# The NeXus Data Format

Mark Könnecke,[1] Frederick Akeroyd,[2] Herbert J. Bernstein,[3] Aaron S. Brewster,[4] Stuart Campbell,[5] Bjørn Clausen,[6] Stephen Cottrell,[2] Jens Uwe Hoffmann,[7] Pete R. Jemian,[8] David Männicke,[9] Raymond Osborn,[10] Peter F. Peterson,[5] Tobias Richter,[11] Jiro Suzuki,[12] Benjamin Watts,[13] Eugen Wintersberger,[14] and Joachim Wuttke[15]
(NeXus International Advisory Committee)

[1] *Laboratory for Development and Methods, Paul Scherrer Institute, 5232 Villigen-PSI, Switzerland*
[2] *ISIS, Rutherford Appleton Laboratory, UK*
[3] *imgCIF, Dowling College, USA*
[4] *Lawrence Berkeley National Laboratory, USA*
[5] *Spallation Neutron Source, USA*
[6] *Los Alamos National Laboratory, USA*
[7] *Helmholtz Zentrum Berlin, Germany*
[8] *Advanced Photon Source, Argonne National Laboratory, USA*
[9] *ANSTO, Australia*
[10] *Argonne National Laboratory, USA*
[11] *Diamond Light Source, UK*
[12] *KEK, Japan*
[13] *Swiss Light Source, Paul Scherrer Institute, 5232 Villigen-PSI, Switzerland*
[14] *Deutsches Elektronen-Synchrotron DESY, 22607 Hamburg, Germany*
[15] *Forschungszentrum Jülich, JCNS at MLZ, 85747 Garching, Germany*

(Dated: 9 September 2014)

NeXus is an effort by an international group of scientists to define a common data exchange format for neutron, X-ray, and muon experiments. NeXus is built on top of the scientific data format HDF5 and adds domain-specific rules for organizing data within HDF5 files in addition to a dictionary of well-defined domain-specific field names. The NeXus data format has two purposes. First, NeXus defines a format that can serve as a container for all relevant data associated with a beamline. This is a very important use case. Second, NeXus defines standards in the form of *application definitions* for the exchange of data between applications. NeXus provides structures for raw experimental data as well as for processed data.

Keywords: NeXus, data format, HDF5, X-ray, neutron, data analysis, data management

## I. INTRODUCTION

Increasingly, major neutron and X-ray facilities have chosen to store data using the NeXus data format. Since 2006, NeXus[1] has undergone substantial refocusing, refinement and enhancement as described in this paper.

Historically, neutron and X-ray facilities have chosen to store data in a plethora of home-grown data formats. This scheme has a number of drawbacks addressed by NeXus:

- It makes the life of traveling scientists unnecessarily difficult as they must deal with multiple files in different formats, file converters and such in order to extract scientific information from the data.
- An unnecessary burden is imposed on data analysis software producers to accommodate many different formats.
- The whole idea of open access to data is sabotaged if the data is in a format which cannot be easily understood.
- Scientific integrity is jeopardized if the data cannot be understood or important elements are missing.
- Modern high speed detectors produce data at such a high rate that many older single image storage schemes have become impractical and an efficient container format is a necessity.

The primary necessity for a data format is to define a physical file format: how is the data written to disk? Rather than invent yet another format, NeXus chose HDF5[2] as the physical file format. HDF5, a binary file format in the public domain that is well supported by both commercial and free software tools, is efficient, self-describing, and platform-independent.

NeXus adds to HDF5:

- Rules for organizing domain-specific data within a HDF5 file
- A link structure to enable quick default visualization
- A dictionary of documented domain-specific field names
- Definitions of standards that can be validated

The development of NeXus is overseen by a committee, the NeXus International Advisory Committee (NIAC)[3].

## II. DESIGN PRINCIPLES

The authors of data-acquisition and instrument-control software are encouraged to generate exactly *one* NeXus container file per measurement (a measurement is either a data accumulation under fixed conditions, or a scan). This file includes not only the detector and monitor data, but also metadata, information on the state of the beamline, parameter logs, and more. Authors of data-reduction and data-analysis software can use NeXus to store processed data along with metadata and a processing log.

NeXus data files are built using basic HDF5 storage elements: data groups (like file system folders), data fields (such as strings, floats, integers, and arrays), attributes (additional descriptors of groups and fields), and links (like file system links). These basic storage elements are used to build the *base classes*, *application definitions*, and *contributed definitions* that elaborate the NeXus standard. As a container format, NeXus allows files to be extended at any moment by additional content, including NeXus base classes, HDF5 groups, and HDF5 datasets.

NeXus can be used for many different experimental techniques, and at different levels of data processing. For each of these different applications, a specific subset of the standardized NeXus entities (data groups and fields) is needed. These subsets, and their hierarchical structure, are standardized in the NeXus application definitions (Sect. VI).

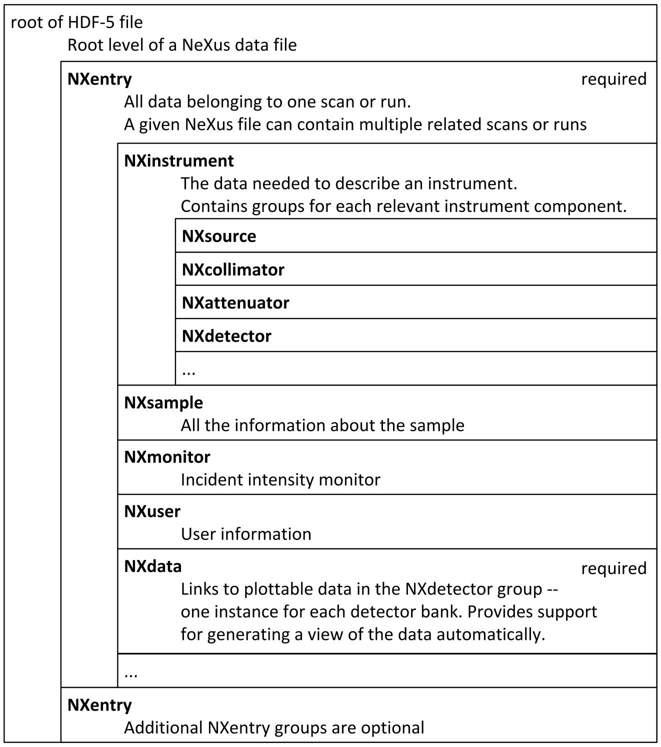
root of HDF-5 file
Root level of a NeXus data file
NXentry
required
All data belonging to one scan or run.
A given NeXus file can contain multiple related scans or runs
NXinstrument
The data needed to describe an instrument.
Contains groups for each relevant instrument component.
NXsource
NXcollimator
NXattenuator
NXdetector
...
NXsample
All the information about the sample
NXmonitor
Incident intensity monitor
NXuser
User information
NXdata
required
Links to plottable data in the NXdetector group --
one instance for each detector bank. Provides support
for generating a view of the data automatically.
...
NXentry
Additional NXentry groups are optional

FIG. 1. Overview of the structure of a NeXus raw data file. Note that only a small part of this structure (the first `NXentry` group and the first `NXdata` group) is actually required. The other content is optional.

## III. FILE HIERARCHIES

NeXus data files are organized into a hierarchy of groups which, in turn, can contain further groups or fields, very much like an internal file system. The content of each NeXus group is defined by a base class, or an application definition, or a contributed definition.

### A. Raw Data File Hierarchy

The major focus of NeXus has been the recording of *raw* experimental data, i.e. information taken directly from the experimental equipment or processed only as required to provide physically meaningful values. The NeXus raw data file hierarchy is the consequence of some practical considerations. An overview of the NeXus data file structure for raw experimental data is shown in FIG. 1.

When looking at a beamline it is easy to discern different components: beam optic components, sample position, detectors and such. It is quite natural to replicate this physical separation with a logical arrangement of storing the data from each component into a separate group. This approach explains the list of beamline components in the `NXinstrument` group presented in FIG. 1. As there can be multiple instances of the same kind of equipment, like slits or detectors, in a given beamline it becomes necessary to add type information to the group name. This type information, the NeXus class name, is provided by a HDF5 attribute. By convention NeXus class names start with the prefix `NX`. Each NeXus group describing a beamline component contains further groups and fields describing the component. A field can contain a single number, a text string or an array, as appropriate to the data to be described.

The requirement to store multiple related scans or runs in the same file or to capture a complete workflow in a file causes the beamline component hierarchy to be pushed one level deeper into an `NXentry` group in the hierarchy. The `NXentry` group thus represents one scan or run (or a processed data entry, as will be discussed later). The `NXentry` group also holds the experiment metadata, such as the date and time at which it was performed.

In the course of NeXus history, the decision was taken to move `NXmonitor` out of `NXinstrument` to the higher hierarchy level of `NXentry`, in order to facilitate quick inspection by humans.

To enable a simple default visualization, a `NXdata` group must be provided at `NXentry` level. It contains information about plot axes and links to the data (which typically reside in the `NXdetector` group). Links are supported by HDF5 and work like symbolic links in the Unix file system.

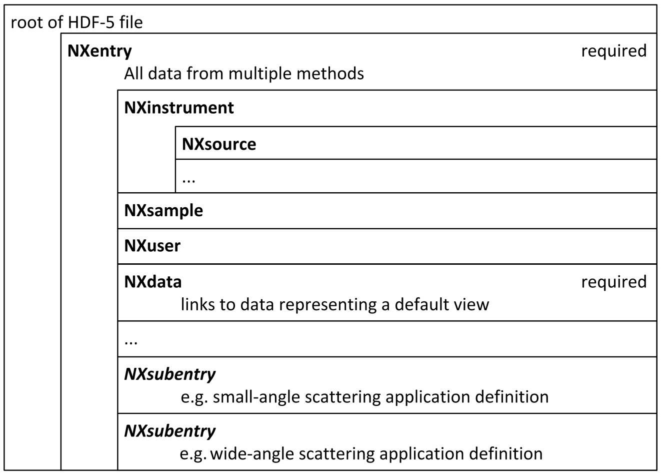
root of HDF-5 file
NXentry
required
All data from multiple methods
NXinstrument
NXsource
...
NXsample
NXuser
NXdata
required
links to data representing a default view
...
NXsubentry
e.g. small-angle scattering application definition
NXsubentry
e.g. wide-angle scattering application definition

FIG. 2. Overview of the structure of a NeXus raw data file for an instrument with multiple methods.

A special base class, `NXcollection`, exempts its contents from validation and thereby allows inclusion of whatever data in arbitrary non-NeXus formats.

### 1. Multiple Method Instruments

Particularly at X-ray sources, some instruments offer multiple techniques that can be used in parallel. For example small-angle scattering and powder diffraction can be measured simultaneously at a SAXS/WAXS beamline. We recommend storing the data from all methods in *one* file, in a *single* `NXentry` hierarchy (FIG. 2). All information from all detectors, logs and such are collected in this one `NXentry` group to keep the data together. Information that is particular for one experimental technique is linked into a `NXsubentry`. The `NXsubentry` follows the hierarchy of `NXentry`. But it will typically only link to the data required by the application definition for the specific experimental technique. The point of this scheme is that both humans and computerized users can easily locate method-specific data while maintaining the full view of the experiment.

### 2. Scans

Scans come in all shapes and sizes. Almost anything can be scanned against anything. An additional consideration is that, in practice, the final number of scan points in the scan cannot be known in advance since it is possible that a scan may be interrupted or terminated before its planned number of observations. Thus, it is a challenge to standardize a scan. NeXus addresses this challenge through use of HDF5 *unlimited dimensions* and additional conventions as described below. With the HDF5 unlimited dimensions feature, one axis of the data is allowed to expand without limit. Thus, the size of a data array does not need to be declared in advance. Data can be appended to an array along the unlimited dimension as required.

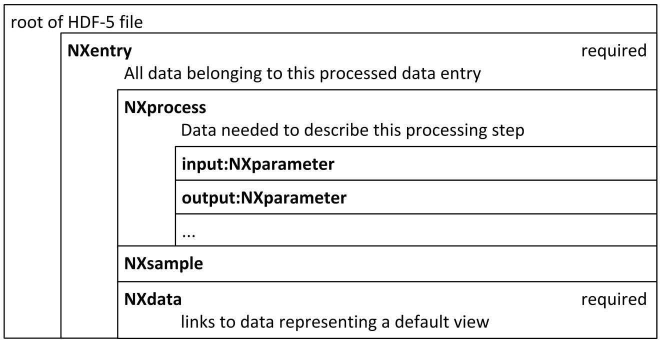
root of HDF-5 file
NXentry
required
All data belonging to this processed data entry
NXprocess
Data needed to describe this processing step
input:NXparameter
output:NXparameter
...
NXsample
NXdata
required
links to data representing a default view

FIG. 3. Overview of the structure of a NeXus processed data file.

Scans are stored in NeXus following these conventions:

- Each variable varied or collected in the scan is stored at its appropriate place in the NeXus beamline hierarchy as an array. The array's first dimension is the scan axis. This is the unlimited dimension in the implementation and data at each scan point is appended to the array.

- The `NXdata` group holds links to all the variables varied or collected during the scan. This creates something equivalent to or better than the tabular representation people are used to for scans. The main detector data can be plotted against any scanned parameter as well as against everything that was deemed worth recording in addition to that. The necessary data is all gathered together in the `NXdata` group either directly or via links, so that other groups do not normally have to be searched to do this plotting.

NeXus allows multi-dimensional scans too. This makes it very simple to produce meaningful slices through data volumes even with NeXus-agnostic software (*e.g.* HDFView[4]).

## B. Processed Data

At the request of the user community, NeXus created a simplified structure for storing the result of data processing: be it reduction or analysis. An overview of the NeXus structure for processed data is given in FIG. 3.

The hierarchy is much reduced as it is not important to carry all experimental information in the data reduction. In contrast to the raw data file structure, `NXdata` in the processed file structure is the place to store the results of the processing, together with its associated axes if the result is a multi-dimensional array.

In addition to the `NXdata` and `NXsample` groups, the `NXprocess` group provides structure to store details about the processing, such as the program used, its version, the date of processing, and other metadata. The NXprocess group can hold additional NXparameter groups which are containers for storing the input and output parameters of the program used to perform the processing.

## IV. COORDINATE SYSTEMS, POSITIONING OF COMPONENTS AND FURTHER RULES

For data reduction, it is often necessary to know the exact position and orientation of beamline components. The first thing needed is a reference coordinate system. NeXus chose to the use same coordinate system as the neutron beamline simulation software McStas[5].

For describing the placement and orientation of components, NeXus stores the same information as is used for the same purpose in the Crystallographic Interchange Format (CIF)[6]. CIF (and NeXus) stores the details of the translations and rotations necessary to move a given component from the zero point of the coordinate system to its actual position. As coordinate transformations are not commutative, the order of transformations must also be stored.

Detailed consideration has been given to the composition of the NeXus format, with a comprehensive set of rules for creating a NeXus file now being well defined. For the sake of brevity here, a complete description of the format is provided elsewhere in the NeXus manual[7] where a full listing of the rules is presented. However, to illustrate the attention to detail given in the definition of the format, the rules include these considerations:

- How axes are associated with data
- That units must be given with the data
- How data is to be stored in NeXus fields
- How to describe array data which is not in ANSI C storage order

## V. BASE CLASSES

As can be seen from the discussion of the NeXus file hierarchy, NeXus arranges data in groups which have a type descriptor and a NeXus base class name associated with them. Technically, the class name is the value of the HDF5 attribute NX_class. The term *base class* is not used in the same sense as in object-oriented programming languages; in particular, there is no inheritance. The NeXus base classes provide a comprehensive dictionary of terms that can be used for each class. The terms in the dictionary comprise concepts and names common to the topic of the base class. The expected spelling and definition of each term is specified in the base classes. It is neither expected nor required to provide all the terms specified in a base class. Terms with other names are permitted but might not be recognized by standard software. These terms designate data fields that can be stored within a group. A data field can have a simple type (like integer, float, date/time, binary), or it can be a NeXus subgroup. The base class definition also contains informal annotations about the semantics of each field.

At base class level, NeXus has no mechanism to mark some fields as obligatory. *All allowed fields are optional.* Which of them are written into data files must be decided according to application needs. These decisions can be standardized in the form of application definitions (see below, Sect. VI).

The NeXus base classes are encoded in NeXus Description Language (NXDL)[7]. NXDL is just another form of an XML file that specifies the content of a NeXus base class. NXDL files may be parsed either by humans or by software and may be validated for syntax and content. The NXDL files are used to validate the structure of NeXus data files. Java source code of a GUI tool has been prepared[8] to perform such validation.

## VI. APPLICATION DEFINITIONS

An application definition, expressed in NXDL, specifies a data structure for a given application domain such as a scientific technique or a specific type of instrument. The data structure consists of a hierarchy of NeXus groups. For each group, a *minimum* content is specified. Application definitions are therefore different than base class definitions, which specify a comprehensive dictionary of terms that can be used.

Historically, an application definition addressed one type of instrument, like X-ray reflectometer, or direct-geometry neutron time-of-flight spectrometer. Thus, application definitions were originally named *instrument definitions*. However, as NeXus can also be used for processed data like a tomography reconstruction or a dynamic scattering law $S(Q, \omega)$, the more generic term *application definition* has been adopted.

The process of drafting and ratifying application definitions is ongoing (see also below, Sect. VII). Currently, scientists representing both the NeXus International Advisory Committee and the IUCr Committee on the Maintenance of the CIF standard are nearly finished with a NeXus application definition for macromolecular crystallography. CBFlib[9] is being extended to work with NeXus-MX format. This work will be published in another paper. Work on another NeXus application definition for reduced small-angle scattering data is also in progress[10] by scientists representing canSAS, NeXus, and the IUCr Commission on Small-Angle Scattering.

## VII. GOVERNANCE

The development of NeXus is overseen by the NeXus International Advisory Committee (NIAC)[3]. The NIAC seeks a balanced representation of the international community. Most major neutron, X-ray, and muon facilities have appointed delegates. Other facilities are invited to join.

The NIAC reviews proposed amendments to the NeXus base classes and application definitions, and holds online votes to ratify changes. A great number of candidate NeXus application definitions exist which were derived from our understanding of the technique described. For each of these, the NeXus team seeks community approval.

## VIII. UPTAKE OF NEXUS

NeXus is already in use as the main data format at many facilities including Soleil, Diamond, SINQ, SNS, Lujan/LANL and KEK. Other facilities including ISIS, DESY and the $\mu$SR community are in the process of moving towards NeXus as their data format. At LBNL, NeXus is currently being adapted for XFEL serial crystallographic data. APS is storing some of its data collection using NeXus. The EPICS[11] area detector software has a plugin to write acquired images into NeXus data files. Also, some commerical manufacturers of area detectors now write acquired images into NeXus data files.

The adoption of NeXus has taken some time. The reason is that NeXus is often chosen whenever a facility starts operation or undergoes major refurbishments. For those facilities where there is an existing and working pipeline from data acquisition to data analysis, the resources are usually lacking to move towards NeXus as the only data file format.

This is reflected in the experience of the muon community. For the ISIS source, the move to a Windows PC-based data acquisition system in 2002 required a new data format, providing an ideal opportunity to exploit the emerging NeXus standard[12]. In contrast, sources at PSI, TRIUMF and KEK continued to make good use of existing formats and software. More recently, funding from the EU has enabled the community to develop the Application Definition as a common exchange format for muon data[13].

Whether used as the main or an intermediate format, users are able to produce compatible NeXus files for data written across all facilities, enhancing the uptake of NeXus within the community.

## IX. BACKWARDS COMPATIBILITY

Historically, NeXus supported reading and writing data files in HDF4, HDF5 and XML formats by use of the NeXus Application Programming Interface (NeXus-API or NAPI). The NAPI is still available, but frozen except for bug fixes. After consultation with the community, the currently recommended use of NeXus is solely in terms of the HDF5 file format, using standard HDF5 tools. That is expected to remain the basis for NeXus software development and file creation in the future.

## X. SUMMARY

NeXus has matured considerably over the last ten years and is now in use in many facilities. NeXus is flexible enough to accommodate a wide variety of instruments and scientific applications, yet efficient enough to handle the data coming from modern high speed detectors. More information, including a full PDF manual, can be found on the project website[14]. Members of the NIAC[3] always welcome correspondence concerning the development of the NeXus data format.

## ACKNOWLEDGMENTS

The NIAC acknowledges the support of all the institutions contributing to NeXus and their respective funding agencies, most notably DOE, NIH and NSF in the US. The development of the muSR NeXus data format was partly funded by the European Commission within the sixth Framework Programme.

[1] M. Könnecke, Physica B **385-386**, 1343 (2006).

[2] Hdfgroup, *HDF-5* (2014 (accessed July 2014)), `http://www.hdfgroup.org/HDF5/`.

[3] NIAC, *NeXus International Advisory Committee (NIAC)* (2014 (accessed July 2014)), `http://wiki.nexusformat.org/NIAC`.

[4] T. H. Group, *HDFView* (2014 (accessed September 2014)), `http://www.hdfgroup.org/products/java/hdfview/index.html`.

[5] P. Willendrup, E. Farhi, and K. Lefmann, Physica B **350**, 735 (2004).

[6] S. Hall and B. McMahon, *International Tables for Cystallography Volume G: Definition and exchange of crystallographic data* (Wiley, 2006).

[7] NIAC, *NeXus Manual* (2014 (accessed July 2014)), `http://download.nexusformat.org/kits/definitions/nexus-manual-3.1.0.tar.gz`.

[8] NIAC, *NXvalidate: a java GUI tool to validate NeXus data files* (2014 (accessed July 2014)), `https://github.com/nexusformat/code/tree/master/applications/NXvalidate`.

[9] H. J. Bernstein and P. J. Ellis, in *International Tables For Crystallography*, Vol. G: Definition and Exchange of Crystallographic Data, edited by S. R. Hall and B. McMahon (Springer, Dordrecht, NL, 2005) Chap. 5.6, pp. 544 – 556, see `http://sf.net.projects/cbflib` and `http://www.bernstein-plus-sons.com/software/CBF/`.

[10] canSAS, *Description of the canSAS2012 data format* (2014 (accessed July 2014)), `http://www.cansas.org/formats/canSAS2012/1.0/doc/`.

[11] M. Rivers, *areaDetector: EPICS software for area detectors* (2014 (accessed September 2014)), `http://cars9.uchicago.edu/software/epics/areaDetector.html`.

[12] P. K. D. Flannery, S.P. Cottrell, Physica B **326**, 238 (2003).

[13] S. P. Cottrell, F. Pratt, A. Hillier, P. King, F. Akeroyd, A. Markvardsen, N. Draper, Y. Yao, and S. Blundell, Physics Procedia **30**, 20 (2012).

[14] NIAC, *NeXus A common data format for neutron, x-ray and muon science* (2014 (accessed July 2014)), `http://www.nexusformat.org/`.