Developing Octopus: an Introduction

Martin Lüders, Micael Oliveira

Octopus Advanced Course 2023, MPSD Hamburg

Scientific Software Development

Unique challenges:

- Translating science into code
- Need to understand the science
- Scientist are often not trained in software engineering
- Software performance is often important
- Many codes need to be enabled for high-performance computing:
  - Parallelism (MPI, OpenMP, etc)
  - GPU's
  - Complex hardware
  - Unusual architectures

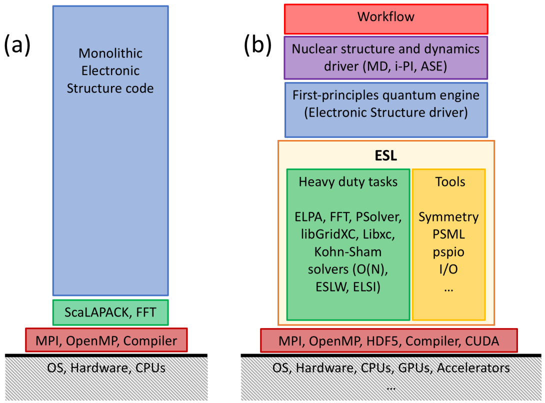
Scientific Software Development

- After a while, cost of maintenance becomes larger than cost of adding new features
- Software engineering good practices are essential!

Some best practices

- Code is the enemy: it can have bugs and it needs maintenance
- Do not reinvent the wheel: reuse code
- Write code that is easy to read and that is mostly self-documented
- Comments about why the code does something are very important
- Test your code
- "Premature optimization is the root of all evil"

Electronic structure “monolithic” and modular coding paradigms



M. J. T. Oliveira, N. Papior, Y. Pouillon, V. Blum, E. Artacho *et al*, J. Chem. Phys. **153**, 024117 (2020)

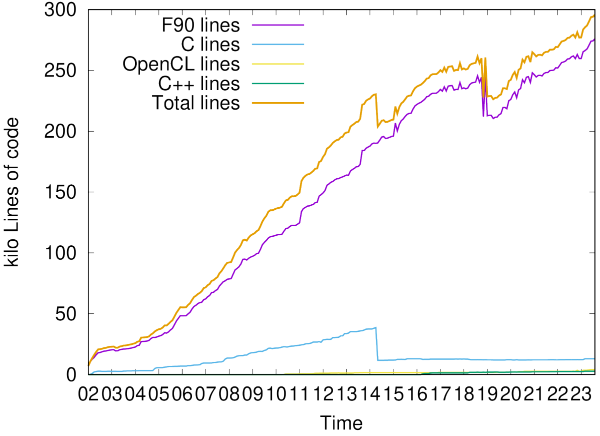
Octopus: Dissecting the Animal

- DFT and TDDFT code
- Some other theories implemented (Hartree-Fock, RDMFT, etc)
- Main focus on excited-state properties
- Real-space representation
- Norm-conserving pseudopotentials

Octopus: Dissecting the Animal

- Project formaly started in 2001
- Free-software (GPL)
- Writen mainly in Fortran 2003 (moving to Fortran 2008)
- Fortran sources are preprocessed with cpp
- Some C, C++, perl and Bison (use the right tool for the job!)
- CUDA/OpenCL for GPU support
- Currently over 250,000 lines of code

Octopus: Dissecting the Animal



Octopus: a code for developers

- Not the fastest code around for most problems, but still quite fast
- Real space grid:
  - Good compromise between plane-waves and localized basis-sets
  - Can be as accurate as any other basis
  - Can easily describe excited states
  - Simple and intuitive
- Lots of “exotic” features (e.g., model systems, arbitrary dimensions, etc)
- A framework to implement, develop and test new ideas

octopus-code.org

- HUGO based website
- Resources for users:
  - Code download
  - Compilation instructions (partially outdated)
  - Manual (partially outdated)
  - Tutorials
  - Input variable reference
  - ...
- Dedicated section for developers

www.octopus-code.org/documentation/main/developers/

- "Starting to develop" guide (must read!)
- Workflow (must read!)
- Coding standards
- Some code documentation (work in progress)
- ...

Git and GitLab

- Octopus uses **git** as version control system
- GitLab provides several important things:
  - Hosts main repository
  - Merge requests
  - Issues

Regression test suite and the Buildbot

- Octopus includes a large collection of regression tests
- Test suite covers $\sim 65\%$ of the code
- Continuous integration (CI) using Buildbot
- Buildbot is interfaced with GitLab

Build system

- Compilation and configuration is based on `autotools`
- Configure script is generated from `configure.ac`
- Makefiles are generated from `Makefile.am` files in each directory
- To generate the configure scripts run `autoreconf -i`
- `VPATH` builds are supported and suggested.
- Currently: transitioning to `CMake`.

External libraries

- We do not like to reinvent the wheel
- We like to share code
- Octopus uses many external libraries, either optional or mandatory:
    - BLAS/LAPACK
    - FFTW
    - MPI
    - GSL
    - Libxc
    - Libvdwxc
    - PSolver
    - ELPA
    - spglib
    - ...

Preprocessor: "templating"

- _inc.F90 files contain code that is independent of data type
- Files are included with the preprocessor in the following way:

```
#include "undef.F90"
#include "real.F90"
#include "my_function_inc.F90"

#include "undef.F90"
#include "complex.F90"
#include "my_function_inc.F90"
...
```

- Several macros are available to use in the _inc.F90 files

Input file variables

- Octopus uses a parser written in Bison
- Input file is fully parsed at the beginning of the calculation:

```
ierr = parse_init('exec/parser.log', mpi_world%rank)
```

- `exec/parser.log` contains all the variables **accessed** during a calculation
- Input variables can be accessed anywhere in the code
- Avoid reading each variable more than once

Input file variables

- All parser interfaces are defined in the parser_oct_m module
- Scalar variables are accessed with the parse_variable function:

```
call parse_variable(global_namespace, 'CalculationMode', OPTION__CALCULATIONMODE__GS,
  inp_calc_mode)
```

- Reading blocks requires to use a block_t data type
- Blocks must be "opened" and "closed":

```
type(block_t) :: blk
...
if (parse_block(namespace, 'Lsize', blk) == 0) then
  ! Lsize is specified as a block
  if (parse_block_cols(blk, 0) < space%dim) then
    call messages_input_error(namespace, 'Lsize')
  end if

  do idir = 1, space%dim
    call parse_block_float(blk, 0, idir - 1, sb%lsize(idir), units_inp%length)
    ...
  end do
  call parse_block_end(blk)
  ...
end if
```

Input variables documentation

- Variables are documented in the source code, just before where they are accessed
- Documentation is parsed by a script that generates HTML and plain text output
- Example:

```
!%Variable CalculationMode
!%Type integer
!%Default gs
!%Section Calculation Modes
!%Description
!% Decides what kind of calculation is to be performed.
!%Option gs 01
!% Calculation of the ground state.
!%Option unocc 02
!% Calculation of unoccupied/virtual KS states. Can also be used for a non-self-consist
!% calculation of states at arbitrary k-points, if <tt>density.obf</tt> from <tt>gs</tt
!% is provided in the <tt>restart/gs</tt> directory.
!% ...
!%End
```

- Options defined in the documentation can be used in the input file

Documentation

- Code documentation
  - Doxygen comments which can include formulas
  - Comment lines for specific constructs
- Feature documentation
  - Publication
  - Manual page on octopus-code.org
  - "Which quantity can Octopus compute" page
- Tutorials

Please, provide documentation to new features with your merge request.