Grenoble, Aug 29th, 2012

From: E. Farhi, ILL
to: FP7/NMI3-II WP6 members and everybody interested by the topic

Data Analysis Standards (NMI3-II WP6)

1st Meeting Minutes (July 5-6th 2012, ILL)

Present:

- Emmanuel FARHI, ILL, France [farhi@ill.fr] +33 47620 7135
- Ricardo Leal, ILL [leal@ill.fr]
- Stig Skelboe, DMSC, Copenhagen University, Denmark [skelboe@nbi.dk]
- Peter Willendrup, DTU, Copenhagen, Denmark [pkwi@fysik.dtu.dk]
- Jon Taylor, ISIS, UK [jon.taylor@stfc.ac.uk]
- Joachim Wuttke, JCNS/FRM-II, Germany [j.wuttke@fz-juelich.de]
- Thomas Gutberlet HZB, Germany [thomas.gutberlet@helmholtz-berlin.de]
- Miriam Forster, ILL [forster@ill.fr]
- *Joachim Kohlbrecher, PSI [joachim.kohlbrecher@psi.ch] (excused)*
- *Mark Johnson, ILL [johnson@ill.fr] (excused)*
- *Laurent Chapon, ILL [chapon@ill.fr] (excused)*
- *Sylvain Petit, LLB, Saclay, France [sylvain.petit@cea.fr] (excused)*
- *Martin Mueller, HZG, Germany [Martin.Mueller@hzg.de] (excused)*

Introduction – presentations to start the discussion
The meeting has started with 3 talks from Emmanuel Farhi, Ricardo Leal and Jon Taylor. Slides are available at <http://nmi3.eu/about-nmi3/other-collaborations/data-analysis-standards/meetings.html> .

Emmanuel has reminded the frame of the workshop, especially in the current software landscape, as well as the tasks of the work-package which should deliver reports and prototype(s). This initial workshop has mainly dealt with Tasks 1 and 2 which should be produced within 4 months from the official start of the project on July 5th, that is for November 5th 2012. The main message is that there are currently many software available, some are alive and some dead. They all represent a significant work, and certainly contain a lot of knowledge, which should not be disregarded. According to Emmanuel, the importance of a given software can be measured in its community of

users (or lack of). Also, in order not to loose the knowledge, it is important to ensure the availability of codes, even when outdated, as well as extensive documentation about the science and the implementation. It has been discussed the importance of the code granularity, that is basically the number of objects/modules which compose the software. Many small modules are meant to be re-used, but may lead to additional complexity in deployment, whereas larger modules (or even a single monolithic program) are easier to deploy, but are not easy to re-use. The number of lines of code (LOC) should be minimized to lower the software complexity, and bugs. Statistically, a programmer writes 12 useful lines/day, and can maintain up to 50-100 kLOC. As the WP6 project has limited resources, we are bound to use a high level language in order to minimize the number of LOC. The final prototype code should then not exceed 50 kLOC, minimize complexity and dependencies.

Ricardo has presented an overview of some selected data analysis/reduction packages, based on a functional and code/dependency analysis, and an initial list of software from the ILL/NMI3 LiveDVD <http://nmi3.eu/about-nmi3/other-collaborations/data-analysis-standards.html>.

Jon has presented the Mantid framework. One of the most promising features of Mantid seems to reside in the VATES part to handle large inelastic data sets in the style of Horace. It was also explained that even though it had been envisaged initially, a Mantid-Matlab interface has been discontinued, arguing that a proper Python/C++ coding of any legacy functionality is preferable to a hack. The Mantid infrastructure currently reaches 1 MLOC (active, removing comments), plus about 500 kLOC for 3rd party software, and about 20 active developers (400+ commits per month). New developments and requests are submitted by scientists to Mantid through tickets, which are then reviewed and ranked by a committee, and then distributed to developers.

It was agreed for this meeting to mainly focus on Tasks 1-2, and consider Task 3.

Task 1 : Review existing data analysis software and practices of software developers (mid-September 2012)

A list of existing software has already been set, and it is enriched with knowledge from practices on ILL instruments. It appears that there is no definitive match between classes of instruments and software, as for similar tasks, many possibilities exist. This is simply the outcome of code duplication in many equivalent software projects. However, most instruments hopefully have a 'working' procedure to treat and analyse data sets. Exceptions arise for e.g. reflectometry where, to our knowledge, unsatisfactory data-analysis software solutions exist for off-specular reflection, spin-echo reflectometry (SERGIS) and GISAN.

Concerning practices, legacy software were often developed by a single author, leading to single points of failure when maintenance ends. Even worse, some old software can not be obtained any more as the links to source code are discontinued.

More recent software (e.g. Mantid, SANSView, FullProf, McStas, iFit, …) tend to make use of development repositories (CVS, SVN, Git, Mercurial) to keep track of code commits and history. This also has the major benefit to gather the project files in a steady location (as long as the repository server runs...). Recent projects tend to prefer Python as interpreted language, and C++ for compiled code. However, there are still active projects in Igor, IDL and Matlab, especially when the developer team is limited.

It was pointed out that Mantid has a similar framework infrastructure as LAMP, that is load

'workspaces', apply scripts/methods/algorithms, with viewing and fitting capabilities. Similarly, the Matlab/iFit (replacement from Mfit) also uses this framework. A data analysis standard may then inspire from this.
In order to extract the relevant information from a variety of software in terms of functionality, an analysis of atomistic process steps should be carried out, per software, per class or instrument. This task can be eased by e.g. the SIMcards in LAMP, which already classify reduction steps per type of data set.

Actions: a report on current software and developer practices is expected by mid-September, written by R. Leal and E. Farhi. It will be submitted to the workpackage members for approval before publication.

Task 2 : Review existing solutions for a common data analysis infrastructure (mid November 2012)
The **Task 1** has already considered a set of software used in neutron facilities. As discussed previously, it appears essential today to provide the following services for modern software projects:

- Dedicated web site with easy to read documents to present the software scope.
- Extended documentation.
- Example data files and tutorials demonstrating their use.
- Mailing list to create a community of users.
- Package installers for common operating systems, and access to source code. Must be straight-forward.
- SVN/CVS/Git/Mercurial repository: the http://forge.ill.fr provides such a service, for e.g. nMoldyn, CrysFML/FullProf, Fable, EDNA, PyMCA, Nomad, ...
- Unit testing: modern software must include a unit testing mechanism, as well as more advanced scientific tests. This requires to provide example data sets for testing.
- Trac/ticket submission: the Trac system allow to register tickets for bug reports and suggestions from users. the http://forge.ill.fr provides such a service.
- Coding conventions: programming style, practices and methods.

However, we may also define some infrastructure to span across a set of software projects (family):

- Common look and feel for interfaces (common logo, layout for interfaces, messages, terminology).
- A LiveDVD for testing and fast deployment (e.g. for tutorials). Such a DVD already exists with a selection of existing software.

Once the development infrastructure is in place, we may focus on the common data analysis functionalities, which define how should be a common data analysis software like:

- supported data formats (load)
- type of displays (plot of the data, the instrument geometry, ...)
- algorithms/methods categorized in generic ones, per type of data set, … (operators)
- supported output formats (save)
- type of fitting routines/optimizers (fit)
- common models used for fitting to data sets (models)
- scripting capability and interfaces
- level of customization (for advanced users)

- work flow from raw data to extracted physical parameters
- contributions from the community: how to foster and include them in the software

One of the most challenging item is to define algorithms/methods. This task may start from a functional analysis of the existing software, set up a list of simple treatment steps, identify what can be re-used from legacy code and concepts, and propose solutions for integration in a modern software infrastructure.

In order to categorize the algorithms and the knowledge work flow, we may inspire from large data set projects which relate the data content to the methodology for treating the data. An example of such data infrastructure can be found in the biology area with the databases interconnecting the gene sequences, their representations, publications, signification, and treatment. In short, any biological data set (sequence, structure, …) must be submitted to data bases prior to being published in scientific journals. Each data base entry, given a unique ID, must contain mandatory information such as references, authors and acquisition conditions, procedure to annotate (interpret) the data. Biology computer systems in fact relate dozens of interconnected data bases, with search engines in the style of Google. Examples of such databases can be found at *Entrez Sequence* <http://www.ncbi.nlm.nih.gov/sites/gquery> (type in 'myoglobin'), and NCBI *MapView* <http://www.ncbi.nlm.nih.gov/projects/mapview/> (genome browser).
Entries can also relate to an ontology, that is a dictionary of common terms to represent biology concepts. All biological data entry contain references to the ontology, in order to standardize the meaning of the data, see for instance *GeneOntology* <http://geneontology.org>. We could envisage, probably beyond the scope of this project, to initiate such an ontology devoted to neutron scattering measurements, with a structure such as :

- data treatment steps (a software/procedure is then an ordered list of such concepts)
    - reduction (instrument corrections)
    - analysis (extraction of physical parameters, that is fitting)
- materials and physical parameters
    - phase
    - structure
    - excitations
- measurements
    - instrument classes
    - measurement configurations

Such an ontology would clarify the conceptual area of measurements and data treatment steps, and allow cross correlation searches in databases (similar measurements treatment procedures, materials, …) and generate dependency trees for neutron scattering measurements and treatments. It is clear that the iCAT project would then be related to such an ontology. The initial definition of the ontology may be inferred from e.g. the SIMcards in LAMP, and conceptual design documents in the Mantid project (URD at <https://github.com/mantidproject/documents/blob/master/Requirements/URD.doc?raw=true> SRD at <https://github.com/mantidproject/documents/blob/master/Requirements/SRD.doc?raw=true>).

Actions: a report on the common data analysis infrastructure is expected by mid-November, written by R. Leal and E. Farhi. It will be submitted to the workpackage members for approval before publication. A Mantid Git branch may be created for us to experiment prototyping.

Task 3 : Develop prototype software in chosen solution for representative applications

The scope of the prototype to write during the workpackage in order to demonstrate Task 2 conclusions was discussed during the meeting. As most facilities taking part in this project are reactor based, it was agreed to focus on data from steady-state neutron scattering instruments. However, as most instruments in this facilities have been operating for many years, the data treatment pipe is usually well defined and functional, using existing software.
As described in the workpackage proposal, the prototype should not implement new functionalities, but rather concentrate on new data analysis methodologies.
Modern instruments tend to make use of white beam trough multiplexed acquisition systems. On reactor based instruments, this may be achieved by mean of a multiple monochromator or analyser. This evolution of classical instrument geometry has been used in multiplexed TAS machines, such as for instance the RITA-2 at PSI, the flat-cone diffractometer at BENSC, the IN8 IMPS and the IN20 flat cone at the ILL. In order to demonstrate how a typical software implementing such feature may look like, the vTAS software was demonstrated <http://www.ill.eu/en/instruments-support/computing-for-science/cs-software/all-software/vtas/>. It was agreed that, in the event of an implementation into Mantid/VATES, the first steps would be to import the data files from e.g. IN8 IMPS or IN20 Flat-Cone, into Mantid and convert it to 4D S(q,w) maps to be processed by VATES. Such an approach minimizes the development. The re-use of legacy code, e.g. vTAS, Restrax, …. must be favoured. The final prototype will gain a lot of credibility if it can be easily inserted/linked into Mantid.

In view to define more accurately the expectations from the TAS instrument users, it was agreed to initially contact TAS instrument responsible staff in reactor based neutron facilities. A list of required functionalities will then be written, to define the prototype functions.

Actions: E. Farhi to contact TAS users/instrument scientists, and define an initial list of functionalities. This list must not implement new functionalities, but only extensions to existing features, on a more complex instrument geometry. R. Leal and E. Farhi to discuss about possible technical implementations.

Next meeting:
As a satellite meeting during the NMI3-I general assembly, final meeting in December 5-6th, 2012 (Garching) <http://nmi3.eu/news-and-media/calendar.html>.