DRAFT Minutes of the EDNA Developers' Meeting held at the DLS March 3rd - 4th 2009

Agenda:

Tuesday March 3rd :

9.30	General EDNA presentation to Diamond Science (Olof)
10.30	DLS developers EDNA training (with a "working" lunch)
14.00 – 18:00	EDNA developers' meeting:

- GUIs for EDNA: ccp4i, GDA, mxCuBE...
- Prioritisation between potential developments for MXV1:
    - Parallel indexing: how to rank different results
    - "Strategy check" using pointless
    - Post refinement of cell using MOSFLM
    - Parallel integration using MOSFLM
    - Scaling using SCALA
- Publications (not discussed due to lack of time)
- Licence for MXV1 - GPL v3?

18:30 -> Meeting dinner

Wednesday March 4th :

9:00 – 12:00 Continued EDNA developers' meeting:

- MXV2 generic data model
- XDS plugin
- Calibration requirements and procedures

Tuesday morning -> 14:00

This time was not a formal part of the EDNA developers' meeting, the idea was to use it as a general introduction of EDNA to Diamond Science and then a detailed introduction to the EDNA framework for new developers at Diamond. The following two feedback were made which is of interest for the EDNA framework:

- Paul Gibbons asked how we could be sure that the EDNA framework can be used for non MX applications when we don't yet have any such application developed. He suggested that a “demo” or “test” application is developed at the same level as “MXV1”, “MXV2” and “TOMOV1”. This test application would serve two purposes:
    - It could be used for testing the workflow part of the framework that cannot be tested unless a plugin is instantiated.
    - I could also server as a prototype and/or as a tutorial for developing other (scientific) applications based on EDNA.
- Graeme Winter pointed out when we browsed through the EDPlugin base classes that it's very easy to accidentally override a method of a super class by giving it a name that already exists. Since Python does not implement the “final” Java method which protects against accidental overriding of methods, Graeme suggested to use a name space coding convention for “final” methods.

Tuesday afternoon session: Developers' meeting part I

Presents: Alun Ashton, Mark Basham, Gleb Bourenkov, Sandor Brockhauser, Gerard Bricogne, Marie-Françoise Incardona, Peter Keller, Andrew Leslie, Karl Levik, Harry Powell, Olof Svensson and Johan Unge

Prioritisation between potential developments for MXV1

The result of discussions concerning new developments for MXV1 can be summarised as follows:

- We agreed that the inclusion of data processing plugins for MXV1 is too optimistic, because the design of the work-flow of these plugins (see next Section) needs more time than available before the planned release in May. This release will therefore be based on the existing prototype with its ccp4i GUI. A part from preparing the release (migration of code to the new project structure, improvements and bug fixes) time will also be used for advancing on the processing side and on MXV2.
- Marie-Françoise has started to work on the SCALA plugin.
- Karl has agreed to write a second ISPyB plugin which give an image filename and directory will be able to retrieve the corresponding “dataCollectionId” needed for storing the results of the characterisation.
- Johan has agreed to start working on the Pointless plugin.
- Harry will work on the workflow / use case for parallel integration using MOSFLM.
- Olof will continue to work on the parallel indexing. The ranking of indexing results is still not well defined and will be the topic of future VCs / meetings.

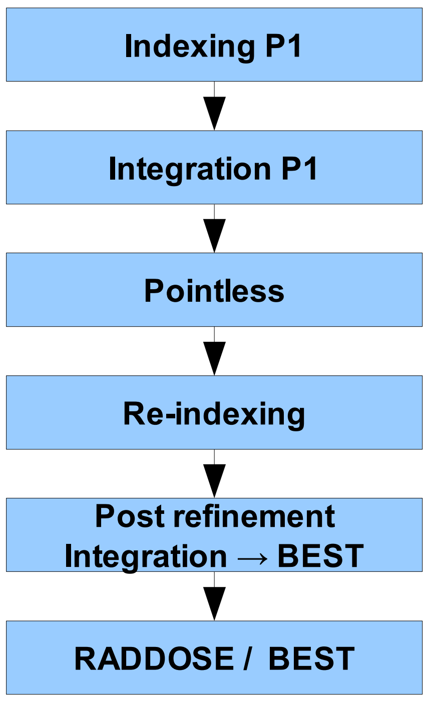
New characterisation work-flow

The main reason for not including any further processing in MXV1 is that we agreed that the current work-flow for characterisation as implemented in the prototype (indexing → integration → BEST strategy, see e.g. the prototype report) is not adequate for accurate determination of the point-group symmetry.

The correct determination of the point-group symmetry is, in the absence of taking into account radiation damage, not of too high importance as it in general results in collecting more data than necessary and hence increases the multiplicity. However, for the absorption calculations performed by RADDOSE, it is of very high importance to know the correct number of molecules in the asymmetric unit, otherwise the calculated absorbed dose rate can be erroneous and hence the BEST data collection strategy not adequate at all to the sample in question.

The CCP4 program **Pointless** is designed to give a probability and confidence values for a range of different point-groups given a data collection. Therefore the inclusion of **Pointless** in the work-flow is of great importance. However, the inclusion of **Pointless** in the work-flow is in it self not sufficient for obtaining accurate determination of the point-group if only a couple of reference images are collected. In general, in order to obtain a high probability and confidence from **Pointless**, one or two sub-wedges with 2-3 degrees of data are necessary.

We therefore agreed to start to work on a characterisation plugin which would implement the following work-flow:



The idea is that if sufficient crystallography data is available by providing two sub-wedges with each 2°-3° of data, this plugin also runs the MOSFLM post-refinement. The refined cell can then be used immediately for integration of images as they are collected during the data collection.

GUI considerations for the EDNA MXV1 / MXV2 projects

The ccp4i GUI for the prototype developed by Gleb and Marie-Françoise has been proven to work very well, however it is not meant to be a substitute for integration of EDNA in BCM GUIs like **mxCuBE** or **GDA**. The major drawback with the ccp4i GUI is the lack of communication back to the BCM for the data collection strategy parameters. However, in the discussions it became rapidly clear that the developers' situation both on the **GDA** and **mxCuBE** sides does not allow for any rapid implementation of an EDNA GUI like the ccp4i.

More, the GUI / EDNA situation is complicated by the fact that we will need to be able to make feedbacks to the GUI, for example for prompting the user to go ahead with a particular choice of a point-group during characterisation. Mark suggested to solve this problem by introducing an optional call-back mechanism that the plugins could use for prompting a user. If no interface is available or if the interface doesn't implement any call-backs, the plugin would go ahead with a default choice.

Here's a non-exhaustive list of other items of interest for the GUI that were discussed – no real conclusions were drawn from these discussions:

- Alun pointed out that the executive summary after a characterisation is quite verbose, and that he would like to have a very short summary of the data collection strategy proposed by the EDNA prototype. Gleb said that there is a “Show summary” button available in the ccp4i “results” widget which allows to reduce the log text, but we would have to introduce HTML tags into the executive summary to make this work. Olof suggested that this could be taken care of by the EDNA ccp4i plugin.

- Sandor suggested that the BCM GUI could implement several characterisation strategies, for example linked to different buttons:
    - A low-dose single image characterisation could be useful especially in the case when the space group is well know.
    - A “traditional” characterisation using two images 90° apart.
    - A two-sub-wedge characterisation, were each sub-wedge would contain 2°-3° of data.
- At the DLS, the prototype is run automatically for each data collection but the results are not used by the BCM (they are kept for the users to analyse them). One possibility could be to run the EDNA characterisation automatically for each characterisation-type data collection, e.g. less than four images. The results of the EDNA characterisation could then be used to inform the user before he starts the data collection with his own parameters what he might risk in terms of statistics (completeness, resolution etc) and radiation damage.
- Another suggestion that was put forward was to implement a “simulate data collection” button in the BCM GUI. (It was also suggested that, since in general data collection parameters are provided by often inexperienced user's, such a functionality could be called WORST since the strategy calculated by ENDA is supposed to be the BEST... ☺)
- Finally, a rapid solution for implementing a GUI in the existing BCMs such as **GDA** and **mxCuBE** would be to simply link the execution of the existing ccp4i GUI to a button, with the possibility of automatically filling in the relevant fields in the ccp4i GUI. [Olof suggested this solution in an ATF (Automation Task Force) meeting at the ESRF but there was a general agreement that this is not a good solution for the ESRF.]

Licence

We discussed briefly the licence for the forthcoming release of MXV1. Olof suggested to use GPL v3 which is the current licence for the EDNA prototype. Alun would have preferred to use LGPL, as this licence is less restrictive in case of redistribution of the EDNA sources.

Olof asked if the DLS is envisaging to redistribute EDNA, and Alun answered that this could be the case if GDA becomes highly dependent on EDNA functionalities. Peter pointed out that if this should ever be a problem, the EDNA collaboration can always provide a separate version of EDNA with a different licence that would allow EDNA to be distributed with GDA under a different licence but would not allow further redistribution.

Finally, we agreed that if we want to use the latest version of AALib, then we have to use GPL v3 since the licence of AALib has changed to be GPL v3, and we need to distribute AALib with EDNA.

Wednesday morning session: Developers' meeting part II – Towards MXV2 characterisation

Presents: Alun Ashton, Gleb Bourenkov, Sandor Brockhauser, Gerard Bricogne, Marie-Françoise Incardona, Peter Keller, Andrew Leslie, Karl Levik, Harry Powell, Olof Svensson Johan Unge, Graeme Winter

Data Processing Plugins (XDS and MOSFLM)

As a result of previous discussions by Olof, Pierre, Sandor and Gleb at a small workshop (September 22-23, 2008), a list of suggested XDS plugins has been presented, discussed and finalized. The general approach is to write a set of plugins implementing a set of defined crystallographic functions that are necessary and sufficient for creating complex workflows (e.g. new characterisation workflow discussed yesterday), The set crystallographic functions includes:

- indexing
- choice of indexing solution (using either of apparent lattice symmetry, pre-defined symmetry, or integrated intensities)
- (post)refinement
- integration
- scaling

In a further discussion, similarities and differences to MOSFLM plugins were identified, and described in a list bellow in a common form.

The Input and Result of these plugins (functions) should be defined in terms of Generic Data Model (in a list bellow the terminology of a current Characterisation data model is used wherever possible). This is opposed to following a preferred workflow of a particular processing program.

Appropriate statistics output will be defined in exactly the same way for equivalent the functions using MOSFLM or XDS.

Spot Search plugin – MOSFLM and XDS

INPUT: SubWedge

RESULT: Spots (list) : detector coordinates, oscillation axis centroid, I/Sigma

{SubWedge, ListOfSpots, Stats(…)}

XDS implements 3D spot search, whereas MOSFLM 2D. For the cases when MOSFLM will be used for spot search on broader subWedges, a post-processor that adds up partials is needed.

Indexing – MOSFLM and XDS

INPUT: spots collected on a consistent of set of subWedges

RESULT IndexingResult

per DEFINITION: consistent (or indexable) are subWedges collected at identical experimentalCondition apart for the “rotationStart” and “oscillationWidth”

In case of XDS, N solution with a penalty bellow threshold (as in MOSFLM) selected, and refined in parallel (one XDS run per solution).

In theory, it is also possible to implement fully generic Indexing without the above “consistency” limitation (this may be a low priority task, for future).

Indexing Solution refinement - XDS (high priority) and MOSFLM (low priority)

INPUT: spots of a consistent set of subWedges; indexing selectedSolution

RESULT: Experimental Condition Refined, Selected indexing

Solution (refined)

This plugin is necessary in case of XDS in order to implement the lattice-symmetry based automatic selection of Indexing Solution as in MOSFLM. It is possible to implement this as a separate function in MOSFLM as well, and will be done later for consistency.

Post-refinement – XDS and MOSFLM

INPUT: images a consistent set of subWedges in + selectedSolution

RESULT: update experimental condition, selected solution refined;

Integration – XDS and MOSFLM

INPUT: images a consistent set of subWedges in + selectedSolution

RESULT: update experimental condition, selected solution refined; Intensity data.

Integtration/postrefinement philosophies are very different in XDS (first integration, then literal postrefinement) and MOSFLM (integration is preformed using fixed postrefined cell). This implementation would permit to combine both approaches transparently.

Generic Penalty

INPUT: generic IndexingResult

OUTPUT: Distortion Index for each solution added to Indexing Results

The generic penalty is used for (A) selecting the indexing solution and (B) a quality indicator of indexing solution

Alternative penalties were discussed :

1. Distorsion Index (DI), in %.
   International Tables for Crystallography (2006). Vol. F, Chapter 11.4, pp. 226–235, used in DENZO
   B – reduced reciprocal cell orthogonalization matrix
   B’ – idealized reduced reciprocal cell orthogonalization matrix
   DI=Sqrt[Sum[Transpose[Inverse[B].B’]-Inverse[B].B’]^2]
2. “Maximum Angular difference” (MA) in degrees, as implemented in cctbx package by Ralf Groesse-Kunstleve.

Whereas MA may be a more contrast absolute indicator of a quality of the solution, DI appears more appropriate for selecting solution (that is the main purpose).

Re-indexing

INPUT: IndexingResult

OUTPUT: IndexingResult (different chosenSolution)

This plugin will implement transformation of orientation matrix consistent with the unit cell transformations. Needs storing 44 transformations of standard basis as defined in Table 9.2.5.1 of Vol. A IT IUCr (2006).

Presentation by Graeme Winter

Graeme gave a concise and useful presentation on his experiences in implementing cross-package processing (like indexing with LABLEIT -> Integration with XDS, and some others). Graeme is preparing a soon publication on that, therefore asks to keep his results confidential.

During the discussion Gleb raised up concerns about fitting LABLEIT into a general EDNA processing schema as outlined in a previous chapter. The end-user interface of LABELIT as used in EDNA prototype and other packages is inflexible and relies on the information from the image headers only, that may not be sufficient for a proper definition of Experimental Condition. Therefore, *e.g.*, the indexing solution may become simply uninterpretable in a Generic Data Model context. More elaborate implementations of LABLEIT would require for MXV2.

Calibration requirements and procedures

The geometrical parameters describing the experimental setup and its state during particular data collection must be provided (typically by a BCM) to EDNA together with the diffraction images. Calibration is a procedure of determining these parameters accurately. There is a variety of calibration methods achieving very high accuracy; these methods can often very different for different sites and instruments. Most of them are not in the scope of EDNA. Also, the problem appears rather in keeping the instrument calibrated under user operation load on a long term. Sandor pointed out to the observations of up to 0.7 degrees variations in axes alignment of MiniKappa between the successive calibrations.

Typical beamline re-alignment procedures (a-la ESRF “Quick-realign”) may induce variation in the beam direction (that is a principle direction in EDNA coordinate frame) of several hundredth of a degree. This would not create convergence problems in a data processing. But, as pointed out by Gerard, this would affect partiality and thus data quality in an inverse-beam experiment for crystals with low mosaicity.

Sandor and Gleb suggested to implement a calibration procedure that is based on the protein crystal diffraction images. It would use the set of data collected on purpose in a specified way (e.g., several specially selected goniostat settings) and the results of the processing plugins (as specified above) plus special post-processors that will derive the necessary information from the results of standard processing plugins. Similar procedures are already implemented by Sandor in STAC. The sample and data requirements will be specified on the documentation level. E.g., calibration using Lysozyme crystal at room temperature may provide angular accuracies within thousands of a degree (or will be limited by the beam divergence and mechanical resolution of the goniostat), typical Trypsin crystal at 100K - to a few hundredths of a degree. The use of the “real” experimental samples for calibration should be discouraged (documentation).

In a regular data processing, significant shifts of the refined parameters should be reported to the user (similar as the beam position shifts are treated in the prototype, but for all refined parameters). The Interpretation of the shifts (outside EDNA) will depend on many factors. In general, it should be noted that the accuracy in parameters attainable in a dedicated calibration experiment is much higher compared to the accuracy of parameters refined using a “real” data.

There is a variation in tolerances of different methods. E.g. XDS can be used with arbitrary spindle orientation. MOSFLM currently requires orthogonality to the beam. Hence, only XDS will be used for PHI scans (at SOLEIL or MAXLAB), or in cases when OMEGA axis is appear not orthogonal (misaligned). In fact, all currently known OMEGAs at ESRF and DIAMOND are aligned sufficiently accurate for MOSFLM processing. Exact value of tolerance will have to be derived, the data can be provided by several beamlines.

Data Model

We discussed the priorities of actions required to implement MXV2. As the plugins are defined and implementations, including MOSFLM- and XDS-specific data models seem rather obvious, the Generic Data Model is a current bottleneck.

The generic Experimental Condition and Indexing Result must be modelled in terms of the basic linear algebra objects (vectors and matrices) of a Common data model, that in turn depend on a choice of an approach to doing linear algebra in EDNA.

Several public domain linear algebra packages have been considered. The general conclusion is that all the solutions are fairly bulky, and “not a native python”; hence, a potential compatibility problem for future. Given a rather compact scope of the problem, we decided to look into implementation of a necessary set of 3D-space liner algebra functions from scratch. There is no necessity in either 4D (rotation-translation) objects and functions in a scope of MXV2 to this point.

We agreed on the following order of steps:

1. Define and implement a (simple) set of 3x3 linear algebra functions.
2. Implement basic objects in a Common data model as required by (1)
3. Implement ExperimentalCondition using (2) and previously defined EDNA generic coordinate system.
   The term ExperimentalCondition may be misleading and needs to be renamed.
   The suggestion was “Instrumenty Configuration”, needs further thought.
4. Proceed with implementations of translations from of INPUT(s)/RESULT(s) from Generic to Specific models, and plugin implementation.

This work will proceed via Friday’s video conferences.