

NanoDome
H2020-NMP-2014-two-stage
NMP-20-2014 Widening materials
models
Grant Agreement n. 646121
DELIVERABLE 4.2
Interface and Linking Description
Document

Contents

1	Introduction	3
2	Overview	3
3	Software Synopsis	4
3.1	Application nanoStreamlineFoam	5
3.2	Application rhoNanoNonreactiveFoamTP	5
3.3	Applications rhoNanoFlowFoam/rhoNanoFlowDyMFoam	5
4	Compilation and Utilization Details	6
4.1	Compilation	6
4.2	Utilization	6

1 Introduction

The purpose of this documentation is to provide a guideline for the source code developed for the deliverable 4.2. The offered software package includes solvers and additional libraries for the open source CFD-library OpenFOAM in the version 2.3.x and is therefore designed to realize the data exchange between *NanoDome* and **OpenFOAM**.

The code is publicly available through the GitHub online repository at this address: https://github.com/nanodome/cfd_linking

2 Overview

As aforementioned in deliverable 4.1, the communication between NanoDome Core and the CFD library OpenFOAM will be realized in two ways, by *linking* and *coupling*. The whole multi scale simulation process is sketched in figure 1.

The *linking process*, which can be understood as a one-way coupling (soft coupling), consists of three basic steps, which are namely the main CFD-computation, the streamline and XML-file generation and finally the XML-file reading meso scale simulation. The first two steps are done by applying the OpenFOAM library, for the last step the NanoDome core is applied. The user can use the software provided in work package 4.2 to post process the converged CFD simulation and extract the required data along the calculated streamlines.

In the *coupling process*, the CFD-computation is also influenced by relevant quantities of the meso scale simulation. Therefore, an additional communication channel from the meso scale simulation to the CFD-simulation is relevant (two-way coupling). The most performant implementation for this process is done by directly implement the NanoDome Core libraries into OpenFOAM. By this kind of implementation, the whole process is embedded in one algorithm and no redundant file sharing algorithms are necessary. The user sets up the simulation, while the solving procedure is controlled by the OpenFOAM solver.

The *linking/coupling* library actually only includes OpenFOAM libraries and solver to generate and save streamline data for the *linking*. The *coupling*, in the aforementioned way, will be implemented during the NanoDome project task 4.3.

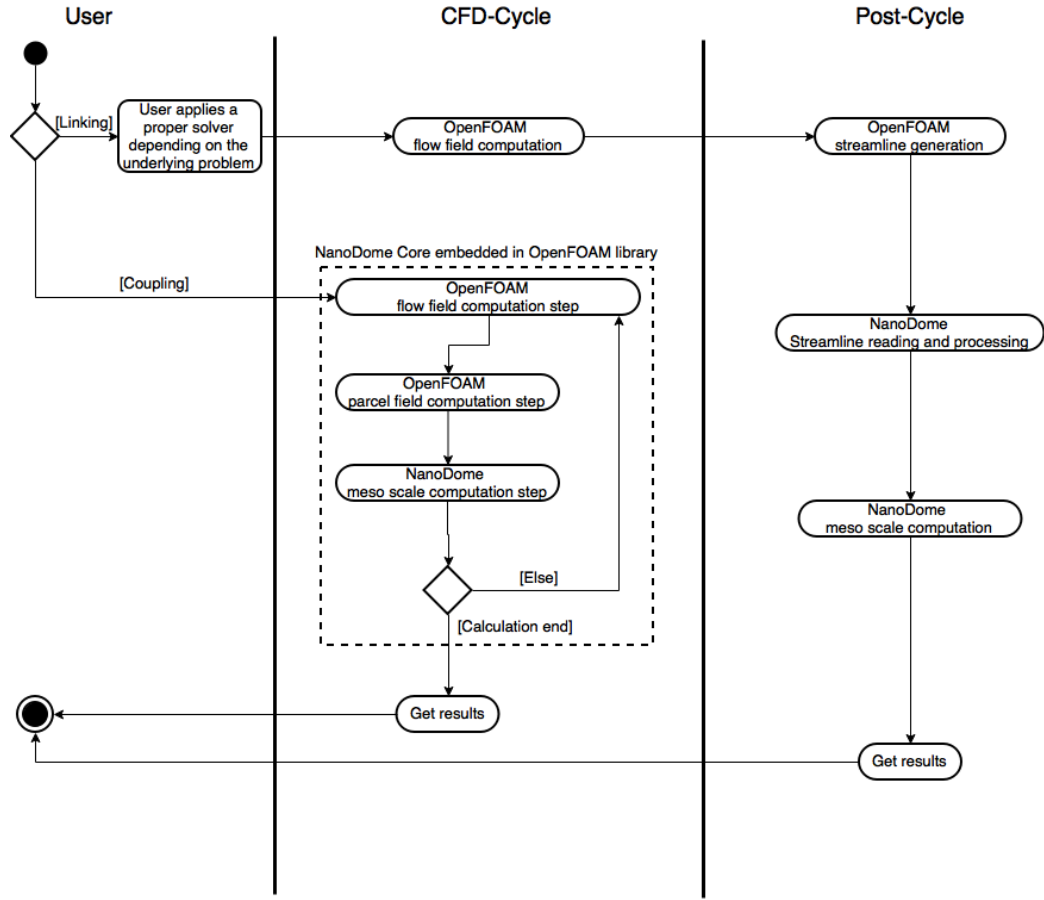


Figure 1: Coupling and linking workflow in the scope of the OpenFOAM implementation

3 Software Synopsis

The software appropriate to this document is an extension for the OpenFOAM CFD-library. It includes additional libraries and solvers to provide the required data in the format described in deliverable 4.1. Additional flow solvers are included to give an idea how to add the calculation and output of the required fields. Also an additional particle solver is offered, which solves a monodisperse nanoparticle model and applies diffusion random walk for more realistic future linking and coupling. As the OpenFOAM CFD-library is a huge and poorly documented collection of algorithms applied in CFD calculations, developing software which implies deeper changes is time consuming and of experimental nature. Therefore, the provided algorithm will change in the future, in order to provide a more robust and cleaner software package.

3.1 Application nanoStreamlineFoam

The OpenFOAM solver *nanoStreamlineFoam* is the main deliverable of work package 4.2 and the main class in the scope of linking with the OpenFOAM CFD-library. For reading and initializing the required volume fields in the OpenFOAM format, the standard libraries provided by OpenFOAM are used. The major object for generating, sampling and saving the streamlines in XML-format is the *StreamCloud* object, which is included via *basicStreamCloud.H*. The *StreamCloud* class includes and manages the generation of the single streamlines by applying the *StreamParcel* class. Furthermore, it includes and controls the XML-sampling and -writing class *XMLSampler*, which is capable to reconstruct and save the streamlines in parallel runs. For constructing and saving the XML-structure *XMLSampler* wraps the *tinyxml2* library.

3.2 Application rhoNanoNonreactiveFoamTP

The main class *rhoNonNonreactiveFoamTP* is a parallel development of the main deliverable for work package 4.2 and is still in development. It basically works like the aforementioned solver *nanoStreamlineFoam*, but uses the *MonteCarloCloud* class for generating tracking data. Inside the *MonteCarloParcel* class, which is embedded in the *MonteCarloCloud* class, a monodisperse nanoparticle method is implemented. This is done in order generate more realistic tracking data with the implemented diffusive random walk model and when applying thermophoresis models – what will be done in the neat future. The sampling and output of the tracking data is realized by the *NanoDomeOutput*, which is derived from the *CloudFunctionObject* class and can be chosen by the user at in the settings via the OpenFOAM run time selection mechanism.

Since this additional tool is still in development, it's usage will not be guided in this document.

3.3 Applications rhoNanoFlowFoam/rhoNanoFlowDyMFoam

This additional solvers were used to test the aforementioned *linking* algorithms in a laminar reacting flow field. An additional conservation equation for the precursor concentration and it's decay is implemented in the header file *CEqn.H*, which is included in the main class. The solver should give an idea, how to construct or change a solver in order to calculate the required quantity fields for applying the *linking*. The *rhoNanoFlowDyMFoam* solver includes an automatic grid refinement in regions of high reaction source terms.

Since this additional solvers are used like basic OpenFOAM CFD-solver, it's usage will not be guided in this document. Users are referred to the OpenFOAM user's manual.

4 Compilation and Utilization Details

For compiling the libraries and solvers, it is mandatory to first properly compile the OpenFOAM CFD-library in version 2.3.x on a Linux system. Only when all OpenFOAM features are given, the provided extensions can be compiled and used.

4.1 Compilation

To compile the source code in the OpenFOAM environment the nanoFoam directory can be copied to any place within the users directory. It is recommended to use the OpenFOAM users directory *username-2.3.x*, which is generally placed in the basis OpenFOAM directory. Then create the *LIB_NANO_SRC* environment variable by adding following line to your .profile (placed in the home directory) `'export LIB_NANO_SRC=$WM_PROJECT_USER_DIR/nanoFoam/src'`. In case the nanoFOAM directory is placed in the user directory, this path is correct, if any other directory is chosen, the part '*\$WM_PROJECT_USER_DIR*' has to be replaced by the correct one. When this is done, following line has to be executed from the home directory: `'source .profile'`.

Now, enter the nanoFoam directory and execute the *wmakeNanoStreamline* shell script for compiling all necessary data for using the *nanoStreamlineFoam* solver. For compiling the additional solvers compile *wmakeAddSolver*, too.

4.2 Utilization

The *nanoStreamlineFoam* solver is applied in the same case directory as the CFD-simulation was conducted before. It is necessary to prolong the simulation by a time frame, which is at least equal to the resistance time of the gas. Furthermore it is necessary to setup the streamline generation and tracking by adding the *stream-CloudProperties* file in the *constant* directory of the case directory. An example setup file is provided in the *exampleSetupFile* directory in the *nanoFoam* base directory.