NeXus: A DataFormat for x-ray, n- and muon Scattering

Mark Könnecke, on behalf of the NIAC

Paul Scherrer Institute
Switzerland

August 5, 2013

The Predicament of the Traveling Scientist

- A different data format wherever she goes

NeXus

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Waists time converting formats or writing readers

NeXus

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Waists time converting formats or writing readers
- Waists time loading data from inefficient data formats

NeXus

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Waists time converting formats or writing readers
- Waists time loading data from inefficient data formats
- Needs multiple files in different formats, local knowledge, lab books etc to analyse data

NeXus

The Predicament of the Traveling Scientist

- A different data format wherever she goes
- Waists time converting formats or writing readers
- Waists time loading data from inefficient data formats
- Needs multiple files in different formats, local knowledge, lab books etc to analyse data
- Cannot read her collaborators data

NeXus Mission

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

NeXus Design

- 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
- Support X-rays, muSR and neutron scattering techniques
- Express validatable standards

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
- Strive to have a representative of each facility in NIAC
- Voluntary effort: 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
- Efficient enough for large data arrays from next generation sources

NeXus

- Portable, self describing, extendable, public domain
- Efficient enough for large data arrays from next generation sources
- HDF-5
- NeXus adds names and rules to HDF-5

Physical File Format

- Portable, self describing, extendable, public domain
- Efficient enough for large data arrays from next generation sources
- HDF-5
- NeXus adds names and rules to HDF-5
- Historical: HDF-4, XML, NeXus-API
- I/O to NeXus files with either HDF-5 or the NeXus-API

- Efficient binary format
- Support for hierarchy: in file filesystem
- On the fly compression
- Reading/writing subsets of data
- Dimensions appendable
- Public domain C, F77 access library
- Well maintained bindings to python, Java
- Well supported by many scientific tools
- Used by: NASA, Boeing, Deutsche Bank, HPC, the weathermen, ...

NeXus

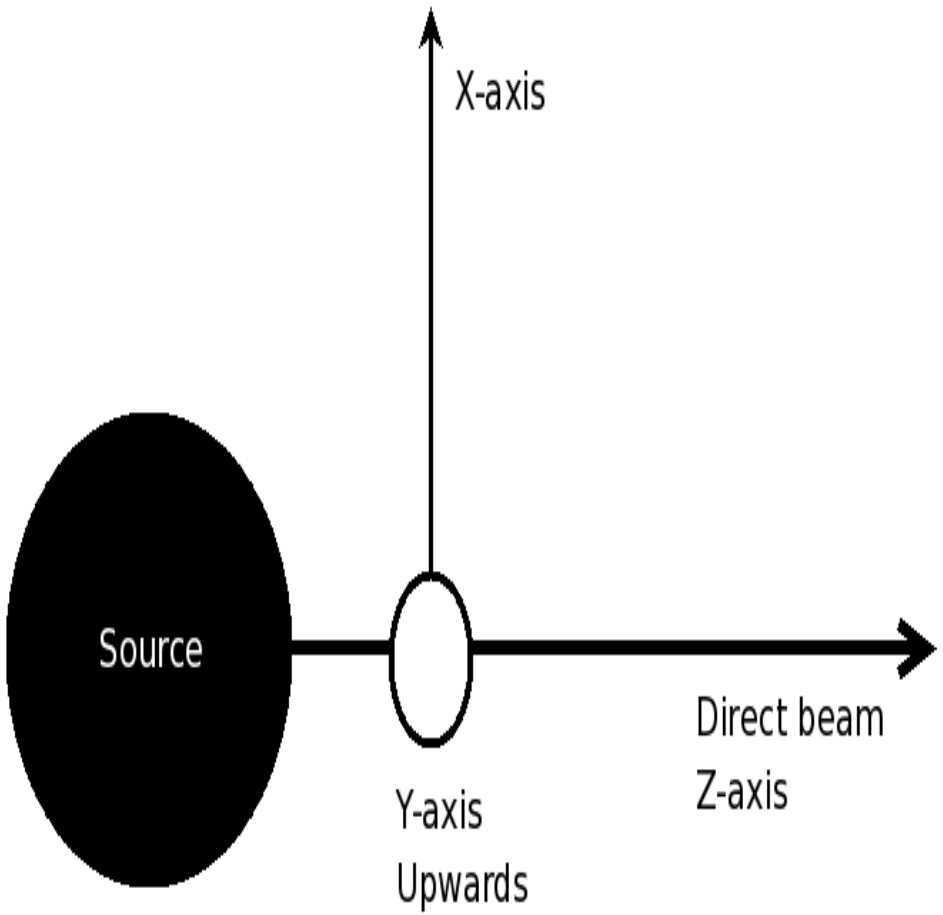
- Support for parallel processing
- Limited image support through conventions
- Has a table API
- User data types analog to C structure
- Supported and maintained by the HDF group
- Rule of thumb: if you need a feature it is cheaper to pay the HDF group rather then write an own format!
- www.hdfgroup.org

NeXus

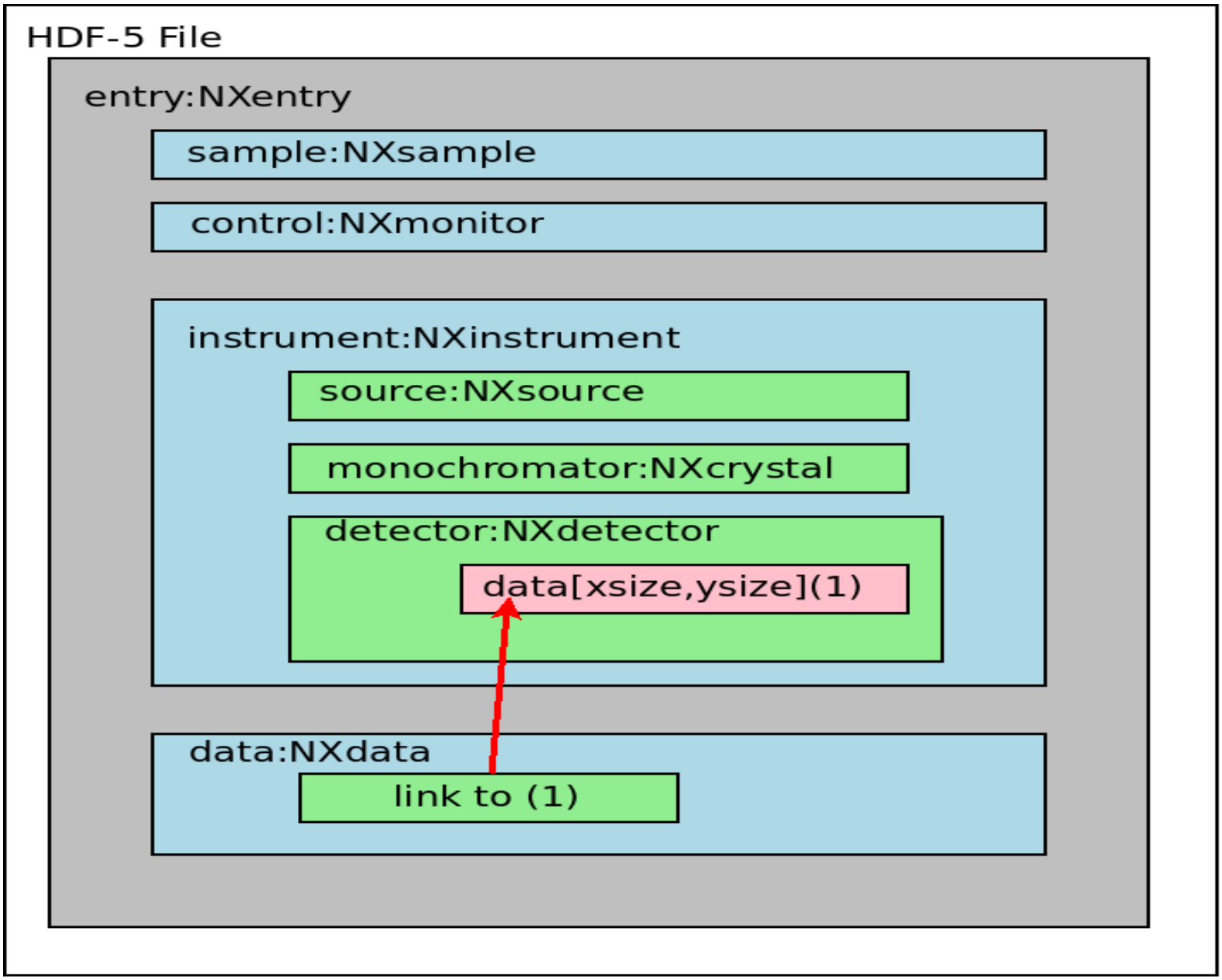
NeXus Objects

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

- McStas Coordinate System
- NeXus stole the CIF way of storing translations, rotations and dependencies

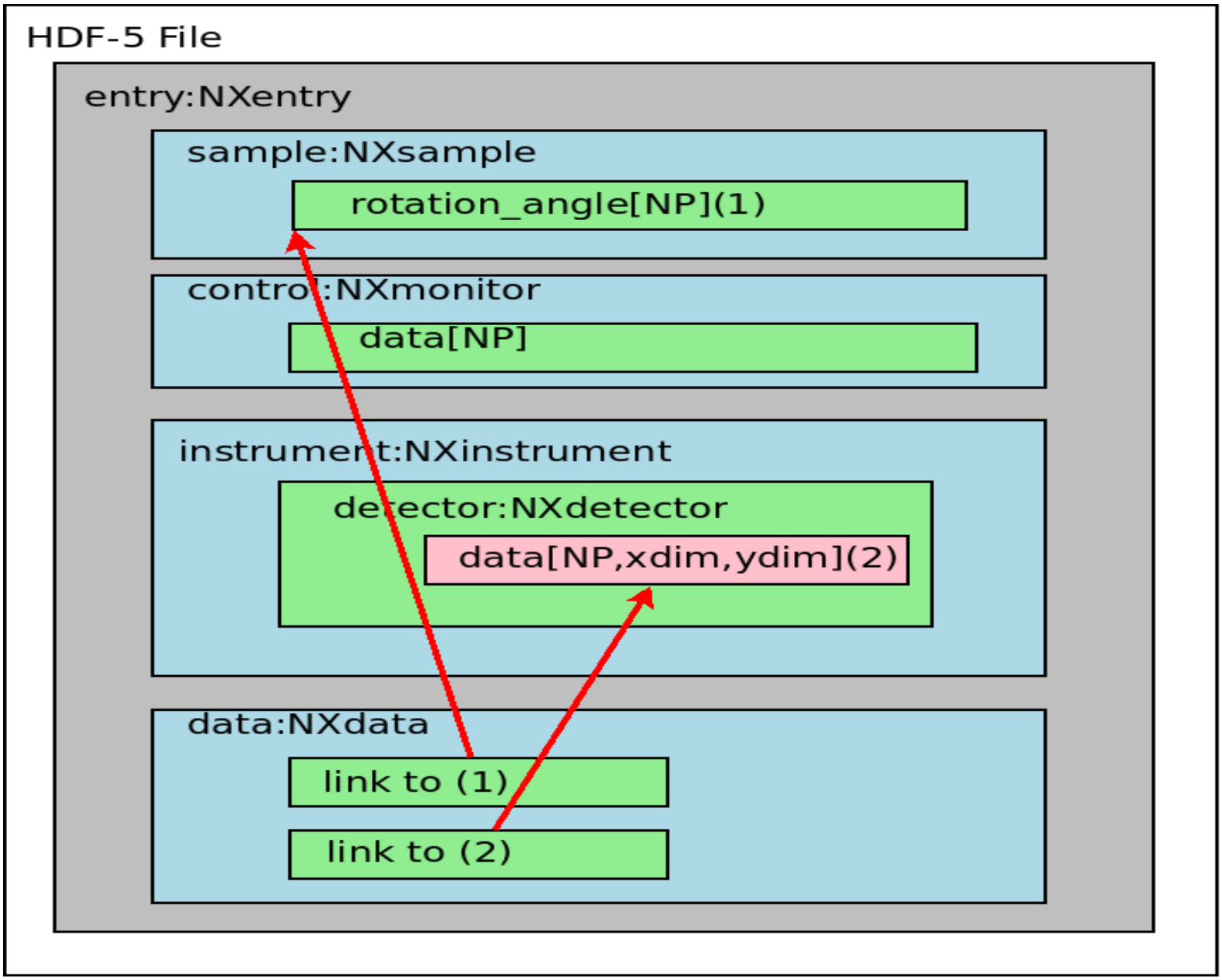


NeXus Raw Data File Structure



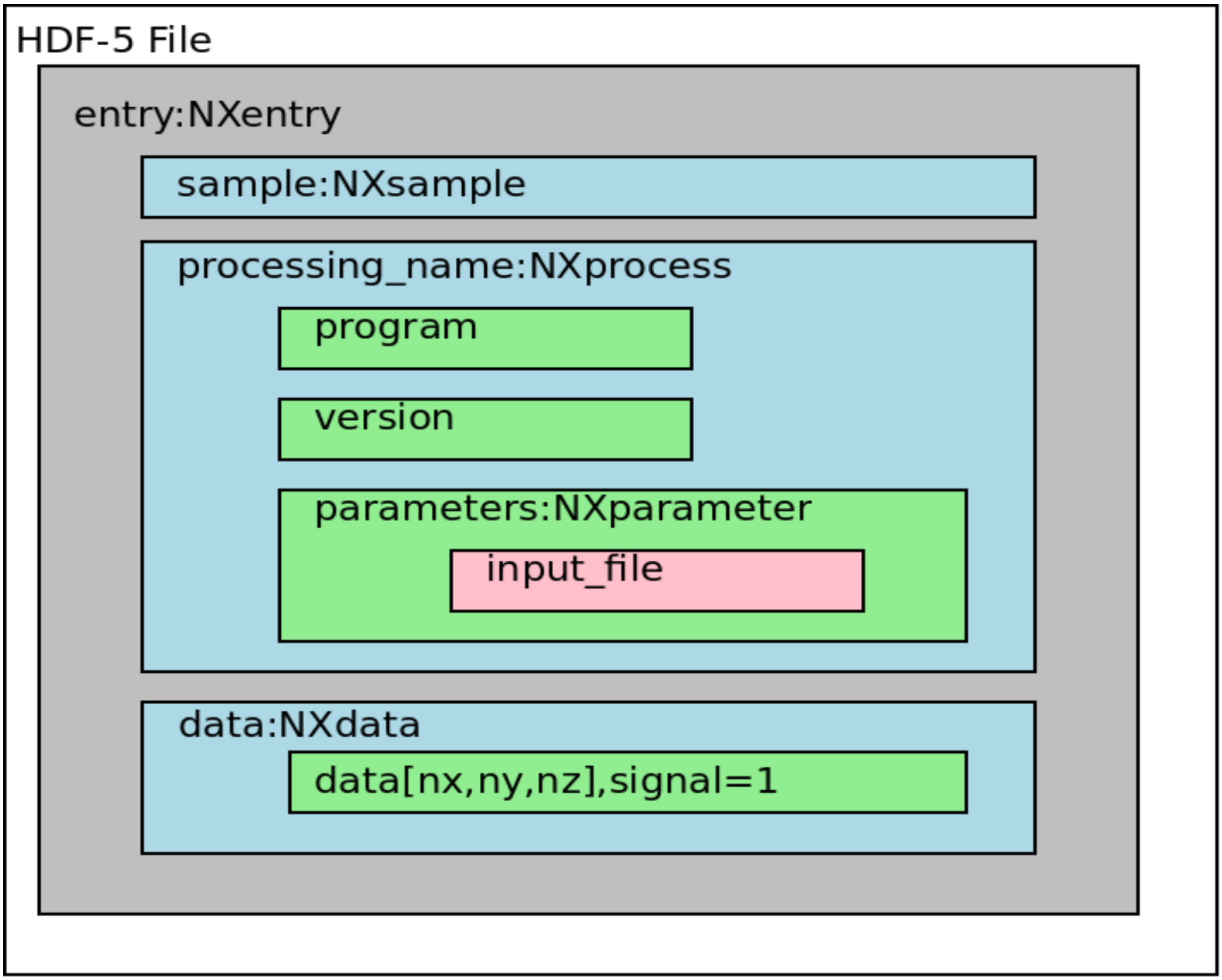
- Come in all shapes and sizes
- Captured by rules:
  - NP is the number of scan points
  - Store all varied parameters as arrays of length NP at the appropriate place in the NeXus hierarchy
  - For area detectors, NP is always the first dimension
  - In NXdata: create links to counts and varied variables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

Scan Example: sample rotation, area decetor



NeXus

Processed Data Structure



NeXus

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

Further NeXus Rules

- Rules for storing single data items on file
- Rules for associating axes with data
- NeXus reserves the prefix NX for group names
- Naming rules

NeXus Base Classes

- Documented dictionaries of allowed names within NeXus groups

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 asks for a FBD
- For a certain purpose only 10-30 data items are really required

NeXus Application Definitions

- NeXus asks for a FBD
- For a certain purpose only 10-30 data items are really required
- The NeXus Application Definition defines these
- Defines a standard for a technique or use case
- Validatable

NeXus Application Definitions

- NeXus asks for a FBD
- For a certain purpose only 10-30 data items are really required
- The NeXus Application Definition defines these
- Defines a standard for a technique or use case
- Validatable
- Application definitions and component definitions written in NXDL

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
nxvalidate	validates NeXus files against a NXDL application definition
nxextract	converts from NeXus to ASCII and binary
nxplot	plots any NeXus file

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

NeXus Benefits

Benefit 1 By using a discoverable data format like NeXus, 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.

NeXus Benefits

Benefit 1 By using a discoverable data format like NeXus, 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.

NeXus Benefits

Benefit 1 By using a discoverable data format like NeXus, 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

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

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

Benefit 9 NeXus allows to store a complete workflow in one file in different NXentries

NeXus Usage

- Soleil: 20 out of 26 instruments do NeXus, 2 mill files
- PSI-SINQ: 11 from 16 instrument on NeXus, 1.4 Mill files
- Lujan/LANL: 11 instruments, no change, 1 million files
- ANSTO: 7 going to 10
- KEK: 10, 6 planned
- SNS: 14,3 in the pipeline
- DESY: 0, 11 in 2 Jahren
- Diamond: 7 NeXus only, 17 writing, moving to 18 as primary format
- ISIS: 8 using, 20 writing, planned: 20 using
- Less intense users:
- PSI-SLS: 2 planned,
- ESRF: 2 beamlines, limited to NXentry, NXcollection, NXdata, moving to 4
- HZB: 3 Neutron, 1 synchrotron, 3 planned
- Muons: 4 instruments

The NIAC's Interest in Meeting with CIF

- NeXus has always tried to be inclusive: what use is a standard if everyone has her own?
- Endorsement by the IUCR would be helpful for the propagation of NeXus
- NeXus raison d'etre is interoperability: collaboration with CIF fits this bill
- In an ideal world CIF, NeXus, Hdf-5 would not matter anymore...

Issues and Differences

- NeXus addresses a far greater range of techniques then CIF
- NeXus is about hierarchies and arrays, CIF about tables
- Both CIF and NeXus have overlapping dictionaries
- NeXus used NXDL, CIF uses many versions of DDL
- Herbert Bernstein has demonstrated that it is possible to map CIF into NeXus. There are issues but no show stoppers
- CIF's ASCII file format hits a limit with current large data sets
- DECTRIS is pushing the PX community towards NeXus/HDF-5 with the upcoming EIGER detectors

Questions

- How are NeXus and CIF to be integrated?
- How far do we go with the integration?
- How are the NeXus and CIF dictionaries to be integrated? Are they to be integrated?
- Is CIF ready to expand towards a more general file format?
- What is the interest of the CIF community to collaborate with NeXus?
- How will NeXus have to change to accomodate CIF?
- How does CIF solve the FBD versus 10-20 required data items problem?
- How will the new integrated format be used?