DECTRIS®

detecting the future

EIGER HDF5 Data and NeXus Format

Andreas Förster – Application Scientist MX

IUCr Workshop on Metadata, ECM, Rovinj, 22 Aug 2015

DECTRIS Ltd.
5400 Baden
Switzerland
www.dectris.com

Outline

1. EIGER data

- *Need for new data format*

2. HDF5 format

- *Concept*
- *Implementation*

3. Metadata considerations

- *What we do*
- *What needs to be done*

PILATUS data – CBF files

- ***Images saved individually***
  Writing 1 frame/file was ok for slow detectors.
  This is reaching its limits now.
- ***Metadata in header of each image***
  Header in mini-CBF format.
  Limited amount of metadata.

EIGER – Coming to a synchroton near you

	Maximum frame rate	Size of dataset (30 s acquisition)	Continuous frame rate
16M	133 Hz	3990 images	65 Hz
9M	238 Hz	7140 images	119 Hz
4M	750 Hz	22.500 images	375 Hz
1M	3000 Hz	90.000 images	1500 Hz

New file format needed

EIGER 1M @ 3 kHz: 90k frames/30 sec

- ***Data container***
  Writing 1 frame/file is not efficient.
- ***Fast and efficient data compression***
  Compressed data stream should be < 800 MB/s.
  Fast decompression for reading.
- ***Flexible storage of header information***
  Frame time, detector name, applied corrections, sample position, …
- ***Real-time logging of beamline parameters***
  Detector distance, oscillation range, goniometer geometry

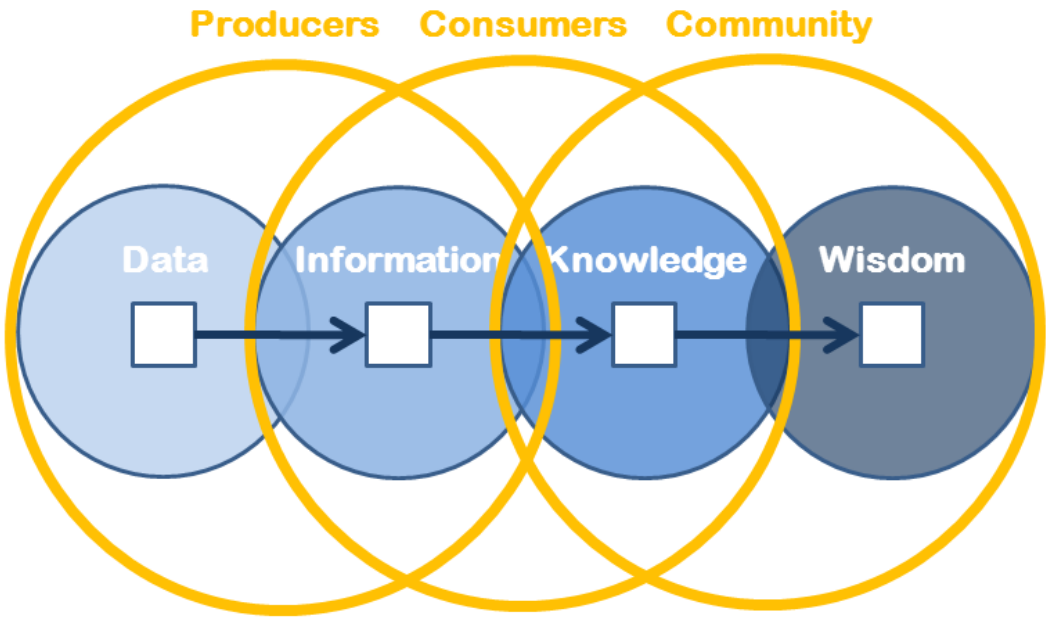
HDF5 file format

Hierarchical Data Format

- ***Container format***
  organizes complex data with metadata.
- ***Scalable and portable***
  no limit on number or size of data objects
  flexible hardware requirements
- ***High-level API***
  numerous interfaces (MatLab, Python, R, …)
- ***Powerful tools and applications***
  for data management, manipulation, viewing and analysis

Continuum of Understanding

We collect data, but
we work with information

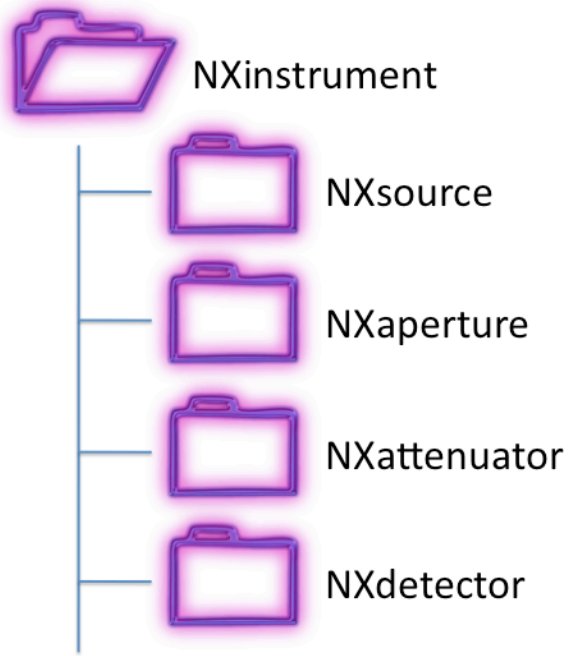


Harlan Cleveland
(Information as a Resource, 1982)

EIGER metadata – NeXus format

NeXus

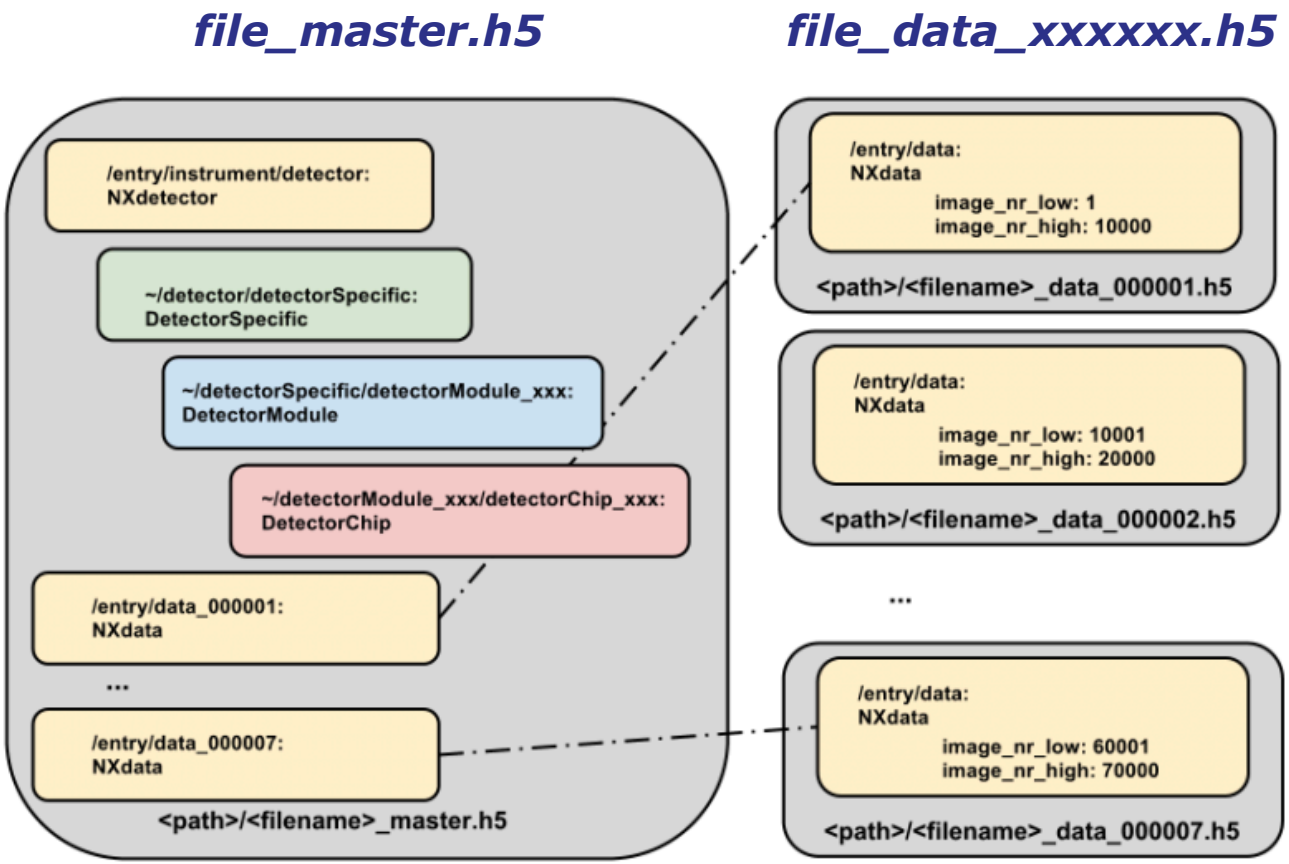
- *common data format for neutron, x-ray and muon science*
- *uses HDF5 as container*
- *Hierarchical representation of experimental and instrumentation metadata*



NXmx

- *Application definition (H. Bernstein, J. Sloan, G. Winter, T. Richter)*
- *CIF can be mapped to NeXus (minicbf2nexus)*

Organization of EIGER data



Organization of EIGER data

```
ich@AirBook:~/Data /bin/ls -lhr *.h5
-rw-rw-r--  1 andreas.foerster  staff   349M May  1 10:42 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_master.h5
-rw-rw-r--  1 andreas.foerster  staff   7.7G May  1 10:42 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000004.h5
-rw-rw-r--  1 andreas.foerster  staff    15G May  1 10:39 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000003.h5
-rw-rw-r--  1 andreas.foerster  staff    13G May  1 10:33 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000002.h5
-rw-rw-r--  1 andreas.foerster  staff    15G May  1 10:28 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000001.h5
-rwx------  1 andreas.foerster  staff    87M May 20 17:40 lyso9_master.h5
-rwx------  1 andreas.foerster  staff   5.0G Apr 21 18:37 lyso9_data_000001.h5
```

Organization of EIGER data

```
ich@AirBook:~/Data /bin/ls -lhr *.h5
-rw-rw-r--  1 andreas.foerster  staff   349M May  1 10:42 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_master.h5
-rw-rw-r--  1 andreas.foerster  staff   7.7G May  1 10:42 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000004.h5
-rw-rw-r--  1 andreas.foerster  staff    15G May  1 10:39 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000003.h5
-rw-rw-r--  1 andreas.foerster  staff    13G May  1 10:33 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000002.h5
-rw-rw-r--  1 andreas.foerster  staff    15G May  1 10:28 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000001.h5
-rwx------  1 andreas.foerster  staff    87M May 20 17:40 lyso9_master.h5
-rwx------  1 andreas.foerster  staff   5.0G Apr 21 18:37 lyso9_data_000001.h5
```

EIGER 4M dataset, 5 GB

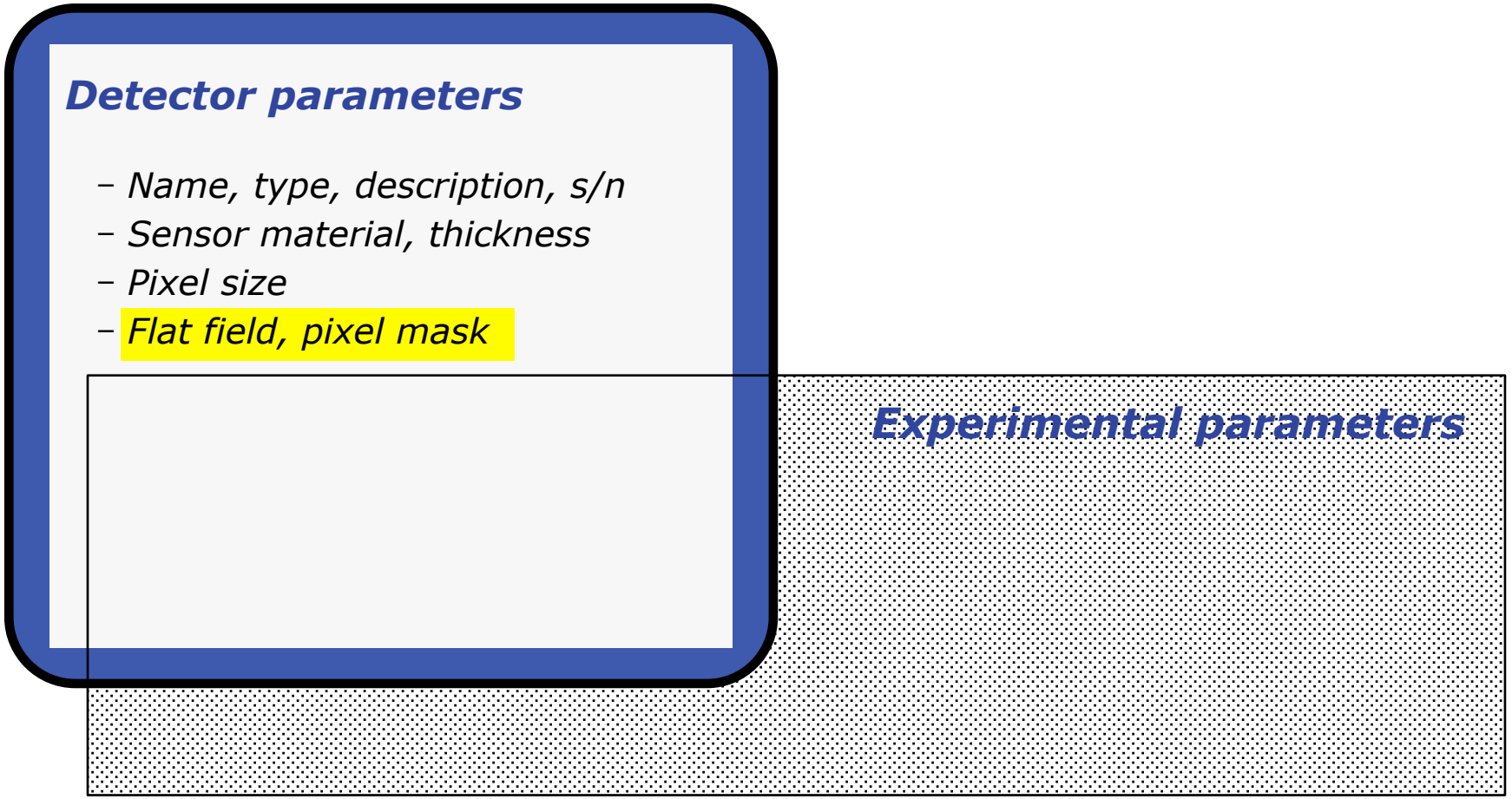
Organization of EIGER data

EIGER 16M dataset, 52 GB

```
ich@AirBook:~/Data /bin/ls -lhr *.h5
-rw-rw-r--  1 andreas.foerster  staff   349M May  1 10:42 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_master.h5
-rw-rw-r--  1 andreas.foerster  staff   7.7G May  1 10:42 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000004.h5
-rw-rw-r--  1 andreas.foerster  staff    15G May  1 10:39 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000003.h5
-rw-rw-r--  1 andreas.foerster  staff    13G May  1 10:33 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000002.h5
-rw-rw-r--  1 andreas.foerster  staff    15G May  1 10:28 lysoHG4_t0p01_0p1d_0p1s_d150_360deg_data_000001.h5
-rwx------  1 andreas.foerster  staff    87M May 20 17:40 lyso9_master.h5
-rwx------  1 andreas.foerster  staff   5.0G Apr 21 18:37 lyso9_data_000001.h5
```

What's up with 350 MB of metadata?

Types of EIGER metadata



Types of EIGER metadata

Detector parameters

- *Name, type, description, s/n*
- *Sensor material, thickness*
- *Pixel size*
- *Flat field, pixel mask*

Essential

- *Photon energy, threshold*
- *Frame time*

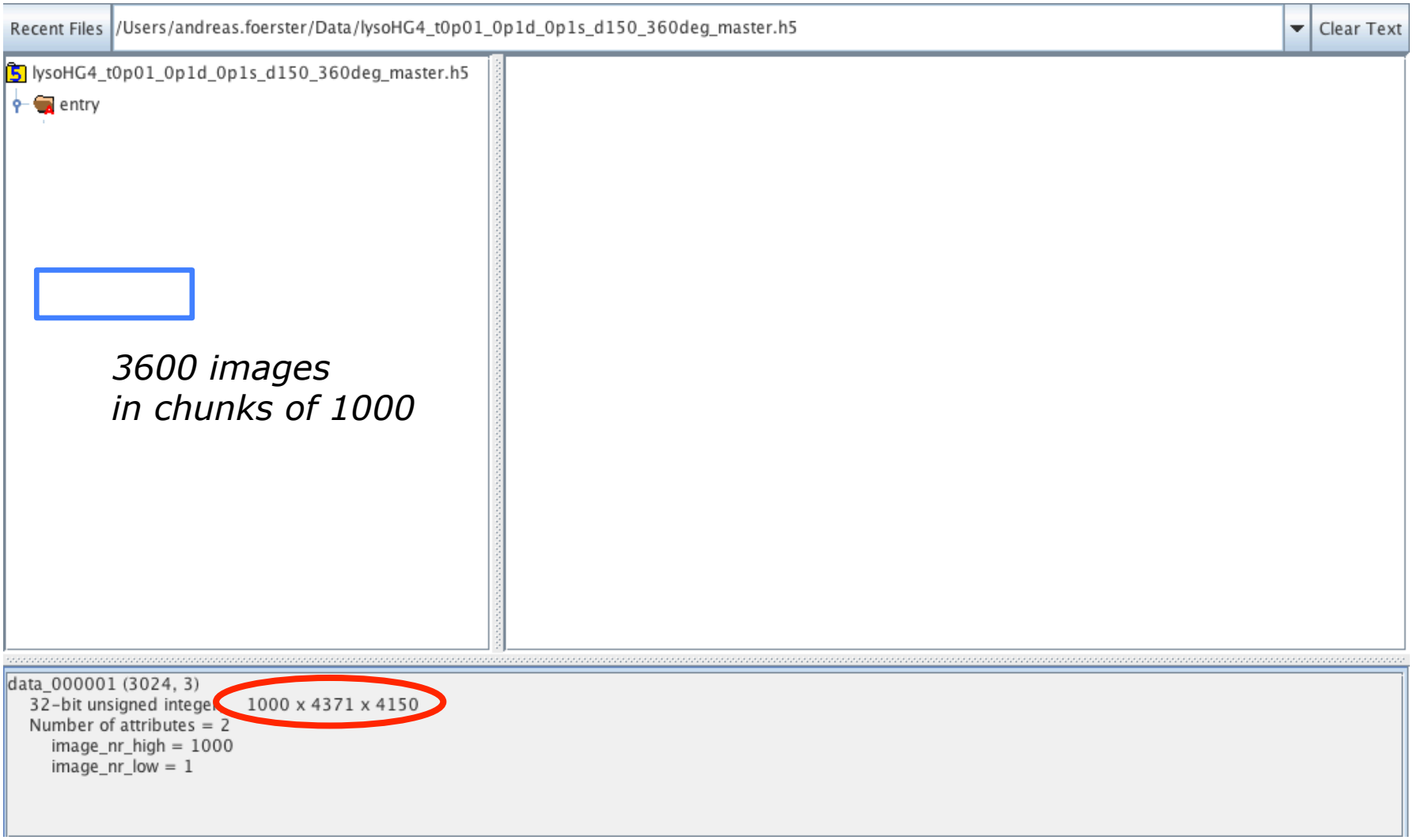
Experimental parameters

Optional

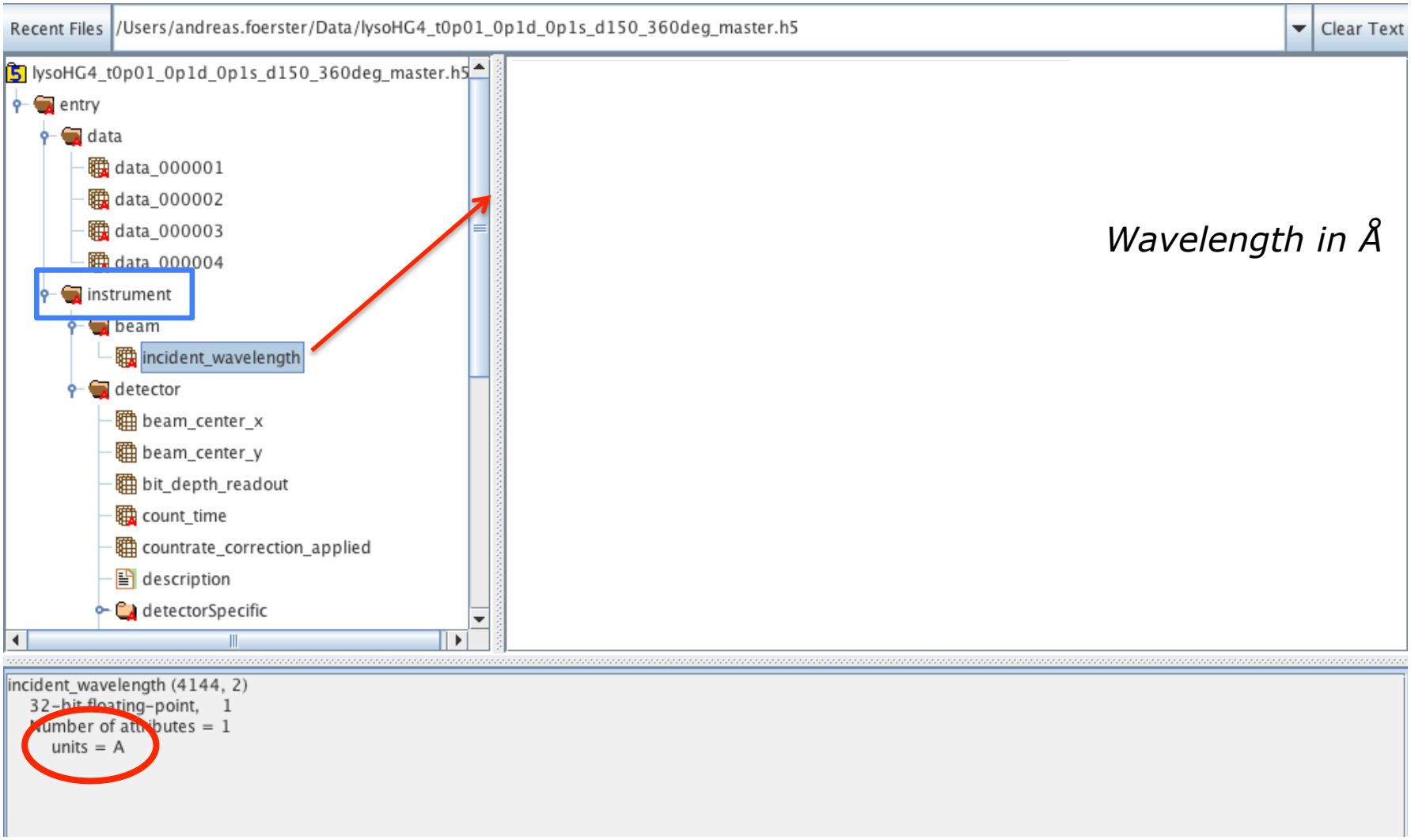
- *Detector distance*
- *Beam center*
- *Oscillation angle*

Experimental values are set (via API), not measured!

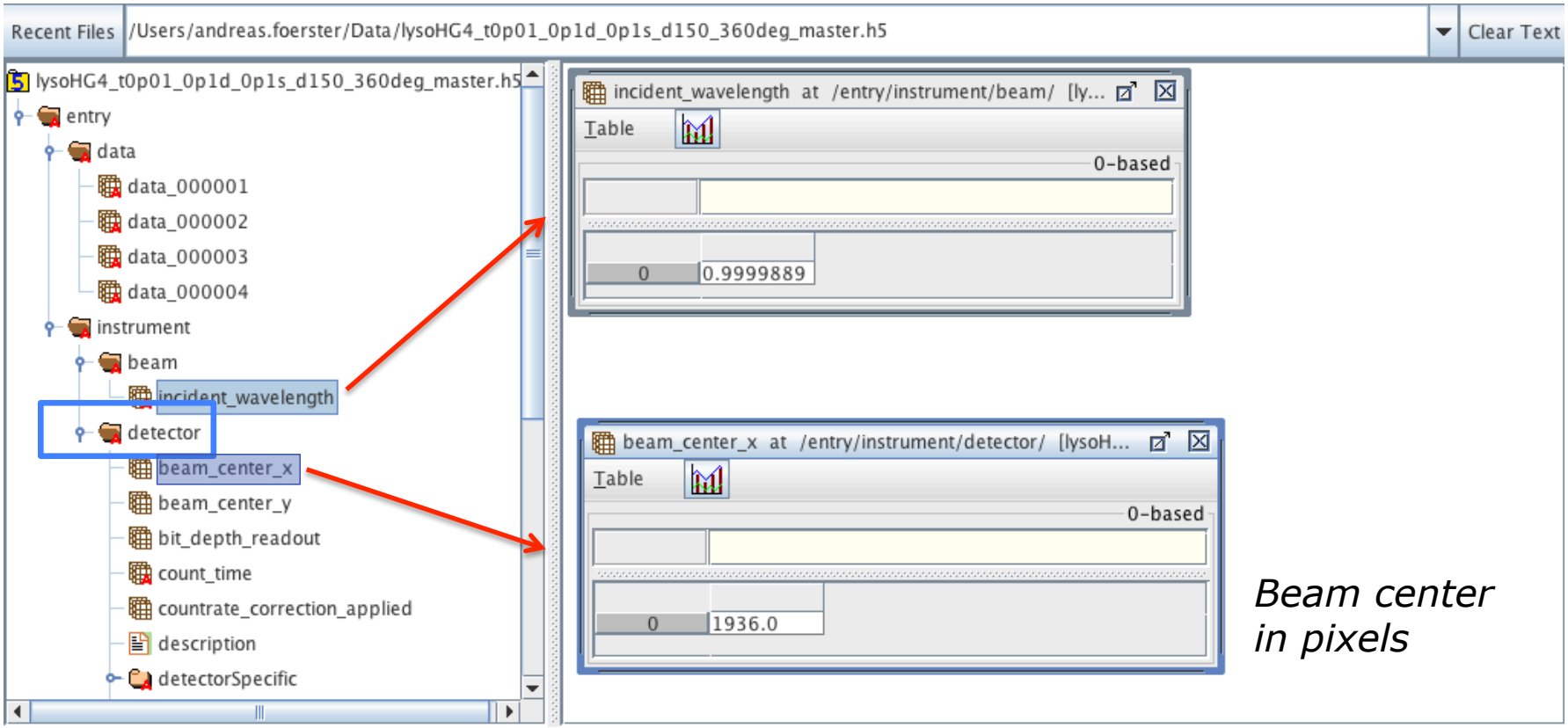
Display metadata with HDFview



Display metadata with HDFview



Display metadata with HDFview

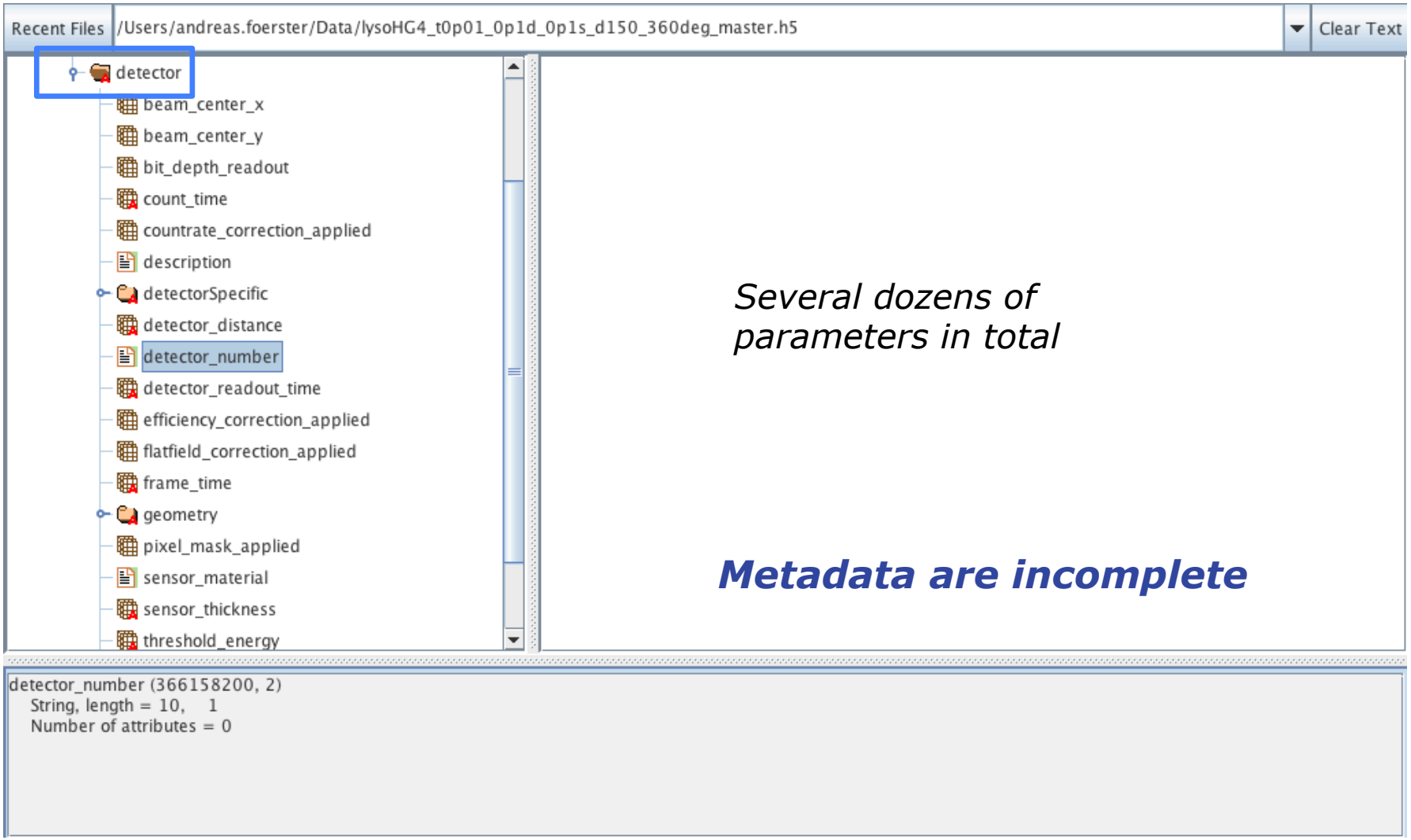



```
incident_wavelength (4144, 2)
   32-bit floating-point,   1
   Number of attributes = 1
      units = A
```

```
h5ls -d master.h5/entry/instrument/detector/beam_center_x
```

*… or use Python API **h5py***

Display metadata with HDFview



Several dozens of parameters in total

Metadata are incomplete

Metadata are incomplete

First law of archiving

- *Some unrecorded details will always become critical later.*

Metadata limitations

- *EIGER metadata are minimal*
  - *Detector-related and (some) experimental values*
  - *Sample and (most) experimental values are missing*

Open and free format allows for extension

- *Standards should be followed.*
- *Automatic processing will be facilitated.*

Extension of metadata

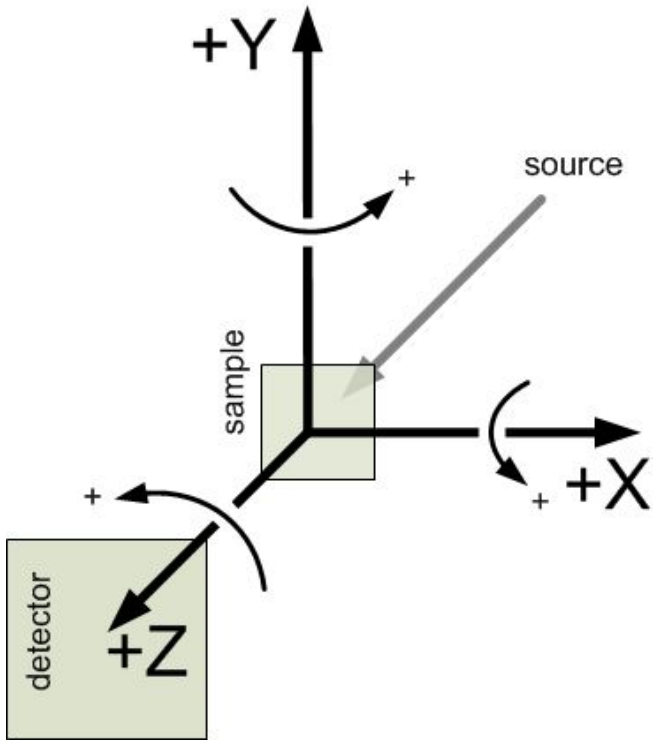
Detector user

- *Sample-related metadata*
  - *Irrelevant for processing*
- *Processed data*

Beamline/OEM engineers

- *Add essential metadata not communicated through detector API*
- *Description of "universal" beamline geometry (Herbert's talk later)*
- *Critical for (automatic) processing*
- *Anything else, but: communicate and share!*

Do not modify existing metadata!



Protection of data and metadata

Avoid manipulation and data loss

- *Checksum to verify entirety of metadata?*
- *Set of checksums to verify identity of associated data files?*
- *Set of checksums to verify integrity of associated data files?*
- *Write protect metadata written at beamline?*

Mechanism inside HDF5?

Mechanism agreed on by NeXus?

Insertion of metadata

Python example – h5py package

```
# open HDF5 file according to input
h5file = h5py.File(input_file)

# create new NeXus entry
sample = h5file.require_group("/entry/sample")
sample.attrs['NX_class'] = ['NXsample']
goniometer = h5file.require_group("/entry/sample/goniometer")
goniometer.attrs['NX_class'] = ['NXtransformations']

# get rotation increment from input
increment = float(sys.argv[2])

# add field and value for rotation increment
omega_range_average = goniometer.require_dataset('omega_range_average', shape=(), dtype="f4")
omega_range_average[()] = increment
omega_range_average.attrs['units'] = 'degrees'

h5file.close()
```

Insertion of metadata

c++ example – DECTRIS packages

```
// set omega_range_average for file passed as first command line argument

int main(int argc, char ** argv) {
  assert(argc >= 2);

  double omegaAvg = 0.1;

  dectris::hdf5::DH5File h5File(argv[1], "a");

  dectris::hdf5::DH5Group omega = h5File.addGroup("/entry/sample/goniometer",true);
  if(omega.attributes().count("NX_class") == 0) {
    omega.addAttribute("NX_class", dectris::hdf5::CHAR, sizeof("NXtransformations")-1, \
    std::vector<ssize_t>()).write((const void*)"NXtransformations");
  }

  if(omega.datasets().count("omega_range_average") == 0) {
    // scalar attribute
    dectris::hdf5::DH5Data h5Data = omega.addDataset("omega_range_average", \
    dectris::hdf5::FLOAT,sizeof(float),std::vector<ssize_t>());
    float omegaAvgFloat = omegaAvg;
    // write expects reference to float value
    h5Data.write(omegaAvgFloat);
  } else {
    std::cerr << "omega_range_average already exists" << std::endl;
  }
}
```

Conclusion

EIGER data written in HDF5

- *Master file and data files*

EIGER metadata conform to NeXus

- *Detector and some experimental metadata*
- *Automated processing requires additional metadata*

Beamline metadata are required

- *Come back after lunch!*

DECTRIS®

detecting the future

Thank you for your attention!

www.dectris.com

andreas.foerster@dectris.com

DECTRIS Ltd.
5400 Baden
Switzerland
www.dectris.com