PaN-data Europe

Deliverable D5.3

Revised specification of data standards

Grant Agreement Number	261537
Project Title	PaN-data Europe Strategic Working Group
Title of Deliverable	Revised specification of data standards
Deliverable Number	D5.3
Lead Beneficiary	STFC
Deliverable Dissemination Level	Public
Deliverable Nature	Report
Contractual Delivery Date	31 May 2011 (Month 12)
Actual Delivery Date	30 July 2011

The PaN-data Europe project is partly funded by the European Commission under the 7th Framework Programme, Information Society Technologies, Research Infrastructures.

Abstract

This document describes the choices to be made in order to arrive at a data standard for the facilities in the PaN-data consortium. In addition a roadmap is presented which details how a common data standard for PaN-data facilities can be achieved.

Keyword list

Data, Format, Standard, X-ray, Neutron, Roadmap.

Document approval

Approved for submission to EC by all partners (July 2011).

Revision history

Issue	Author(s)	Date	Description
0.1	Mark Koennecke	25 Mar 2011	Initial version
0.2	Mark Koennecke	17 May 2011	Updated with changes from Armando Sole, Carlos Izarra
0.3	Mark Koennecke	3 Jun 2011	Major rework after Diamond meeting
0.4	Mark Koennecke	29 Jun 2011	Minor additions
0.5	Mark Koennecke	4 Jul 2011	Minor structural changes as suggested by Michael Wilson
1.0	Simon Lambert	30 Jul 2011	Complete version ready for submission

Table of contents

Page

1 Introduction

The aim of this workpackage is to:

- identify areas where improved data sharing is desirable
- identify mechanisms by which scientific data can be more easily shared between different facilities
- propose a roadmap of how improved data sharing may be achieved

As of now, most participating facilities use a plethora of home grown data formats for storing experimental data. This causes major problems when trying to reuse data as data conversion tools need to be written or data analysis programs need to be modified in order to digest data coming from other institutions. Quite often the information required to perform data analysis on a given data set is distributed across several files. This poses the problem of keeping all the required files together and in synchronisation.

Data formats are important within the PaN-data project in two areas: firstly PaN-data strives to encourage collaboration in data analysis software. Such collaboration becomes much easier when software developed at one partner's facility can immediately work with data created at another partner’s facility. Another aim of PaN-data is the creation of a pan-European data catalogue. This catalogue will enable scientists to search for data they are interested in. Now consider the situation when a scientist has found some interesting data in the database and has been able to download it. If the data was in a well defined format, the scientist could immediately use the data. In the current situation, with multiple incompatible data formats, they first have to investigate the format the data has been written in and find a way to read it. Depending on their computing skill level they may even have to enlist professional help in order to get the job done.

In order to facilitate both the collaboration on data analysis software and in order to gain maximum use of the pan-European data database, PaN-data sets out to define a shared data format.

2 Data format standardisation

There are a couple of choices to be made when defining a data format. These choices and their consequences are discussed in this section.

2.1 Data format use cases

When discussing data formats it helps to look at the typical usage scenarios for data files first.

The first and basic use case is that numbers and arrays have to be written to files and to be read back correctly from it. This is the purpose of a physical file format. The physical file format should be accessible from a wide range of computing platforms and programming languages. It is also advantageous if this physical file format is supported by many general purpose scientific tools. In order to support long term maintainability of the physical file format both its specification and the application programmer interfaces necessary to access files written in the physical file format should be open source. Standardising on a physical file format allows the data file contents to be shared correctly – this still leaves the question of how the data should be interpreted, but if the data has not been read correctly in the first place "interpretation" would not be valid anyway. A good choice of physical file format would allow data to be structured appropriately and appropriate meta data to be added. In such situations an expert in the scientific domain the data belonged to could make a good attempt at interpreting it.

In the context of physical file formats it is necessary to consider what to do with proprietary data files written by some equipment. This actually depends on the length of the intended storage period for the data files. If the storage period is short, such proprietary formats may be used. For long term storage, however, a conversion into a open source physical file format with characteristics as described above must be performed. The reason is that the long term availability of the tools to access proprietary file formats is not guaranteed.

The second use case of a data file is as an interface to data reduction or data analysis software. This interface is of course specific for each experimental technique. Typically between 10-20 parameters are required to perform data processing. It makes the life of a programmer writing data reduction software much easier if they can rely on finding the necessary data under a specific name and at a specific position in a data file. The set of technique specific parameters will be called an "application definition" in the rest of this document. Defining such application definitions supports software reuse. This is one of the goals of PaN-data, especially of work package 6. It should be noted that an "application definition" is not specific to a particular data format – it just encapsulates the information required for a particular application of the data.

A third use case of a data file is as an interface to data file search software. The data policy agreed upon in work package 2 of PaN-data calls for a searchable data file index. In order to support such an index a data file must contain a minimum set of metadata to be used in searching for data files. This includes things such as title, sample name, user information, proposal ID, time of the experiment etc.

A fourth use case of a data file is as a container for long term data archival. This use implies that a user of an archived file finds enough information about the experiment in the file in to be able to reconstruct what had been done. A full description of the beamline (FBD) is desirable. This can easily be hundreds of parameters. Long term archival also implies that a possible user has to able to give meaning to the data fields in the file. The transportation of meaning can happen in various ways. One is a document giving details about data fields which is supplied with the data file. The other option is to use an agreed upon vocabulary of field names. This vocabulary can be reused across techniques. The NeXus format provides a suitable vocabulary through its rules for structuring data files and its base classes. Having a full beamline description in the data file is desirable for two further reasons:

- It increases the chances that enough data is available to support not yet foreseen data analysis or data mining techniques in the future. Thus the FBD enables new science.
- The FBD allows others to inspect and re analyse data files. This supports scientific integrity.

2.2 Forms of data

Data can come in various forms, from raw data through reduced to processed data and finally published data. When defining a data standard the level at which data is shared has to be considered.

The basic data is raw data from the instrument. The writing of such data happens usually automatically from the data acquisition (DAQ) software. Thus raw data is not subject to any interpretation and the most trustworthy. Raw data provides an interface to data reduction programs and, sometimes, directly to data analysis software.

In some cases the correction of the raw data is very detector or instrument specific. Such data reduction software cannot be shared. In these cases it makes sense to share data which has also been reduced at the facility. Sharing such data requires trust into the providing facility to get the initial data reduction right. Reduced data provides an interface to data analysis software.

The other level of data is processed data. This is data which has been corrected and converted into some physical meaningful space. Examples include I(Q), S(Q,OM) or integrated Intensities in protein crystallography. Typically processed data files are used to exchange data between various data analysis programs. Even more trust into the processing of the data is required to use such data. Still it makes sense to define data formats for such data too. It is desirable however that such data contains a record of the files, parameters and programs which were used to arrive at the processed data set so that a re-run of the data reduction could be performed with either the same or different parameters.

The last level of data is the data resulting from data processing. This can be crystal structures etc. Usually standards for the exchange of such data already exist, for example the CIF format (International Union of Crystallography). Thus they will not be considered any further in the PaN-data standardization process.

2.3 Choice of data formats

Finally a data format needs to be chosen. Contenders are:

- HDF-5 stands for Hierarchical Data Format Version 5. HDF-5 was developed by the hdfgroup. It is purely a physical file format and provides for the efficient storage of numbers and arrays. HDF-5 fulfils all requirements for a physical file format as detailed in section 2.1. HDF-5 does not itself provide any naming or file structuring conventions.
- HDF-5 with no specified hierarchy, but following NeXus conventions for e.g. units and axis association. Users will be free to define their own data field names. Only the names required by an application definition are defined.
- Full NeXus on top of HDF-5 as physical file format. This includes application definitions. NeXus rules for a full beamline description and the NeXus vocabulary for field names is used.

3 Standardisation roadmap

The PaN-data consortium already agreed upon some basic properties of the PaN-data data format.

There is a consensus that the physical file format will be HDF-5. It is an efficient binary platform independent format and is open source. Among scientific data formats HDF-5 is the clear market leader. Most proprietary and open source tools support reading and writing of HDF-5 files. HDF-5 alone already facilitates dictionary based file reading approaches as suggested by the Common Data Model proposal. So the first stage of the standardisation roadmap is agreed and many partners are already using HDF5 (either natively or via the NeXus API).

The next stage on from standardising on the physical format is standardise on items of the file content. This will require discussion and agreement between interested parties and it is important to choose areas where agreement, adoption and scientific gain will be the highest. It was also decided to limit the PaN-data effort to a number of commonly used techniques. These are:

- Protein Crystallography (PX)
- Tomography
- Tomography processed data
- Small angle scattering
- Monochromatic powder diffraction
- X-ray fluorescence

NeXus structures and application definitions will serve as a starting point. Therefore more details on NeXus are given in an appendix. At this stage it must be emphasized that the application definitions developed for NeXus are actually file format **independent**. They just detail what sort of information needs to be stored for the various techniques to facilitate data exchange/reuse.

In order to achieve the goals of PaN-data standard data files must at least fulfil the conditions of the use cases 2 and 3 from section 2.1. This means that they fulfil some form of an application definition in order to support data exchange and contain enough metadata for indexing files into a file database.

The standardization process will have the following steps:

1. For each technique interested partners will come together and detail the file format to use
2. The interested partners will then implement the file format in their DAQ and DA software.
3. When it is shown that the standard works, it will be ratified by the other partners.

4. The PANDATA standard will also be circulated to the wider NeXus community and NeXus International Advisory Committee (NIAC) for comment

It must be emphasised that this process has to be performed for each of the techniques the PaN-data consortium tries to standardize. Different partners may be involved for different techniques.

3.1 Partner Selection and Technique Specific Standard Evaluation

For each of the techniques which were selected for standardization a number of PaN-data facilities have to be found which are interested to develop the standard. At least two partners need to work together. These partners then get together and decide upon a number of properties of the data standard for the chosen technique. These properties are:

- The use cases from section 2.1 to be supported by the standard
- Which form of data, raw, reduced or processed to standardize upon
- Which data format to use, section 2.3
- The number and names of data items to store in a standard compliant file. The ex-isting application definitions from NeXus can serve as a starting point. Some coordination with the other partners is required in this step in order to minimize the number of different names for the same thing.

The partners will also study and document the technique specific use cases of data. The outcome could also be a set of standards for the technique. This process should be finished within 3-6 months.

3.2 Implementation

With a candidate data standard established as described in section 3.1 the partners working on a technique specific standard implement the standard. They modify their DAQ system to produce standard compliant files and adapt their preferred data analysis tools to read standard compliant files. Then the whole data analysis pipeline will be tested and the standard and the tools improved until the following goals are met:

1. At least 80-90% of the technique specific data uses cases identified in 3.1 will be met
2. Each partner can process data generated at the other partner facilities. This is why at least two partner should collaborate in the development of a technique specific standard.

This process should be accomplished within 3-6 months.

3.3 Review and Adoption

The outcomes of section 3.2 are technique specific standards which are known to work. These technique specific standards will then be reviewed by the other partners and ratified. Then the other partners can choose to implement the technique specific standards at their facilities. In this stage and other stages consultations with the NeXus International Advisory Committee (NIAC) should take place.

4 Summary

Given the diversity of data formats currently used by European synchrotron and neutron sources, it is not realistic to to establish or impose a standard right now. The project recommends that the proposed roadmap towards creating a standard is adopted in order to arrive at a standard which is proven to be practical. And then to arrive at a consensus on a data standard.

5 References

NeXus data format http://www.nexusformat.org/
The NeXus manual: http://download.nexusformat.org/doc/NeXusManual.pdf
HDF group website http://www.hdfgroup.org/

6 Annex: Description of NeXus

The PaN-data data standardization process will take advantage of the work already per-formed within the NeXus data format. Thus some details of NeXus are given below.

6.1 Brief Introduction to NeXus

NeXus is based on the HDF-5 file format, a binary file format proposed to become an ISO standard by the European Union. In an HDF-5 file there are different kinds of objects:

- **Groups** group together both other groups and scientific data sets. They can be compared to directories in a hierarchical file system. Groups in NeXus have both a name and a class name.
- **Datasets** are multi-dimensional datasets of various numeric types. Datasets represent the actual payload of the file.
- **Attributes** can be used at group or dataset level to store meta data about the dataset or the group.
- **Links** sometimes it is convenient to refer to the same dataset at two or more places within the file hierarchy. Links allow one to do this without duplicating datasets. Such links are very similar to symbolic links as used in Unix file systems.

NeXus data files have a hierarchical structure. There are slightly different file structures for raw data files and processed data files. An example of a raw data file structure is shown in Figure 1. At the root level of each NeXus file there are one to many groups of class NXentry. NXentries allow NeXus files to store multiple related experiments or a whole workflow in one file. A raw data NXentry holds some fields and the following other groups:

- **NXinstrument** This group contains further groups which describe the beamline components which make up the instrument.
- **NXsample** This group holds information on the sample
- **NXmonitor** This group holds information on the counting parameters of the instrument: counting mode, preset, elapsed time etc.
- **NXdata** This groups holds links to those data items which an automatic plotting program like NXplot would use in order to generate a default plot of the data in the file.

```
entry:NXentry
          title
          start_time
          definition
          instrument:NXinstrument
              source:NXsource
                  type
                  name
                  probe
              monochromator:NXcrystal
                  wavelength[i]
              detector:NXdetector
                  polar_angle[ndet]
                      @axis=1
                  data[ndet]
                      @signal=1
          sample:NXsample
              name
              rotation_angle
          control:NXmonitor
              mode
              preset
              integral
          data:NXdata
              polar_angle --> /NXentry/NXinstrument/NXdetector/polar_angle
              data --> /Nxentry/NXinstrument/NXdetector/data
```

Fig. 1: Example NeXus raw data file hierarchy. Hierarchy levels are indicated by indentation. Attributes are prefixed with @. Links are shown as -->.

For some techniques it may be more appropriate to standardize on reduced data. One such example is tomography. Standardization of a raw data file format allows for collaboration on data reduction software; standardization on the resulting reconstructed volume data allows for collaboration on data visualisation and analysis software. NeXus stores processed data in a simplified NXentry hierarchy as seen in Figure 2. A processed data NXentry contains the following subgroups:

- **NXdata** The result of the data reduction or analysis is stored directly in a NXdata group together with the data sets describing the axes of a potentially multi dimensional dataset.
- **NXsample** contains the sample details
- **NXprocess** contains a description of the kind processing which took place: the program and version used etc. Moreover the NXprocess group can contain NXparameter groups which describe the input and output parameters used for this run.
- **NXinstrument** and other NeXus groups are optional in processed data files. If information about the instrument is required in a processed data file it should be stored in an NXinstrument hierarchy.

The possible fields in each group are defined in dictionaries documented in the NeXus manual and the NeXus WWW-site. Further NeXus rules allow the association of axes with multi dimensional datasets or specifying how individual data items have to be stored.

```
entry:NXentry
    title
    definition
    instrument:NXinstrument
        source:NXsource
            type
            name
            probe
    sample:NXsample
        name
    reconstruction:NXprocess
        program
        version
        date
        parameters:NXparameters
            raw_file
    data:NXdata
        data[nx,nx,nz]
            @signal=1
            @axes =x,y,z
        x[nx]
        y[ny]
        z[nz]
```

Fig. 2: Example processed data NeXus file.

A common approach in experimental techniques is to perform scans to collect data at various positions in a sample. This is very hard to standardize as in principle any parameter can be scanned. NeXus solves this problem through a set of rules which describe how scans and rasterisations are to be stored in a file. These rules can be applied to all the data file standards given in the annex. The rules to apply are:

- The scan dimension NP is always the first dimension in any multi dimensional dataset.
- All data is stored as arrays of dimension NP, original dimensions of the data at the appropriate position in the NeXus hierarchy.
- The NXdata group must contain all the parameters varied throughout the scan and the detector data either directly or linked. Thus the NXdata group mimics the popular tabular representation of a scan.

An example of such a file structure is given in Figure 3.

```
entry:NXentry
     instrument:NXinstrument
        detector:NXdetector
            data[np,number of x pixels,number of y pixels]
                @signal=1
                @interpretation=image
    sample:NXsample
        rotation_angle[np]
            @axis=1
            @interpretation=scalar
    data:NXdata
        rotation_angle --> /NXentry/NXsample/rotation_angle
        data --> /NXentry/NXinstrument/NXdetector/data
```

Fig. 3: Example NeXus scan. The scan was rotation of the sample versus an area detector.

For more detailed information on scans and other advanced topics of NeXus usage please consult the NeXus manual.

6.2 NeXus Application Definitions

The actual data file content for the proposed PaN-data standards is given in the appendix in the form of NeXus application definitions. A NeXus application definition specifies which minimum data content and what structure a file must have in order to be standard compliant for a given technique. This is actually the strict definition of a standard for a specific application of NeXus. Another way to look at a NeXus application definition is as a contract between file writers and file users about the content of a file for a specific use case. The beauty of the NeXus approach is that additional data can be added to the file without breaking the standard.

NeXus application definitions are written in an application of XML called the NeXus Definition Language (NXDL). As raw XML is difficult to read, we choose to document the application definitions in the same simplified form as used in Figure 1 and 2. In this form:

- Indentation describes hierarchy level
- NeXus groups are annotated in the form name:class name. This is name and class name separated by a colon, with name being optional. Groups can nevertheless be recognised as their names always start with NX.
- Datasets are represented as simple names. If the dataset is multi dimensional, the dimensions are given in square brackets
- Attributes are prefixed with an 'at' sign, @
- Links are annotated as name --> path to the linked data item.

Please keep in mind that NeXus application definitions are being used here as a means to communicate the required content of a data file for a specific technique.

There is yet another special case to consider: this is the case of the multi method instrument. Some instruments apply multiple techniques at the same time to the same sample. The rule for such cases is to build a NXentry containing the full instrument. Within this NXentry, additional hierarchies of type NXsubentry are constructed which contains links to the technique specific data. See Figure 4 for an example.

```
entry:NXentry
    user:NXuser
    sample:NXsample
    instrument:NXinstument
        SASdet:NXdetector
            data:[,]
                @signal = 1
        fluordet:NXdetector
            data:[,]
                @signal = 1
        large_area:NXdetector
            data:[,]
    SAS:NXsubentry
        definition = "NXsas"
        instrument:NXinstrument
            detector:NXdetector
                data --> /entry/instrument/SASdet/data
        data:NXdata
            data --> /entry/instrument/SASdet/data
    Fluo:NXsubentry
        definition = "NXFluo"
        instrument:NXinstrument
            detector --> /entry/instrument/fluordet/data
            detector2 --> /entry/instrument/large_area/data
        data:NXdata
            detector --> /entry/instrument/fluordet/data
```

Fig. 4: Example NeXus hierarchy using the NXsubentry structure for multi method instruments.

7 Annex: NeXus Application Definitions

7.1 Protein Crystallography

This is a simple rotation scan around the sample rotation axis. Changes by Frank Schluenzen are included.

```
entry:NXentry
   definition
   instrument:NXinstrument
     detector:NXdetector
       polar_angle
       beam_center_x
       beam_center_y
       data[np,number of x pixels,number of y pixels]
       @signal=1
       x_pixel_size
       y_pixel_size
       distance
       frame_start_number
      timestamp[np]
   attenuator:NXattenuator
     attenuator_transmission
   source:NXsource
     type
     name
     probe
   monochromator:NXmonochromator
     wavelength
   sample:NXsample
     rotation_angle[np]
     @axis=1
     rotation_angle_step[np]
     @axis=1
     name
     orientation_matrix[3,3]
     unit_cell[6]
     temperature[np]
     x_translation[np]
     y_translation[np]
     z_translation[np]
   data:NXdata
     rotation_angle --> /NXentry/NXsample/rotation_angle
     data --> /NXentry/NXinstrument/NXdetector/data
   title
   start_time
   control:NXmonitor
     mode
     preset
     integral
```

7.2 Kappa Geometry Diffractometer

Many PX beamlines now use kappa geometry diffractometers.

```
entry:NXentry
    definition
    instrument:NXinstrument
        detector:NXdetector
            polar_angle[np]
            data[np,number of x pixels,number of y pixels]
                @signal=1
                @signal=1
            x_pixel_size
            y_pixel_size
            distance
            frame_start_number
        source:NXsource
            type
            name
            probe
        monochromator:NXmonochromator
            wavelength
    sample:NXsample
        rotation_angle[np]
            @axis=1
        kappa[np]
            @axis=1
        phi[np]
            @axis=1
        alpha
        name
        orientation_matrix[3,3]
        unit_cell[6]
        temperature[NP]
        x_translation
        y_translation
        distance
    name:NXdata
        polar_angle --> /NXentry/NXinstrument/NXdetector/polar_angle
        rotation_angle --> /NXentry/NXsample/rotation_angle
        kappa --> /NXentry/NXsample/kappa
        phi --> /NXentry/NXsample/phi
        data --> /NXentry/NXinstrument/NXdetector/data
    title
    start_time
    control:NXmonitor
        mode
        preset
        integral
```

7.3 Tomography Raw Data

This is again a scan around the sample rotation axis. However, in tomography it is common to collect dark field and bright field images before, during and after the actual sample scan in order to be able to correct the data for detector effects. For data reduction the order in which those images have been collected is important.

```
entry:NXentry
    title
    start_time
    end_time
    definition
    instrument:NXinstrument
        NXsource
            type
            name
            probe
         detector:NXdetector
            data[nFrames,xsize,ysize]
                @signal=1
            image_key[nFrames]
            x_pixel_size
            y_pixel_size
            distance
    sample:NXsample
        name
        rotation_angle[nFrames]
            @axis=1
        x_translation[nFrames]
        y_translation[nFrames]
        z_translation[nFrames]
    control:NXmonitor
        data[nFrames]
    data:NXdata
        data --> /NXentry/NXinstrument/data:NXdetector/data
        rotation_angle --> /NXentry/NXsample/rotation_angle
```

Where image_key is an array which holds for each image either 0,1 or 2 depending if it is a sample, bright field or dark field image.

7.4 Tomography Processed Data

Processed data for tomography is the reconstructed sample volume. This is described in a suitable NeXus processed data definition.

entry:NXentry

```
    title
    definition
    NXinstrument
        NXsource
            type
            name
            probe
    NXsample
        name
    reconstruction:NXprocess
        program
        version
        date
        parameters:NXparameters
            raw_file
    data:NXdata
        data[nx,nx,nz]
            @signal=1
        x[nx]
            @axis=1
        y[ny]
            @axis=2
        z[nz]
            @axis=3
```

7.5 Small Angle Scattering

It is proposed to use the same file format for all small angle techniques: small angle x-ray scattering, small angle neutron scattering, wide angle scattering and grazing angle small angle scattering. This was done by storing the appropriate rotation angles at the sample and detector positions.

```
NXentry
    title
    start_time
    end_time
    definition
    instrument:NXinstrument
        source:NXsource
            type
            name
            probe
        monochromator:NXmonochromator
            wavelength
            wavelength_spread
        collimator:NXcollimator
            geometry:NXgeometry
                shape:NXshape
                    shape
                    size
        detector:NXdetector
            data[nXPixel,nYPixel]
                @signal=1
            distance
            x_pixel_size
            y_pixel_size
            polar_angle
            azimuthal_angle
            rotation_angle
            meridional_angle
            beam_center_x
            beam_center_y
        name
    sample:NXsample
        name
        meridional_angle
    control:NXmonitor
        mode
        preset
        integral
    data:NXdata
        data --> /NXentry/NXinstrument/NXdetector/data
```

7.6 Powder Diffraction

This is the simple case of a x-ray or neutron powder diffractometer at a monochromatic source.

```
entry:NXentry
    title
    start_time
    definition
    instrument:NXinstrument
        source:NXsource
            type
            name
            probe
        monochromator:NXcrystal
            wavelength
        detector:NXdetector
            polar_angle[ndet]
                @axis=1
            data[ndet]
                @signal=1
    sample:NXsample
        name
    control:NXmonitor
        mode
        preset
        data
    data:NXdata
        polar_angle --> /NXentry/NXinstrument/NXdetector/polar_angle
        data --> /NXentry/NXinstrument/NXdetector/data
```

7.7 X-ray Fluorescence

```
entry:NXentry
    title
    start_time
    definition
    instrument:NXinstrument
        source:NXsource
            type
            name
            probe
        monochromator:NXmonochromator
            wavelength
        fluorescence:NXdetector
            data[nenergy]
                @axes=energy
                @signal=1
                @interpretation=spectrum
            energy[nenergy]
                @interpretation=scalar
            calibration_type
            calibrartion_parameters[]
    sample:NXsample
        name
    control:NXmonitor
        mode
        preset
        elapsed_time
        live_time
        data
    data:NXdata
        energy --> /entry/instrument/fluorecence/energy
        data --> /entry/instrument/fluorecence/data
```

This definition includes additions by Armando Sole. Please note that the definition is for a single spectrum, NeXus scan rules apply.

7.7.1 X-ray Fluorescence Short Version

```
entry:NXentry
    title
    start_time
    MCA_1(This name is free):NXsubentry
        definition="NXfluo"
        (*)counts[npoints, nchannels]
        channels[nchannels]
        calibration_parameters[]
        calibration_type
        preset_time[npoints]
        elapsed_time[npoints]
        live_time[npoints]
        excitation_energy[npoints]
    MCA_2(This name is free):NXsubentry
        definition="NXfluo"
        (*)counts[npoints, nchannels]
        channels[nchannels]
        calibration_parameters[]
        calibration_type
        preset_time[npoints]
        elapsed_time[npoints]
        live_time[npoints]
        excitation_energy[npoints]
    ...
```

This is an example of a flattened application definition, without hierarchy as suggested by Armando Sole.