# The Scientific Data Exchange Introductory Guide

http://www.aps.anl.gov/DataExchange

Version 0.9.5

May 4, 2014

**Table 1:** Version history

| Version | Date | Notes |
|---|---|---|
| 0.8.9 | Feb. 25, 2012 | de carlo: first document draft. |
| 0.9.0 | May 25, 2012 | saunders: Extensive reworking of document structure and some definitions. |
| 0.9.1 | April 20, 2013 | de carlo: added Python examples and corrected minor tomography definitions. Changed provenance definition. |
| 0.9.2 | June 27, 2013 | de carlo: added sample_x and sample_y in the raw data exchange definition to store sample projection position at each point during data collection. Modified Python example 5. |
| 0.9.3 | Nov 18, 2013 | de carlo: replaced python examples with one using data_exchange python class wrote by David Vine. |
| 0.9.4 | Dec 7, 2013 | de carlo: clean transfer of data_exchange latex documents from svn (now obsolete) to https://github.com/data-exchange/data-exchange |
| 0.9.5 | May 3, 2014 | de carlo: added SLS definitions for DPC tomography |

# Contents

# 1 Introduction

This document is a guide to the basic design principles and core guidelines for the Data Exchange file format. Briefly, Data Exchange is a set of guidelines for storing scientific data and metadata in a Hierarchical Data Format 5 (HDF5) file (`http://www.hdfgroup.org/HDF5`). The design guidelines for Data Exchange provide a starting point to which one may add custom data and metadata to solve particular problems.

HDF5 has many important characteristics for scientific data storage. It offers platform-independent binary data storage with optional compression, hierarchical data ordering, and support for MPI-based parallel computing. Data are stored with alphanumeric tags, so that one can examine a HDF5 file's contents with no knowledge of how the file writing program was coded. Tools for this examination include the HDF5-supplied command-line utility `h5dump` to examine the contents of any HDF5 file, or the freely-available Java program `HDFView` to interactively examine the file.

Why not just use the NeXus format? While we employ some of the naming conventions of NeXus, the format has not been as widely adopted in the synchrotron radiation community as it has in the neutron beam research community. NeXus is normally based on use of an Application Programming Interface (API) for all file access; this API has not in the past had well-debugged versions available for all platforms and languages. NeXus has not supported recent HDF5 extensions such as dimension scales, and it has not had definitions for tracking the provenance of subsequent analysis steps. "Straight" HDF5 calls can make use of the HDF5 group's support of a wide variety of computer platforms, and built-in support in MATLAB, IDL, and Python; it also has support for MPI-based parallel access to files. As a result, we have sought to follow many naming conventions from NeXus, but to aim for a more streamlined file structure with necessary definitions added.

This reference guide describes the basic design principles of Dat Exchange, examples of their application, a core reference for guidelines common to most uses, and coding examples.

# 2 The Design of Data Exchange

For various reasons, many x-ray techniques developed at synchrotron facilities around the world are unable to directly compare results due to their inability to exchange data and software tools. The aim of Data Exchange is to define a simple file format offering few basic rules and allowing each community to extend and add technique specific components. The goal is to provide extensibility in defining data, meta data and provenance information in a simple way that can be easily adopted by various x-ray techniques.

The Data Exchange format is implemented using Hierarchical Data Format 5 (HDF5), which offers platform-independent binary data storage with optional compression, hierarchical data ordering, and self-describing tags so that one can examine a HDF5 file's contents with no knowledge of how the file writing program was coded.

The aim and the scope of Data Exchange is very similar to the Coherent X-ray Imaging Data Bank file format (CXI), so whenever possible we will use the same conventions, name tags, and reference system. This document is using a similar diagram definition set by Filipe R. N. C. Maia in "CXI file format" (http://cxidb.org/cxi.html), with a few minor modifications for Data Exchange definitions.

The core principle of Data Exchange is that it must be simple enough that it is not necessary to use a support library beyond core HDF5. The simplicity of Data Exchange makes it easy for anyone to either look at an example file using h5dump or HDFView, or to look at example code in language X, and then create their own read and write routines in language Y.

The simplest Data Exchange file provides information sufficient to share a multidimensional data array. In this simplest form, Data Exchange implements only one "exchange" group. The "exchange" definition is designed to allow for simple exchange of images, spectra, and other forms of beamline detector data with a minimum of fields. This definition is essentially a technique-agnostic format for exchanging data with others. Data Exchange is also designed to be extended to include technique-specific data and metadata. This is achieved by providing optional, but clearly defined, metadata components to the base definition.

## 2.1 HDF5

The HDF5 format is the basis of the Data Exchange format. Data Exchange, like CXI, is not a completely new file format, but simply a set of rules designed to create HDF5 files with a common structure to allow a uniform and consistent interpretation of such files.

HDF5 was chosen as the basis because it is a widely used high performance scientific data format which many programs can already, at least partially, read and write. It also brings with it the almost automatic fulfillment of the Data Exchange requirements, i.e. simplicity, flexibility and extensibility. HDF5 version

1.8 or higher is required as previous versions don't support all features required by Data Exchange.

## 2.2 Data types

Data Exchange uses the same CXI convention for data types as defined at `http://cxidb.org/cxi.html` using HDF5 native datatypes. The data should be saved in the same format as it was created/acquired. For example CCD images acquired as 16 bit integers should be saved using the `H5T_NATIVE_SHORT` HDF5 type. In this way all cross platform big-little endian issues reading and writing files are eliminated.

## 2.3 Root Level Structure

While HDF5 gives great flexibility in data storage, straightforward file readability and exchange requires adhering to an agreed-upon naming and organizational convention. To achieve this goal, Data Exchange adopts a layered approach by defining a set of *mandatory* and `optional` fields.

The general structure of a Data Exchange file is shown in table **??**. The most basic file must have an "implements" string, and an "exchange" group at the root level/group of the HDF5 file. Optional "measurement" and "provenance" groups are also defined. Beyond this, additional groups may be added to meet individual needs, with guidelines suggesting the best structure.

**Table 2:** Data Exchange Top Level Members

| Member | Type | Example |
|---|---|---|
| *implements* | string dataset | "exchange:measurement:provenance" |
| *exchange* | group | |
| `measurement` | group | |
| `provenance` | group | |

***implements***

Mandatory scalar string dataset in the root of the HDF5 file whose value is a colon separated list that shows which components are present in the file. All components listed in the *implements* string are to be groups placed in the HDF5 file at the root level/group. In a minimal Data Exchange file, the only mandatory item in this list is *exchange*. A more general Data Exchange file also contain `measurement` and possibly `provenance`, in which case the implements string would be: "*exchange*: `measurement`: `provenance`"

***exchange***

Mandatory group containing one or more arrays that represent the most basic version of the data, such as raw or normalized optical density maps or a

elemental signal map. *Exchange_N* is used when more than one core dataset or derived datasets are saved in the file. The *exchange* implementation for specific techniques are defined in separate sections in the Reference Guide.

`measurement`
: Optional group containing the measurement made on the sample. `Measurement` contains information about the `sample` and the `instrument`. `Measurement_`*N* is used when more than one measurement is stored in the same file.

`provenance`
: Optional group containing information about the status of each processing step.

In a Data Exchange file, each dataset has a unit defined using the `units` attribute. `units` is not mandatory - if omitted, the default unit as defined in Appendix **??** is used.

The detailed rules about how to store datasets within the exchange group are best shown through examples in the next section. Detailed reference information can be found in the Data Exchange Core Reference section.

# 3 Data Exchange by Example

The examples in this section show how one can store data for imaging experiments using the Data Exchange format. It is general enough, however, to show how Data Exchange can be extended or adapted to other techniques. These examples are meant to give a flavor for our approach. A complete reference to the core structure can be found in Section **??**. Technique specific extensions to the core structure can be found at the end of the Reference Guide.

## 3.1 Diagram color code

All the diagrams in this section follow the color conventions shown in Figure **??**. The basic elements are HDF5 datasets, attributes, and groups. We also support internal references to elements in the file by a simple scalar string that holds the path of the dataset within the file. On the diagram, this is shown as a reference dataset that points to the referred-to dataset. Note that we use this mechanism rather than HDF5 hard or soft links

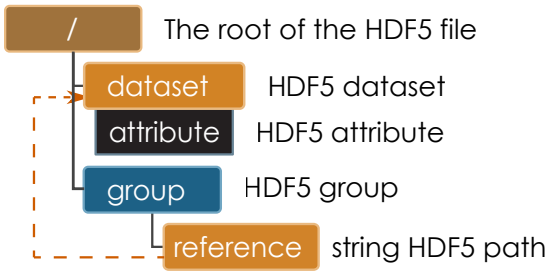


**Figure 1:** Explanation of the color code used in the diagrams

## 3.2 A minimal Data Exchange file for imaging

Figures **??** shows a diagram of a minimal Data Exchange file to store a single projection image. It is strongly encouraged that all datasets shall have a units attribute. The axes of the dataset are not specified in this minimal case, and can be assumed to be x and y with a zero-based integer sequence, or more simply, pixels.

## 3.3 Storing and describing a multidimensional dataset

A multidimensional dataset should be described as fully as possible, with units for the dataset as well as dimension descriptors (that also have units defined). There are also additional descriptive fields available such as title and description. The order of dimensions in the dataset should put the slowest changing dimension first, and the fastest changing dimension last.

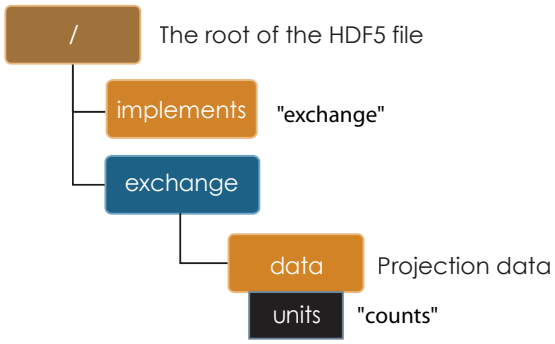


**Figure 2:** Diagram of a minimal Data Exchange file for a single image.

It is strongly encouraged that all datasets have a units attribute. The string value for units should preferably be an SI unit, however well understood non-SI units are acceptable, in particular ”degrees”. The units strings should conform to those defined by UDUNITS at http://www.unidata.ucar.edu/software/udunits. While UDUNITS is a software package, it contains simple XML files that describe units strings and acceptable aliases.

The axes of a multidimensional dataset are described through the use of additional one-dimensional datasets (dimension descriptors), one for each axis in the main dataset. Take for example a 3-dimensional cube of images, with axes of x, y, and z where z represents the angle of the sample when each image was taken. There should be 3 additional one-dimensional datasets called x, y, and z where x and y contain an integer sequence, and z contains a list of angles. X and y have units of ”counts” and z has units of ”degrees”. To simplify, it is acceptable to omit x and y, since the default interpretation will always be an integer sequence.

The dimension descriptors (x, y, and z) can be associated with the main dataset through two mechanisms. The HDF5 libraries contain a function call H5DSattach_scale to ”attach” a dimension descriptor dataset to a given dimension of the main dataset. HDF5 takes care of entering several attributes in the file that serve to keep track of this association. If the particular programming language you work in does not support this HDF5 function, then you can instead add a string attribute to your main dataset called axes. The axes attribute is simply a colon separated string naming the dimension descriptor datasets in order, so ”z:y:x” in this case. Additional examples below show this in action.

## 3.4 Storing projections, dark fields, and white fields

A tomographic data set consists of a series of projections, dark and white field images. The dark and white fields must have the same projection image dimensions and can be collected at any time before, after or during the projection data collection. The angular position of the tomographic rotation axis, theta, can be used to keep track of when the dark and white images are collected. These

examples show projection, dark, and white images saved in three 3D arrays as shown in Figure **??** and **??** using, by default, the natural HDF5 order of the a multidimensional array (rotation axis, ccd y, ccd x), i.e. with the fastest changing dimension being the last dimension, and the slowest changing dimension being the first dimension. If using the default dimension order, the axes attribute ”theta:y:x” can be omitted. The `axes` attribute is mandatory if the 3D arrays use a different axes order. This could be the case when, for example, the arrays are optimized for sinogram read (`axes` = ”y:theta:x”). As no units are specified the data is assumed to be in “counts” with the axes (x, y) in pixels.

If the positions of the rotation axis for each projection, dark, and white images are not specified via theta dimension scale datasets, it is assumed that the raw projections are taken at equally spaced angular intervals between 0 and 180 degree, with white and dark field collected at the same time before or after the projection data collection.

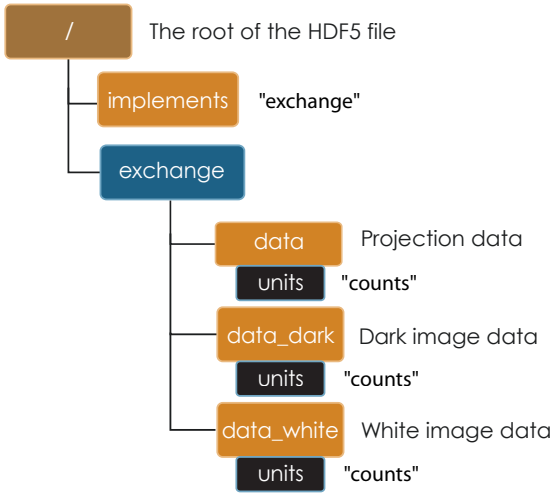


**Figure 3:** Diagram of a minimal Data Exchange file for a single tomographic data set including raw projections, dark, and white fields.

## 3.5 A typical Data Exchange file for tomography

A series of tomographic data sets are typically collected changing the instrument status (energy, detector or optics position) or changing the sample status (position, environment etc.). Figure **??**, **??** and **??** show the content of files changing the sample temperature, the x-ray source energy and detector-sample distance.

### 3.5.1 Sample Temperature Scan

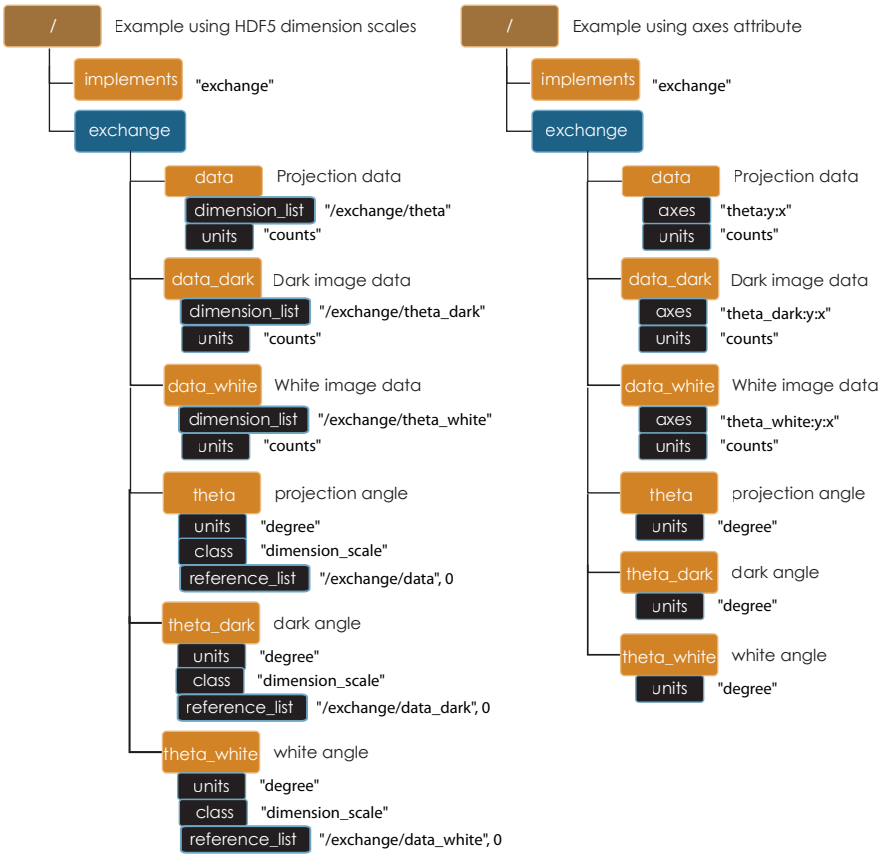


**Figure 4:** Diagram of a single tomographic data set including raw projections, dark and white fields. In this case, there are additional dimension descriptor datasets theta, theta_dark, and theta_white that contain the positions of the rotation axis for each projection, dark, and white image. The lefthand example shows this as it would appear using the HDF5 H5DSattach_scale function. The righthand example shows this as it would appear by manually adding an axes attribute (for cases where H5DSattach_scale is unavailable).

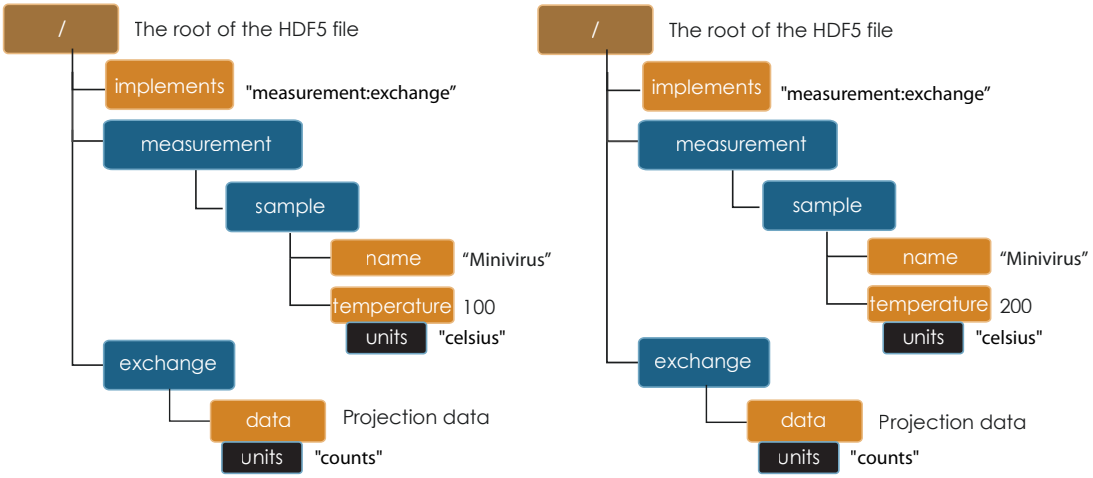


**Figure 5:** Diagram of two tomographic data sets taken at two different sample temperatures (100 and 200 celsius).

### 3.5.2 X-ray Energy Scan

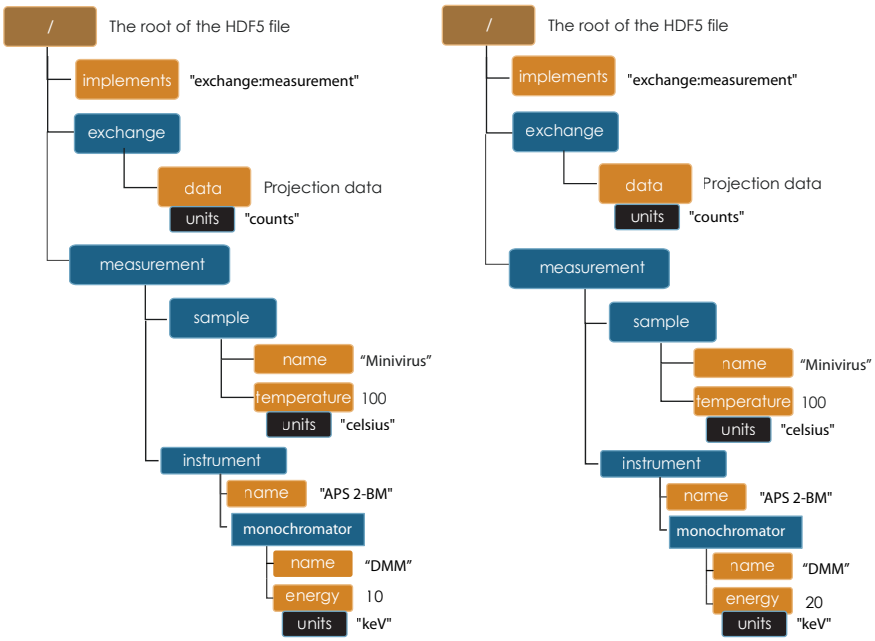


**Figure 6:** Diagram of two tomographic data sets taken at two different energy (10 and 20 keV).

### 3.5.3 Detector-sample Distance Scan

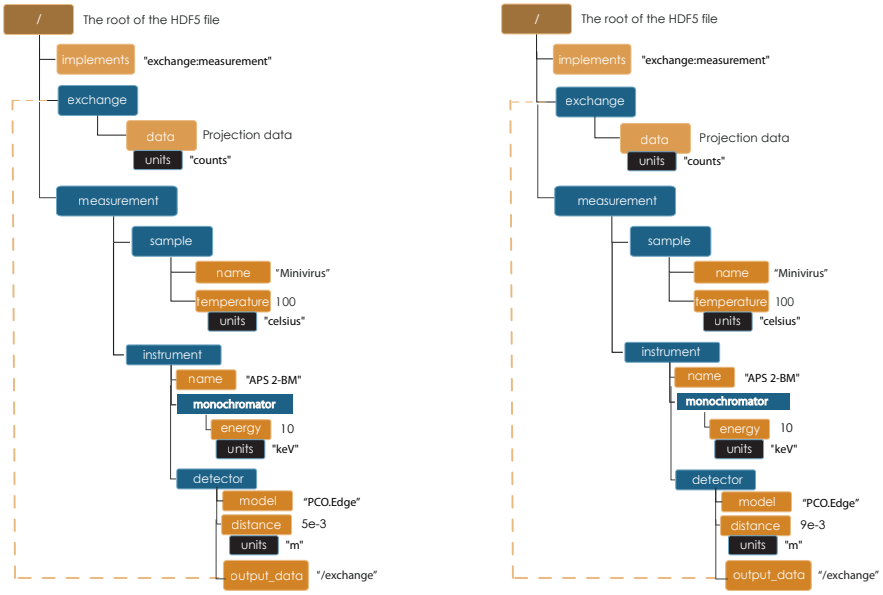


**Figure 7:** Diagram of two tomographic data sets collected with two different detector-sample distances (5 and 9 mm). Note the use of `output_data` dataset to associate the detector with the exchange group generated from the acquisition.

### 3.5.4 Series of Tomographic Measurements

A series of tomographic measurements, when relevant, can be stored in the same file appending $\_N$ to the measurement tag. In nano tomography experiments, for example, the detector field of view is often smaller than the sample. To collect a complete tomographic data set, it is necessary to raster the sample across the field of view moving its x and y location. Figure **??** shows a file from a nano tomography experiment when the sample rasters through the field of view.

There are limits to this approach, as one clearly does not want to have hundreds of measurement groups in a file (or multiple files) where most of the metadata is the same. For measurements where there are many ”positioner” values (aka a ”scan”), it is more sensible to add dimension(s) to the exchange dataset, and describe the ”positioner” values as dimension scales. This is a judgement left to the user.

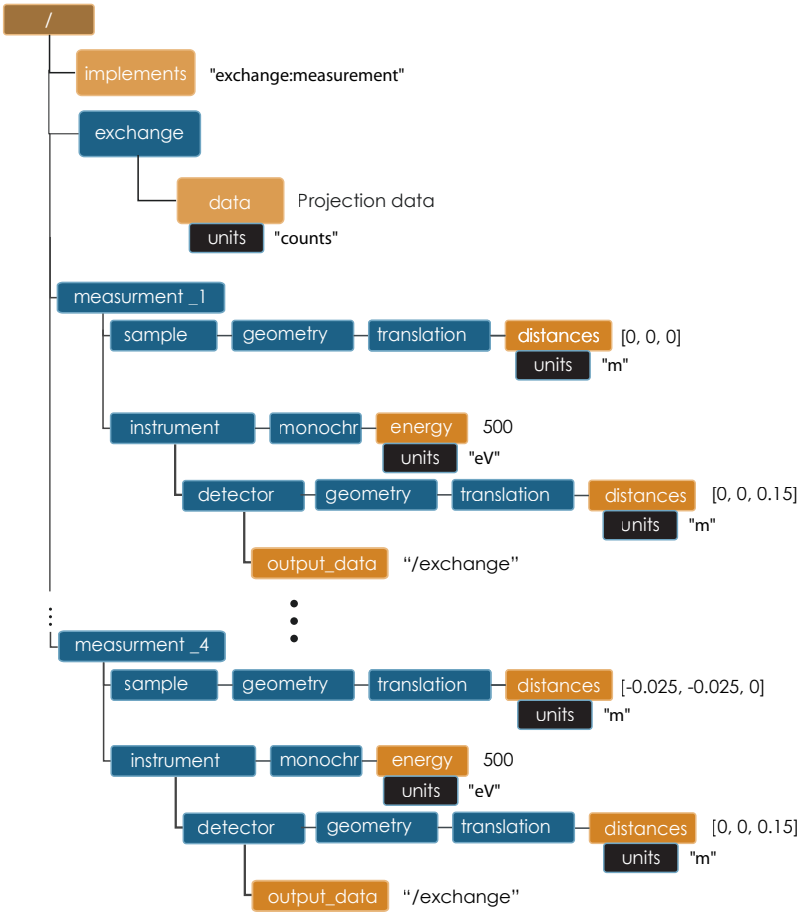


**Figure 8:** Diagram of a file with 4 tomographic data sets from a nano tomography experiment.

# 4 Data Exchange Core Reference

## 4.1 Top level (root)

This node represents the top level of the HDF5 file and holds some general information about the file.

**Table 3:** Top Level Members

| Member | Type | Example |
|---|---|---|
| *implements* | string dataset | "exchange:measurement:provenance" |
| *exchange_N* | group | |
| measurement_*N* | group | |
| provenance | group | |

***implements***
: A colon separated list that shows which components are present in the file. The only *mandatory* component is *exchange*. A more general Data Exchange file also contains `measurement` and `provenance` information, if so these will be declared in implements as ”exchange:measurement:provenance”

***exchange*_*N***
: The data taken from measurements or processing. Dimension descriptors within the group may also serve to describe ”positioner” values involved in a scan.

`measurement`_*N*
: Description of the sample and instrument as configured for the measurement. This group is appropriate for relatively static metadata. For measurements where there are many ”positioner” values (aka a ”scan”), it is more sensible to add dimension(s) to the exchange dataset, and describe the ”positioner” values as dimension scales rather than record the data via multiple matching measurement_N and exchange_N groups. This is a judgement left to the user.

`provenance`
: The Provenance group describes all process steps that have been applied to the data.

## 4.2 Exchange

The exchange group is where scientific datasets reside. This group contains one or more array datasets containing n-dimensional data and optional descriptions of the axes (dimension scale datasets). Exactly how this group is used is dependent on the application, however the general idea is that one exchange group contains one cohesive dataset. If, for example, the dataset is processed into some other form, then another exchange group is used to store the derived data.

Multiple exchange groups are numbered consecutively as exchange_$N$. At a minimum, each exchange group should have a primary dataset named data. The title is optional.

**Table 4:** Exchange Group Members

| Member | Type | Example |
|---|---|---|
| `title` | string dataset | "absorption_tomography" |
| *data* | array dataset | n-dimensional dataset |

`title`
: Descriptive title for data dataset. Current types include: absorption_tomography, phase_tomography, dpc_tomography

***data***
: The primary scientific dataset. Additional related datasets may have any arbitrary name. Each dataset should have a units and description attribute. Discussion of dimension descriptors and optional axes attribute is covered in Section **??**.

**Table 5:** data attributes

| Attribute | Type | Example |
|---|---|---|
| `description` | string attribute | "transmission" |
| `units` | string attribute | "counts" |

### 4.2.1 Storing and describing a multidimensional dataset

A multidimensional dataset should be described as fully as possible, with units for the dataset as well as dimension descriptors (that also have units defined). There are also additional descriptive fields available such as title and description. The order of dimensions in the dataset should put the slowest changing dimension first, and the fastest changing dimension last.

It is strongly encouraged that all datasets have a units attribute. The string value for units should preferably be an SI unit, however well understood non-SI

units are acceptable, in particular ”degrees”. The units strings should conform to those defined by UDUNITS at http://www.unidata.ucar.edu/software/udunits. While UDUNITS is a software package, it contains simple XML files that describe units strings and acceptable aliases.

The axes of a multidimensional dataset are described through the use of additional one-dimensional datasets (dimension descriptors), one for each axis in the main dataset. Take for example a 3-dimensional cube of images, with axes of x, y, and z where z represents the angle of the sample when each image was taken. There should be 3 additional one-dimensional datasets called x, y, and z where x and y contain an integer sequence, and z contains a list of angles. X and y have units of ”counts” and z has units of ”degrees”. To simplify, it is acceptable to omit x and y, since the default interpretation will always be an integer sequence.

The dimension descriptors (x, y, and z) can be associated with the main dataset through two mechanisms. The HDF5 libraries contain a function call to ”attach” a dimension descriptor dataset to a given dimension of the main dataset. HDF5 takes care of entering several attributes in the file that serve to keep track of this association. If the particular programming language you work in does not support this HDF5 function, then you can instead add a string attribute to your main dataset called axes. The axes attribute is simply a colon separated string naming the dimension descriptor datasets in order, so ”z:y:x” in this case.

## 4.3 Measurement

This group holds sample and instrument information. These groups are designed to hold relatively static data about the sample and instrument configuration at the time of the measurement. Rapidly changing ”positioner” values (aka scan) are better represented in the exchange group dataset.

There is a geometry group common to many of the subgroups under measurement. The intent is to describe the translation and rotation (orientation) of the sample or instrument component relative to some coordinate system. Since we believe it is not possible to determine all possible uses at this time, we leave the precise definition of geometry up to the technique. We do encourage the use of separate translation and orientation subgroups within geometry. As such, we do not describe geometry further here.

**Table 6:** Measurement Group Members

| Member | Type | Example |
|---|---|---|
| sample | group | |
| instrument | group | |

`sample`
: The sample measured.

`instrument`
: The instrument used to collect this data.

### 4.3.1 Sample

This group holds basic information about the sample, its geometry, properties, the sample owner (user) and sample proposal information. While all these fields are optional, if you do intend to include them they should appear within this parentage of groups.

`name`
: Descriptive name of the sample.

`description`
: Description of the sample.

`preparation_date`
: Date and time the sample was prepared.

`chemical_formula`
: Sample chemical formula using the CIF format.

**Table 7:** Sample Group Members

| Member | Type | Example |
|---|---|---|
| name | string dataset | "cells sample 1" |
| description | string dataset | "malaria cells" |
| preparation_date | string dataset (ISO 8601) | "2012-07-31T21:15:22+0600" |
| chemical_formula | string dataset (abbr. CIF format) | "(Cd 2+)3, 2(H2 O)" |
| mass | float dataset | 0.25 |
| concentration | float dataset | 0.4 |
| environment | string dataset | "air" |
| temperature | float dataset | 25.4 |
| temperature_set | float dataset | 26.0 |
| pressure | float dataset | 101325 |
| thickness | float dataset | 0.001 |
| position | string dataset | "2D" APS robot coord. |
| geometry | group | |
| experiment | group | |
| experimenter | group | |

`mass`
: Mass of the sample.

`concentration`
: Mass/volume.

`environment`
: Sample environment.

`temperature`
: Sample temperature.

`temperature_set`
: Sample temperature set point.

`pressure`
: Sample pressure.

`thickness`
: Sample thickness.

`position`
: Sample position in the sample changer/robot.

`geometry`
: Sample center of mass position and orientation.

`experiment`
: Facility experiment identifiers.

`experimenter`
: Experimenter identifiers.

#### 4.3.1.1 Geometry

This class holds the general position and orientation of a component. We do not define this further here.

**Table 8:** Geometry Group Members

| Member | Type | Example |
|---|---|---|
| *translation* | group | |
| *orientation* | group | |

***translation***
: The position of the object with respect to the origin of your coordinate system.

***orientation***
: The rotation of the object with respect to your coordinate system.

#### 4.3.1.2 Experiment

This provides references to facility ids for the proposal, scheduled activity, and safety form.

**Table 9:** Experiment Group Members

| Member | Type | Example |
|---|---|---|
| proposal | string dataset | "1234" |
| activity | string dataset | "9876" |
| safety | string dataset | "9876" |

`proposal`
: Proposal reference number. For the APS this is the General User Proposal number.

`activity`
: Proposal scheduler id. For the APS this is the beamline scheduler activity id.

`safety`
: Safety reference document. For the APS this is the Experiment Safety Approval Form number.

#### 4.3.1.3 Experimenter

Description of a single experimenter. Multiple experimenters can be represented through numbered entries such as experimenter_1, experimenter_2.

**Table 10:** Experimenter Group Members

| Member | Type | Example |
|---|---|---|
| name | string dataset | "John Doe" |
| role | string dataset | "Project PI" |
| affiliation | string dataset | "University of California, Berkeley" |
| address | string dataset | "EPS UC Berkeley CA 94720 4767 USA" |
| phone | string dataset | "+1 123 456 0000" |
| email | string dataset | "johndoe@berkeley.edu" |
| facility_user_id | string dataset | "a123456" |

`name` User name.

`role` User role.

`affiliation` User affiliation.

`address` User address.

`phone` User phone number.

`email` User e-mail address

`facility_user_id` User badge number

### 4.3.2 Instrument

The instrument group stores all relevant beamline components status at the beginning of a measurement. While all these fields are optional, if you do intend to include them they should appear within this parentage of groups.

**Table 11:** Instrument Group Members

| Member | Type | Example |
|---|---|---|
| `name` | string dataset | "XSD/2-BM" |
| source | group | |
| shutter_*N* | group | |
| attenuator_*N* | group | |
| monochromator | group | |
| detector_*N* | group | |

`name`
: Name of the instrument.

`source`
: The source used by the instrument.

`shutter_N`
: The shutter(s) used by the instrument.

`attenuator_N`
: The attenuators that are part of the instrument.

`monochromator`
: The monochromator used by the instrument.

`detector_N`
: The detectors that compose the instrument.

#### 4.3.2.1 Source

Class describing the light source being used.

**Table 12:** Source Group Members

| Member | Type | Example |
|---|---|---|
| name | string dataset | "APS" |
| datetime | string dataset (ISO 8601) | "2011-07-15T15:10Z" |
| beamline | string dataset | "2-BM" |
| current | float dataset | 0.094 |
| energy | float dataset | 4.807e-15 |
| pulse_energy | float dataset | 1.602e-15 |
| pulse_width | float dataset | 15e-11 |
| mode | string dataset | "TOPUP" |
| beam_intensity_incident | float dataset | 55.93 |
| beam_intensity_transmitted | float dataset | 100.0 |
| geometry | group | |

`name`
: Name of the facility.

`datetime`
: Date and time source was measured.

`beamline`
: Name of the beamline.

`current`
: Electron beam current (A).

`energy`
: Characteristic photon energy of the source (J). For an APS bending magnet this is 30 keV or 4.807e-15 J.

`pulse_energy`
: Sum of the energy of all the photons in the pulse (J).

`pulse_width`
: Duration of the pulse (s).

`source`
: Beam mode: TOPUP.

`beam_intensity_incident`
: Incident beam intensity in (photons per s).

`beam_intensity_transmitted`
: Transmitted beam intensity (photons per s).

#### 4.3.2.2 Shutter

Class describing the shutter being used.

**Table 13:** Shutter Group Members

| Member | Type | Example |
|---|---|---|
| name | string dataset | "Front End Shutter 1" |
| status | string dataset | "OPEN" |
| geometry | group | |

`name`
: Shutter name.

`status`
: "OPEN" or "CLOSED" or "NORMAL"

#### 4.3.2.3 Attenuator

This class describes the beamline attenuator(s) used during data collection. If more than one attenuators are used they will be named as attenuator_1, attenuator_2 etc.

**Table 14:** Attenuator Group Members

| Member | Type | Example |
|---|---|---|
| thickness | float dataset | 1e-3 |
| attenuator_transmission | float dataset | unit-less |
| type | string dataset | "Al" |
| geometry | group | |

`thickness`
: Thickness of attenuator along beam direction.

`attenuator_transmission`
: The nominal amount of the beam that gets through (transmitted intensity)/(incident intensity).

`type`
: Type or composition of attenuator.

#### 4.3.2.4 Monochromator

Define the monochromator used in the instrument.

**Table 15:** Monochromator Group Members

| Member | Type | Example |
|---|---|---|
| type | string dataset | "Multilayer" |
| energy | float dataset | 1.602e-15 |
| energy_error | float dataset | 1.602e-17 |
| mono_stripe | string dataset | "Ru/C" |
| geometry | group | |

`type`
: Multilayer type.

`energy`
: Peak of the spectrum that the monochromator selects. Since units is not defined this field is in J and corresponds to 10 keV.

`energy_error`
: Standard deviation of the spectrum that the monochromator selects. Since units is not defined this field is in J.

`mono_stripe`
: Type of multilayer coating or crystal.

#### 4.3.2.5 Detector

This class holds information about the detector used during the experiment. If more than one detector are used they will be all listed as detector_$N$.

**Table 16:** Detector Group Members

| Member | Type | Example |
|---|---|---|
| manufacturer | string dataset | "CooKe Corporation" |
| model | string dataset | "pco dimax" |
| serial_number | string dataset | "1234XW2" |
| geometry | group | |
| output_data | string dataset | "/exchange" |

`manufacturer`
: The detector manufacturer.

`model`
: The detector model.

`serial_number`
: The detector serial number .

`output_data`
: String HDF5 path to the exchange group where the detector output data is located.

## 4.4 Provenance

Data provenance is the documentation of all transformations, analyses and interpretations of data performed by a sequence of process functions or `actors`.

Maintaining this history allows for reproducible data. The Data Exchange format tracks provenance by allowing each actor to append provenance information to a process table. The provenance process table tracks the execution order of a series of processes by appending sequential entries in the process table.

Scientific users will not generally be expected to maintain data in this group. The expectation is that analysis pipeline tools will automatically record process steps using this group. In addition, it is possible to re-run an analysis using the information provided here.

**Table 17:** Process table to log actors activity

| actor | start_time | end_time | status | message | reference | description |
|---|---|---|---|---|---|---|
| gridftp | 21:15:22 | 21:15:23 | FAILED | auth. error | /provenance/griftp | transfer detector to cluster |
| gridftp | 21:15:26 | 21:15:27 | FAILED | auth. error | /provenance/griftp | transfer detector to cluster |
| gridftp | 21:17:28 | 22:15:22 | SUCCESS | OK | /provenance/griftp | transfer detector to cluster |
| norm | 22:15:23 | 22:30:22 | SUCCESS | OK | /provenance/norm | normalize the raw data |
| rec | 22:30:23 | 22:50:22 | SUCCESS | OK | /provenance/rec | reconstruct the norm. data |
| convert | 22:50:23 | | RUNNING | OK | /provenance/export | convert reconstructed data |
| gridftp | | | QUEUED | | /provenance/griftp_2 | transfer data to user |

`actor`
: Name of the process in the pipeline stage that is executed at this step.

***start_time***
: Time the process started.

***end_time***
: TIme the process ended.

***status***
: Current process status. May be one of the following: QUEUED, RUNNING, FAILED, or SUCCESS.

***message***
: A process specific message generated by the process. It may be a confirmation that the process was successful, or a detailed error message, for example.

***reference***
: Path to a process description group. The process description group contains all metadata to perform the specific process. This reference is simply the HDF5 path within this file of the technique specific process description

group. The process description group should contain all parameters necessary to run the process, including the name and version of any external analysis tool used to process the data. It should also contain input and output references that point to the exchange_$N$ groups that contain the input and output datasets of the process.

***description***
Process description.

# A Appendix

## A.1 Default units for Data Exchange entries

The default units for Data Exchange entries follow the CXI entries definition, i.e. are SI based units (see table **??**) unless the ”units” attribute is specified. Data Exchange prefers to use the default SI based units whenever possible.

**Table 18:** SI (and common derived) base units for different quantities

| Quantity | Units | Abbreviation |
|---|---|---|
| length | meter | m |
| mass | kilogram | kg |
| time | second | s |
| electric current | ampere | A |
| temperature | kelvin | K |
| amount of substance | mole | mol |
| luminous intensity | candela | cd |
| frequency | hertz | Hz |
| force | newton | N |
| pressure | pascal | Pa |
| energy | joule | J |
| power | watt | W |
| electric potential | volt | V |
| capacitance | farad | F |
| electric resistance | ohm | Ω |
| absorbed dose | gray | Gy |
| area | square meter | m^2 |
| volume | cubic meter | m^3 |

### A.1.1 Exceptions

Angles are always defined in degrees *not* in radians and use the abbreviation ”degree”.

### A.1.2 Times and Dates

Times and Dates are always specified according to the ISO 8601. This means for example “1996-07-31T21:15:22+0600”. Note the “T” separating the data from the time and the “+0600” timezone specification.

## A.2 Geometry

### A.2.1 Coordinate System

The Data Exchange uses the same CXI coordinate system. This is a right handed system with the z axis parallel to the X-ray beam, with the positive z direction pointing away from the light source, in the downstream direction. The y axis is vertical with the positive direction pointing up, while the x axis is horizontal completing the right handed system (see Fig. **??**). The origin of the coordinate system is defined by the point where the X-ray beam meets the sample.

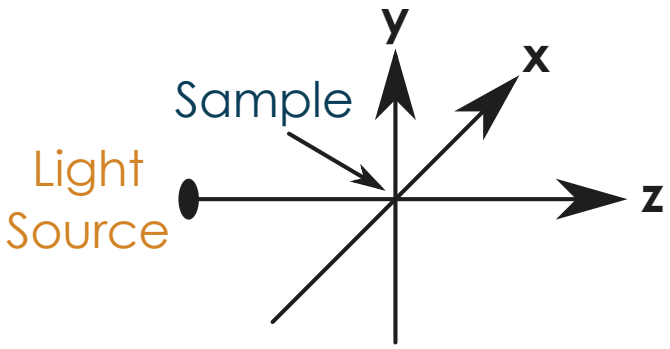


**Figure 9:** The coordinate system used by CXI. The intersection of the X-ray beam with the sample define the origin of the system. The z axis is parallel to the beam and points downstream.

### A.2.2 The local coordinate system of objects

For many detectors their location and orientation is crucial to interpret results. Translations and rotations are used to define the absolute position of each object. But to be able to apply these transformations we need to know what is the origin of the local coordinate system of each object. Unless otherwise specified the origin should be assumed to be the geometrical center of the object in question. The default orientation of the object should have the longest axis of the object aligned with the x axis, the second longest with the y axis and the shortest with the z axis.