NeXus: show-and-tell meeting

Reflectometry at P08

13 Feb 2024 Indico page.
Next meeting (ALPS II).

Attendees:

- Christoph Rosemann (DESY)
- Jan Kotanski (DESY)
- George O'Neill (ESS)
- Niels Lefeld (DESY)
- Sven Karstensen (DESY)
- Hans-Peter Schlenvoigt (HZDR)
- Regina Hinzmann (DESY)
- Chen Zhang (ORNL)
- Rachel Wolf (DESY)
- Peter Chang (RAL/DIAMOND)
- Paul Millar (DESY)

Notes

Christoph: 15 years experience in HEP (various frameworks with different file formats) and 8 years experience in photon science (more basic "things", several designs and implementations on – small processing tools, algorithms, prototypes; data formats for 1–4D data; loosely based on HDF5 and NeXus.

The basic idea: standard data format for intermediate processing, between DAQ and final analysis.

The obvious/only choice is NeXus (HDF5).

LISA ("Liquid Interface Scattering Apparatus") experiment is located at P08 (High Res Diffraction), at PETRA-III.

Geometry setup and motor names – many angles.
Motor angles start with "m"
Sample stage (angles start with "s")
Detector part (angles start with "d")
Motor, sample and detector are decoupled.

DAQ is driven by TANGO/SARDANA. Imaging detector (Lamda) writes NeXus and everything else can be ASCII. Jan Kotanski is working on standardising this.

Challenges:
Brittle setup: any change of a name, cable or address will alter the output without warning. (swapping photo diode cables could lead to valid looking data, but analysis would lead to invalid results)
No import/export of this data, requires local expert knowledge.
Data flow: initial data from control system, transient formulation, persistency.

Example data is shown, using the FIO (“header and data”) format.
A dataset is a set of scans, typically the data over a whole range of qz values.
Sample and detector are constant over the dataset.
Scan is a list scan points.

Christoph showed a proposed NeXus definition.

(Q. How to show a NeXus structure in a nice tree?)

Questions:

Is there a transient class instantiation?

Mapping of classes, mapping of hierarchy?

Is there a better way to represent the hierarchy of data in NeXus?

Hard to understand without

Developing a NXDL definition

(manual process currently)

How to extend NeXus with functionality

Geometry in NeXus

Resources

Q NXposition?

Still struggling to do it in the “standard way”. The (x,y,z) coordinate.

Several NXentry nested. This could be problematic.

All physical items (the NXattenuator) are located under the NXinstrument.
Double-crystal deflector: do you describe the beam in terms of its geometry: its position in space that impinges on the sample.

The people doing the experiment have the detector beam following the incident beam.

Still need to talk with the people conducting the experiments.

Setup is in some kind of calibrated fashion? So wouldn’t we need this information for the data to FAIR.

Christoph: thought about using links?

Peter: an entry is one measurement. All scanpoints should be within one NXentry. I think either there’s a single NXentry for the whole measurement or several entries - one for each scan command.

Christoph: At the beginning, there’s high intensity so high level of attenuation (and control exposure time). For larger angles, tess attenuation is needed (and longer exposure time).

Sure, if your data acquisition can write the attenuation and exposure time per scan point in the same fields then you could use a single NXentry.

Initial question about the structure:

JanK: did you look at other NeXus files?

George: Have a look at “NXCollection“ general catch-all to group together data.
Christoph: trying to avoid NXCollectin due to its loose semantics.
Perhaps subclassing NXCollection to provide more specific semantics.

Transient class:

NeXpy has python classes. Auto-generating class definitions through XSLT. DIAMOND currently have examples of this that transform NXDL to Java classes.

https://github.com/DawnScience/scisoft-core/tree/master/org.eclipse.dawnsci.nexus

FAIRMAT have recently added a translation between NXDL and YAML. This supports both directions, allowing authoring base classes in YAML.

In the definitions repo, there’s a target (nyaml) in the make file that generates YAML and another target (nxdl) to transform back.

In MX land, there’s already people using multi-axis . They have already understood how to transform from the lab frame to the detector frame. The same applies for a sample. Axis values are written to fields in NXtransformations groups and each field is linked by a depends_on attribute to create dependency chains from detector module or sample to the lab frame.

Have a look at the NXmx class. This provides an example of how transformations can be chained.

George: “depends on” can be useful in linking different axis.