The Current State of NeXus

Mark Könnecke, on behalf of the NIAC

Paul Scherrer Institute
Switzerland

October 21, 2011

The Predicament of the Traveling Scientist

- A different data format wherever she goes

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Spends lots of time converting formats or writing readers

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Spends lots of time converting formats or writing readers
- Waits even longer to load data from inefficient data formats

NeXus

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Spends lots of time converting formats or writing readers
- Waits even longer to load data from inefficient data formats
- DA requires N files in different formats, notes, local knowledge

NeXus

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Spends lots of time converting formats or writing readers
- Waits even longer to load data from inefficient data formats
- DA requires N files in different formats, notes, local knowledge
- Cannot read her collaborators data

NeXus

NeXus Mission

- Definition of a standard data format
  - Rules
  - Validation tools
- Promotion of NeXus
  - Documentation
  - NeXus API
  - Outreach to the scientific community

- Complete data for typical use
- Full Beamline Description (FBD)
- Extendable, add additional data as you please
- Self describing
- Easy automatic plotting
- Platform independent, public domain, efficient
- Suitable for a wild variety of applications

NeXus History

- Devised from three independent proposals by Jonathan Tischler, APS, Przemek Klosowski, NIST and Mark Koennecke, ISIS, PSI in 94-96
- Improved during various NOBUGS conferences
- NeXus International Advisory Committee, NIAC, since 2003
- Since 2003 yearly meetings of the NIAC
- We already considered many issues!
- Except for one year, we never had money to develop NeXus

NeXus Levels

1. Physical file format and API for accessing files
2. Rules for storing data in files
3. Component and application definitions
4. NeXus Utilities

Physical File Format

- Portable, self describing, extendable, public domain
- Hierarchical data format, NCSA, HDF-4, later HDF-5
- HDF-5:
  - grouping support
  - on the fly compression
  - reading/writing subsets
  - Dimensions appendable
  - Public domain C, F77 access library
  - Used by: NASA, Boing, the weathermen, ...
- XML for those who wish to edit their data

NeXus

NeXus API

- NeXus-API hides complex HDF API
- Transparent access to all three supported physical file formats
- ANSI-C implementation
- Bindings: C++, F77, Java, python, IDL, SWIG
- January, 4, 2010: 1311217 files processed at PSI alone
- NAPI use not mandatory!

NeXus API Example

```
nxfile = nxs.open('hrpt2008n152088.hdf','r')
nxfile.openpath('/entry1/data1/two_theta')
x = nxfile.getdata()
nxfile.openpath('/entry1/data1/counts')
y = nxfile.getdata()
nxfile.openpath('/entry1/title')
txt = nxfile.getdata()
nxfile.close()

plot(x,y)
xlabel('two theta')
ylabel('counts')
title(txt)
show()
```

NeXus Objects

- Files
- Groups identified by name and a classname beginning with NX
- Scientific data sets
- Attributes
- Links

- McStas Coordinate System
- Angle based polar coordinate system
- General axis and transformations
- Full mapping imageCIF - NeXus now possible

entry:NXentry
- sample:NXsample
- instrument:NXinstrument
  - source:NXsource
  - velocity_selector:NXvelocity_selector
  - detector:NXdetector
    - data[xsize,ysize], signal=1 (1)
- control:NXmonitor
  - data
- data:NXdata
  - link to (1)

NeXus Processed Data File Structure

entry:NXentry
- sample:NXsample
- processing_name:NXprocess
  - program
  - version
  - parameters:NXparameter
    - raw_file
- data:NXdata
  - data[nx,ny,nz], signal=1

entry:NXentry
- sample:NXsample
- instrument:NXinstrument
- .....
- sas:NXsubentry
  - sample:NXsample
  - instrument:NXinstrument
    - source:NXsource
    - velocity_selector:NXvelocity_selector
    - detector:NXdetector
      - data[xsize,ysize], signal=1 (1)
  - control:NXmonitor
    - data
  - data:NXdata
    - link to (1)

entry,NXentry
- measurement:NXcollection
  - positions:NXcollection
    - om
    - two_theta
  - scalars:NXcollection
    - title
    - wavelength
  - data:NXdata
    - detector1
    - mca5

Why the Hierarchy??

- Supports self description and allows short names in components

Why the Hierarchy??

- Supports self description and allows short names in components
- Name, classname pair allows for multiple components of the same type

Why the Hierarchy??

- Supports self description and allows short names in components
- Name, classname pair allows for multiple components of the same type
- NXentry allows for multiple datasets in the same file

Why the Hierarchy??

- Supports self description and allows short names in components
- Name, classname pair allows for multiple components of the same type
- NXentry allows for multiple datasets in the same file
- NXdata supports automatic plotting

Why the Hierarchy??

- Supports self description and allows short names in components
- Name, classname pair allows for multiple components of the same type
- NXentry allows for multiple datasets in the same file
- NXdata supports automatic plotting
- Hierarchy makes namespace manageable for full beamline descriptions

Why the Hierarchy??

- Supports self description and allows short names in components
- Name, classname pair allows for multiple components of the same type
- NXentry allows for multiple datasets in the same file
- NXdata supports automatic plotting
- Hierarchy makes namespace manageable for full beamline descriptions
- Effort required once when writing, use n times

Rules for Storing Data Items

- Store physical values in C storage order; exceptions allowed
- Use NeXus components and dictionary names
- Missing names will be quickly accepted by the NIAC
- Names: full words separated by _
- Specify units in same format as used by UDunits
- Application definitions may restrict units
- There are rules which associate axis with data

- Come in all shapes and sizes
- Captured by rules:
  - Store all varied parameters as arrays of length NP at the appropriate place in the NeXus hierarchy
  - For multi detectors, NP, number of scan points is always the first dimension
  - In NXdata: create links to counts and varied variables
- Rasterisation is treated similar to scans

Scan Example: sample rotation, area detector

entry:NXentry
- sample:NXsample
  - rotation_angle[NP], axis=1 (1)
- instrument:NXinstrument
  - detector:NXdetector
    - data[NP,xsize,ysize],signal=1 (2)
- control:NXmonitor
  - data[NP]
- data:NXdata
  - link to (1)
  - link to (2)

NeXus

NeXus Component and Application Definitions

- Component definitions: dictionaries of allowed field names for the various NeXus groups
- Application Definitions
  - Define what has to be in a NeXus file for a certain application
  - Defines standards
  - Another view: Contract between file producers and users about what has to be in a NeXus file for a well defined purpose
  - Validation by NXvalidate
- Written in NeXus Definition Language, NXDL

All Base Classes

NXaperture
NXattenuator
NXbeam_stop
NXbeam
NXbending_magnet
NXcharacterization
NXcollimator
NXcrystal
NXdata
NXdetector
NXdisk_chopper
NXentry
NXenvironment
NXevent_data
NXfermi_chopper
NXfilter
NXflipper
NXgeometry
NXguide
NXinsertion_device
NXinstrument
NXlog
NXmirror
NXmoderator
NXmonitor
NXmonochromator
NXnote
NXorientation
NXparameters
NXpolarizer
NXprocess
NXsample
NXsensor
NXshape
NXsource
NXtranslation
NXuser
NXvelocity_selector
NXbending_magnet
NXxraylens
NXcapillary

NeXus

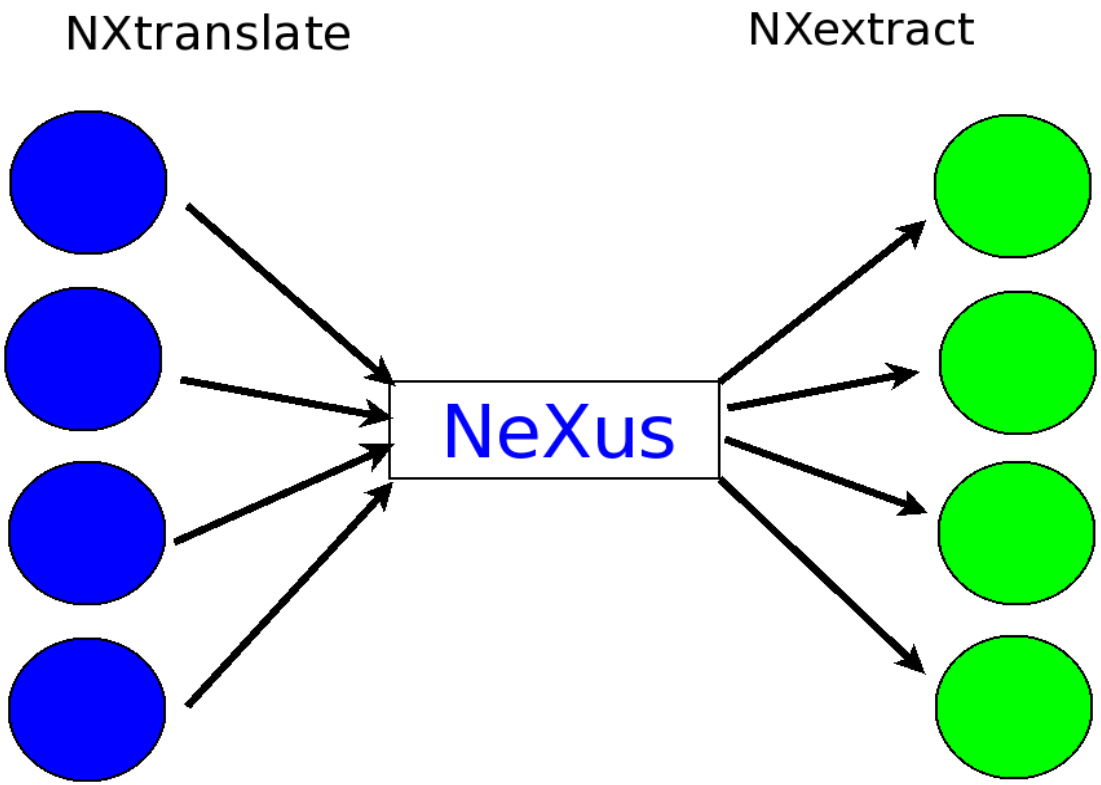
Available NeXus Application Definitions

NXarchive
NXreftof
NXtas
NXtomophase
NXxnb
NXtomoproc
NXindirectof
NXsastof
NXtofsingle
NXmonopd
NXsas
NXtofraw
NXxeuler
NXxrot
NXtofsingle
NXiqproc
NXsqom
NXxas
NXrefscan
NXscan
NXtomo
NXxkappa
NXiqproc
NXdirectof
NXlauetof
NXtofraw
NXxasproc

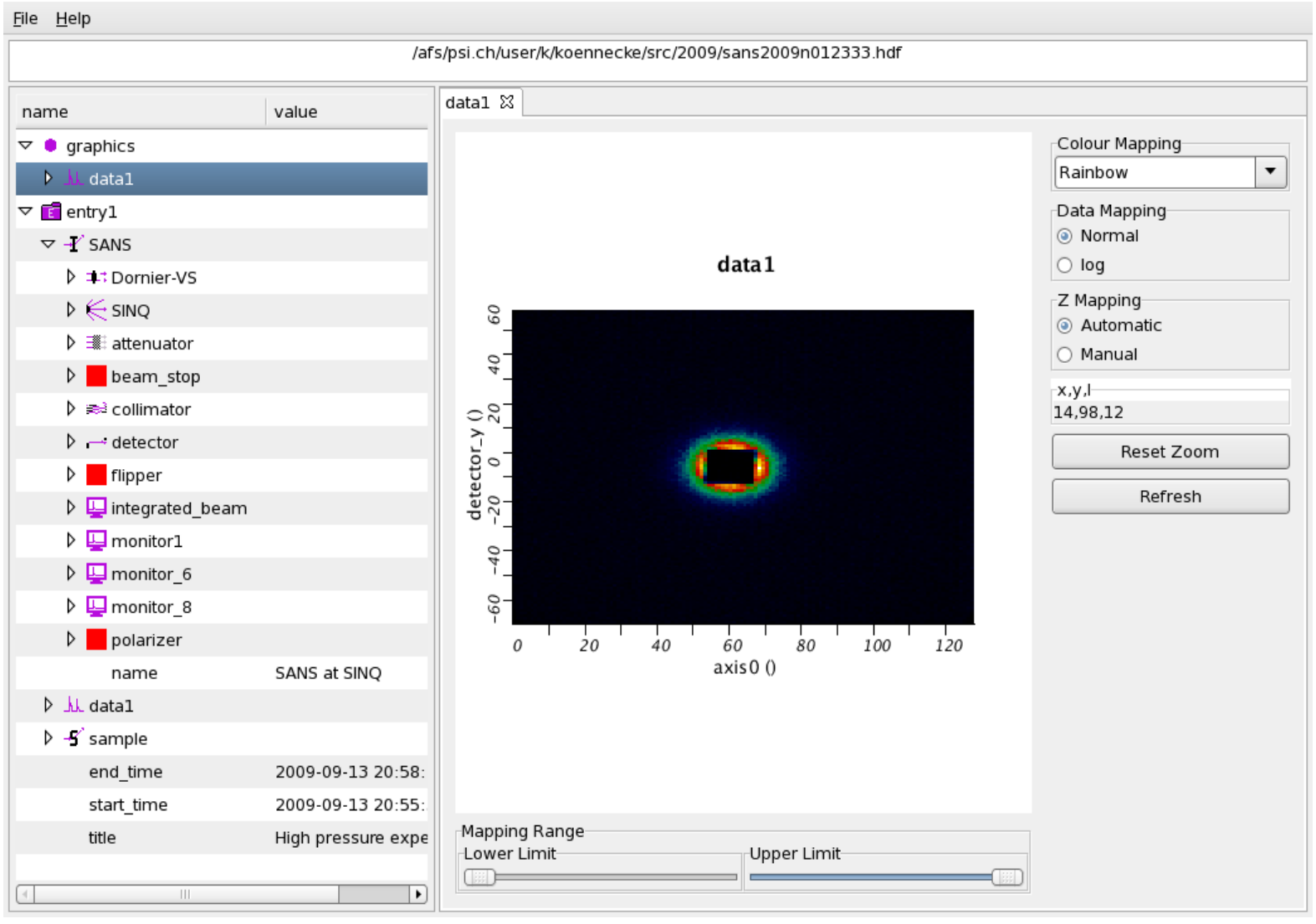
NeXus Tools

nxbrowse	CLI NeXus browser
nxtree	prints NeXus tree
NXmeta	dumps all NeXus meta data
nxtranslate	transforms into NeXus
nxvalid	validates NeXus files
nxextract	converts from neXus to ASCII and binary
nxplot	plots any NeXus file

NeXus Conversions



NXplot


File Help
/afs/psi.ch/user/k/koennecke/src/2009/sans2009n012333.hdf
name
value
graphics
data1
entry1
SANS
Dornier-VS
SINQ
attenuator
beam_stop
collimator
detector
flipper
integrated_beam
monitor1
monitor_6
monitor_8
polarizer
name
SANS at SINQ
data1
sample
end_time
2009-09-13 20:58:
start_time
2009-09-13 20:55:
title
High pressure expe
data1
data1
detector_y ()
axis0 ()
Colour Mapping
Rainbow
Data Mapping
Normal
log
Z Mapping
Automatic
Manual
x,y,I
14,98,12
Reset Zoom
Refresh
Mapping Range
Lower Limit
Upper Limit

Other Systems Using NeXus

- DANSE
- DAVE
- FABLE (ESRF)
- ISAW
- LAMP
- openGenie
- ICAT
- Mantid
- openGDA
- All HDF tools

Challenge 1 in science you are supposed to do new, non standard, things. These of course cannot be easily cast into a standard.

Data Format Challenges

Challenge 1 in science you are supposed to do new, non standard, things. These of course cannot be easily cast into a standard.

Challenge 2 in order to establish a standard a lot of people need to agree

NeXus

Data Format Challenges

Challenge 1 in science you are supposed to do new, non standard, things. These of course cannot be easily cast into a standard.

Challenge 2 in order to establish a standard a lot of people need to agree

Challenge 3 a standard requires scarce scientific programming resources for adoption

NeXus

Benefit 1 By using a discoverable data format like NeXus, XML, HDF-5, people can at least figure out what is in the data file.

Data Format Benefits

Benefit 1 By using a discoverable data format like NeXus, XML, HDF-5, people can at least figure out what is in the data file.

Benefit 2 Using predefined names from a dictionary gives meaning to the data in a file.

Data Format Benefits

Benefit 1 By using a discoverable data format like NeXus, XML, HDF-5, people can at least figure out what is in the data file.

Benefit 2 Using predefined names from a dictionary gives meaning to the data in a file.

Benefit 3 Using a shared API reduces learning costs and increases application stability.

Data Format Benefits

Benefit 1 By using a discoverable data format like NeXus, XML, HDF-5, people can at least figure out what is in the data file.

Benefit 2 Using predefined names from a dictionary gives meaning to the data in a file.

Benefit 3 Using a shared API reduces learning costs and increases application stability.

Benefit 4 With NeXus, HDF-5, plus professional programming techniques a DA application can read any file which contains the required data.

Benefit 5 Storing as much data as possible increases the likelihood that the needed data is actually on file, even for unforeseen uses.

NeXus

More Benefits

Benefit 5 Storing as much data as possible increases the likelihood that the needed data is actually on file, even for unforeseen uses.

Benefit 6 Storing as much data as possible allows to track down causes of funny results

NeXus

More Benefits

Benefit 5 Storing as much data as possible increases the likelihood that the needed data is actually on file, even for unforeseen uses.

Benefit 6 Storing as much data as possible allows to track down causes of funny results

Benefit 7 Storing as much data as possible helps to protect against scientific fraud

NeXus

More Benefits

Benefit 5 Storing as much data as possible increases the likelihood that the needed data is actually on file, even for unforeseen uses.

Benefit 6 Storing as much data as possible allows to track down causes of funny results

Benefit 7 Storing as much data as possible helps to protect against scientific fraud

Benefit 8 Application Definitions make all data handling problems go away

NeXus

Who commits to NeXus?

What You Can Do with NeXus

1. Store and archive data from a wild variety of instruments
2. Store processed data
3. Store a complete workflow from raw data to publication ready data in several NXentries in one file
4. Store a set of related experiments in one file
5. Define strict and validatable standards

What the NIAC has done for You

1. Define and documented hundreds of data item names
2. Developed structures to store complex instruments
3. Considered coordinate systems at great length
4. Developed standards for many instrument types
5. Developed tools to validate standards
6. Developed a simplified API to protect you from the HDF APIs

NeXus

Breaking News

- Dectris has dediced to store Eiger data in HDF-5 using NeXus conventions
- Programming model:
  - Dectris writes HDF-5 file with NXdetector
  - Local DAQ-system adds beamline metadata

Breaking News

- Dectris has dediced to store Eiger data in HDF-5 using NeXus conventions
- Programming model:
  - Dectris writes HDF-5 file with NXdetector
  - Local DAQ-system adds beamline metadata
- For Mark Rivers: there will be an API too

Conclusion

- NeXus is a mature and capable data format
- There is no other standard then NeXus on the horizon
- New things are developed with NeXus everywhere, uptake at established sites is slow
- You are invited to join the NIAC and contribute to NeXus
- More information: http://www.nexusformat.org