The NeXus Data Format

Mark Könnecke,[1] Frederick Akeroyd,[2] Herbert J Bernstein,[3] Aaron S. Brewster,[4] Bjoern Clausen,[5] Stephen Cottrell,[6] Jens Uwe Hoffmann,[7] Pete Jemian,[8] David Männicke,[9] Raymond Osborn,[10] Peter F. Peterson,[11] Tobias Richter,[12] Jiro Suzuki,[13] Bejmanin Watts,[14] Eugen Wintersberger,[15] and Joachim Wuttke[16]
(NeXus International Advisory Committee)
[1)] *Laboratory for Development and Methods, Paul Scherrer Institute, 5232 Villigen-PSI, Switzerland*
[2)] *ISIS, Rutherford Appleton Labortaory, UK*
[3)] *imgCIF, Dowling College USA*
[4)] *Lawrence Berkeley National Laboratory, USA*
[5)] *Los Alamos National Laboratory, USA*
[6)] *ISIS, Rutherford Appleton Laboratory, UK*
[7)] *Helmholtz Zentrum Berlin, Germany*
[8)] *Advanced Photon Source, USA*
[9)] *ANSTO Australia*
[10)] *Argonne National Laboratory, USA*
[11)] *Spallation Neutron Source, USA*
[12)] *Diamond Light Source, UK*
[13)] *KEK, Japan*
[14)] *Swiss Light Source, Paul Scherrer Institute, 5232 Villigen-PSI, Switzerland*
[15)] *DESY, Germany*
[16)] *Forschungszentrum Jülich, JCNS at MLZ, Garching, Germany*

(Dated: 14 August 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 as well as 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, an increasingly important function. 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 choose to store their data in 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 choose to store their 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 have to 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 as they have to accommodate so 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.
- Modern high speed detectors produce data at such a rate that many older single image storage schemes become impractical and an efficient container format is a necessity.

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

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).

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

NeXus can be used for many different experimental techniques, and at different levels of data processing. For each of these different application, 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 V).

III. NEXUS 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 either a base class, application definition, or contributed definition. For an overview of the NeXus data file structure for raw experimental data see FIG. 1.

In the following sections, we will describe some of the rules that define the overall structure of NeXus files. Many fine details of the NeXus format have been thoroughly discussed and are now well defined, but for the sake of brevity, we will not present an exhaustive view of NeXus here. A full listing of NeXus rules are given in the NeXus manual[3]. Some examples of these additional rules include those governing:

- 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

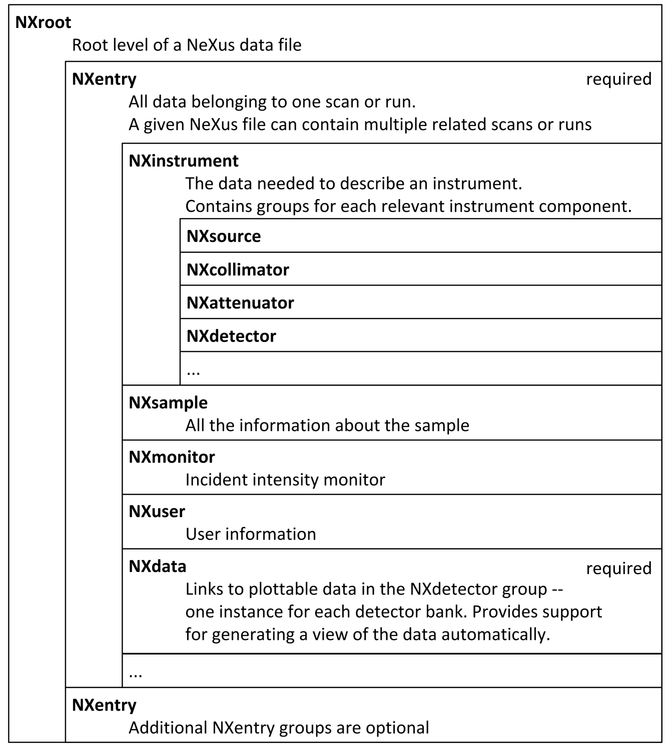

FIG. 1. Overview of the structure of a NeXus raw data file.

A. NeXus 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. 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 mirror this physical separation with a logical arrangement of storing the data from each component in 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, which is provided by a HDF5 attribute, the NeXus class name. 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

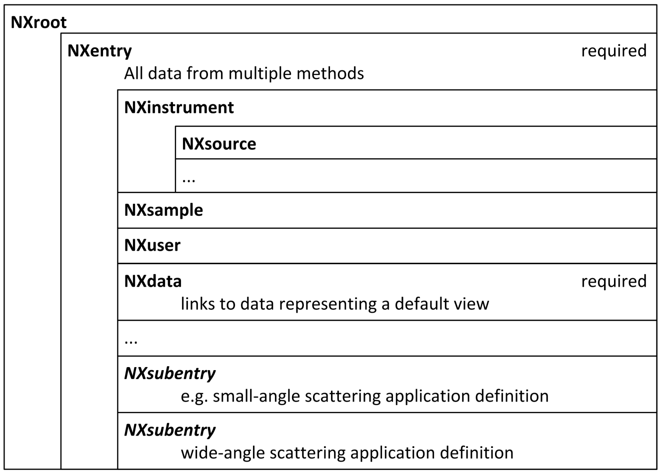

FIG. 2. Overview of the structure of a NeXus raw data file for an instrument with multiple methods.

`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 debatable 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.

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 to store the results from all subinstruments in *one* file. All data are stored in one and the same `NXentry` hierarchy (FIG. 2). Information that applies to all subinstruments is stored at `NXentry` level as before. Information that is peculiar for one experimental technique is moved down into a `NXsubentry`. Aside from this rearrangement, each subentry follows the application definition that is pertinent for its experimental technique.

2. Scans

Scans come in all shapes and sizes. Almost anything can be scanned against anything. An additional difficulty is that in practice, the 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 solves these difficulties through a couple of conventions and the use of a HDF5 feature called unlimited dimensions. With the HDF5 unlimited dimensions feature, one axis of the data is allowed to expand without limit and 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.

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 is appended at each scan point to the array.

- The `NXdata` group holds links to all the variables varied or collected during the scan. This creates something equivalent or better than the tabular representation people are used to for scans. The main detector data scan be plotted against any scanned parameter as well as against everything that was deemed worth recording in addition to that, reading the `NXdata` group alone.

NeXus allows multi-dimensional scans too. This makes it very simple to produce meaningful slices through data volmes even with NeXus-agnostic software (like hdfview). Interrupting a multi-dimensional scan may, depending on the software used, leave some of the data in an uninitialised state (usually the HDF5 fill value).

3. Coordinate System and Positioning of Components

For analysing data 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 use same coordinate system as the neutron beamline simulation software McStas[4].

For describing the placement and orientation of components, NeXus stores the same information as used for the same purpose in the Crystallographic Interchange Format (CIF)[5]. 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.

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. In many cases even the reduced data is big enough to need an efficient binary representation. A good example is a tomography

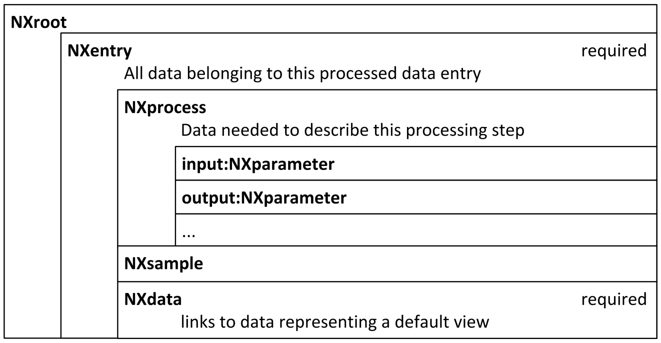

FIG. 3. Overview of the structure of a NeXus processed data file.

reconstruction. A tabular representation of the NeXus processed data file structure 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.

Information about the sample and instrument can be stored in `NXsample` and `NXinstrument` groups as required.

In addition, there is a new structure to store details about the processing such as the program used, its version, the date of processing, and other metadata in the `NXprocess` group. 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. NEXUS BASE CLASSES

As can be seen from the discussion of the NeXus file hierarchy, NeXus arranges data into groups which have a type descriptor, a NeXus base class name, associated with them (technically, this name is the value of the HDF5 attribute `NXclass`). The term *base class* is not meant in the sense of object-oriented programming languages; in particular, there is no inheritance. A NeXus base class is rather a dictionary of allowed keywords. These keywords 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 form of application definitions (see below, Sect V).

NeXus does not specify the *rank* of a data field. For many, but not all fields it is possible to infer from the field name or from an explicit annotation whether the field is a scalar or what its array dimensionality is. The character data type `NXchar` is almost always to be understood as string type.

The NeXus base classes are encoded in NeXus Description Language (NXDL)[3]. NXDL is just another application of XML. Thus a NXDL file is an XML file that specifies the content of the NeXus base classes.

V. NEXUS APPLICATION DEFINITIONS

An application definition specifies a data structure for a given application domain. The data structure consists of a hierarchy of NeXus groups. For each group, a *minimum* content is specified. Application definitions are therefore complementary to base class definitions, which specify the *maximum* content of groups.

Typically, an application definition addresses one type of instrument, like x-ray reflectometer, or direct-geometry neutron time-of-flight spectrometer. Therefore 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* is preferred.

NeXus application definitions are expressed in NXDL. They may be parsed either by humans or by software and they may be validated for syntax and content. The NXDL files are used to validate the structure of NeXus data files. A tool exists to perform such validation.

The process of drafting and ratifying application definitions is ongoing (see also below, Sect VII). Currently scientists from NeXus and the IUCr are nearly finished with a NeXus application definition for macromolecular crystallography. CBFlib[6] 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[7] by members of canSAS, NeXus, and the IUCr Commission on Small-Angle Scattering.

VI. UPTAKE OF NEXUS

NeXus is already in use as the main data format at facilities like Soleil, Diamond, SINQ, SNS, Lujan/LANL and KEK. Other facilities like ISIS, DESY and the μSR community are in the process of moving towards NeXus as data format. At LBNL, NeXus is currently being adapted for XFEL serial crystallographic data.

The adoption of NeXus took 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.

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[8]. 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[9]. Whether used as the main or an intermediate format, users will soon be able to produce compatible NeXus files for data written across all facilities, enhancing the uptake of NeXus within the community.

VII. NEXUS GOVERNANCE

The development of NeXus is overseen by the NeXus International Advisory Committee (NIAC). The NIAC seek balanced representation of the international community. Most major neutron, x-ray, and muon facilities have already 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. BACKWARDS COMPATIBILITY

In the past, NeXus supported data files in HDF-4, HDF5, and XML file formats. To support writing software to write and read these file formats, the NeXus application/programmer interface (NeXus-API) was provided. This API is maintained at a bug fix level. On request of the community, we now concentrate our efforts only on the HDF5 files and tools. **What API do we recommend now? The frozen NeXus API, or a vanilla HDF5 emitter/parser?**

IX. SUMMARY

NeXus has matured considerably over the last 10 years and is now in use in many facilities. NeXus is flexible enough to accommodate a wide variety of instruments and scientific applications. Yet it is 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 web site.[10] Also, do not hesitate to contact members of the NIAC.

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

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

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

[6] 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/`.

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

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

[9] 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).

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