Top-Level Code Walk-Through: scalarTransportFoam and magU

Henrik Rusche

`h.rusche@wikki-gmbh.de`

Wikki, United Kingdom and Germany

OpenFOAM-Workshop Training Sessions

7th OpenFOAM Workshop

24.6.2012, Darmstadt, Germany

Outline

Objective

- Detailed source code walk through simple executables
  - ○ `scalarTransportFoam`: scalar transport equation with prescribed velocity
  - ○ `magU`: velocity field magnitude utility

Topics

- Types of source files: headers, include files and compiled files
- `scalarTransportFoam` walk-through
- `magU` walk-through
- `wmake` build system

Top-Level Solver

Structure of a Top-Level Solver

- **Header files**
  - Located before the `main(int argc, char *argv[])` statement
  - Contain class definition for all classes used in the solver code
  - Packed for convenience, *e.g.* `#include "fvCFD.H"` for the FVM
- **Include files**
  - Code snippets that are repeated in many places are packed into files
  - Example:

```
#   include "setRootCase.H"
#   include "createTime.H"
#   include "createMesh.H"
```

- **createFields.H**
  - Contains field and material property definition used by the equation
- **Time loop and equations**
  - Contains equation set to be solved, together with auxiliary functionality
  - Best documentation for implemented algorithm
  - `runTime.write();` triggers database I/O for automatic objects

Top-Level Solver Walk-Through

`scalarTransportFoam` Walk-Through

- Create a field by reading it from a file

```
volScalarField T
(
    IOobject
    (
        "T",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

- Read options: `MUST_READ`, `READ_IF_PRESENT`, `NO_READ`
- Write options: `NO_WRITE`, `AUTO_WRITE`
- `IOobject` and `regIOobject`: registration with object database for automatic read-write operations

Top-Level Solver Walk-Through

`scalarTransportFoam` Walk-Through

- Retrieving data from a dictionary

```
IOdictionary transportProperties
(
    IOobject
    (
        "transportProperties",
        runTime.constant(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
);

dimensionedScalar DT
(
    transportProperties.lookup("DT")
);
```

Top-Level Solver Walk-Through

`scalarTransportFoam` Walk-Through

- Dictionary format: file header (IOobject) and keyword-value entry pairs

```
FoamFile
{
    version             2.0;
    format              ascii;

    root                "";
    case                "";
    instance            "";
    local               "";

    class               dictionary;
    object              transportProperties;
}

// Diffusivity
DT           DT [0 2 -1 0 0 0 0] 0.01;
```

Top-Level Solver Walk-Through

WIKKI

`scalarTransportFoam` Walk-Through

- Time loop: note consistent naming of objects: `mesh`, `runTime` etc.

```
for (runTime++; !runTime.end(); runTime++)
{
    Info<< "Time = " << runTime.timeName() << nl << endl;

#   include "readSIMPLEControls.H"

    for (int nonOrth=0; nonOrth<=nNonOrthCorr; nonOrth++)
    {
        solve
        (
            fvm::ddt(T)
          + fvm::div(phi, T)
          - fvm::laplacian(DT, T)
        );
    }

    runTime.write();
}
```

Utility Walk-Through

`magU` Walk-Through

- Example of a utility performing post-processing on data written out in files
- Algorithm: go through all time directories, read velocity field if present, write out its magnitude
- `applications/utilities/postProcessing/velocityField/magU`
- Add option to operate only on chosen time directory:
  `Usage: magU <root> <case> [-parallel] [-constant] [-latestTime] [-time time]`
- The loop now involves data directories found in the `case`, rather than advancing through time

```
    instantList Times = runTime.times();

// set startTime and endTime depending on -time and -latestTime
#   include "checkTimeOptions.H"

    runTime.setTime(Times[startTime], startTime);
    for (label i=startTime; i<endTime; i++)
    {
        runTime.setTime(Times[i], i);
        ...
```

Utility Walk-Through

Example: Calculate and Write Velocity Magnitude

- Attempt to read the velocity

```
IOobject Uheader
(
    "U",
    runTime.timeName(),
    mesh,
    IOobject::MUST_READ
);

if (Uheader.headerOk())
{
    mesh.readUpdate();

    Info<< "    Reading U" << endl;
    volVectorField U(Uheader, mesh);
    ...
else
{
    Info<< "    No U" << endl;
}
```

Utility Walk-Through

Example: Calculate and Write Velocity Magnitude

- Calculate and write velocity magnitude: `mag(U)`!
- Note the use of alternative constructor and read/write options

```
Info<< "    Calculating magU" << endl;
volScalarField magU
(
    IOobject
    (
        "magU",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mag(U)
);

Info << "mag(U): max: " << gMax(magU.internalField())
    << " min: " << gMin(magU.internalField()) << endl;

magU.write();
```

Build System

Using `wmake` Build System

- Build system controlled by files in `Make` directory
- Sub-directories organised by platform type and options: `WM_OPTIONS`
- `Make/files` lists source files and location of executable or library

```
    scalarTransportFoam.C
    EXE = $(FOAM_APPBIN)/scalarTransportFoam
```

- `Make/options` lists include paths and library dependencies

```
    EXE_INC =  -I$(LIB_SRC)/finiteVolume/lnInclude
    EXE_LIBS = -lfiniteVolume
```

- Relevant Make system variables
  - `EXE`: Location for the executable. Use `FOAM_APPBIN` or `FOAM_USER_APPBIN`
  - `LIB`: location for the library. Use `FOAM_LIBBIN` or `FOAM_USER_LIBBIN`
  - `EXE_INC`: location of include paths. Use `LIB_SRC`; each library soft-lints all files into `lnInclude` directory for easy inclusion of search paths
  - `EXE_LIBS`, `LIB_LIBS`: Link libraries for executables or other libraries

Summary

Summary

- A bulk of OpenFOAM executables follow the same pattern
- Top-level objects used repeatedly are consistently named: `mesh`, `runTime`
- Top-level physics solver codes operate in a time- or iteration loop
- Post-processing utilities operate in a loop over existing data directories
- `wmake` build system is controlled by files in the `Make` directory