048,)    46       0.2      45       115      94      4.08706
5        0.25     157     (2048,)    112      0.2      72       144      63      4.82019
6        0.3       133     (2048,)    210      0.2      40       144      39      5.58663
7        0.35      0       (2048,)    231      0.2      9        121     121     6.34575
8        0.4       0       (2048,)    206      0.2      0        83      211     7.08857
9        0.45      0       (2048,)    141      0.2      0        49      205     7.8449
10       0.5       0       (2048,)    56       0.2      0        22      86      8.58351
11       0.55      0       (2048,)    115      0.2      0        8        18      9.33992
12       0.6       1       (2048,)    179      0.2      7        33      2       10.0941
13       0.65      8       (2048,)    226      0.2      34       67      0       10.8332
14       0.7       25      (2048,)    247      0.2      90       98      0       11.5913
15       0.75     60      (2048,)    204      0.2     138      124      0       12.3432
16       0.8       116     (2048,)    122      0.2     132      136      0       13.0831
17       0.85     170     (2048,)    66       0.2      76      124      0       13.8369
18       0.9       190     (2048,)    28       0.2      30       99      38      14.5933
19       0.95     182     (2048,)    8        0.2      5        61      60      15.3306
20       1         0       (2048,)    2        0.2      0        31      80      16.0883

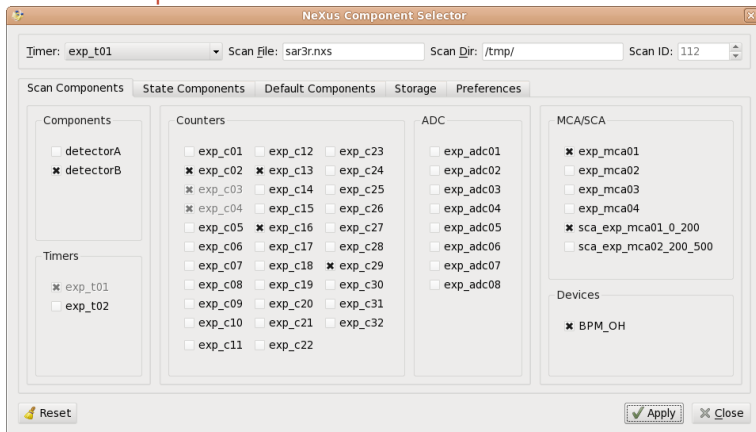
Operation saved in /tmp/sar3r_00078.nxs (nxs)
Scan #78 ended at Fri May 2 15:05:51 2014, taking 0:00:16.599937. Dead time 74.7% (motion dead time 32.6%)

Door_demo1_1 [65]: █

```

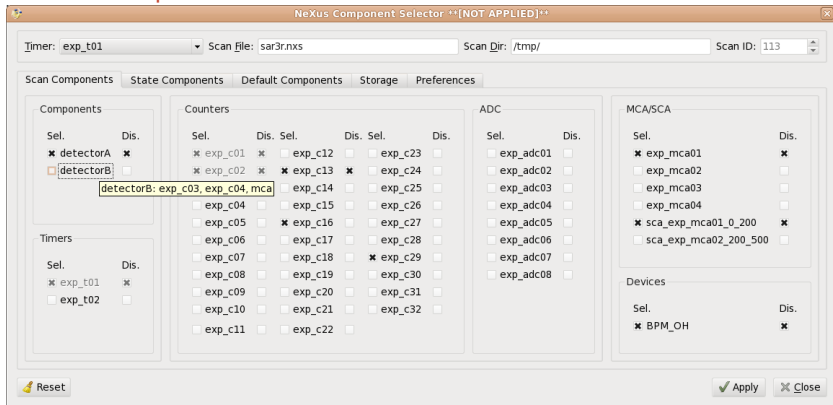
For NeXus the file extension is `.nxs`

Scan Components



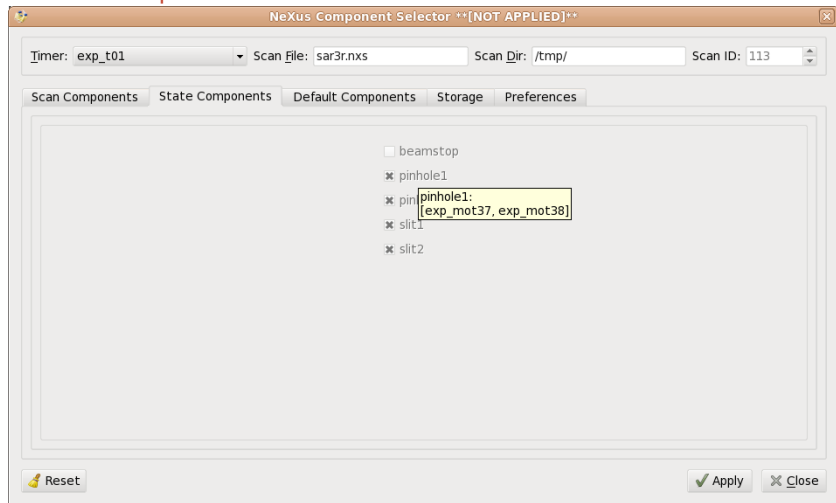
Selecting components containing other devices.

Scan Components



One can also disable display for Taurus.
Searching for the best user interface.

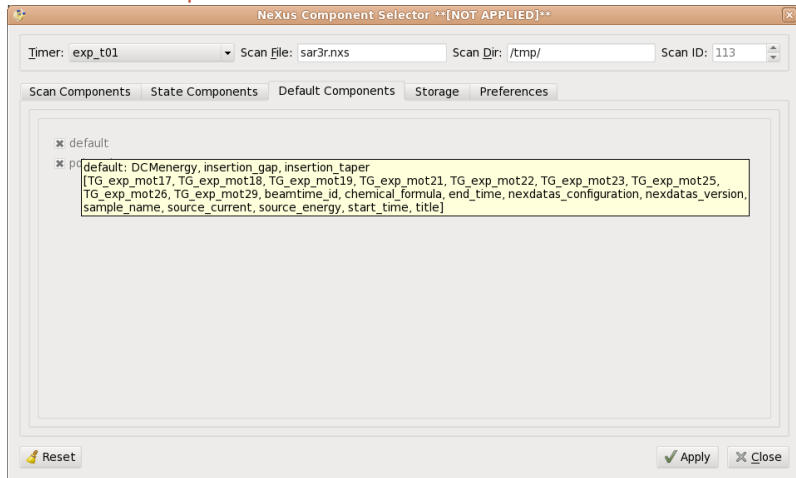
State Components



State components describe **experimental setup**.

They are **automatically deselected** if related to them **motors** are switch **off**.

Default Components



Default components describe mandatory data,
e.g. title, sample_name, beamtime_id, start_time, end_time,...

User scan in spock with the exp_mot04 motor.

```

jkotan@haso228k:jkotan
File Edit View Search Terminal Help

Door_demo1_1 [7]: ascan exp_mot04 0 1 20 0.2
Operation will be saved in /tmp/sar3r.nxs (nxs)
Scan #114 started at Wed May 7 09:21:53 2014. It will take at least 0:00:04.300006
Moving to start positions...
#Pt No   exp_mot04  exp_mca01  exp_c29  exp_t01  exp_c16  sca_exp_mca01_0_200  exp_c13  exp_c01  exp_c02  exp_c03  dt
0        0          (2048,)    0         0.2      3         3064             6        208      84       24       1.61741
1        0.05      (2048,)    0         0.2      0          2              15       166     35       37       2.3801
2        0.1       (2048,)    0         0.2      0        20176           31       108     62       50       3.14396
3        0.15      (2048,)    0         0.2      0        6408            58       63      93       53       3.87836
4        0.2       (2048,)    0         0.2      0          0             82       33     121      44       4.65853
5        0.25      (2048,)    1         0.2      0         0             107      16     134     30       5.48928
6        0.3       (2048,)    11        0.2      0          0             108      6     140     14       6.13104
7        0.35      (2048,)    46        0.2      0        3760            97      2     114     41       6.80056
8        0.4       (2048,)    109       0.2      0          0             66      0     88     93       7.62838
9        0.45      (2048,)    167       0.2      0        11600           105     9     57     146     8.3797
10       0.5       (2048,)    156       0.2      1          0             112     20     32     208     9.12957
11       0.55      (2048,)    0         0.2      12         0             104     44     15     223     9.87812
12       0.6       (2048,)    0         0.2      50        13040           74      7     6     179    10.6106
13       0.65      (2048,)    0         0.2      124        0              49     115    0     121    11.3805
14       0.7       (2048,)    0         0.2      187        0              28     147    2     64     12.124
15       0.75      (2048,)    0         0.2      163        9162            13     178    9     26     12.8862
16       0.8       (2048,)    2         0.2      89         0              5     170    30    8     13.636
17       0.85      (2048,)    7         0.2      27         0              1     155    73    2     14.3866
18       0.9       (2048,)    19        0.2      5          14292           0     111   121    3     15.135
19       0.95      (2048,)    39        0.2      0         13404           0     74    150    26    15.8856
20       1         (2048,)    66        0.2      0          0              0     41    135    59    16.6334

Operation saved in /tmp/sar3r.nxs (nxs)
Scan #114 ended at Wed May 7 09:22:10 2014, taking 0:00:17.231842.Dead time 75.6% (motion dead time 18.5%)

Door_demo1_1 [8]: █

```

Here, scans are stored in one file: sar3r.nxs

NeXus Component Selector – for EXPERTS

Storage parameters

The screenshot shows the 'NeXus Component Selector' application window. At the top, there are input fields for 'Timer: exp_t01', 'Scan File: sar3r.nxs', 'Scan Dir: /tmp/', and 'Scan ID: 113'. Below these are tabs for 'Scan Components', 'State Components', 'Default Components', 'Storage', and 'Preferences'. The 'Storage' tab is active, showing several sections: 'Measurement:' with 'Door: Door/demo1/1' and 'MntGrp: nxsmntgrp'; 'Devices:' with 'Config Device: p09/mcs/r228' and 'Writer Device: p09/tdw/r228'; 'Configuration:' with 'Load ...' and 'Save ...' buttons; 'Dynamic Components:' with a checked 'Links' option and a 'Default NeXus Path:' field containing '/entry\$var.serialno:NXentry/NXinstrument/NXcollection'; and 'Others:' with a checked 'Append Scans' option and a 'Time Zone: Europe/Berlin' field. At the bottom, there are 'Reset', 'Apply', and 'Close' buttons.

Append Scans – scans stored in one NeXus file

NeXus Component Selector – for EXPERTS

Preferences

The screenshot shows the 'NeXus Component Selector' application window with the 'Preferences' tab selected. The window title is 'NeXus Component Selector'. At the top, there are input fields for 'Timer: exp_t01', 'Scan File: sar3r.nxs', 'Scan Dir: /tmp/', and 'Scan ID: 113'. Below these are five tabs: 'Scan Components', 'State Components', 'Default Components', 'Storage', and 'Preferences'. The 'Preferences' tab is active and contains several sections:

- View:** A dropdown menu set to 'CheckBoxes (NL)' and a 'Row Max.: 11' spinner.
- Selector Server:** A text input field containing 'test/nxsrecselector/01'.
- Frames:** A text input field containing the JSON array: `[[["Components",1],["Timers",5]],[[["Counters",4],["ADC",3]],[[["MCA/SCA",6],["Devices",0]]]]`
- Groups:** A text input field containing the JSON object: `{ "3":["exp_adc",0], "4":["exp_c",0], "5":["exp_t",0], "6":["exp_mca",0], "sca_exp_*.0" }`
- Profile:** Two buttons labeled 'Load' and 'Save'.

At the bottom of the dialog, there are three buttons: 'Reset' (with a trash icon), 'Apply' (with a checkmark icon), and 'Close' (with an 'X' icon).

Changing the scan components layout.

NeXus Storage Software

- Creating fully describing files
- Configuration in components with datasources and strategy
- User Interfaces:
 - Sardana + GUI, Sardana + Macros, Python scripts
- Deployment:
 - roll-out software (FS-EC)
 - provide information about experiment (Beamline Scientists)
 - provide components (FS-EC)
 - select devices/components in Selector GUI and run scans (Users)

NeXus Storage Software

- Creating fully describing files
- Configuration in components with datasources and strategy
- User Interfaces:
 - Sardana + GUI, Sardana + Macros, Python scripts
- Deployment:
 - roll-out software (FS-EC)
 - provide information about experiment (Beamline Scientists)
 - provide components (FS-EC)
 - select devices/components in Selector GUI and run scans (Users)

NeXus Storage Software

- Creating fully describing files
- Configuration in components with datasources and strategy
- User Interfaces:
 - Sardana + GUI, Sardana + Macros, Python scripts
- Deployment:
 - roll-out software (FS-EC)
 - provide information about experiment (Beamline Scientists)
 - provide components (FS-EC)
 - select devices/components in Selector GUI and run scans (Users)

NeXus Storage Software

- Creating fully describing files
- Configuration in components with datasources and strategy
- User Interfaces:
 - Sardana + GUI, Sardana + Macros, Python scripts
- Deployment:
 - roll-out software (FS-EC)
 - provide information about experiment (Beamline Scientists)
 - provide components (FS-EC)
 - select devices/components in Selector GUI and run scans (Users)

NeXus Storage Software

- Creating fully describing files
- Configuration in components with datasources and strategy
- User Interfaces:
 - Sardana + GUI, Sardana + Macros, Python scripts
- Deployment:
 - roll-out software (FS-EC)
 - provide information about experiment (Beamline Scientists)
 - provide components (FS-EC)
 - select devices/components in Selector GUI and run scans (Users)

Thank You