IAEA
International Atomic Energy Agency

Joint ICTP-IAEA Workshop on Reactor Physics, Thermal Hydraulics and Plant Design Engineering of Small Modular Reactors

13th April 2026 – 17th April 2026

Introduction to OpenFOAM and installation

15th April 2026

Tommaso Pernatsch – Politecnico di Milano (tommaso.pernatsch@polimi.it)
Stefano Lorenzi – Politecnico di Milano (stefano.lorenzi@polimi.it)

The OpenFOAM software and how can we model MSRs

We will have a look on:

- What is OpenFOAM?
- Introduction to Finite Volumes methods
- How is structured an OpenFOAM case and solver?
- Installation of OpenFOAM

The OpenFOAM software and how can we model MSRs

We will have a look on:

- What is OpenFOAM?
- Introduction to Finite Volumes methods
- How is structured an OpenFOAM case and solver?
- Installation of OpenFOAM

What is OpenFOAM?

- OpenFOAM is an open-source C++ library for numerical simulation in continuum mechanics, i.e., is used to solve (coupled) PDEs and ODEs

- **Open-source** environment, the source code is available, and you **can customize** it

- Field Operation and Manipulation: all the tools required to **discretise** and **solve transport equations** on a geometry domain are available.

- **Multiphysics simulation capabilities**: CFD, Computational heat transfer and conjugate heat transfer, combustion and chemical reactions, multiphase flows and mass transfer, neutronics

Open∇FOAM

The OpenFOAM software and how can we model MSRs

We will have a look on:

- What is OpenFOAM?
- Introduction to Finite Volumes methods
- How is structured an OpenFOAM case and solver?
- Installation of OpenFOAM

Introduction on FV

The purpose of any discretisation practice is to transform one or more partial differential equations into a corresponding system of algebraic equations.

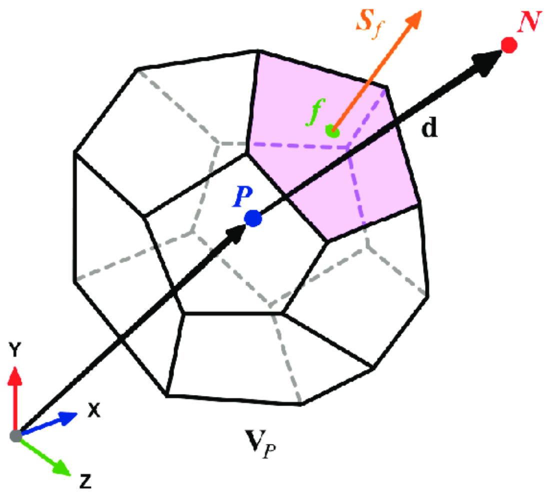
$$\underbrace{\int_{V_P} \frac{\partial \rho \phi}{\partial t} dV}_{\text{Temporal derivative}} + \underbrace{\int_{V_P} \nabla \cdot (\rho \mathbf{u} \phi) dV}_{\text{Convective term}} - \underbrace{\int_{V_P} \nabla \cdot \left(\rho \Gamma_\phi \nabla \phi\right) dV}_{\text{Diffusion term}} = \underbrace{\int_{V_P} S_\phi(\phi) dV}_{\text{Source term}}$$

Taken from Joel Guerrero, Finite Volume Method: A Crash introduction. https://wiki.openfoam.com/Theory_by_Joel_Guerrero

Introduction on FV

Two ingredients: **domain discretization** and equation discretization

The domain is divided in **control volumes V_p** where the solution is calculated by integrating the PDE over V_p.

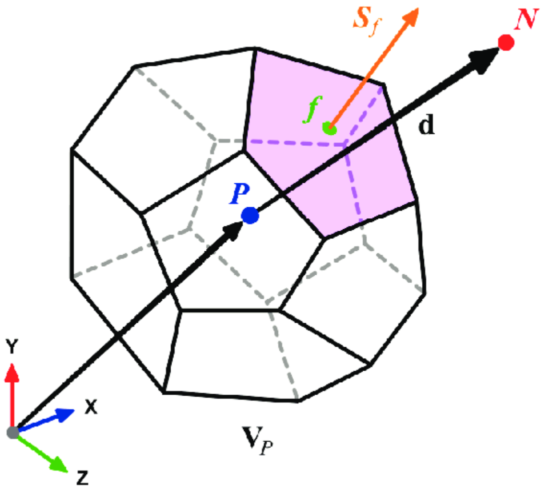


- **P** is the centroid of the corresponding CV
- **N** is the centroid of the neighbor CV
- **d** is the distance from P to N
- **f** is the control volume face
- **S_f** is the face area vector pointing outwards from the control volume, normal to the face with magnitude equal to the area of the face

Taken from Joel Guerrero, Finite Volume Method: A Crash introduction. https://wiki.openfoam.com/Theory_by_Joel_Guerrero

Introduction on FV

Two ingredients: domain discretization and **equation discretization**



$$\phi(\mathbf{x}) = \phi_P + (\mathbf{x} - \mathbf{x_P}) \cdot (\nabla\phi)_P$$

$$\phi(t + \delta t) = \phi^t + \delta t \left(\frac{\partial \phi}{\partial t}\right)^t$$

HP: linear variation in space and time

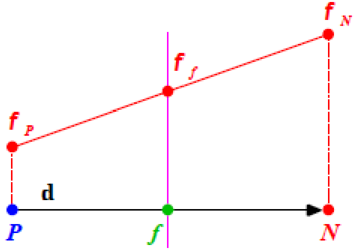
$$\int_{V_P} \nabla \cdot \mathbf{a} dV = \oint_{\partial V} d\mathbf{S} \cdot \mathbf{a}$$

Gauss (divergence) theorem to convert the volume integrals into surface integrals.

Taken from Joel Guerrero, Finite Volume Method: A Crash introduction. https://wiki.openfoam.com/Theory_by_Joel_Guerrero

Introduction on FV

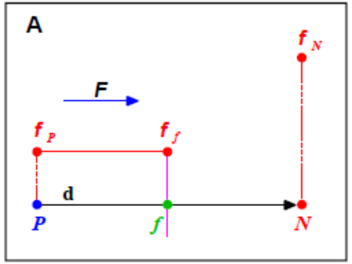
Convective flux



$$\phi_f = f_x \phi_P + (1 - f_x)\phi_N$$

$$f_x = \frac{f\,N}{P\,N} = \frac{|\mathbf{x}_f - \mathbf{x}_N|}{|\mathbf{d}|}$$

Linear interpolation
(central differencing)
. II order accurate
. May create oscillatory solution



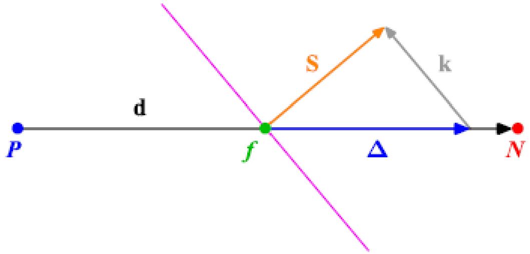
$$\phi_f = \begin{cases} \phi_f = \phi_P & \text{for} \quad \dot{F} \geq 0 \\ \phi_f = \phi_N & \text{for} \quad \dot{F} < 0 \end{cases}$$

Upwind differencing
. I order accurate
. Bounded solution
. Diffusive

Taken from Joel Guerrero, Finite Volume Method: A Crash introduction. https://wiki.openfoam.com/Theory_by_Joel_Guerrero

Introduction on FV

Diffusive flux



$$\mathbf{S} = \nabla_{\perp} + \mathbf{k}$$

$$\mathbf{S} \cdot (\nabla\phi)_f = \underbrace{|\nabla_{\perp}| \frac{\phi_N - \phi_P}{|\mathbf{d}|}}_{\text{Orthogonal contribution}} + \underbrace{\mathbf{k} \cdot (\nabla\phi)_f}_{\text{Non-orthogonal contribution}}$$

Linear interpolation
(central differencing)
. II order accurate
. Truncation error introduced due to the mesh non-orthogonality

Introduction on FV

Time discretisation

$$\left(\frac{\partial \rho \phi}{\partial t}\right)_P = \frac{\rho_P^n \phi_P^n - \rho_P^0 \phi_P^0}{\Delta t}$$

$$\int_t^{t+\Delta t} \phi(t) dt = \frac{1}{2}(\phi^0 + \phi^n)\Delta t$$

Euler explicit (very bad)
. I order accurate
. Possibly unstable

Euler implicit
. I order accurate
. Stable
. Leads to a linear system

Crank Nicholson
. II order accurate
. Stable
. Leads to a linear system

Taken from Joel Guerrero, Finite Volume Method: A Crash introduction. https://wiki.openfoam.com/Theory_by_Joel_Guerrero

Introduction on FV

The time discretisations lead to a linear system to be solved

$$a_P \phi_P^n + \sum_N a_N \phi_N^n = R_P$$

$$[A][\phi] = [R]$$

$$\begin{pmatrix} a_{11} & a_{12} & & \ddots & & & & \\ a_{21} & a_{22} & a_{23} & & \ddots & & & \\ & \ddots & \ddots & \ddots & & \ddots & & \\ \ddots & & \ddots & \ddots & \ddots & & \ddots & \\ & a_S & & a_W & a_P & a_E & & a_N \\ & & \ddots & & \ddots & \ddots & \ddots & \\ & & & \ddots & & \ddots & \ddots & \ddots \\ & & & & \ddots & & \ddots & a_{PP} \end{pmatrix} \times \begin{pmatrix} \\ \phi_S \\ \\ \phi_W \\ \phi_P \\ \phi_E \\ \\ \phi_N \\ \\ \end{pmatrix} = \begin{pmatrix} \\ b_S \\ \\ b_W \\ b_P \\ b_E \\ \\ b_N \\ \\ \end{pmatrix}$$

Taken from Joel Guerrero, Finite Volume Method: A Crash introduction. https://wiki.openfoam.com/Theory_by_Joel_Guerrero

The OpenFOAM software and how can we model MSRs

We will have a look on:

- What is OpenFOAM?
- Introduction to Finite Volumes methods
- How is structured an OpenFOAM case and solver?
- Installation of OpenFOAM

OpenFOAM case

How to perform simulation with OpenFOAM:

1) Select the solver, i.e., the modelling

2) Case preparation
   - Geometry and mesh
   - Initial and boundary conditions
   - Physical models/parameters
   - Discretization
   - Solver parameters
   - Simulation parameters

3) Post-processing
   - Visualization
   - Data analysis

OpenFOAM case

How to perform simulation with OpenFOAM:

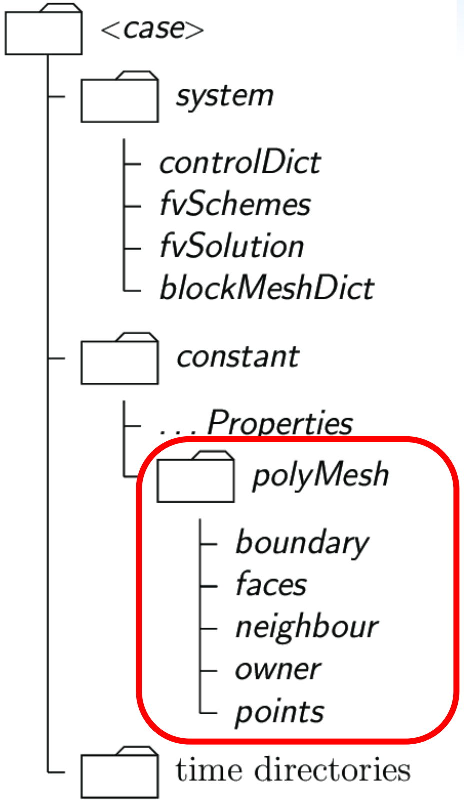
1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis



OpenFOAM case

How to perform simulation with OpenFOAM:

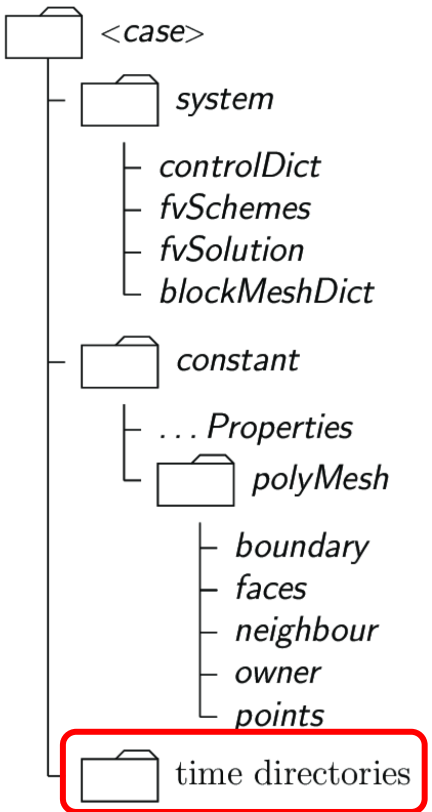
1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis



OpenFOAM case

How to perform simulation with OpenFOAM:

1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis

IC

BC

```
dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    movingWall
    {
        type            zeroGradient;
    }

    fixedWalls
    {
        type            zeroGradient;
    }

    frontAndBack
    {
        type            empty;
    }
}
```

OpenFOAM case

How to perform simulation with OpenFOAM:

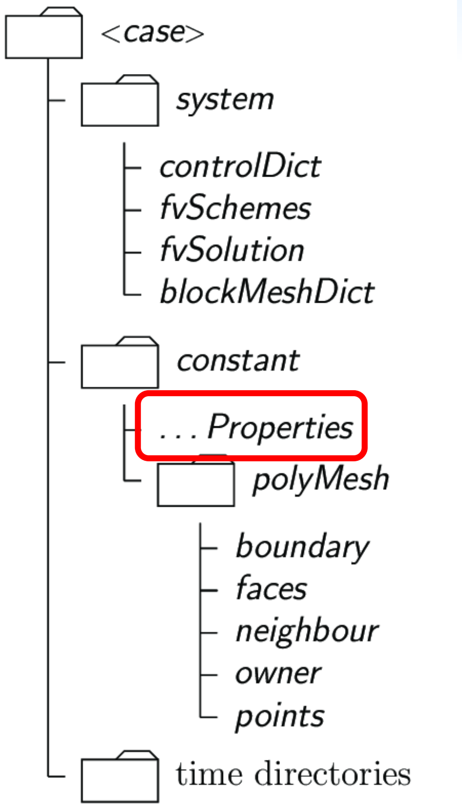
1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis



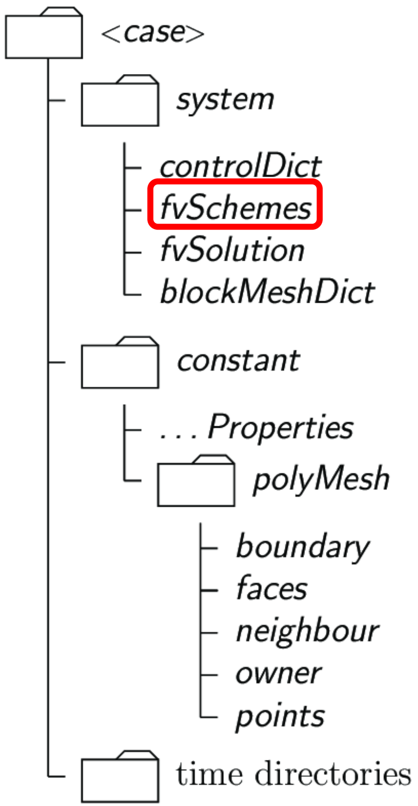
OpenFOAM case

How to perform simulation with OpenFOAM:

1) Select the solver, i.e., the modelling

2) Case preparation
- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing
- Visualization
- Data analysis



OpenFOAM case

How to perform simulation with OpenFOAM:

1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis

```
ddtSchemes
{
    default         backward;
}
```

$$\frac{\partial \phi}{\partial t}$$

```
gradSchemes
{
    default         Gauss linear;
    grad(p)         Gauss linear;
}
```

$$\nabla \phi_P$$

```
divSchemes
{
    default         none;
    div(phi,U)      Gauss linear;
}
```

$$\nabla \cdot (\mathbf{U}\phi)$$

```
laplacianSchemes
{
    default         Gauss linear orthogonal;
}
```

$$\nabla \cdot \Gamma \nabla \phi$$

```
interpolationSchemes
{
    default         linear;
}
```

$$\left\{ \begin{array}{l} \phi_f = f_x \phi_P + (1 - f_x)\phi_N \\ f_x = \frac{fN}{PN} = \frac{|\mathbf{x}_f - \mathbf{x}_N|}{|\mathbf{d}|} \end{array} \right.$$

```
snGradSchemes
{
    default         orthogonal;
}
```

$$\mathbf{n}_f \cdot \nabla \phi_f$$

OpenFOAM case

How to perform simulation with OpenFOAM:

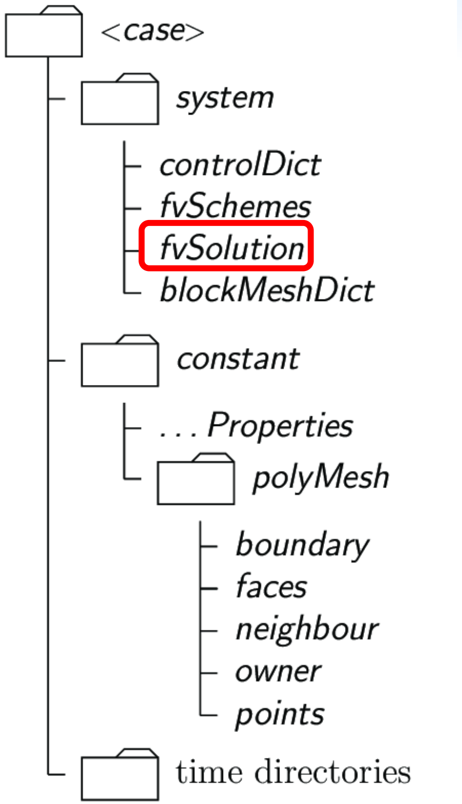
1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis



OpenFOAM case

How to perform simulation with OpenFOAM:

1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis

Matrix solver

NS equation

```
solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }
    pFinal
    {
        $p;
        relTol    0;
    }
    U
    {
        solver          PBiCGStab;
        preconditioner  DILU;
        tolerance       1e-08;
        relTol          0;
    }
}

PISO
{
    nCorrectors    2;
    nNonOrthogonalCorrectors    1;
}
```

OpenFOAM case

How to perform simulation with OpenFOAM:

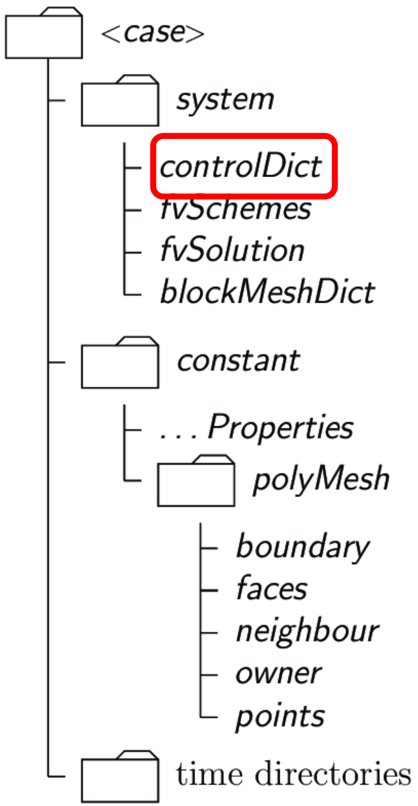
1) Select the solver, i.e., the modelling

2) Case preparation

- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing

- Visualization
- Data analysis



OpenFOAM case

IAEA

How to perform simulation with OpenFOAM:

1) Select the solver, i.e., the modelling

2) Case preparation
- Geometry and mesh
- Initial and boundary conditions
- Physical models/parameters
- Discretization
- Solver parameters
- Simulation parameters

3) Post-processing
- Visualization
- Data analysis

```
startFrom       latestTime;
startTime       0;
stopAt          endTime;
endTime         10000;
deltaT          1;
writeControl    runTime;
writeInterval   100;
purgeWrite      10;
writeFormat     ascii;
writePrecision  8;
writeCompression off;
timeFormat      general;
timePrecision   6;
runTimeModifiable   yes;
```

OpenFOAM solver

Let's take a laminar fluid dynamics problem…

Physical modelling = Incompressible Navier – Stokes equations

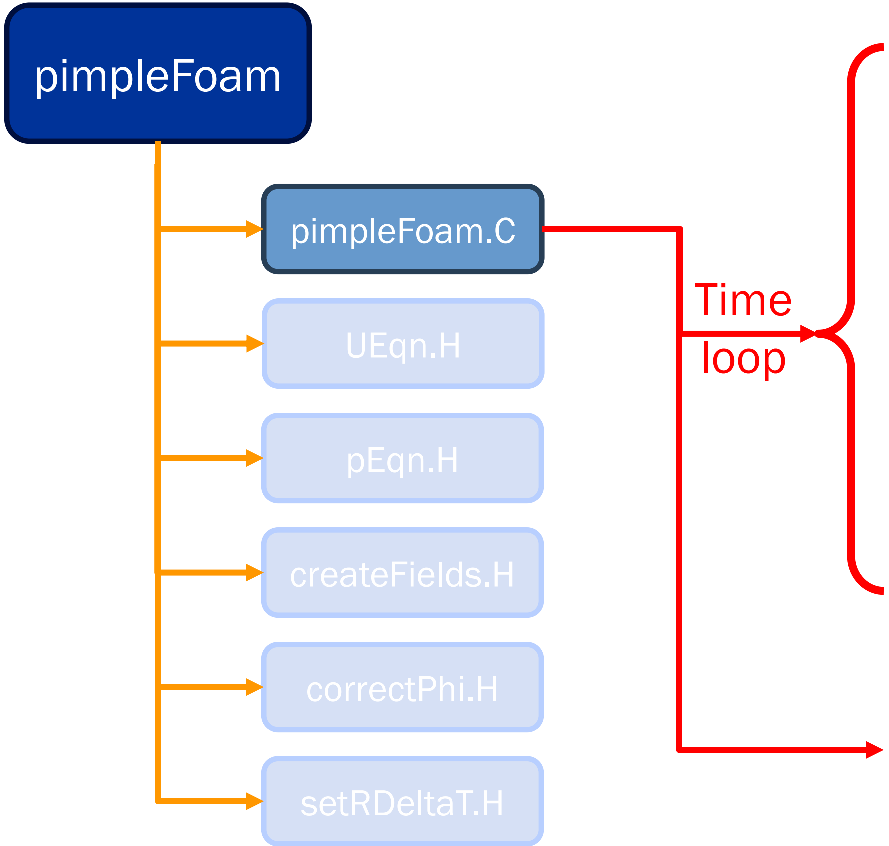
$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \dfrac{\partial \mathbf{u}}{\partial \mathrm{t}} + (\mathbf{u} \cdot \nabla)\mathbf{u} = \nu \nabla^2 \mathbf{u} - \dfrac{\nabla \mathrm{p}}{\rho_0} + \mathbf{F} \end{cases}$$

Which is the structure of an hypotetical OF solver for Incompressible NS?

OpenFOAM solver

- pimpleFoam
  - pimpleFoam.C
  - UEqn.H
  - pEqn.H
  - createFields.H
  - correctPhi.H
  - setRDeltaT.H

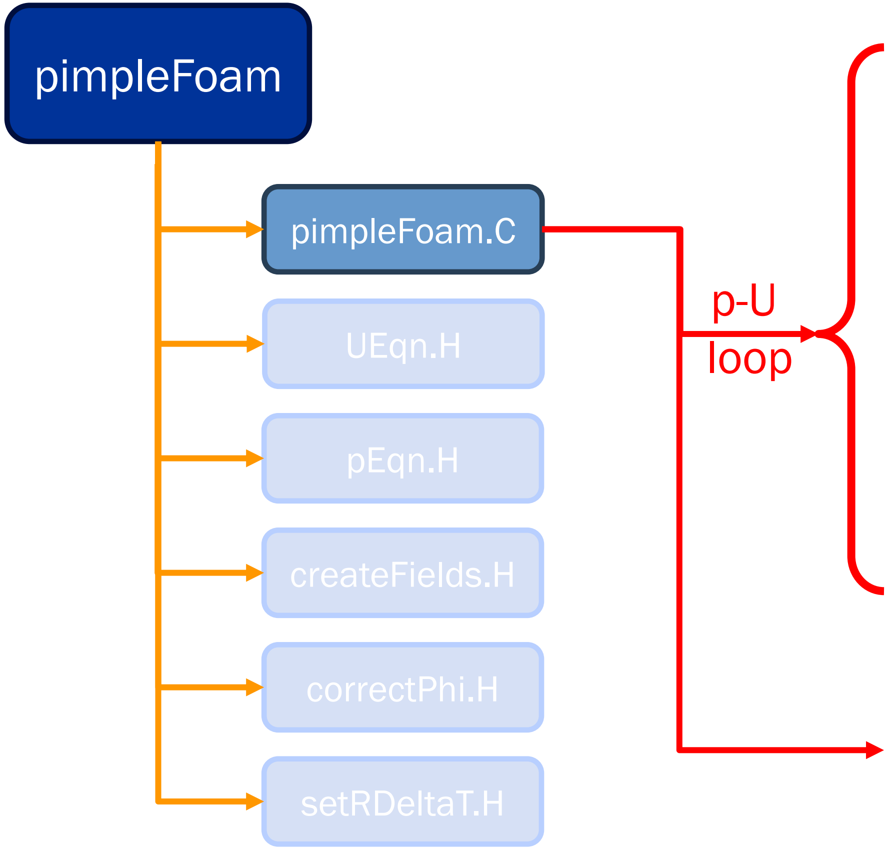
OpenFOAM solver




```
Info<< "\nStarting time loop\n" << endl;

while (runTime.run())
{
    #include "readDyMControls.H"

    if (LTS)
    {
        #include "setRDeltaT.H"
    }
    else
    {
        #include "CourantNo.H"
        #include "setDeltaT.H"
    }

    ++runTime;

    Info<< "Time = " << runTime.timeName() << nl << endl;
```

3 loops in the main solver file

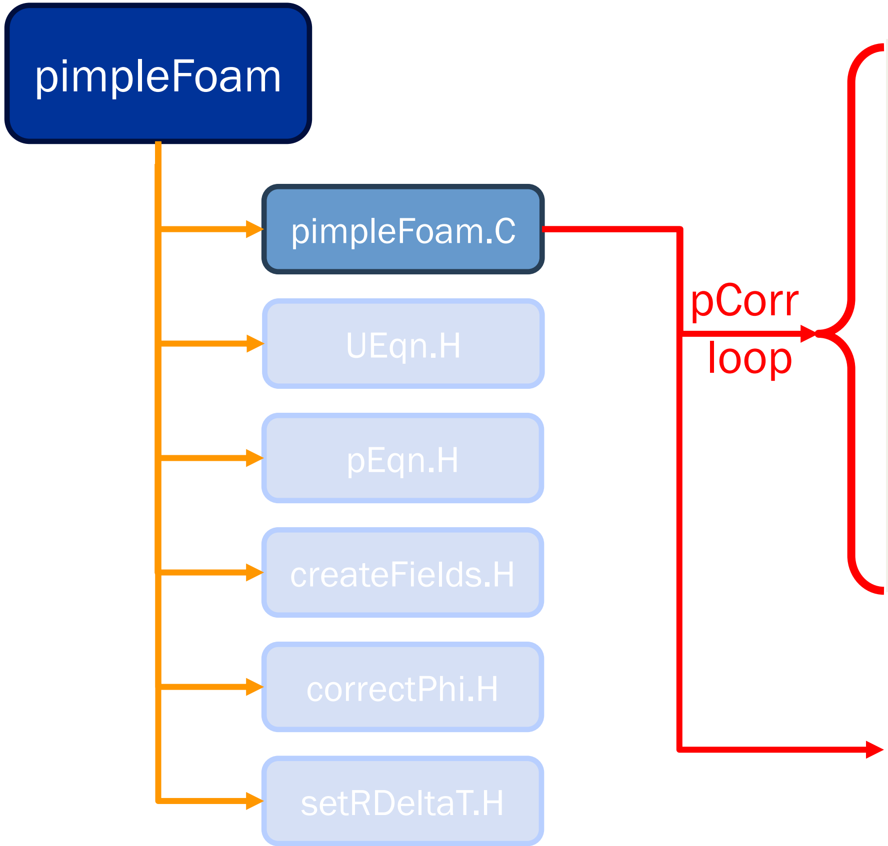
OpenFOAM solver




```
// --- Pressure-velocity PIMPLE corrector loop
while (pimple.loop())
{
    if (pimple.firstIter() || moveMeshOuterCorrectors)
    {
        // Do any mesh changes
        mesh.controlledUpdate();

        if (mesh.changing())
        {
            MRF.update();

            if (correctPhi)
            {
                // Calculate absolute flux
                // from the mapped surface velocity
                phi = mesh.Sf() & Uf();

                #include "correctPhi.H"
```

3 loops in the main solver file

OpenFOAM solver




```
#include "UEqn.H"

// --- Pressure corrector loop
while (pimple.correct())
{
    #include "pEqn.H"
}

if (pimple.turbCorr())
{
    laminarTransport.correct();
    turbulence->correct();
}
```

3 loops in the main solver file

OpenFOAM solver

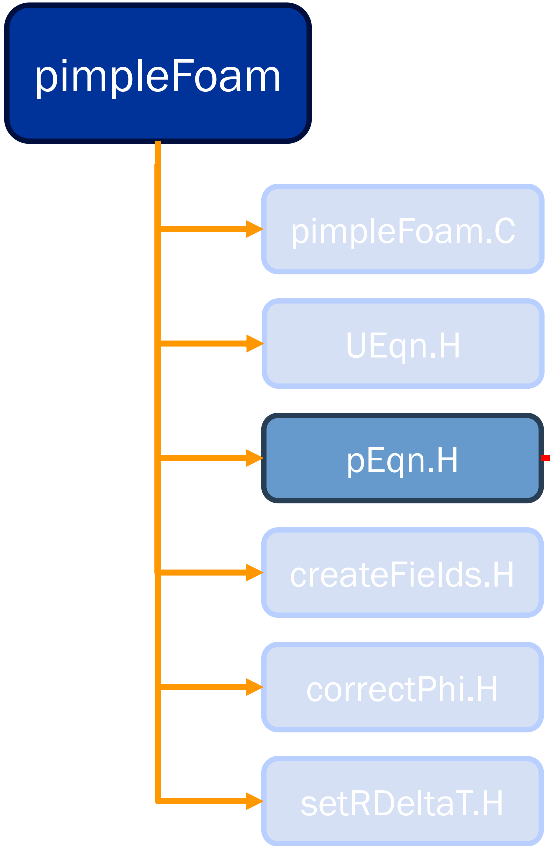
- pimpleFoam
  - pimpleFoam.C
  - UEqn.H
  - pEqn.H
  - createFields.H
  - correctPhi.H
  - setRDeltaT.H

```
// Solve the Momentum equation

MRF.correctBoundaryVelocity(U);

tmp<fvVectorMatrix> tUEqn
(
    fvm::ddt(U) + fvm::div(phi, U)
  + MRF.DDt(U)
  + turbulence->divDevReff(U)
 ==
    fvOptions(U)
);
fvVectorMatrix& UEqn = tUEqn.ref();

UEqn.relax();

fvOptions.constrain(UEqn);

if (pimple.momentumPredictor())
{
    solve(UEqn == -fvc::grad(p));

    fvOptions.correct(U);
}
```

Equation mimicking

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\mathbf{u} \cdot \boldsymbol{\nabla})\boldsymbol{u} - \nu \Delta \boldsymbol{u} + \boldsymbol{\nabla} p = 0$$

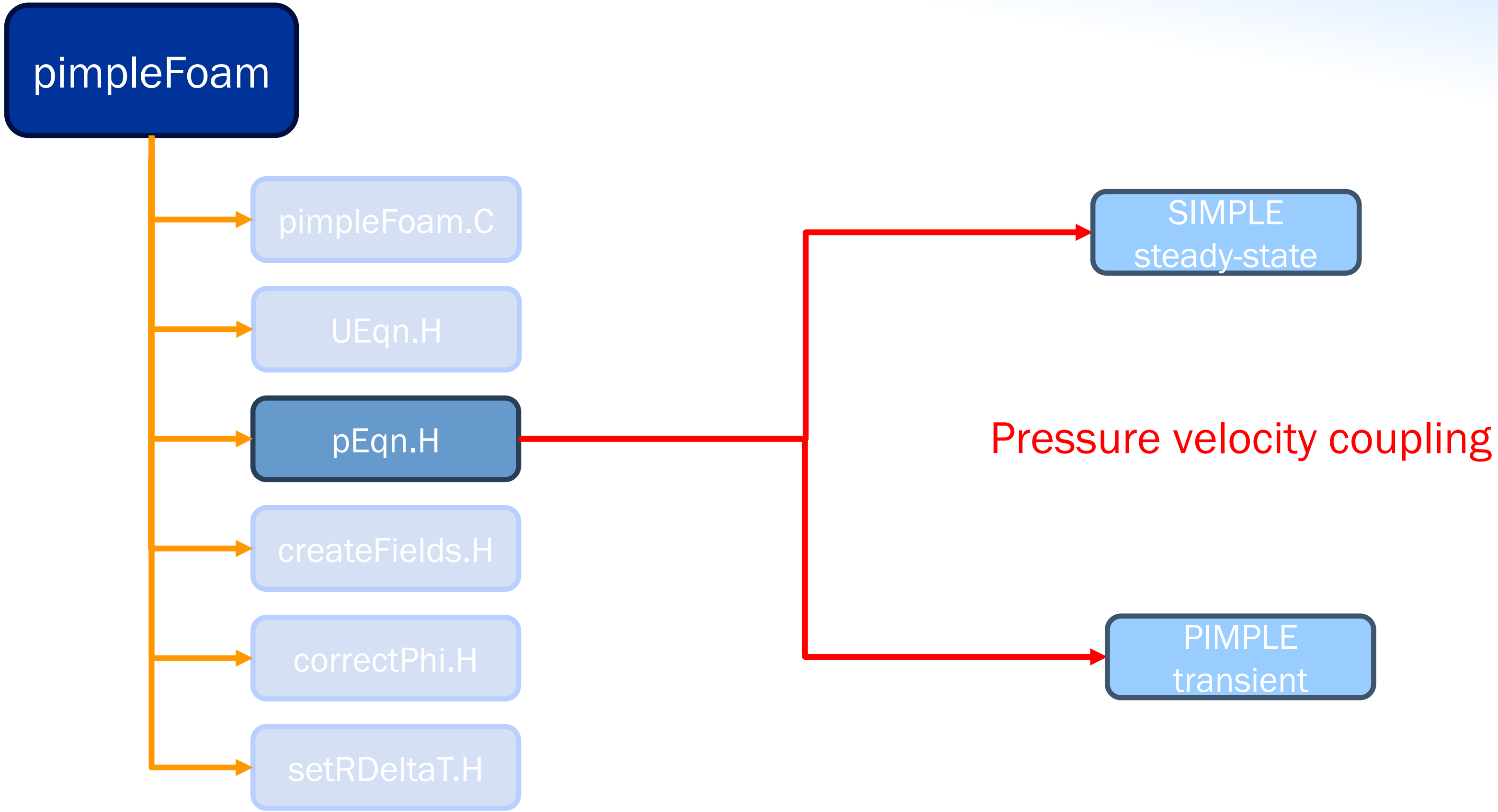
OpenFOAM solver




```
// Non-orthogonal pressure corrector loop
while (pimple.correctNonOrthogonal())
{
    fvScalarMatrix pEqn
    (
        fvm::laplacian(rAtU(), p) == fvc::div(phiHbyA)
    );

    pEqn.setReference(pRefCell, pRefValue);

    pEqn.solve(p.select(pimple.finalInnerIter()));

    if (pimple.finalNonOrthogonalIter())
    {
        phi = phiHbyA - pEqn.flux();
    }
}
```

No real pressure eq. in the NS

OpenFOAM solver



First steps with OF

I am curious about OpenFOAM… but which version?

openfoam.*com*

This is the version we will use OpenFOAM-v2312

openfoam.*org*

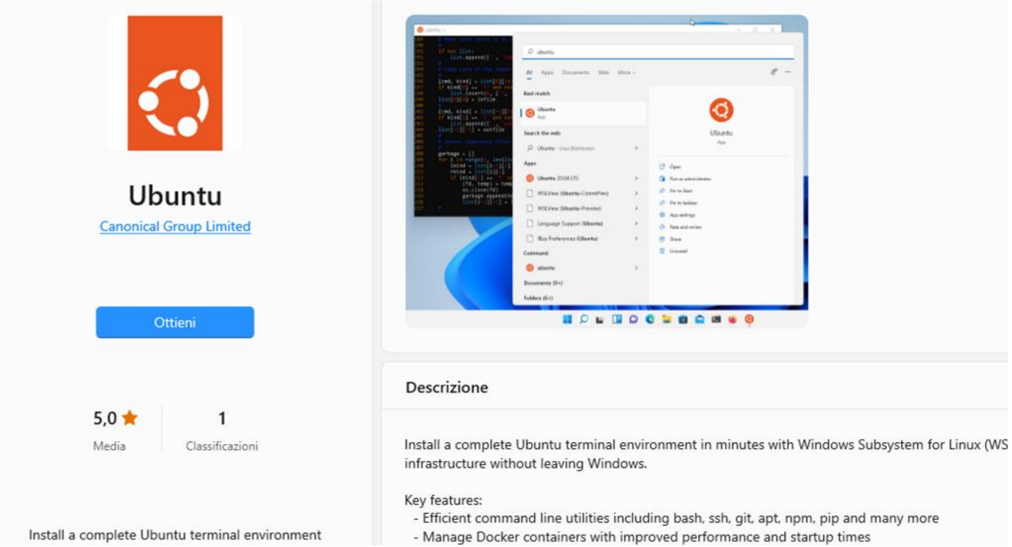
IMPORTANT

If you want to use an available solver or take features from available solvers for your own solver, be very careful and select the right OF version!

First steps with OF

Can I use it on my computer?

OpenFOAM runs natively on Linux systems...



MAC, or the Linux subsystem for Windows can be used

First steps with OF

IAEA

How to install it?

- Windows → Download Ubuntu and then install OpenFOAM
- macOS → Download Docker with OpenFOAM distribution
- Linux → Install directly OpenFOAM

First steps with OF

Learn OpenFOAM – Presentations and tutorial from OF wiki

https://wiki.openfoam.com/index.php?title=%223_weeks%22_series

3-weeks-series

Day 1	Day 2	Day 3	Day 4	Day 5
install - first steps	steps - visualization	introductory course	discretization	theory - fun simulations - tips
Day 6	**Day 7**	**Day 8**	**Day 9**	**Day 10**
geometry and meshing	turbulence 1	turbulence 2	multiphase	parallelization
Day 11	**Day 12**	**Day 13**	**Day 14**	**Day 15**
programming 1	programming 2	programming 3	programming 4	programming 5

IAEA
International Atomic Energy Agency

Questions?