Coder dans YALES2

V. Moureau, G. Lartigue, P. Bénard
CNRS-CORIA, UMR 6614, Rouen
http://www.coria-cfd.fr

Installation, Compilation, Debugging, ...

Installation (voir site web)

- Variables d'environnement

```
setenv YALES2_HOME /home/user/Devel/YALES2/trunk # installation path
setenv YALES2_HOSTTYPE myplatform                # name of the platform
set path = ( $path $YALES2_HOME/tools )          # path to the tools (optional)
```

- Récupérer les sources

```
git clone git@gitlab.coria-cfd.fr:yales2/yales2.git
```

- Créer une plateforme dans /platforms
- Compiler les librairies (HDF5, METIS, FFTW)
- Compiler les sources

```
cd $YALES2_HOME/src ; make –j 4
```

Structure des sources

- /src/main
  - Ce sont les sources principales du code
  - Un module par fichier
  - Les fichiers *_defs_m.f90 définissent les structures
  - Les fichiers *_numerics_m.f90 et *_solver_m.f90 définissent les solveurs

- /src/c3sm
  - L'interface graphique liée au moteur C3SM développé par le CERFACS

- /src/python
  - Interface python (nécessite f90wrap)

- Versioning : yales2-date

Structure des sources : /src/main

- Modules de définition des objets
  - *_defs_m.f90
  - Contiennent les constructeurs, destructeurs, …

- Modules de la bibliothèque numérique
  - grid_io_*_m.f90 : entrées/sorties
  - grid_*_m.f90 : interpolation, raffinement, …
  - bnd_*_m.f90 : frontières
  - data_*_m.f90 : données
  - scalar_*_m.f90 : scalaires (ensembles de données)

- Modules de définition des solveurs
  - *_solver_m.f90 : initialisation et boucle temporelle
  - *_numerics_m.f90 : méthodes numériques

Le Makefile principal et les dépendances

- Options du Makefile dans /src
  - `make –j 4`
  - `make debug`
  - `make doc`

- Gestion des dépendances entre modules dans le Makefile
  - Gestion automatique qui reconnaît les modules et les « use »
  - Il suffit de rajouter un nouveau fichiers dans /src/main
  - Il faut ensuite ajouter les « use module_m » où c'est nécessaire
  - Les dépendances sont refaites automatiquement lors du `make`

Le debugging

- Débugging élémentaire
  - `setenv Y2_DEBUG_MODE TRUE`
  - Ou `make debug` dans /src (avec `make clean` avant si nécessaire)
  - Puis `make` dans le répertoire de run

- Possibilité de faire du debugging en parallèle avec gdb

```
mpirun –np 4 xterm –e gdb –ex run ./exec
```

Quelques conseils

- Mettre régulièrement sa version à jour

```
git fetch --all
git pull
```

- Respecter l'indentation et le formatting
- Ecrire la documentation en même temps que les sources
- Fournir des cas test
- Partagez vos intentions de développement (les besoins des uns et des autres peuvent se recouper)
- Participez aux comités de pilotage