Paper 9.5 CECAM Software Development Workshop for an Electronic Structure Library

Following strong recommendations from the Border of Directors CECAM is proposing to start a line of novel workshops with an extended event on software development for electronic structure in 2014.

Software development workshops: The proposal is to get assemble between 12 and 15 people to work on a topic for a period of up to 6 weeks. Initially, we propose to hold the workshop in Lausanne during the summer of 2014. The workshop would start with a "conventional" meeting (small conference) of duration 3 days with representatives from the community, establishing (or more likely, tuning) the agenda for the following weeks. During these weeks, 12-15 committed people would stay to work for the whole period and achieve agreed goals. The participants in the extended workshop would be young but reasonably experienced programmers in the software of interest. They must be endorsed by their supervisors, but will be financially supported by CECAM.

Software development is a topic in which CECAM is genuinely interested (see Software Development Working Group, and the document on Open Innovation approved by the Council. It is much at the heart of the CECAM community.

Proposed topic for 2014: *Electronic Structure Libraries*

Electronic Structure, both in physics and chemistry (but also of use in Materials Science, Earth Sciences, Bio Sciences and Engineering) is one of the key pillars of the CECAM community. The idea is to kick-start a community-maintained open Electronic Structure Library, as an important tool within the Open Innovation ideas proposed for this community and endorsed by CECAM's council.

(The ideas outlined here are equally applicable to a library on molecular dynamics (solvers, integrators, thermostats, and barostats). A possible consideration is to enlarge the scope and have a common CECAM library, with electronic structure and molecular dynamics).

The ambition is to segregate a layer of software in terms of modules of general purpose, so that new ideas, and new science, can be coded by scientists using the library without needing to rewrite functionalities that are well established in the community, and without needing to know more software engineering than science. That should allow separating the coding needed for cutting edge research from the software infrastructure, which needs maintaining and rewriting at every step of the hardware race.

The proposed delivery mechanism would be a Wikipedia-like site (media-wiki server maintained at CECAM), in which every entry is the description of a subroutine, module, etc. The important part of the entry is the definition of the interface (what goes in, what comes out) and the main operations inside, as essential documentation. Implementations can be then attached, which can be different (different coding language, different license, different algorithmic implementation etc.). A good definition could allow software engineers to write new implementations for new and different platforms whenever timely. Some useful characteristics would include:

- Parallelisation (and thus data decompositions) should be hidden below these modules: many demanding tasks can be heavily abstracted and made common (e.g. domain decompositions for grids or local bases, sparse matrices operations based on space decomposition). The user reads documentation on how to define the data structure and does not worry about the rest.
- Modularity
- Pragmatism and not over-ambition. We will start will existing codes or sections of codes where possible. We will seek and exploit commonalities, but if some tasks are different for different methods, then we will create different routines. With appropriate modular standards and documentation, the code can be readily extended in the future. If we insist in just incorporating the commonest operations, we could be attempting too little, and we would not achieve a qualitative change in our way of operating. Serious, realistic thinking on what goes into the library must be undertaken at the start of the six-week period.

- Creating a living library. The library should be continuously open to contributions. We need to establish a realistic model for curation.

Kick-starting the event:

Make certain that the library and protocols started in the planed software development workshop in 2014 can be completed in good order.

- Assemble key players in the electronic-structure software community for the starting 3-day workshop, to:
    - finalise the working agenda for the following weeks;
    - establish key targets for the end of the workshop (list of software examples);
    - agree/select/negotiate ready-made packages to be incorporated from existing programs
    - establish protocols and methods for the workshop and beyond;
    - consider the future: funding for software infrastructure; funds going to HPC in some European Councils should be accessible (either funds or their workforce) to re-code numerically-intensive modules of the library for new hardware architectures; relate to HPC developments.
- Working period (6 weeks) to reach the targets.
    - Start by making a definition of immediate and longer term goals;, try to abstract commonalities in routines and data structure (be pragmatic);
    - standardise data as required, using the Nanoquanta definition if deemed useful and if available;
    - work on documentation of the defined packages;
    - adapt pre-existing code into the library and library style;
    - code new modules;
    - parallelise key modules with help of IBM Zürich (Curioni) and NVIDIA (for GPUs)

Support: CECAM has a computer and programming officer who can help; a sensible development environment will be provided. IBM Zürich is interested in contributing and helping. NVIDIA have also expressed an interest in helping (to be confirmed).

Preparation of the workshop:

This document will be iterated and circulated in the relevant community to define both teams (3-day workshop, 6-week one), the general scope, and to prepare the workshop. In the 3-day workshop there will be opportunity to specify and define many aspects of the work, but there may be need to have several aspects well defined before getting there (key packages to incorporate will benefit from having spoken previously with teams that have already coded them; this means that such packages should be identified in advance).

Preparatory meeting early in 2014 using e-mail and Skype to facilitate

Licensing:

There are license issues to be considered. These are not necessarily pressing, but it may be important to be clear on our approach when and if talking to companies, or exploring future relations with companies or start-ups. It will need to be clear under what conditions can the library routines available on the web be used by the general user.

GPL has many convinced practitioners (ABINIT, ESPRESSO, CPMD, CP2K). It is possible, however, that its infective character could be in the way of many initiatives. FreeBSD or similar approaches are interesting alternatives.

Since for entries with well-defined tasks and definition, there can be several attached implementations (e.g. from different sources, for different architectures) different implementations could have different licenses attached. This will need to be resolved from the start

Other possibilities: If CECAM becomes a depository of the rights of the library, it can issue the library under different licenses: e.g. it could be publicly distributed with GPL or other approaches but also offer other licenses to companies that would like to use it with their own software (may be even for fees in some cases that could be used to further CECAM's activities in this kind of space).

Present software (and groups) that could contribute (with people, ideas and code)

Electronic structure

- SIESTA (Soler, Garcia),
- ABINIT (Gonze),
- CAMPOS (Jacobsen),
- BigDFT (Goedecker, Deutsch, Genovese)
- ADF (Baerends)
- Quantum-Espresso (Laio),
- CP2K (Hutter, Vandevondele)
- CPMD (Curioni)
- CONQUEST (Bowler).
- ONETEP/CASTEP (Mostofi, Haynes, Payne)
- FHI-AIMS (Blum, Rinke, Scheffler)
- VASP (Kresse)
- WIEN (Blaha)
- CASINO (Needs)

Molecular dynamics

- Trocadero (Hernandez),
- GROMACS (Londhal)
- DL_POLY (Daresbury)
- LAMMPS
- CP2K
- CAMPOS

Examples of modules, routines, packages (to be revised, expanded, and prioritised)

- Domain decomposition of data among processors (following space, ideas about work balance, space-filling curves, inhomogeneous cases such as surfaces, tubes, clusters)
- Parallelisation of sparse (block-sparse) algebra library.
- XC functionals in real-space grid of arbitrary symmetry (cubic to triclinic) Marqués's and Soler's.
- Parallel FFTs
- Multi-grid Poisson solver in arbitrary grids (including variable epsilon, Poisson Boltzmann…)
- LCAO Solvers
- Iterative minimization schemes for PWs
- Linear-scaling solvers
- Finding neighbours
- Geometry dealings: lattice vectors, atomic positions, format conversions, set up grid for a cell for a given cutoff, k-grid (Monkhorst-Pack)
- Symmetry (identify, exploit).
- Atomic dealings: Solving atomic DFT for pseudo/PAW related things. Basis sets. Atomic quantities (populations)
- Xml-f90 lib (ready)
- Post-processing tools (DOS, bands, Fermi surface..)

- GUI, automatisation of system preparation
- Visualisation tools and/or drivers to connect with existing ones (XCrysDen, VMD, GDIS)
- Analysis of MD trajectories. (g(r), correlation functions….)
- MD drivers for the many flavours available
- Free energy wrappers for MD codes
- Task farming parallelisation