EDF R&D

Fluid Dynamics, Power Generation and Environment Department
Single Phase Thermal-Hydraulics Group

6, quai Watier
F-78401 Chatou Cedex

Tel: 33 1 30 87 75 40
Fax: 33 1 30 87 79 16

SEPTEMBER 2017

Code_Saturne documentation

Code_Saturne version 4.0 tutorial: Turbulence simulation in a mixing tee

contact: saturne-support@edf.fr

TABLE OF CONTENTS

Chapter I

Introduction

1 Tutorial components

This tutorial makes use of:

- The SALOME [1] platform for geometry generation, meshing, and post-processing
- *Code_Saturne* [2], [3] for CFD calculations
- Reference [4] for comparison with published results

To work through this tutorial you will need a computer on which these two software applications are already available or on which you have permission to install them.

2 Tutorial structure

In PWR plants, thermal fatigue can occur in energy cooling systems which are subjected to cyclic stresses. These cyclic stresses are generally found in T-Junctions where cold and hot water streams mix and the resulting turbulent fluctuations create thermal fluctuations at the walls which can lead to thermal fatigue.

This tutorial focuses on modelling such a T-Junction, using the RANS and the LES approach and is composed of five complementary parts:

- Part 1 (Section II) presents the T-Junction case experimental set-up.
- Part 2 (Section III) illustrates how to create the geometry in blocks, how to create groups and how to mesh the T-junction.
- Part 3 (Section IV) shows how to set-up the RANS case in *Code_Saturne* and the results of the RANS simulation compared with experimental data [4].
- Part 4 (Section V) presents the results of the RANS simulation and compares them with experimental data.
- Part 5 (Section VI) describes how to set-up a LES case.

3 What you will learn

Through this tutorial, you will learn how to:

- Create a computational domain and a hexahedral mesh for a T-junction with SALOME.
- Setup an unsteady, viscous, turbulent, CFD simulation with variable fluid properties and no-slip walls, using *Code_Saturne*
- Program user coding to:
  - Compute user defined variables and extract quantities of interest for post-processing
  - Model the inlet turbulence for the LES computation using the Synthetic Eddy Method
  - Compute the time averages of variables for the LES simulation
- Control and run the *Code_Saturne* simulations from the CFDStudy module
- Examine the *Code_Saturne* output and results files, including data along specified profiles and at monitoring points
- Visually and quantitatively compare the predicted results with experimental data

Chapter II

Part 1 - Vattenfall T-Junction Benchmarking Experiment

1 Case description

The OECD/NEA-Vattenfall T-Junction benchmark [4] was initiated to test the ability of state-of-the-art CFD codes to predict the important parameters affecting high-cycle thermal fatigue in mixing tees or T-Junctions. The experiments were carried out at the Älvkarleby Laboratory at the Vattenfall R&D facility in Sweden.

The test rig used to complete the experiments is made from Plexiglas® and consists of two main pipe sections mounted horizontally upstream and downstream of a T-Junction with a branch pipe mounted vertically above this junction. This set-up is illustrated in Figure II.1.

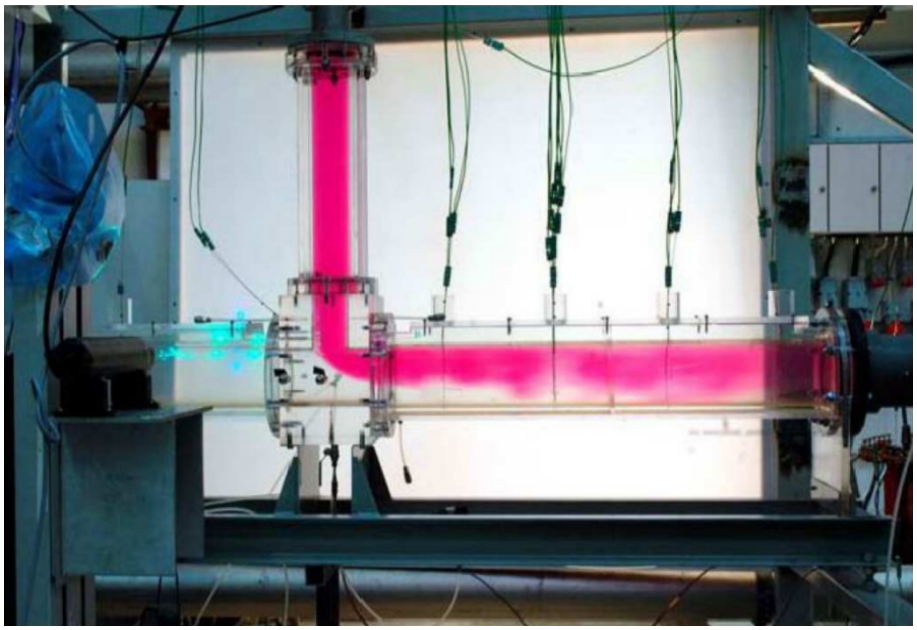


Figure II.1: Photograph of the Vattenfall test section [4].

The working fluid is deionised tap water with cold water flowing in the main horizontal pipe which is fed via a long inlet pipe from a high level reservoir. Hot water is pumped into the vertical pipe via a shorter inlet section and the two fluid streams mix in and downstream of the T-Junction.

The test rig dimensions are described next.

2 Test rig dimensions

The horizontal and vertical pipes in the experiment have differing diameters and lengths. The horizontal pipes upstream and downstream of the T-Junction have an internal diameter of 140mm and a length of 1070mm. The vertical pipe has an internal diameter of 100mm and a length of 470mm. These dimensions, as well as those of the T-Junction, are shown in Figure II.2.

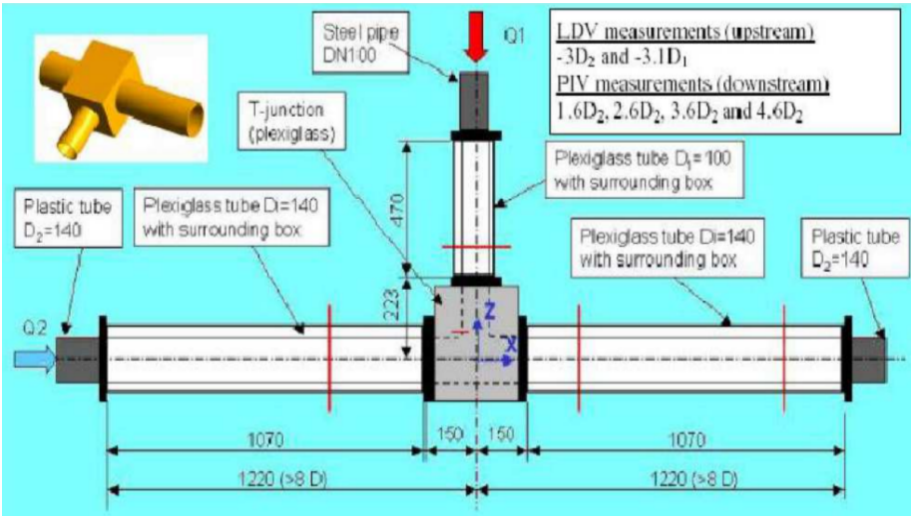


Figure II.2: Schematic of the mixing Tee in Plexiglas [4].

The flow physics are described next.

3 Flow physics

The flow physics involve the mixing of hot and cold fluid streams in a T-junction. As can be seen in Figure II.1, the flow field is steady in the pipes upstream of the T-junction but unsteady downstream of the meeting point of the hot and cold streams due to the mixing process of the two streams. The mixing process produces complex flow patterns, with non-uniform temperatures that will have an impact on local wall temperatures. The dominant physics is forced convection with unsteady mixing in and downstream of the T-junction.

The test rig operating conditions are described next.

4 Operating conditions

The test rig operating conditions are as follows:

- The cold water is entering the test section at a temperature of $19.0°C$ and with a velocity $\vec{U} = (0.585, 0.0, 0.0)$
- The hot water is entering the test section with a temperature of $36.0°C$ and with a velocity $\vec{U} = (0.0, 0.0, -0.764)$

The fluid properties are described next.

5 Fluid properties

The difference in temperature between the cold and hot water streams is approximately $17°C$. This temperature difference is sufficient to result in property variations in the fluid streams. Table II.1 presents the approximate temperature dependence of the fluid properties of water [4].

$T(°C)$	$\rho(kg/m^3)$	$k(W/mK)$	$\mu(PA.s)$	$C_p(J/kg.K)$	$Pr(-)$
15	999.2	0.5911	1.1380×10^{-3}	4186	8.058
20	998.3	0.5996	1.0020×10^{-3}	4182	6.988
25	997.2	0.6076	0.8904×10^{-3}	4179	6.125
30	995.8	0.6151	0.7977×10^{-3}	4178	5.419
35	994.1	0.6221	0.7196×10^{-3}	4178	4.833
40	992.3	0.6287	0.6533×10^{-3}	4178	4.342

Table II.1: Variation of water properties with temperature.

5.1 Geometry

The schematic of the computational domain is presented in Figure II.3 below.

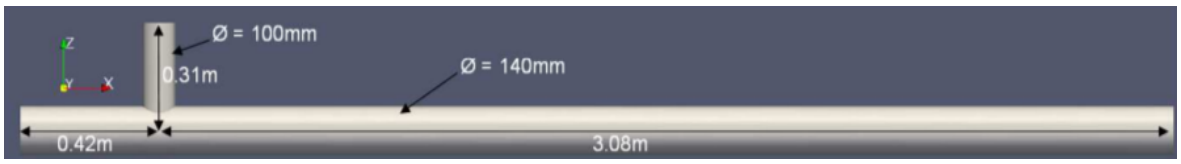


Figure II.3: Schematic of the mixing Tee.

As per the experimental setup, the horizontal pipe has a diameter of 140mm and the vertical pipe a diameter of 100mm. The reference frame is located at the cross section of the two pipes. The cold and the hot inlets are respectively located at -0.42m and 0.31m from this reference frame. The outlet is situated at 3.5m from the cold inlet.

5.2 Boundary conditions

Four boundary conditions are used in this study:

- Cold inlet : $\vec{U}$ = (0.585, 0.0, 0.0) with a temperature of 19.0°C
- Hot inlet : $\vec{U}$ = (0.0, 0.0, -0.764) with a temperature of 36.0°C
- Outlet : The standard outlet condition is used
- Walls : The wall boundary are assumed to be no-slip, smooth and adiabatic

6 Creating the *Code_Saturne* Study and Cases

A *Code_Saturne* study called *Mixing_Tee* and two cases are created. One for the first set of calculations, which we call *RANS*, and the other for the LES simulation, which we call *LES*.

The study and the cases are created using the procedure described in Part I of tutorial 1 [5]. Start SALOME with the command *code_saturne salome*, select the CFDStudy module, and go through all the steps detailed in [5] to:

- Create the CFD case structure with the CFDStudy module
- Save the new file as 'Mixing_Tee'
- Customise the background settings

Once the study and cases have been created, you should end up with the directory structure (Figure II.4) shown in the Object Browser tab, displaying the study and the two cases.

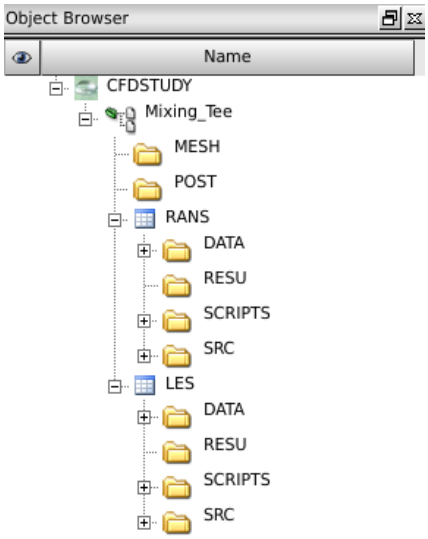


Figure II.4: *Mixing_Tee* Study, *RANS* and *LES* case File Structure

The cases are then ready to be set up.

Chapter III

Part 2 - Creating the Computational Domain

1 What you will learn

In this part of the tutorial you will use the geometry, GEOM, and meshing, MESH, modules of SALOME to create the computational domain. Through the different steps, you will learn to:

- Create a decomposition in blocks of the geometry in order to have a conformal hexahedral mesh by executing basic operations
- Create groups in order to prepare the meshing process and the setting-up of the boundary condition
- Specifiy different mesh sizes by applying different laws to mesh your domain.

If you wish you can skip this part by executing the script supplied in Appendix 2 to obtain directly the final mesh with the groups.

2 Creating the Geometry

The method chosen to build the T junction geometry makes it possible to create different meshes which can be used either for RANS or LES calculations by modifying only the refinement parameters. The geometry is made of a butterfly decomposition in 8 parts for both the inlet and the outlet. The final geometry is shown in Figure III.1 and Figure III.2.

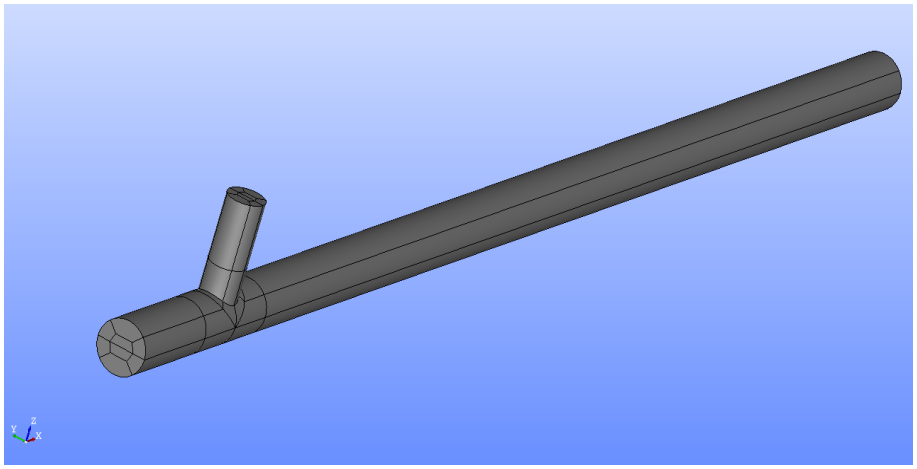


Figure III.1: View of the final geometry with block decomposition.

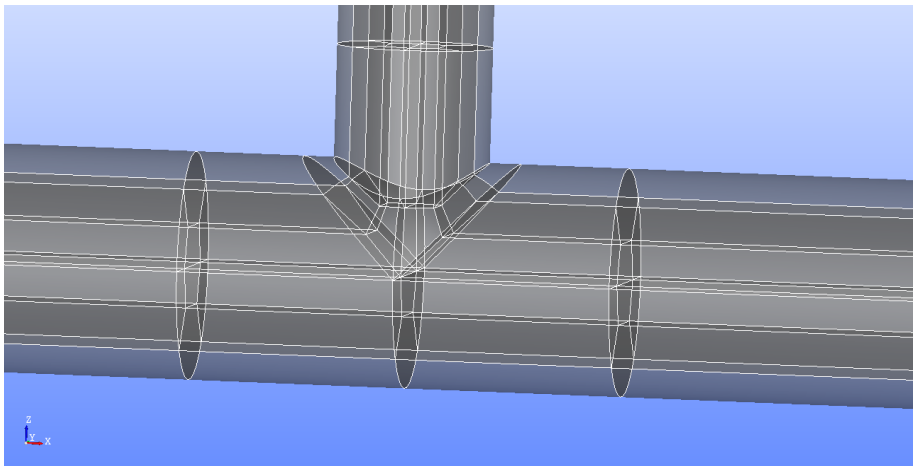


Figure III.2: Zoom in transparency of the T junction.

The process of creating this geometry is presented step by step in the sections below. First, only half of the geometry will be build. Then, in the final step the other half will be build by symmetry using a mirror operation.

2.1 Inlet Plane

The first step consists in creating every quadrangle faces which will make the inlet in to order to later have generate a good butterfly mesh. The final decomposition of the inlet is presented in Figure III.3.

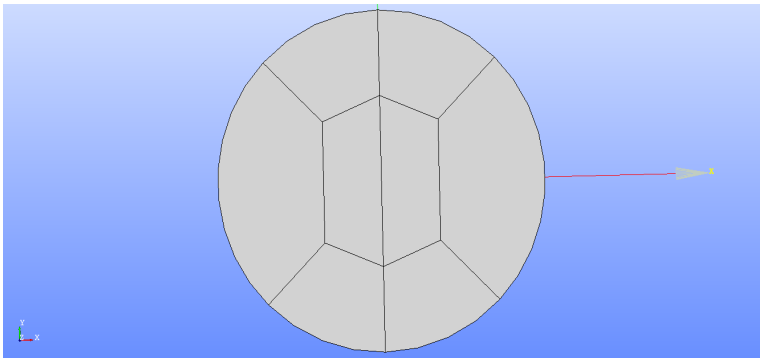


Figure III.3: View of the inlet made of 8 parts.

Start by creating two vertices (***New Entity*** ⇒ ***Basic*** ⇒ ***Point***) with the following coordinates: *Vertex 1* (0,0.05,0.14) and *Vertex 2* (0,0.025,0.14).

Then, join the two vertices with a line (***New Entity*** ⇒ ***Basic*** ⇒ ***Line***) and rotate the line (***Operations*** ⇒ ***Transformation*** ⇒ ***Rotation***) three times around the Z axis with an angle of $45°, 135°$ and $180°$.

Create a third vertex, *Vertex 3*, at (0,0,0.14), and name it "center_inlet". Create three arcs (***New Entity*** ⇒ ***Basic*** ⇒ ***Line***) linking the exterior vertices of the four straight lines by selecting the second option in the ***Arc Construction*** window. Create the arcs using the vertex "center_inlet" as the centre point and, for each, a start point and an end point corresponding to the exterior vertices of the four lines.

To complete the structure of half of the inlet plane, create the 3 remaining lines corresponding to

half of the hexagon at the center of the inlet as shown in Figure III.4.

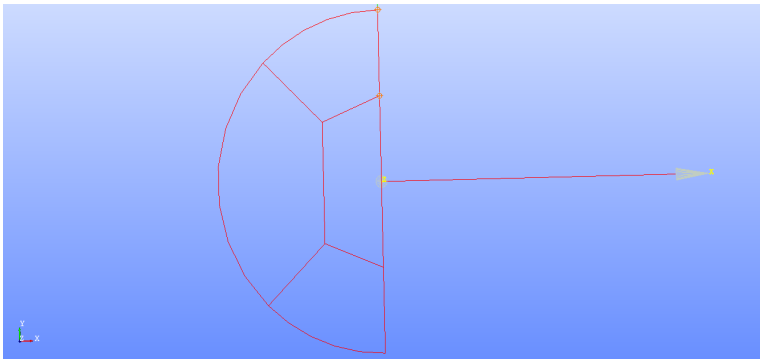


Figure III.4: Building the inlet pattern.

Now, build the four faces (***New Entity*** ⇒ ***Build*** ⇒ ***Face***) by selecting the 4 edges corresponding to each face.

As the entire inlet will be used to do a scaling operation in part 2.3, its second half is created now. To construct the other half of the inlet, you will use a mirror operation. First, generate the face which will act as a mirror plane. Build this face with an "OYZ" orientation, ***New Entity*** ⇒ ***Primitives*** ⇒ ***Rectangle***.

Then perform a mirror operation (***New Entity*** ⇒ ***Transformation*** ⇒ ***Mirror Image***). In the ***Mirror Image*** menu, select the third option, add the four faces in the "Objects" field, and the mirror face you have just created in the "Plane Mirror" field.

Upon executing the mirror operation, you should now see the 8 faces, as shown in Figure III.3.

2.2 Vertical Pipe

Select the four faces corresponding to half of the inlet, created in 2.1, and perform an extrusion (***New Entity*** ⇒ ***Generation*** ⇒ ***Extrusion***) along the Z axis with the height of -0.14.

Create a disk (***New Entity*** ⇒ ***Primitives*** ⇒ ***Disk***) with a radius of 0.07 and an "OYZ" orientation. Do an extrusion (***New Entity*** ⇒ ***Generation*** ⇒ ***Extrusion***) in both direction of this disk along the X axis with the height of 0.42. Name this extrusion "horizontal_pipe_tool".

Then do a cut operation, (***New Entity*** ⇒ ***Boolean*** ⇒ ***Cut***), between "horizontal_pipe_tool" and the extrusion of the four faces. The result of this operation is presented in Figure III.5.

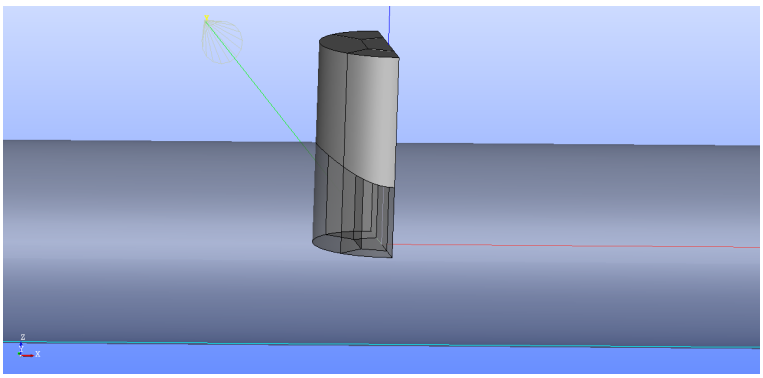


Figure III.5: Visualization of the cut operation (in opaque) between the cylinder and the extrusion of the four faces (both in transparency).

The previous operation allow to get the edges of the junction between the vertical pipe and the horizontal pipe with an explode operation (***New Entity*** ⇒ ***Explode*** ⇒ ***Edges***) on the result of the cut operation. You can visualize the edges belonging to the vertical and horizontal pipe in Figure III.6 (curves in red).

Do an extrusion of the face corresponding to the half of the inlet hexagon and build three lines between the vertices of the extrusion and the junction edges as shown in Figure III.6.

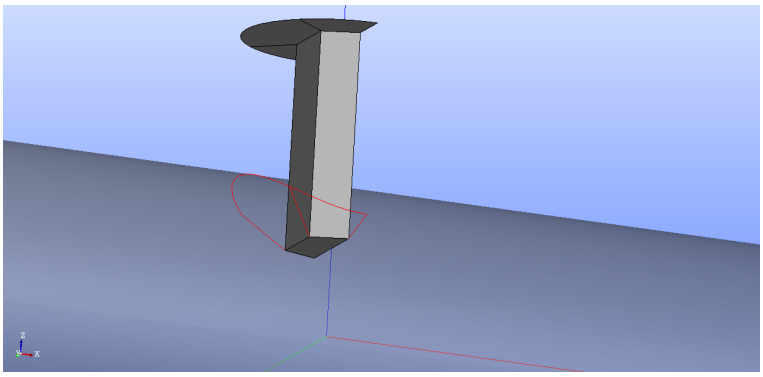


Figure III.6: Visualization of the joining edges.

In order to create the two solids presented in Figure III.7. Create the remaining lines between the inlet and the juction area and build the faces by selecting the edges created and the arcs belonging to the junction. Create also two shells, (***New Entity*** ⇒ ***Build*** ⇒ ***Shell***), each containing six faces corresponding to the two solids. Then create the two solids, (***New Entity*** ⇒ ***Build*** ⇒ ***Solid***), from the shells previously created.

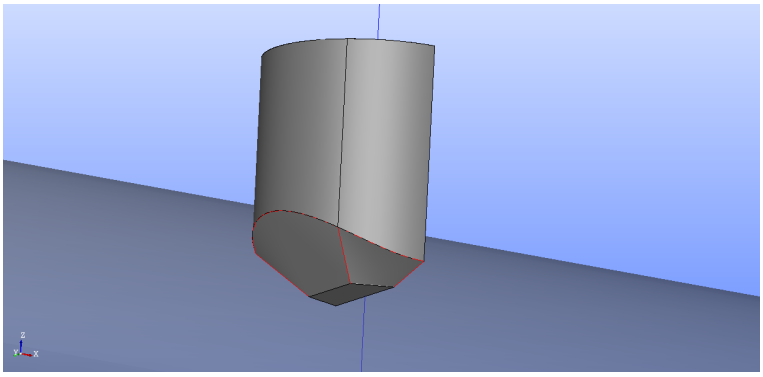


Figure III.7: Creating the two solids.

One solid of the vertical pipe is still missing and can be built with a mirror operation in the plane "OXZ" by selecting the right solid from the two previously created as shown in Figure III.8.

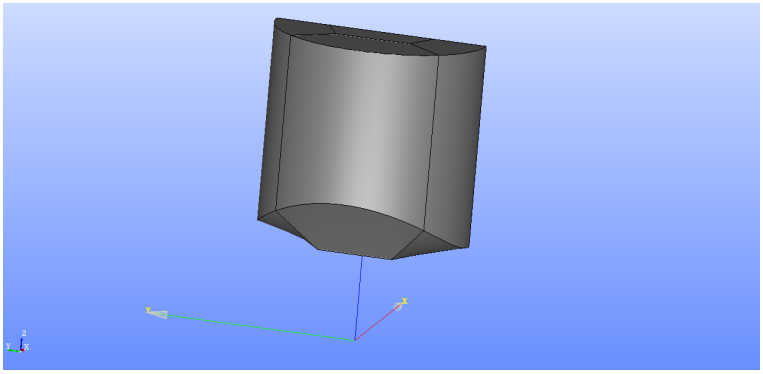


Figure III.8: Half of the vertical pipe.

2.3 Building half of the horizontal pipe

Do a scaling operation (***Operation*** ⇒ ***Transformation*** ⇒ ***Scale Transform***) of the 8 faces which made the inlet created in part 2.1. Select the 8 faces in the "Objects" field, select also the *Vertex 3* (build in part 2.1) in the "Central Point" field and set 1.4 in the "Scale Factor" field. Name the result of this scalling operation "Face_scaled". The Figure III.9 shows in red the original 8 faces and in grey the result of the scaling operation.

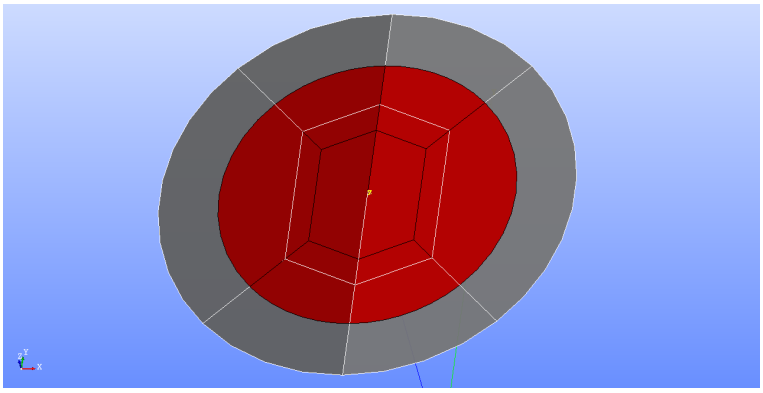


Figure III.9: Scaling operation.

Next do a translation operation (***Operation*** ⇒ ***Transformation*** ⇒ ***Translation***) of "Face_scaled" to the origin and then rotate at 90° degrees the result of the translation around the axis Y. Finally do an extrusion of this face rotated along the X axis with the height parameter set at -0.14, name this extrusion "Extrusion_pipe_hori". The result is shown in Figure III.10.

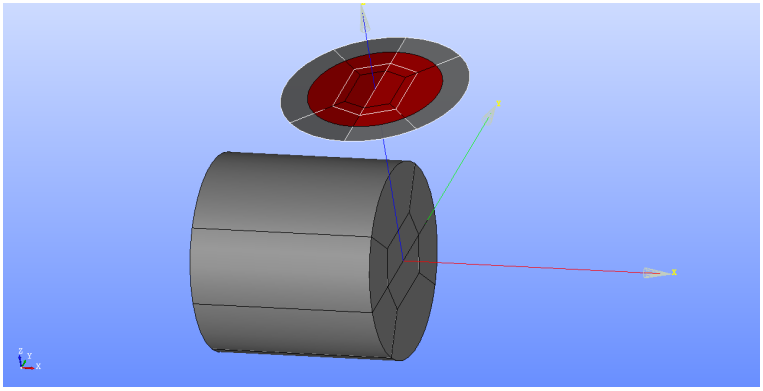


Figure III.10: Extrusion of the "Face_scaled" translated and rotated.

Create a box (***New Entity*** ⇒ ***Primitives*** ⇒ ***Box***) with a dimension of 0.21 for "Dx ,Dy and Dz". Do a translation of -0.105 of this box in the Y direction. Then rotate it around the Y axis with an angle of $-45°$ degrees.

Do a common operation, (***New Entity*** ⇒ ***Boolean*** ⇒ ***Common***), between the box and "horizontal_pipe_tool" as presented in Figure III.11. Name the result of the common operation "Common_junction".

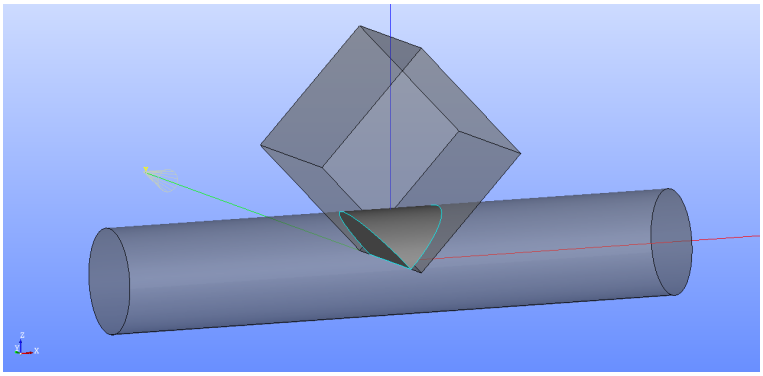


Figure III.11: Visualization the result of the common operation (in opaque) between the box and the cylinder (both in transparency).

Finally realize a cut operation between the "Extrusion_pipe_hori" and "Common_junction". You have now half of your horizontal pipe as shown in Figure III.12.

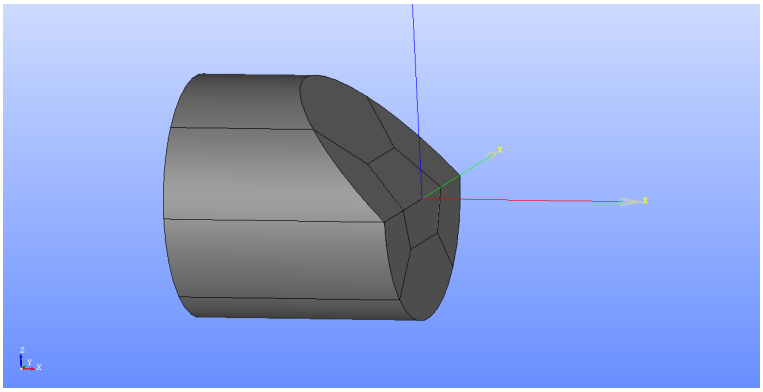


Figure III.12: Horizontal pipe

2.4 Building the junction part

Realize a cut operation between the "Common_junction" and the vertical part realized in 2.2. Name the result of this operation "Common_mid_cut", it can be visualize in Figure III.13.

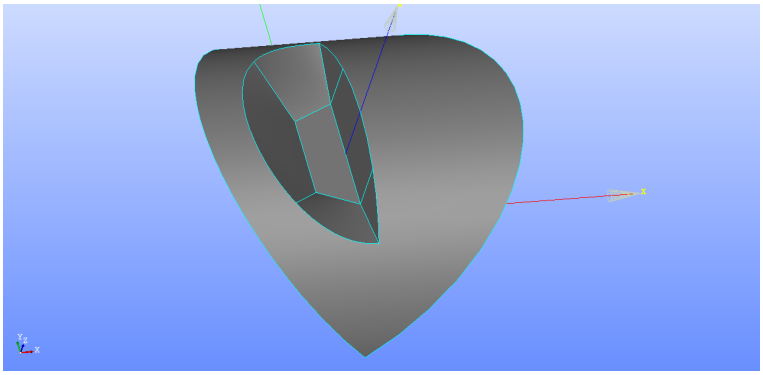


Figure III.13: Result from the cut operation between the vertical part and "Common_junction".

Create a cylinder (***New Entity*** ⇒ ***Primitives*** ⇒ ***Cylinder***) with a radius of 0.1, activate the "Angle" option and set an angle of 45 degrees. Rotate this cylinder (to get the side aligned with the "OYZ" plane) as shown in Figure III.14 and do a common operation between the cylinder and "Common_mid_cut".

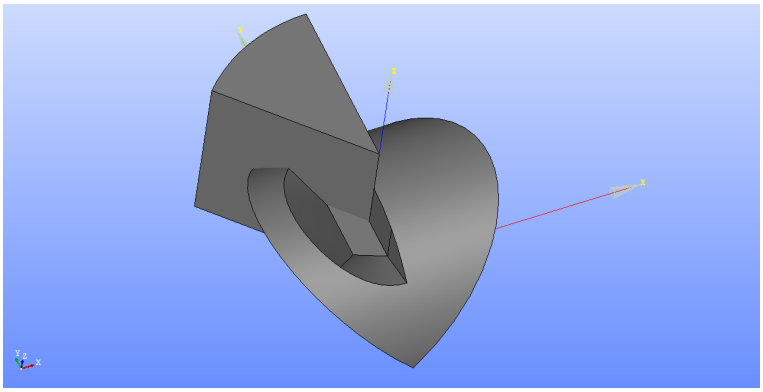


Figure III.14: Common operation between "Common_mid_cut" and a cylinder.

Repeat the common operation with a cylinder of 90 degrees as you can visualize in Figure III.15.

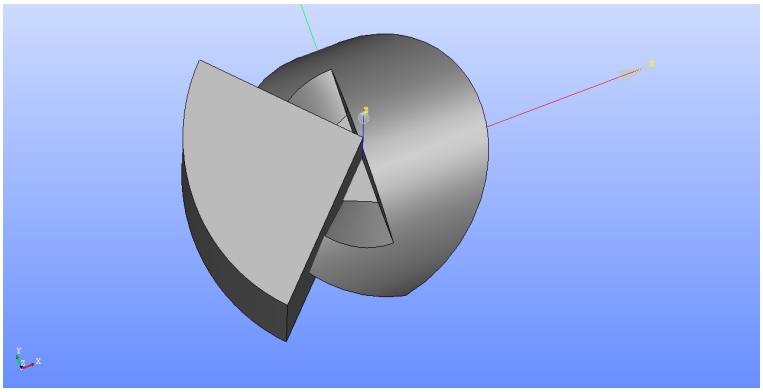


Figure III.15: Common operation between "Common_mid_cut" and a cylinder.

Finally, repeat once again the common operation between "Common_mid_cut" and a cylinder with an angle of 45 degrees. You can visualize the result of the three common operations in Figure III.16.

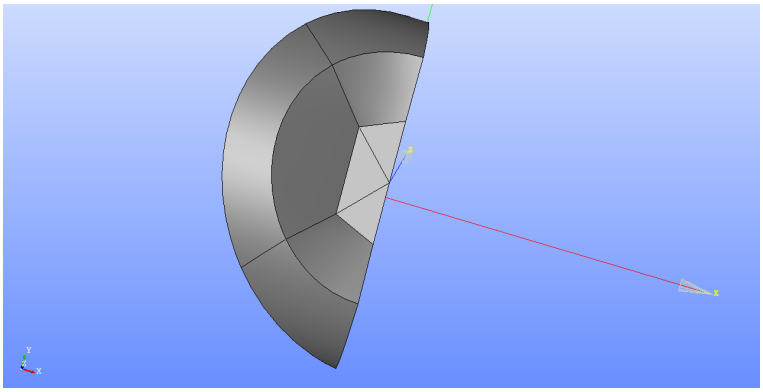


Figure III.16: Results of the different common operations.

Display the vertical part realized in 2.2 and the horizontal part realized in 2.3. Then create the solid between the two red faces in Figure III.17, by creating the lines, faces, shell and solid.

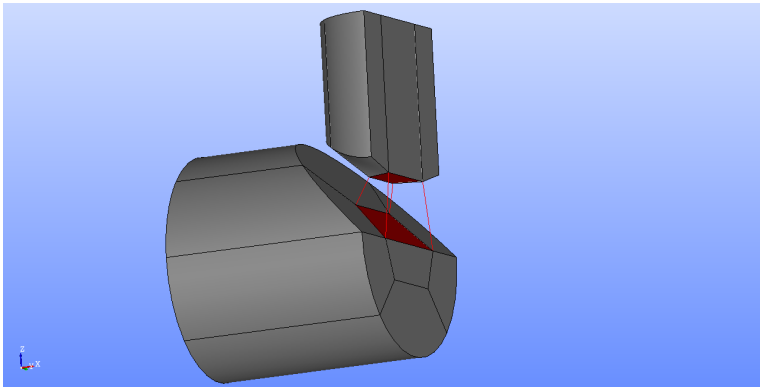


Figure III.17: Creation of a solid linking the vertical and horizontal part.

You can see the solid created in Figure III.18.

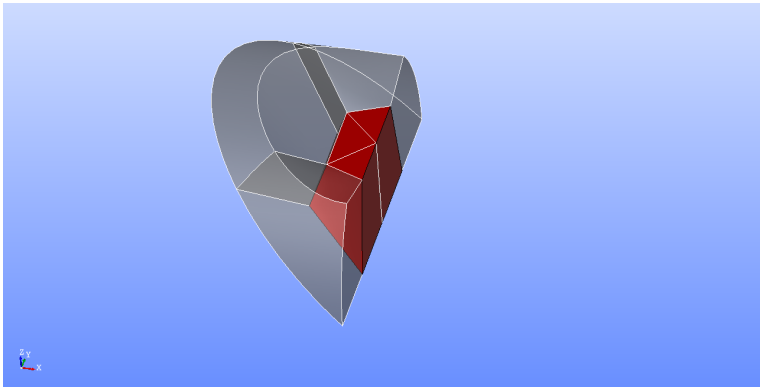


Figure III.18: Solid created (in red) with the three common parts in transparency

Do a cut operation between the solid and the three common parts (Figure III.19).

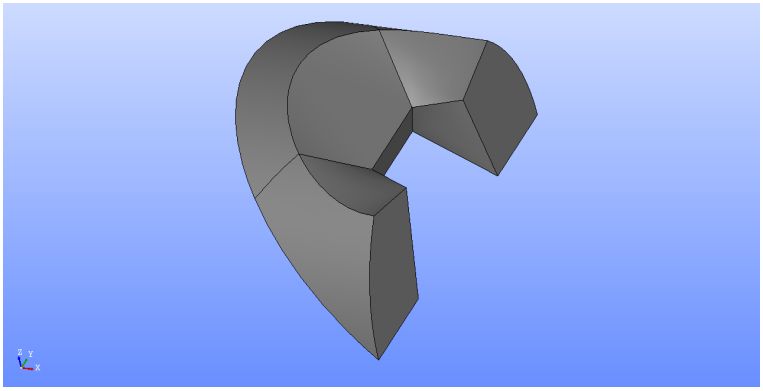


Figure III.19: Result of the cut operation

You have now obtained your junction part, Figure III.20.

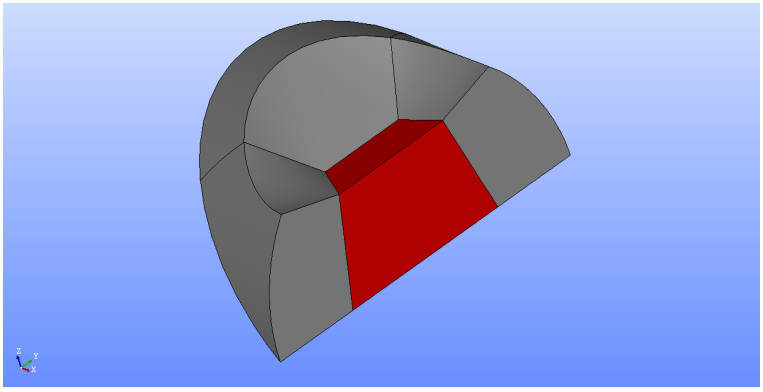


Figure III.20: Visualization of the junction part.

2.5 Building the remaining geometry

Display the 3 geometry created in 2.2, 2.3, 2.4 as shown in Figure III.21.

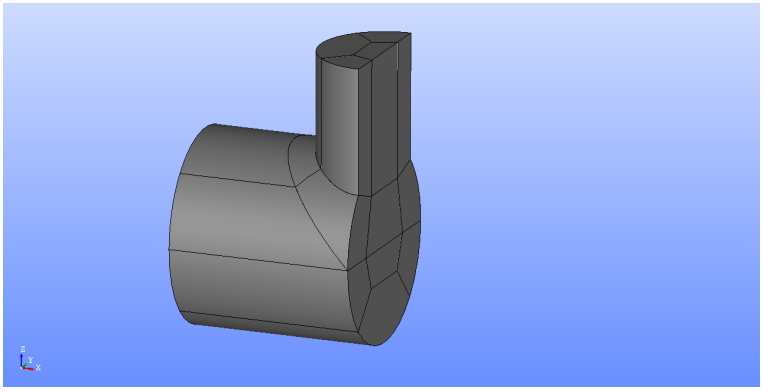


Figure III.21: Visualization of the 3 parts created.

Realize a mirror operation in order to get the same result as shown in Figure III.22.

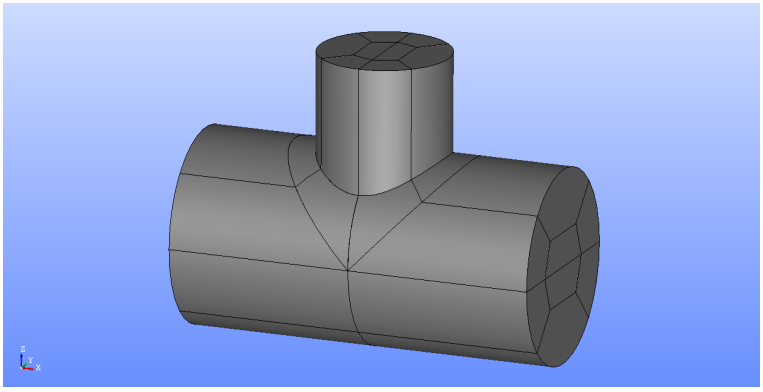


Figure III.22: Visualization of the T junction.

Finally in order to get the real geometry, realize 3 extrusion operations by selection the faces of each extremity. Do an extrusion with the height of -0.28 along the X on the left side according to Figure III.23, a vertical extrusion along the Z axis with the height of 0.17 and an extrusion along the X axis with the height of 2.94.

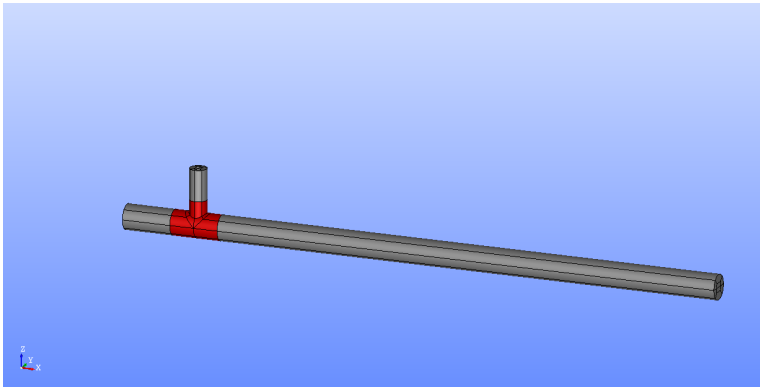


Figure III.23: Geometry of the T junction pipe.

The last step consist in suppressing the unnecessary edges and faces of the geometry. In order to to that SALOME possessed two useful tools. Create a compound of the different part of your geometry

to have only one object for the whole geometry. Then realize a glue edges operation of your compound with a precision of 1×10^{-4} (***Repair*** $\Rightarrow$ ***Glue Edges***). Next do a glue faces operation with a precision of 1×10^{-4} on the glue edges result (***Repair*** $\Rightarrow$ ***Glue Faces***).

Finally if you have followed all the steps carefully you should obtain the same result as Figure III.24 by doing ***Mesure*** $\Rightarrow$ ***What is***.

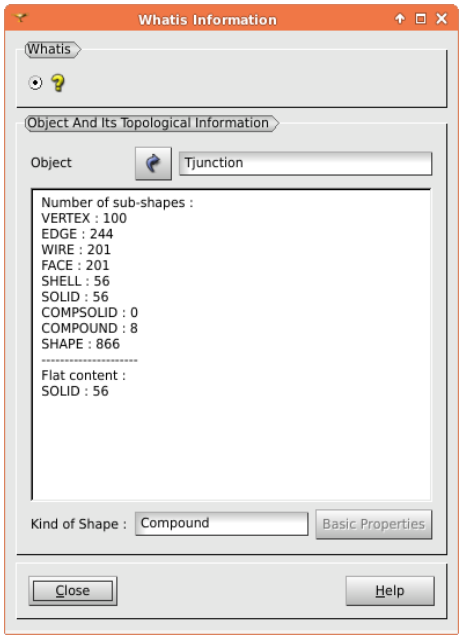


Figure III.24: Features of the geometry.

3 Creating groups

In order to set up the boundary conditions later, it is necessary to create several groups of faces.

3.1 Creating groups of faces

To create groups of faces, right click on your geometry in "Object Browser" then click on "Create group".

In the "Create Group" pop-up window, select "faces" in "Shape type", and name and select your faces for the horizontal pipe's inlet, *inlet_1*, Figure III.25, the vertical pipe's inlet, *inlet_2*, Figure III.26, the horizontal pipe's outlet, *outlet*, Figure III.27 and, lastly, the wall boundayr, *wall*, Figure III.28. To select several faces keep the shift key pressed.

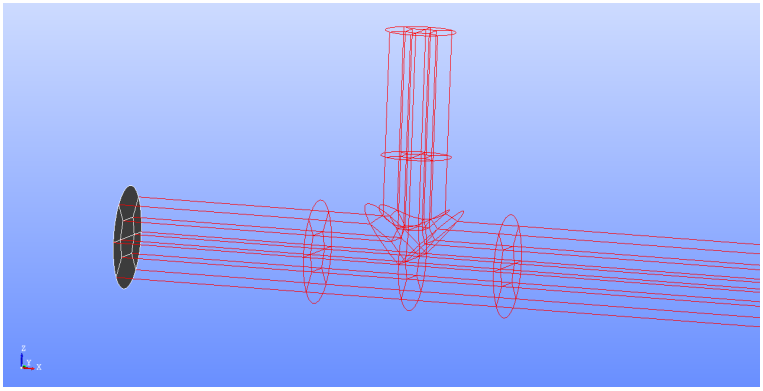


Figure III.25: inlet_1.

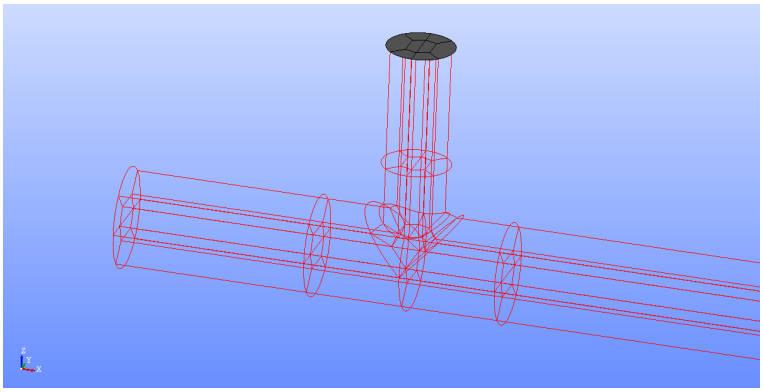


Figure III.26: inlet_2

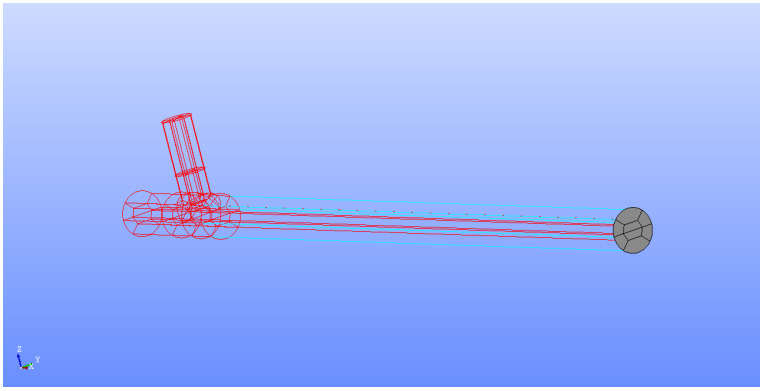


Figure III.27: outlet.

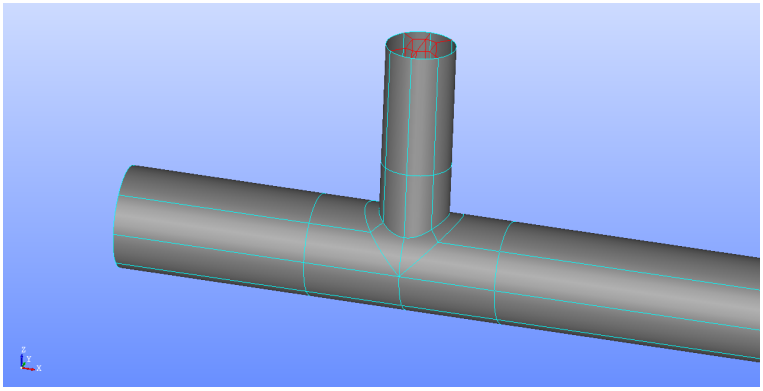
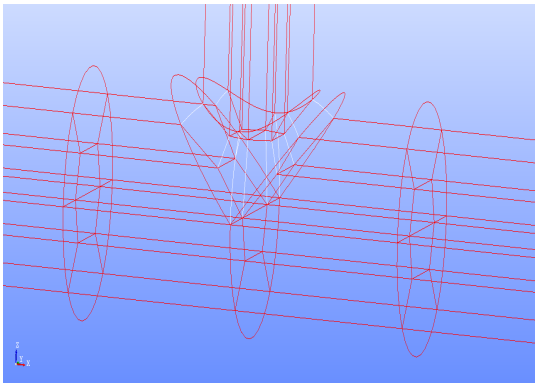


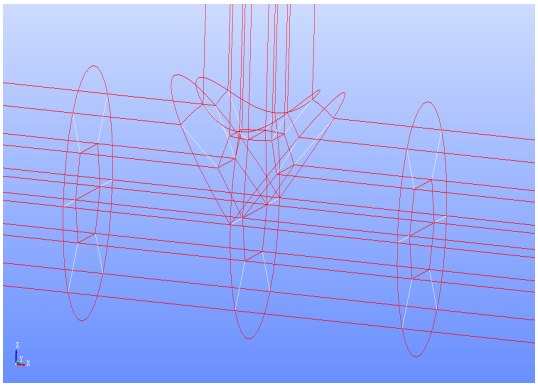
Figure III.28: wall.

3.2 Creating group of edges

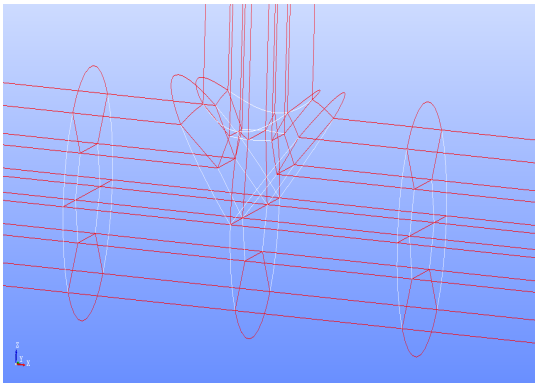
In order to mesh the computational domain easily it is necessary to create several groups of edges. To create a group of edges, right click on your geometry in "Object Browser" and then click on "Create group". A pop-up window called "Create Group" appears, select edges in "Shape type", name and select your groups as shown in Figure III.29a to III.29p. To select several edges keep the shift key pressed.



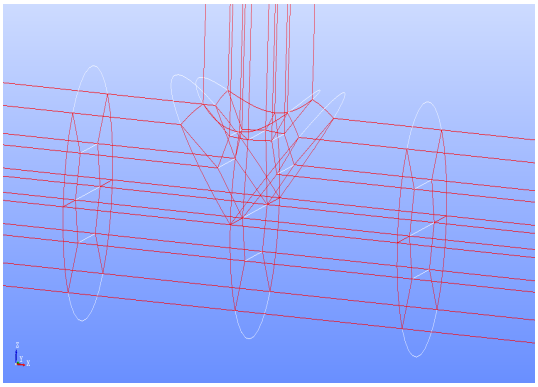
(a) groupe_h_mid



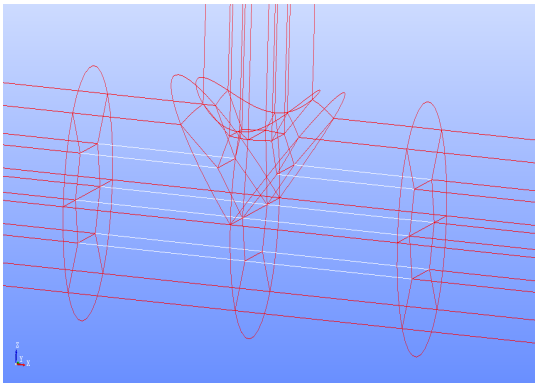
(b) groupe_cote



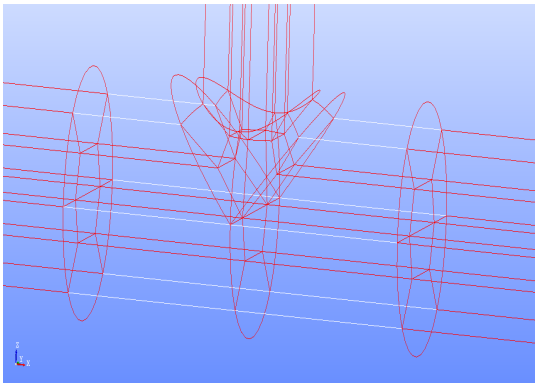
(c) groupe_petit_arc



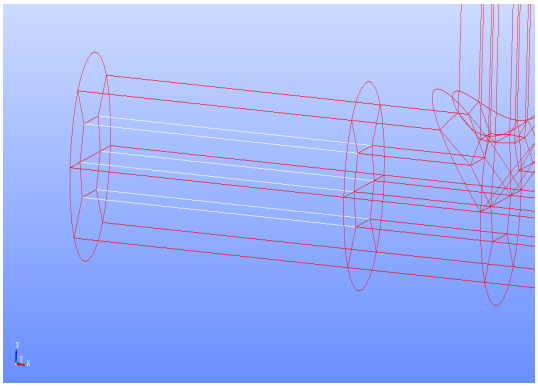
(d) groupe_grand_arc



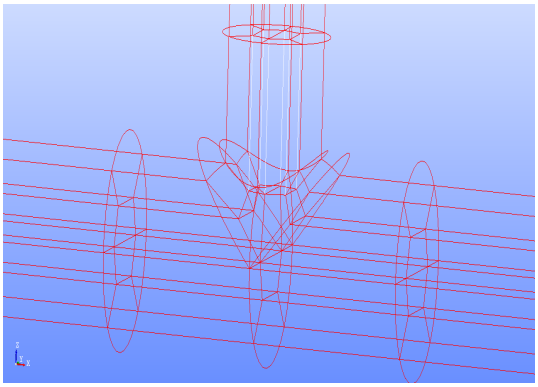
(e) groupe_proff_mid_hori_int



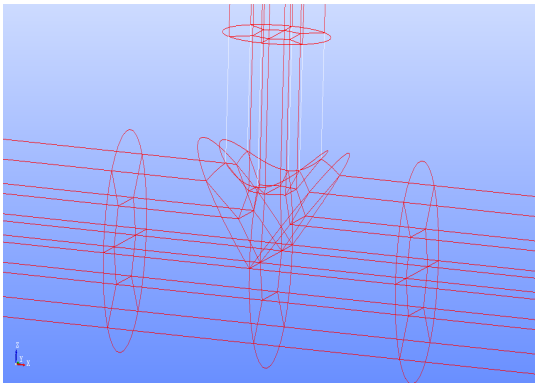
(f) groupe_proff_mid_hori_ext



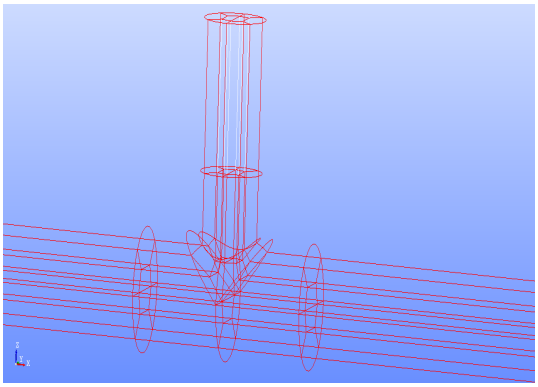
(g) groupe_proff_small_int



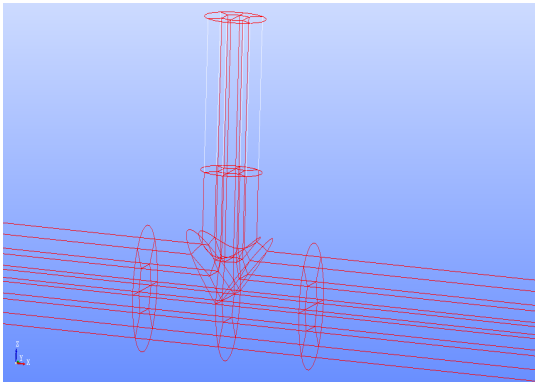
(h) groupe_proff_mid_verti_int



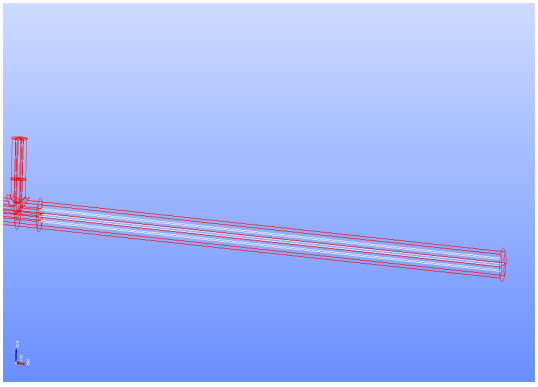
(i) groupe_proff_mid_verti_ext



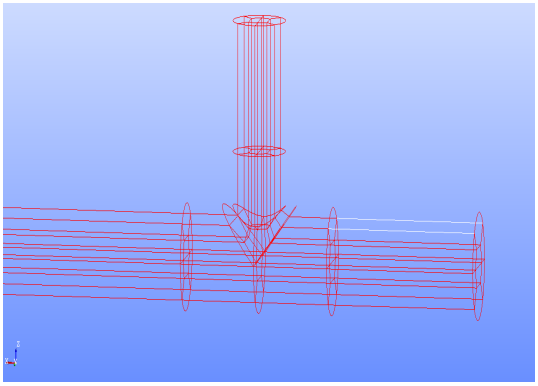
(j) groupe_proff_verti_int



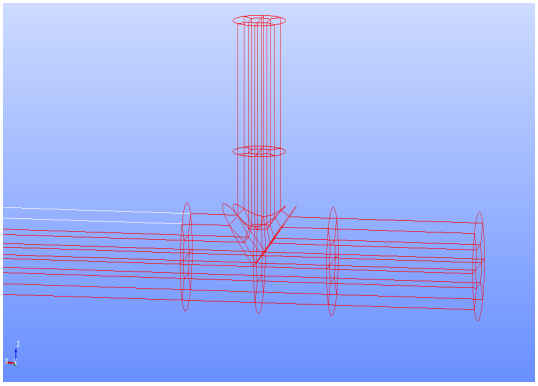
(k) groupe_proff_verti_ext



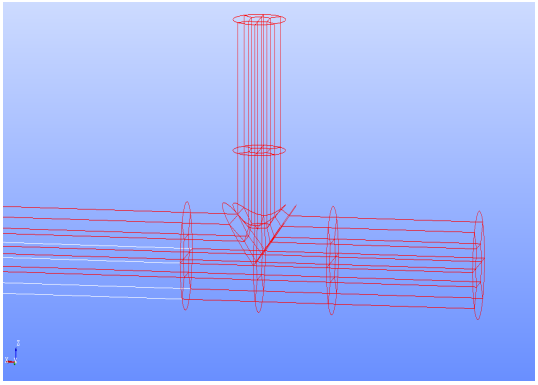
(l) groupe_proff_long_int



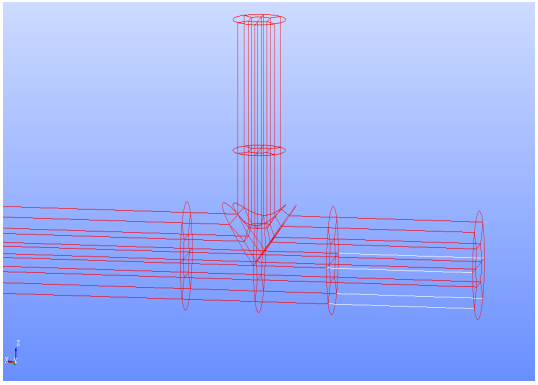
(m) proff_small_ext_top



(n) proff_long_ext_top



(o) groupe_proff_long_ext



(p) groupe_proff_small_ext

Figure III.29: Defining the groups of edges.

The geometry model is now ready to be meshed in the MESH module.

4 Meshing

Move to the MESH module of SALOME.

In order to keep a conform and hexahedral mesh, different sets of hypotheses are needed to discretise the different groups of edges.

First, create a mesh of your geometry: ***Mesh*** $\Rightarrow$ ***Create Mesh***. In the pop-up, click on "Assign a set of hypotheses" and choose "3D: Automatic Hexahedralization" as shown in Figure III.30. Then, press "Apply and Close".

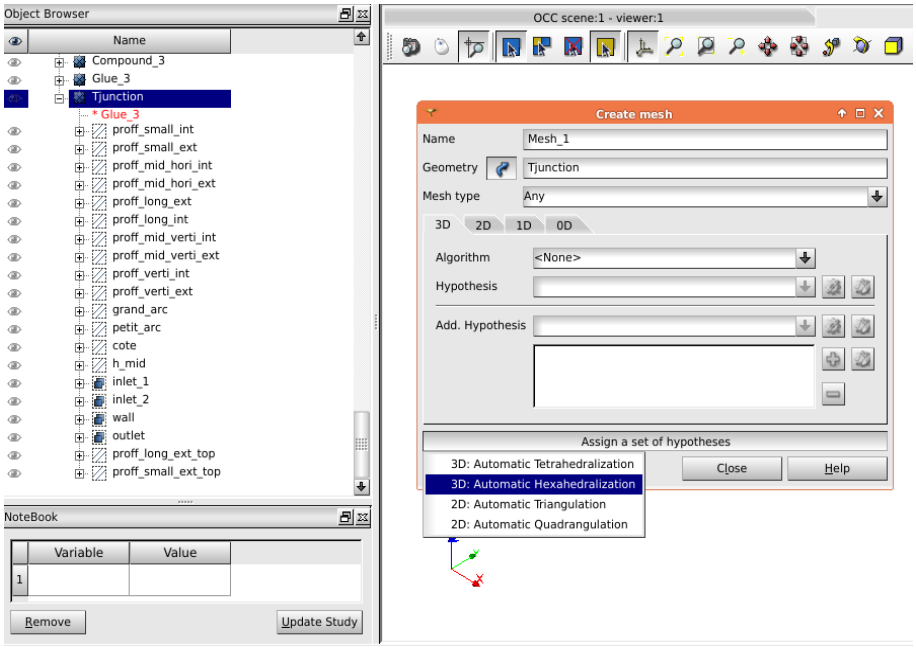


Figure III.30: Selecting the hexahedral hypothesis.

The next step is to create a sub-mesh for each group of edges. Right click on your mesh previously created, called "Mesh_1" by default, and select "Create sub-mesh". Then select a group of edges by clicking on a group in the tree of the "Object Browser" as shown in Figure III.31. Then select "Wire Discretisation" for the algorithm and select the hypothesis "Nb Segments" that you can renamed. In Figure III.31 is an example of how to proceed for the group of edges "proff_small_int".

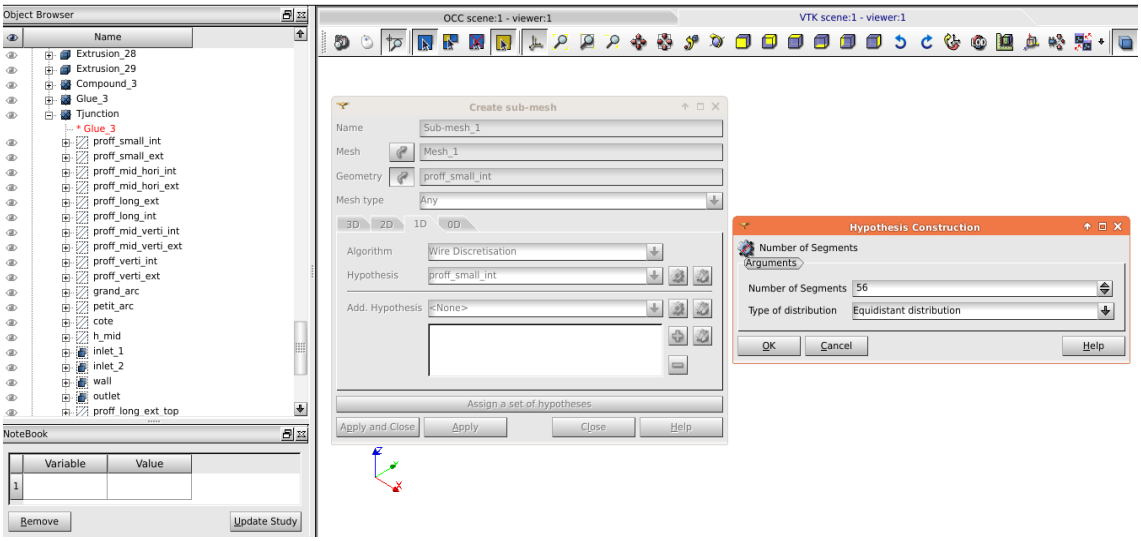


Figure III.31: Creating a Sub-mesh for proff_small_int.

Repeat the same operation for the others groups of edges with the parameters given in Table III.1.

Group	Hypothesis	Type of distribution	Nb. Segments
proff_small_int	Nb. Segments	Equidistant distribution	56
proff_small_ext	Nb. Segments	Equidistant distribution	56
proff_mid_hori_int	Nb. Segments	Equidistant distribution	28
proff_mid_hori_ext	Nb. Segments	Equidistant distribution	28
proff_long_ext	Nb. Segments	Equidistant distribution	588
proff_long_int	Nb. Segments	Equidistant distribution	588
proff_mid_verti_int	Nb. Segments	Equidistant distribution	20
proff_mid_verti_ext	Nb. Segments	Equidistant distribution	20
proff_verti_int	Nb. Segments	Equidistant distribution	34
proff_verti_ext	Nb. Segments	Equidistant distribution	34
grand_arc	Nb. Segments	Equidistant distribution	14
petit_arc	Nb. Segments	Equidistant distribution	6
h_mid	Nb. Segments	Equidistant distribution	7

Table III.1: Meshing parameters.

For the group named "cote" instead of selecting "Equidistant distribution" select "Scale distribution" and set a scale factor of 1.4 as shown in Figure III.32. Check the orientation of every edges is going from the exterior to the interior of the pipe. Otherwise add the edges where the orientation is wrong in "Reversed Edges" field (see Figure III.32).

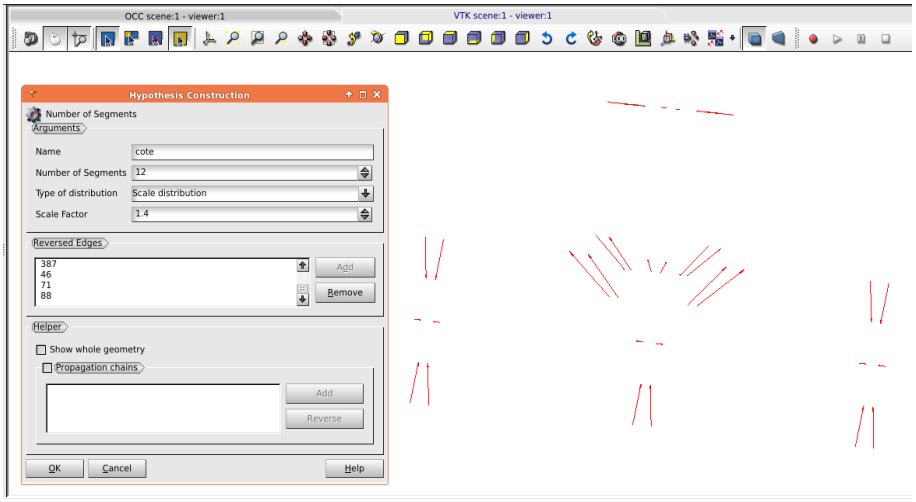


Figure III.32: Scaling law and reversed edges.

For the last two group of edges set "Fixed Point 1D" for the hypothesis and set the parameters as shown in Figure III.33 and Figure III.34.

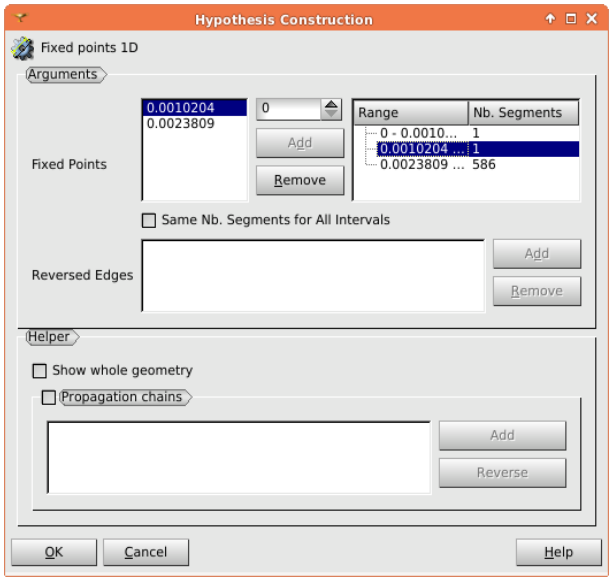


Figure III.33: Fixed_Point_1d hypothesis for the group "proff_long_ext_top".

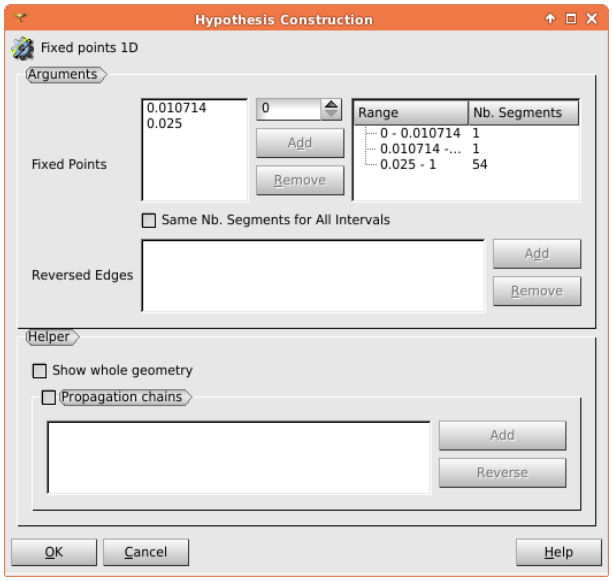


Figure III.34: Fixed_Point_1d for the group "proff_small_ext_top".

You should have now a mesh with 16 sub-mesh. Right click on your mesh "Mesh_1" by default and select "compute". Your mesh will be created, if you right click on "Mesh_1" and select "Mesh information" you should have the same criteria as Figure III.35.

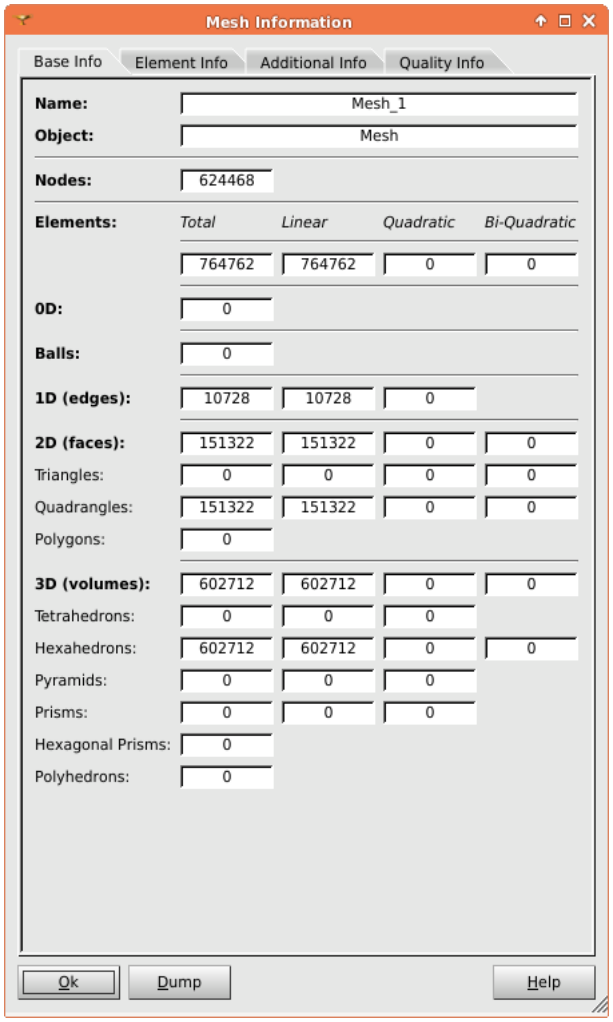


Figure III.35: Mesh information.

The Figure III.36 to Figure III.38 are showing different views of the mesh and Figure III.39 and Figure III.40 are showing clipping of the mesh.

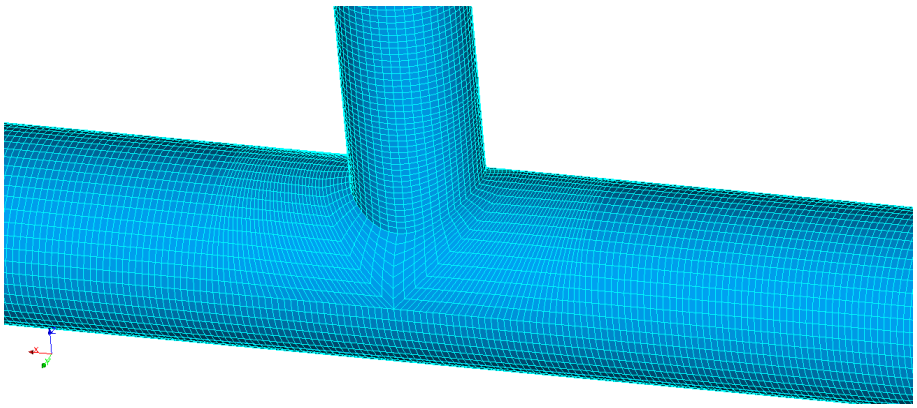


Figure III.36: Junction between the cold and hot pipe.

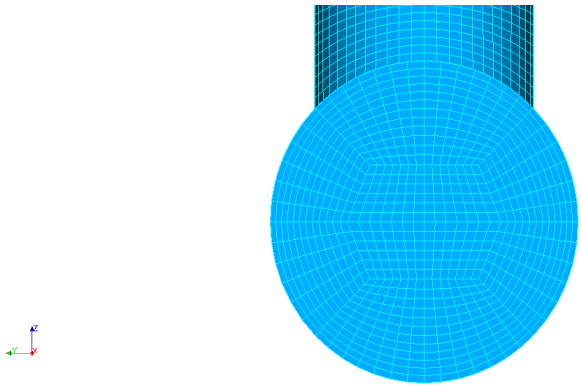


Figure III.37: Cold inlet.

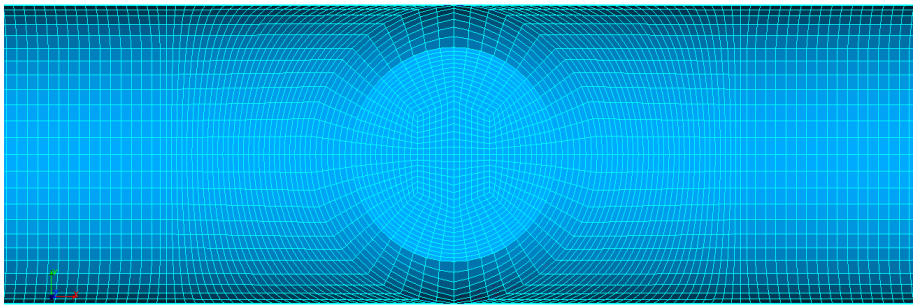


Figure III.38: Hot inlet.

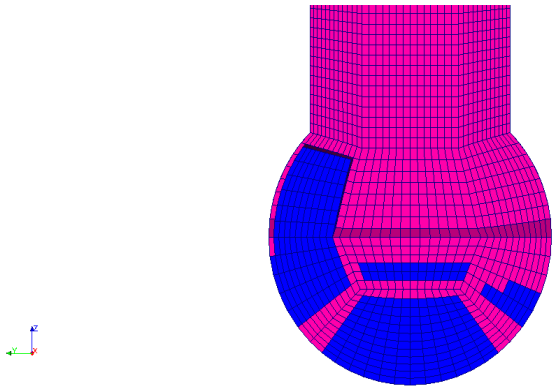


Figure III.39: Slice in the plane (yz).

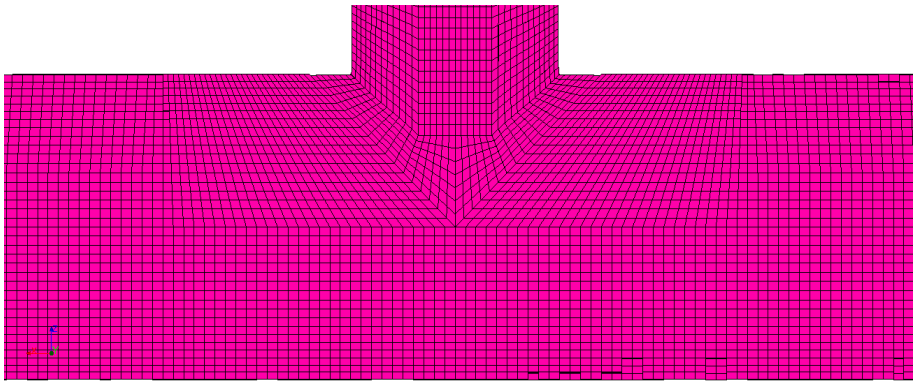


Figure III.40: Slice in the plane (xz).

Chapter IV

Part 3 - RANS Computation

1 What you will learn

In this fourth part of the tutorial you will learn how to set-up, run and post-process the results of a steady-state RANS calculation for the T-Junction generated in Part 3 (Section IV). The setting up and running will be completed using the *Code_Saturne* GUI. Post-processing will be undertaken using ParaViS in SALOME where you will learn how to generate section plots in the flow domain and generate 2D line plots in order to compare predicted with experimental data.

2 Setting up the CFD simulation

The CFD case is set-up and run from the CFDStudy module (Section II). In the CFDStudy module, create a 'New File' and verify that the case directory structure has been correctly recognised by clicking on the 'Identity and Paths' folder in the tree menu. If the case directory is correct you can continue. If not, you will need to set the correct directory. Then, save the file as 'RANS.xml'. You can now proceed with setting up the case, following the top down order of the folders in the left-hand column, starting with the mesh.

2.1 Selecting the volume mesh

Open the 'calculation Environment' folder and in the 'Meshes' panel of the 'Meshes Selection' sub-folder, add the 'Mesh_RANS_1.med' to the initially empty list of meshes (Figure IV.1). This is done by clicking on the "+" icon shown in Figure IV.1 and selecting the appropriate mesh in the MESH directory.

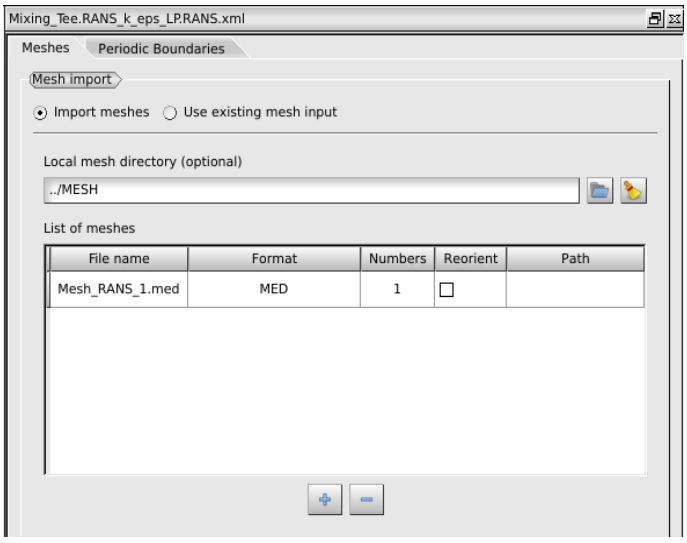


Figure IV.1: Selecting the 'Mesh_RANS_1.med' file for the calculations.

No further input is necessary for the volume mesh.

You can now go to the 'Thermo-physical Models' folder in order to specify the flow physics for the calculation.

2.2 Thermo-physical models

In the 'Calculation features' sub-folder, change the algorithm to 'steady flow' in the drop down menu. Leave all the other default values unchanged: multiphase flow, atmospheric flows, combustion and the electrical and compressible models are all inactive.

In the 'Turbulence models' sub-folder, change 'Turbulence model' to 'k-epsilon Linear Production'. In the 'Advanced Options' sub-folder, ensure that the wall function type is set to 'Two scale model' and that 'Gravity terms in the turbulence equations' is selected.

In 'Thermal model' sub-folder choose 'Temperature (Celsius)' for the 'Thermal scalar'. This will activate the solution of the temperature equation and designate Temperature as the scalar specified at the boundary conditions (Figure IV.2).

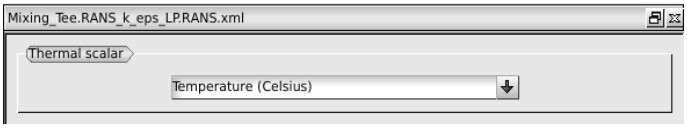


Figure IV.2: Activating solution of the Temperature equation

No other settings are required for the thermo-physical models. You can now move to the 'Physical properties' folder.

2.3 Physical properties

Due to the mixing of the hot and cold fluid streams, all the physical properties of the fluid are temperature dependant. The fluid is water at atmospheric pressure where the density, viscosity, specific heat and thermal conductivity are considered as function of the local temperature using the following relations [7]:

$$C_p(T) = -1.0224 \times 10^{-4} T^3 + 2.9201 \times 10^{-2} T^2 - 1.822T + 4209.9 \tag{IV.1}$$

$$\mu(T) = -1.9296 \times 10^{-9} T^3 + 4.7256 \times 10^{-7} T^2 - 4.2088 \times 10^{-5} T + 1.6947 \times 10^{-3} \tag{IV.2}$$

$$\frac{\lambda}{C_p(T)} = -3.0374 \times 10^{-13} T^3 - 2.1701 \times 10^{-9} T^2 + 4.7970 \times 10^{-7} T + 1.3538 \times 10^{-4} \tag{IV.3}$$

$$\rho(T) = 1.4078 \times 10^{-5} T^3 - 5.5855 \times 10^{-3} T^2 - 2.8886 \times 10^{-3} T + 1000.4 \tag{IV.4}$$

For this tutorial, these physical properties are coded in GUI. Initialise all parameters corresponding to the intermediate temperature 27.5°C and set the option "variable" for all the quantities which will be calculated with the set of equations defined above. The final set-up for this panel is shown in Figure IV.3.

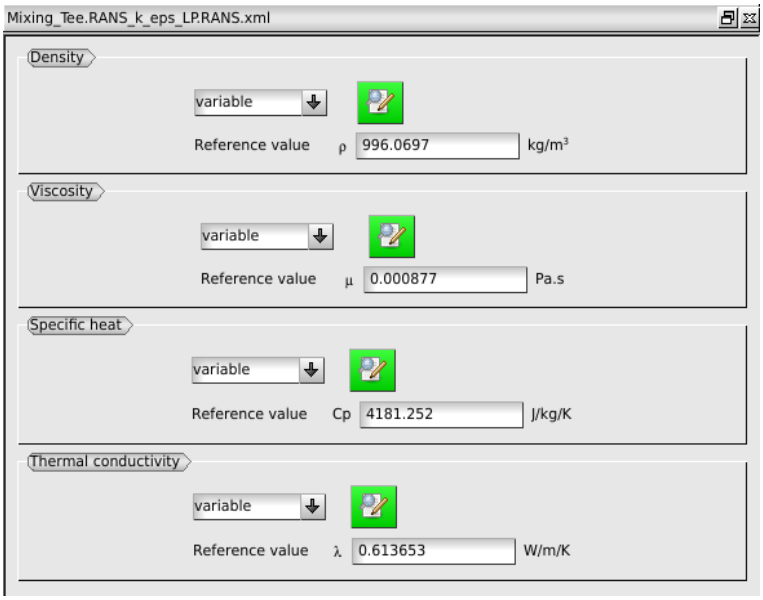


Figure IV.3: Selection of fluid properties.

The implementation of the different laws for the properties are shown in Figure IV.4 to Figure IV.7.

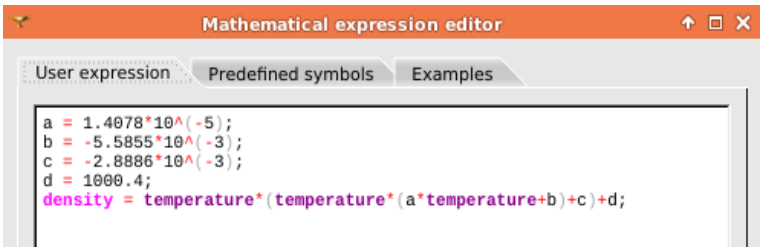


Figure IV.4: Coding the density as a function of the temperature.

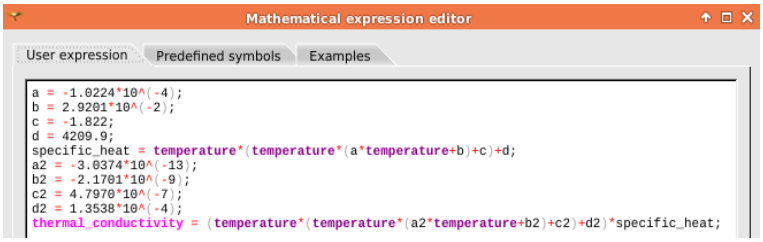


Figure IV.5: Coding the thermal conductivity as a function of the temperature.

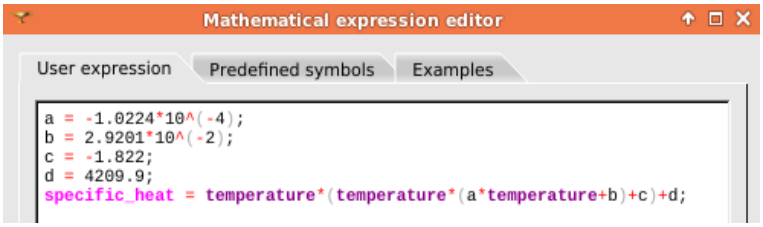


Figure IV.6: Coding the specific heat as a function of the temperature.

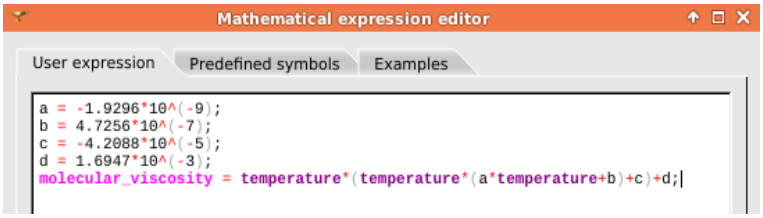


Figure IV.7: Coding the viscosity as a function of the temperature.

In the 'Reference values' sub-folder, set the reference pressure to 101325.0Pa and the reference velocity to 0.5m/s in the 'Reference values' panel.

In the 'Gravity' sub-folder, set the acceleration of gravity by entering the value '$-9.81 m/s^2$' for its component in the vertical (Z) direction in the 'Gravity' panel, Figure IV.8.

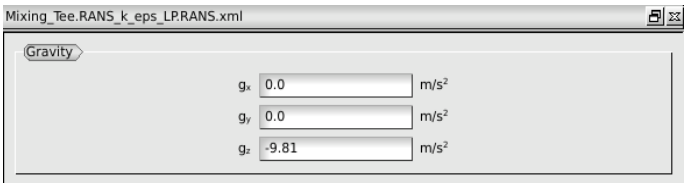


Figure IV.8: Gravity and Hydrostatic pressure specification.

No other settings are required in this folder. You can now move to the 'Volume conditions' folder.

2.4 Volume conditions

The initial values for the velocity and temperature are defined in the 'Initialization' sub-folder of the 'Volume conditions' folder. The flow is initially stagnant by default. To set the initial temperature, click on the 'Mathematical Expression Editor' button marked 'Thermal' in the 'Initialization' panel and enter the temperature of the cold inlet in the pop-up editor panel (Figure IV.9).

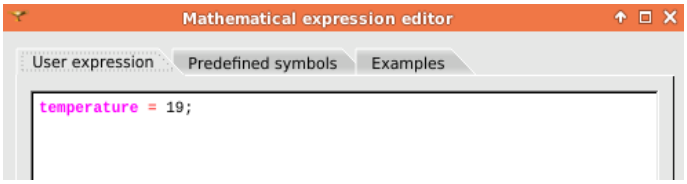


Figure IV.9: Initial temperature specification.

No other settings are required in this folder. You can now move to the 'Boundary conditions' folder.

2.5 Boundary conditions

Four boundary conditions are used in this study:

- Cold inlet : U (0.585, 0.0, 0.0) with a temperature of 19.0°C
- Hot inlet : U (0.0, 0.0, -0.764) with a temperature of 36.0°C
- Outlet : The standard outlet condition is used

- Walls : The wall boundary are assumed to be no-slip, smooth and adiabatic

For the LES computation, the Synthetic Eddy Method [8] is used to artificially generate turbulence at the domain inlets.

Two methods may be used to specify the boundary conditions. The user can create them manually or they can be partially completed by *Code_Saturne* itself.

For the first method, in the 'Boundary Conditions' folder, select 'Definition of Boundary Regions' sub-folder and in its panel click on the 'add' button four times to manually create the four boundary conditions generated in Phase II, namely 'inlet_1', 'inlet_2', 'outlet' and 'wall'. Change the 'Selection criteria' name of each boundary to reflect exactly the name of groups defined in Phase II. Then, change the nature to 'inlet' in the 'Nature' drop-down menu for 'inlet_1' and 'inlet_2', the nature to 'outlet' for the boundary 'outlet' and leave the nature 'wall' for the boundary 'walls'. The boundary regions are now fully defined (Figure IV.10).

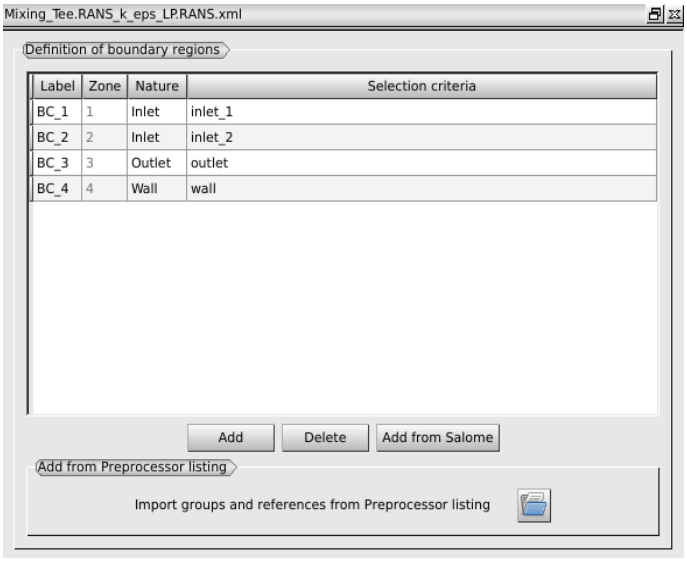


Figure IV.10: Boundary definition.

For the second method, execute the 'check mesh' in the 'Mesh quality criteria' sub-folder from the 'Calculation environment' folder as shown in the tutorial 1 Part II [5]. Then, go back into the 'Boundary conditions' folder and select 'Definition of Boundary Regions' sub-folder. Then, click on the icon in the panel to import groups and references from the pre-processor output file called 'listing'. In the pop-up window select the file 'check_mesh.log' as shown in Figure IV.11 and click on 'Open'.

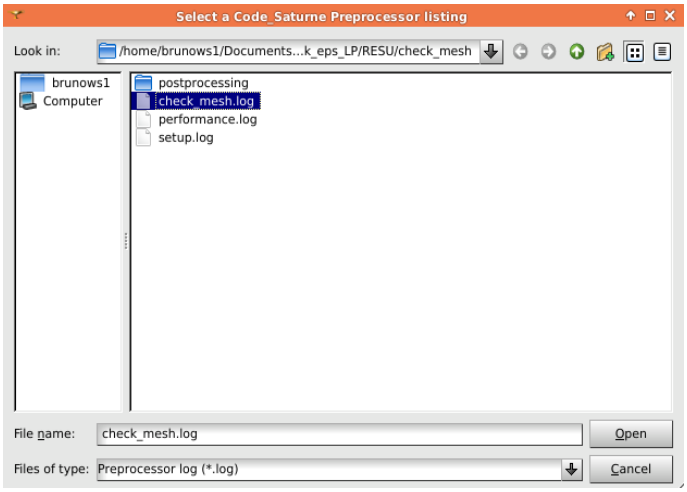


Figure IV.11: Preprocessor listing for boundary definitions.

Then change the nature of each boundary condition as explained above in order to obtain an identical boundary set-up to that shown in Figure IV.10.

Having defined their type, the boundary values can now be specified. Select the 'Boundary conditions' sub-folder and click on the boundary 'inlet_1'. Specify all boundary values as given in Figure IV.12. Boundary values for 'inlet_2' are presented in the Figure IV.13.

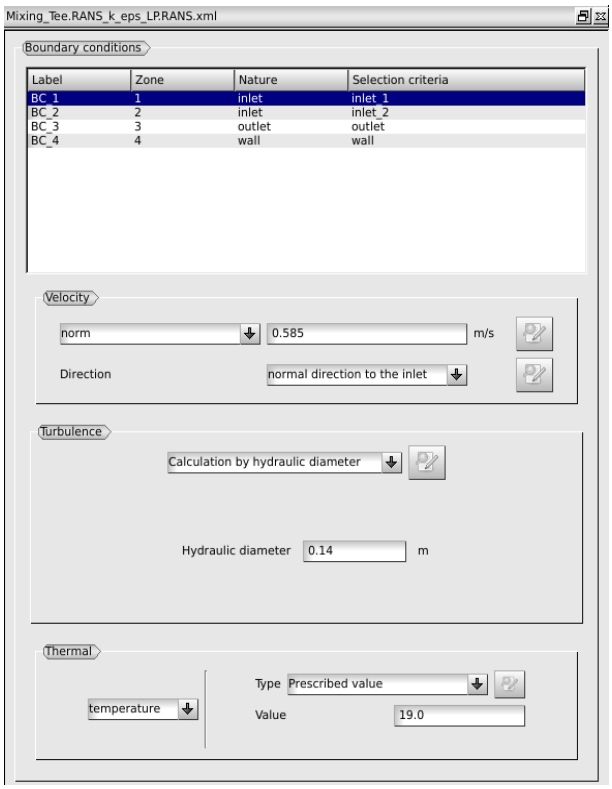


Figure IV.12: Inlet_1

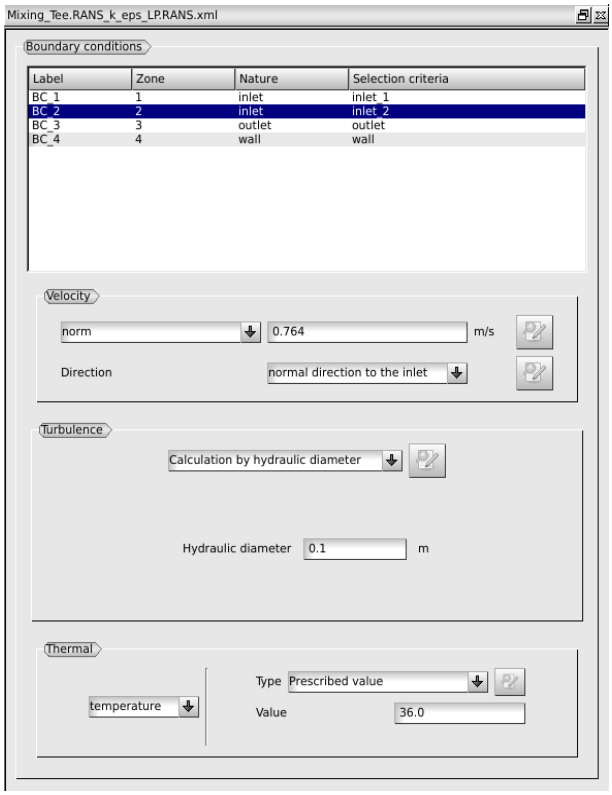


Figure IV.13: Inlet_2

Leave the default parameters for the 'outlet' and 'walls' boundary conditions.

No other settings are required in this folder. You can now go to the 'Numerical parameters' folder.

2.6 Numerical Parameters

In the 'Numerical Parameters' folder and in the 'Global parameters' sub-folder set the 'Velocity-Pressure algorithm' to SIMPLE. Leave as default all other settings.

In the 'Equation parameters' sub-folder, the 'Solver' panel shows that pressure, velocity and temperature are solved for. In order to decrease overall computation time, it is possible to reduce the maximum iteration number to 1000 for each variable and decrease the solver precision to 10^{-5} as shown in Figure IV.14.

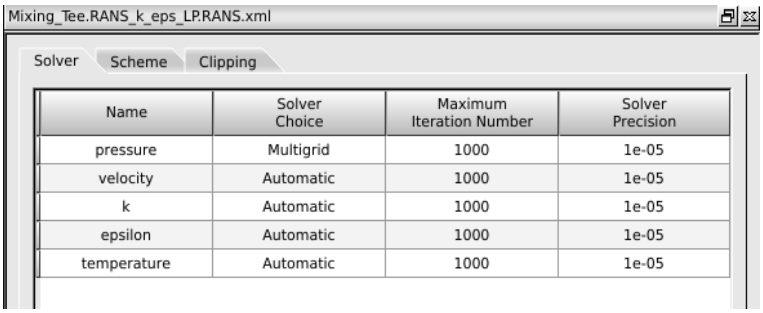


Figure IV.14: Solver parameters.

Leave as default the parameters in the 'Scheme' panel. However, if you have limited computational

resources, you can change the second Centered scheme to the first order Upwind scheme for the convective variables.

The 'Clipping' panel is used to set the temperature bounds thereby making it possible to instruct the code to clip the temperature to these values if it strays outside this defined range. This is done in order to help improve solution stability. For this tutorial you can enter the cold and hot inlet temperature for clipping the predicted scalar temperature, as shown in Figure IV.15.

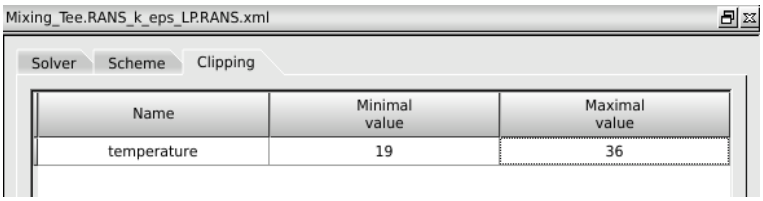


Figure IV.15: Clip of the scalar temperature.

In the 'Steady flow management' panel, keep the relaxation coefficient at 0.7 but change the number of iterations to 3,000.

No other settings are required in this folder. You can now move to the 'Calculation control' folder.

2.7 Calculation control

In the 'Calculation control' folder, select the 'Output control' sub-folder and go into the 'Monitoring Points' panel. Click on the '+' icon to add a probe then enter the coordinate of this first probe as shown in Figure IV.16. Repeat this procedure for other probes of your own choice. Monitoring probes can be useful to check the convergence of the simulation.

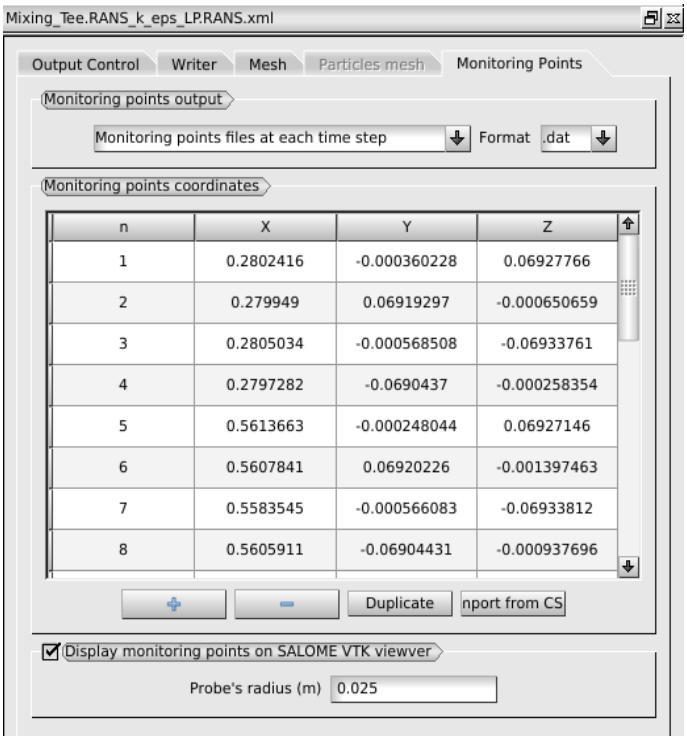


Figure IV.16: Monitoring points.

The *Code_Saturne* calculation is now fully specified from the standpoint of the GUI and the xml file should be saved.

3 Running and analysing the simulation

3.1 Running the simulation

In the 'Calculation management' folder click on the 'Start/Restart' sub-folder and check that the 'calculation restart' option is off. It is possible to specify a checkpoint frequency by clicking on the 'Advanced options' which launches and 'Advanced options' panel as shown in the Figure IV.17.

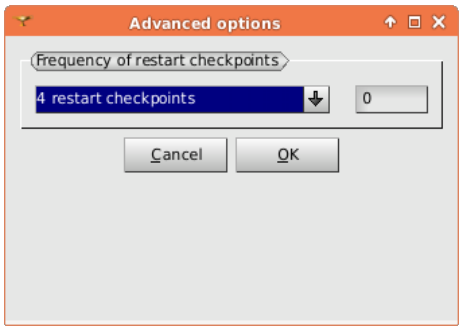


Figure IV.17: Checkpoint frequency.

In the panel of 'Prepare batch calculation' sub-folder (Figure IV.18), the default selections of 'runcase' for the 'Script file', 'standard' (straight calculations without mesh or partitions import or processing) and '1' processor for the 'Calculation script parameters' are set as default. Change the 'Number of processes' to those that you require for running the simulation.

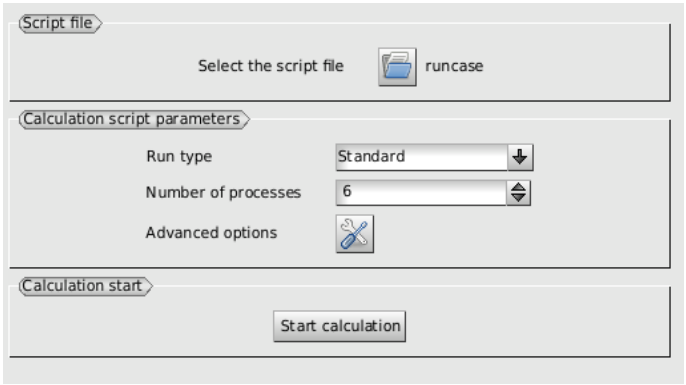


Figure IV.18: Batch calculation settings.

Press the 'Start calculation' button to run *Code_Saturne*. The pop-up panel for the run opens, listing in real time the different stages of the calculation, from user-subroutines compilation to saving the results.

Wait for the calculation to complete and open the 'listing' file in your 'Mixing_Tee/RANS/RESU/DateOfRunTimeOfRun/' directory. Verify that the residuals listed under 'derive' in the 'Information on Convergence' table have dropped several orders of magnitude for all variables (pressure, velocity, temperature), showing that the calculations have fully converged to a steady-state solution (Figure IV.19).

** INFORMATION ON CONVERGENCE

Variable	Rhs norm	N_iter	Norm. residual	derive	Time residual
c Velocity	0.14523E+01	27	0.27371E-01	0.37695E+04	0.43777E+04
c Velocity[X]				0.36134E+04	
c Velocity[Y]				0.70732E+00	
c Velocity[Z]				0.15546E+03	
c Pressure	0.32305E+00	2238	0.54861E+00	0.99999E+00	0.49392E+04
c k	0.87829E-01	21	0.17092E-03	0.78012E-05	0.35884E+03
c epsilon	0.72240E+00	31	0.54678E-02	0.15008E+00	0.38016E+04
c TempC	0.83303E+07	16	0.38866E-04	0.86257E+02	0.40963E+02

(a) After 1 iteration.

** INFORMATION ON CONVERGENCE

Variable	Rhs norm	N_iter	Norm. residual	derive	Time residual
c Velocity	0.90496E+01	15	0.70589E-02	0.77416E-03	0.78021E+00
c Velocity[X]				0.26096E-03	
c Velocity[Y]				0.63613E-04	
c Velocity[Z]				0.44959E-03	
c Pressure	0.12350E-01	82	0.53598E-02	0.98606E-01	0.29537E+00
c k	0.13416E+00	11	0.24571E-04	0.92884E-08	0.18812E+00
c epsilon	0.10081E+01	10	0.65732E-04	0.10838E-05	0.24369E+00
c TempC	0.10253E+07	14	0.41000E-02	0.24363E+00	0.45131E+00

(b) After 3000 iterations.

Figure IV.19: Convergence history from 'listing' file.

You can now proceed with examining and post-processing the results by returning to the SALOME platform.

4 Results analysis

In this section, the results obtained from the steady-state simulation using k-ϵ LP turbulence model are compared to experimental data [4]. In the first instance, the y+ is checked along the wall to ensure that y+ is within the limits of the law of the wall model then a clip plane is used to cut the volume mesh along the main pipe in order to visualise the mixing between the hot and the cold fluid streams. Finally, the velocity and the normalised temperature are extracted along measurement lines after the T-Junction in order to compare predicted and experimental data.

4.1 Importing *Code_Saturne* Results into SALOME/ParaViS

Start a SALOME session in the usual manner and select 'ParaViS' from the drop-down module selector in the top menu bar. The name of the module will add itself to the 'Object Browser' list and the ParaView-specific panels and menus will be activated, including a new 'ParaView scene viewer' window.

Before loading the run data in ParaViS, follow the steps described in Tutorial 1 Part II [5], to modify the default colour schemes.

In the 'Pipeline Browser' panel on the left-hand side, right click and select 'Open' in the drop-down menu. Point to the 'RESULTS.case' file in the RESU directory for the run that has just finished:

/../Mixing_Tee/RANS/RESU/DateOfRunTimeOfRun/postprocessing/RESULTS.case

Then, follow the steps described in Tutorial 3 Part I [9], to create the 'ExtractBlock' and 'CellData-PointData' object used for post-processing.

4.2 Checking the y+ at the boundaries

The listing file after 3000 iterations is showing a y+ in a range between 22 and 176 as shown in Figure IV.20.

```
** BOUNDARY CONDITIONS FOR SMOOTH WALLS
   -------------------------------------

-------------------------------------------------------------
                                          Minimum     Maximum
-------------------------------------------------------------
   Rel velocity at the wall uiptn :  0.13849E-01 0.11978E+01
   Friction velocity        uet   :  0.14613E-02 0.10954E+00
   Friction velocity        uk    :  0.19455E-01 0.74442E-01
   Dimensionless distance   yplus :  0.22559E+02 0.17619E+03
   Friction thermal sca.    tstar :  0.00000E+00 0.00000E+00
   Rough dim-less th. sca.  tplus :  0.42264E+02 0.58702E+02
   ------------------------------------------------------
   Nb of reversal of the velocity at the wall   :          0
   Nb of faces within the viscous sub-layer     :          0
   Total number of wall faces                   :      39572
-------------------------------------------------------------
```

Figure IV.20: Boundary information after 3000 iterations.

A visualization on Paravis enable to locate where the low y+ are located as well as the high y+ (Figure IV.21).

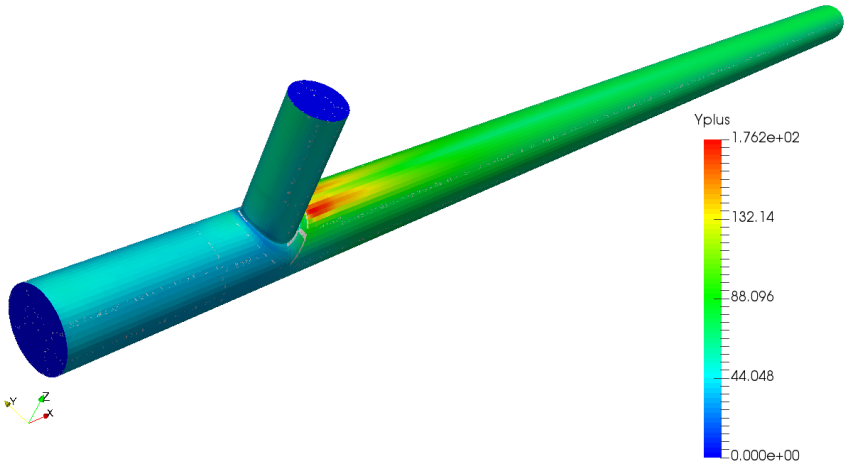


Figure IV.21: Visualisation of the y+.

4.3 Visualising data on a slice plane

In the 'Pipeline Browser' select the 'CellDataPointData' object and click on ***Filters*** ⇒ ***Common*** ⇒ ***Slice*** in the top menu or click on the 'slice' icon. In the 'Object Inspector' ⇒ 'Properties' tab choose the 'Y Normal' direction and leave the default coordinate of the slice origin, then press 'Apply'. Click on the 'Display' tab of the 'Object Inspector' to choose the 'TempC' as 'Color by' as shown in Figure IV.22. Then click on the 'Edit Color Map' button to pop-up the 'Color Scale Editor'. In this panel, click on 'Color Legend' to launch the 'Color legend' window and tick the 'Show Color Legend' box to add a colour legend to the scene.

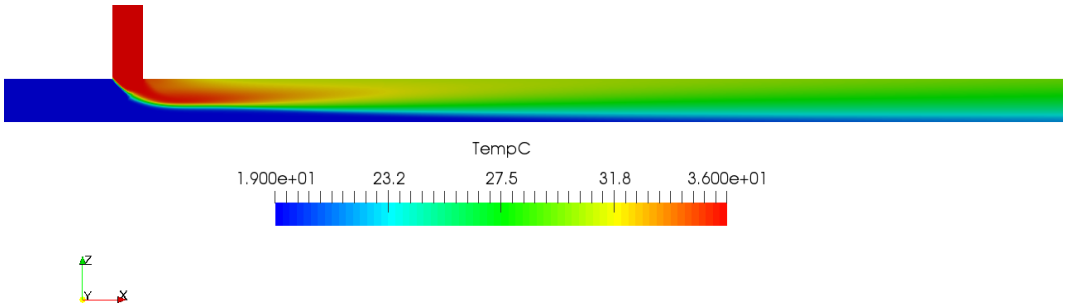


Figure IV.22: Temperature field in the plane (xz).

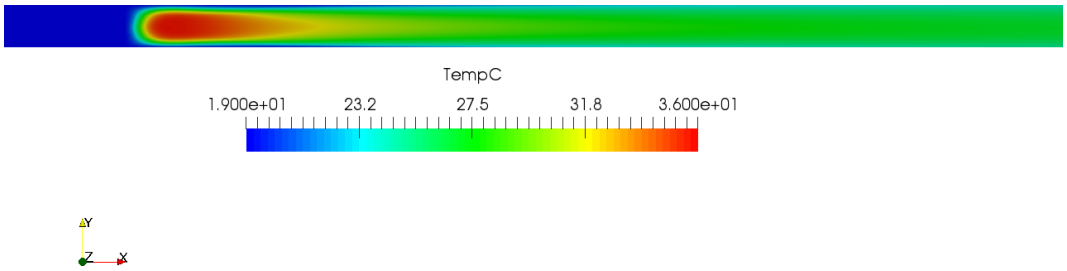


Figure IV.23: Temperature field in the plane (xy).

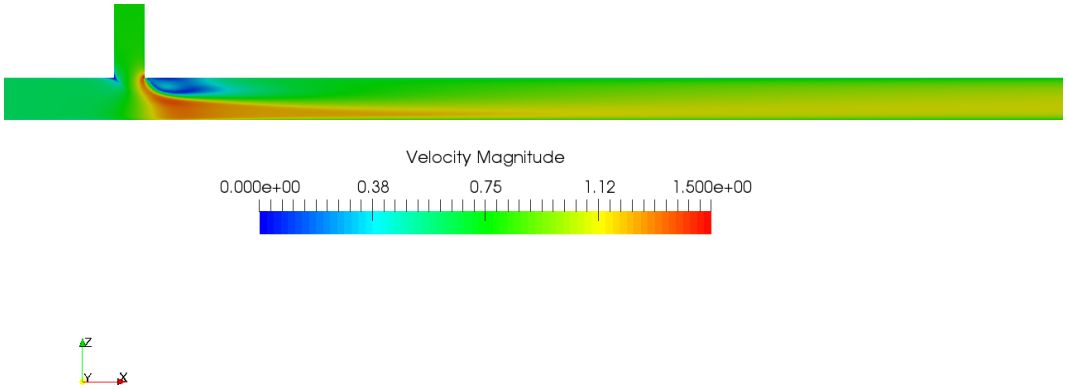


Figure IV.24: Velocity magnitude field in the plane (xz).

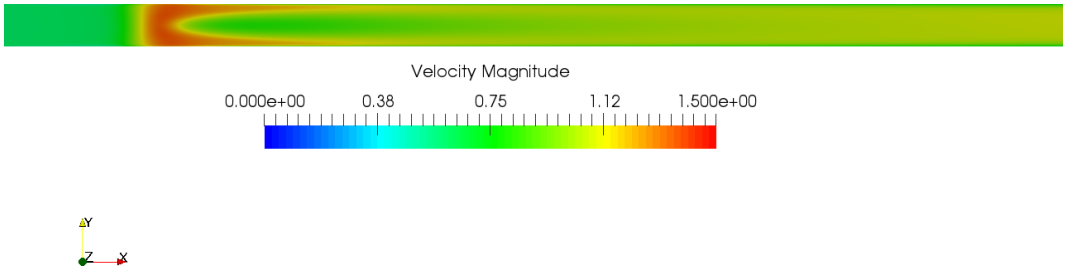


Figure IV.25: Velocity magnitude field in the plane (xz).

4.4 Extracting line data

To prepare the extraction of the data, select the 'CellDataPointData' object and click on ***Filters*** ⇒ ***Common*** ⇒ ***Calculator***. Set the calculator with the "y" coordiantes by clicking on "Scalars" and selecting "coordsY", Figure IV.26. For the post-processing of the vertical data, create a calculator for the "z" coordinates.

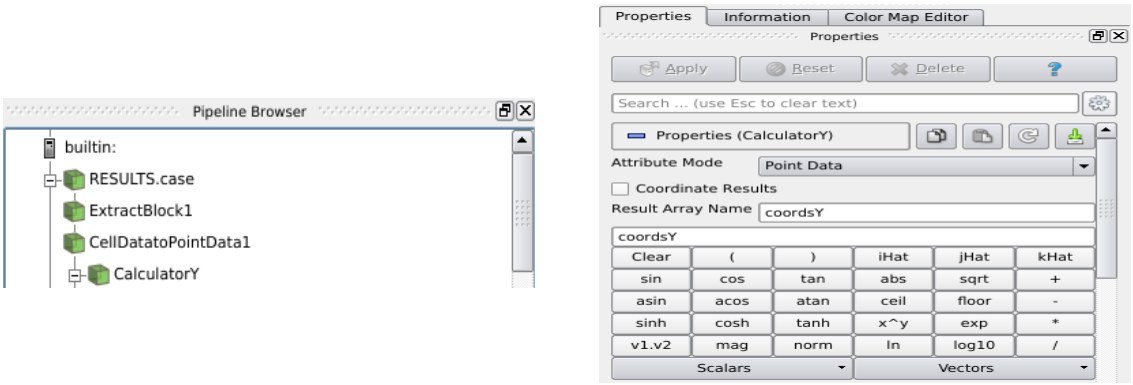


Figure IV.26: Creating a calculator for y coordinates.

In order to plot the data along a line going through the computational domain, select the 'Calculator' object and click on ***Filters*** ⇒ ***Data Analysis*** ⇒ ***Plot Over Line***. In the 'Object Inspector' ⇒ 'Properties' tab enter the coordinates of the two points which will define the line as shown in the Figure IV.27. This line will go through the main pipe along the horizontal diameter at x=0.224m after the Tee.

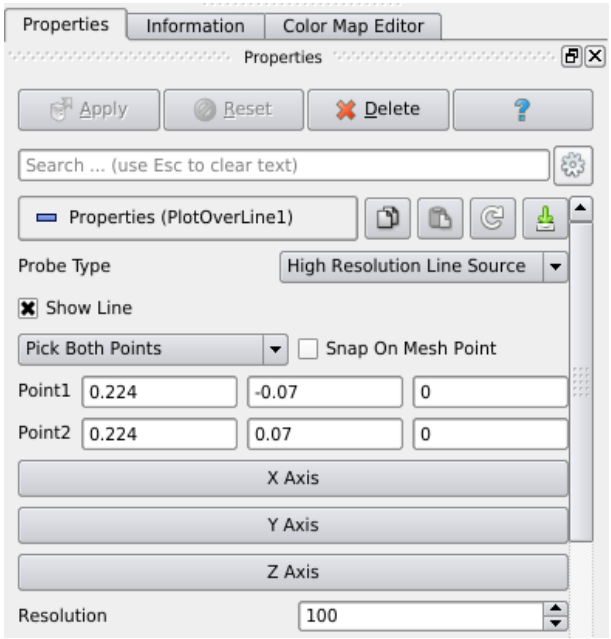


Figure IV.27: Properties specification for the PlotOverLine.

After pressing 'Apply', you will see that the scene window which is named 'Layout #1' is cut into two parts. On the left is the 3D colour view and on the right is the 2D graph. In this tutorial we want to have one 'Layout' per graph so close the graph view by clicking on the cross button at the top right hand corner of the view. Next, click on the '+' icon just next to the 'Layout #1' tab to create the

'Layout #2', as shown in Figure IV.28.

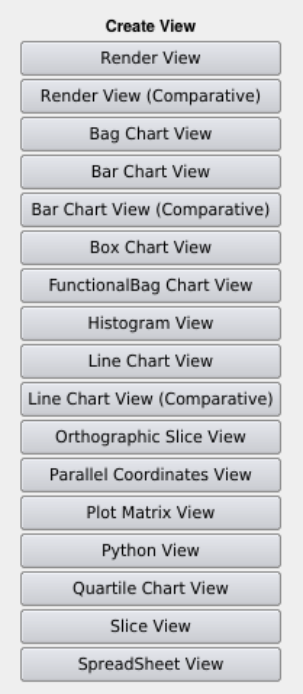


Figure IV.28: ParaView 'Create View' menu.

Click on the 'Line Chart View' button in 'Layout #2' and click on the 'eye' icon at the left hand of the 'PlotOverLine1' in 'Pipeline Browser'. As a result, all data available in the computational domain are plotted in the 'Layout #2' panel. In order to plot only the x-component of the velocity, in the 'Display' tab of the 'Object Browser' select only the 'Velocity' variable in the 'Series Parameters' area and choose 'coordsY' which corresponds to the y-coordinate created by the calculator (Figure IV.26).

In order to add the experimental data to the graph, file 'exp_u16Dh.csv', please follow the instructions of tutorial 1 Part II [5]. All the data necessary to create the different files containing the experimental data are listed in Appendix 1. By way of an example, take the data of the Table VIII.2. Follow the instructions in Tutorial 1 Part II [5], in order to create the *.csv files. When the 'exp_u16Dh.csv' is in the 'Pipeline Browser', repeat the procedure as for the 'PlotOverLine1' (close the default view after pressing 'Apply', select the 'Layout #2', click on the 'eye' icon of the 'exp_u16Dh.csv' object and choose the correct data to plot).

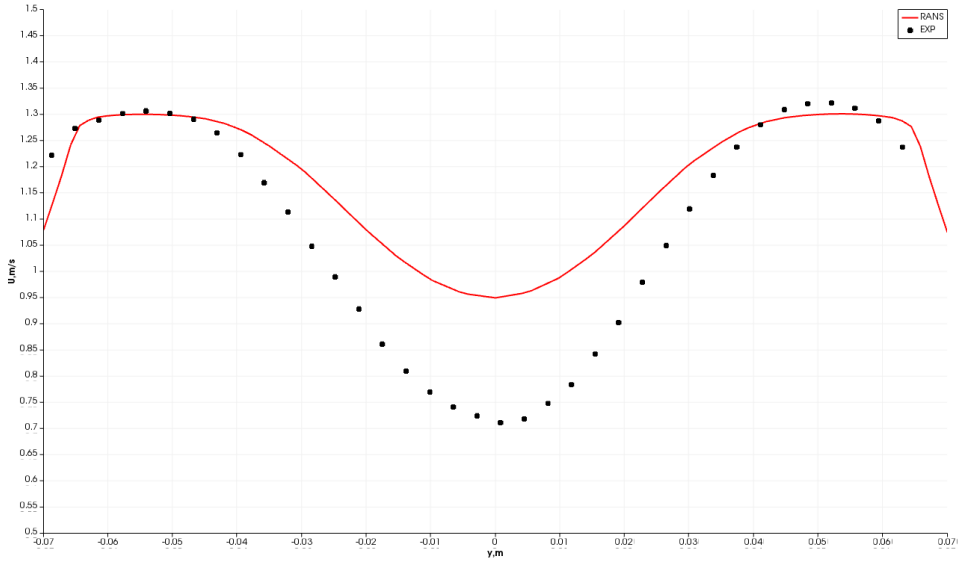
Chapter V

Part 4 - Comparison of Predicted and Experimental Data

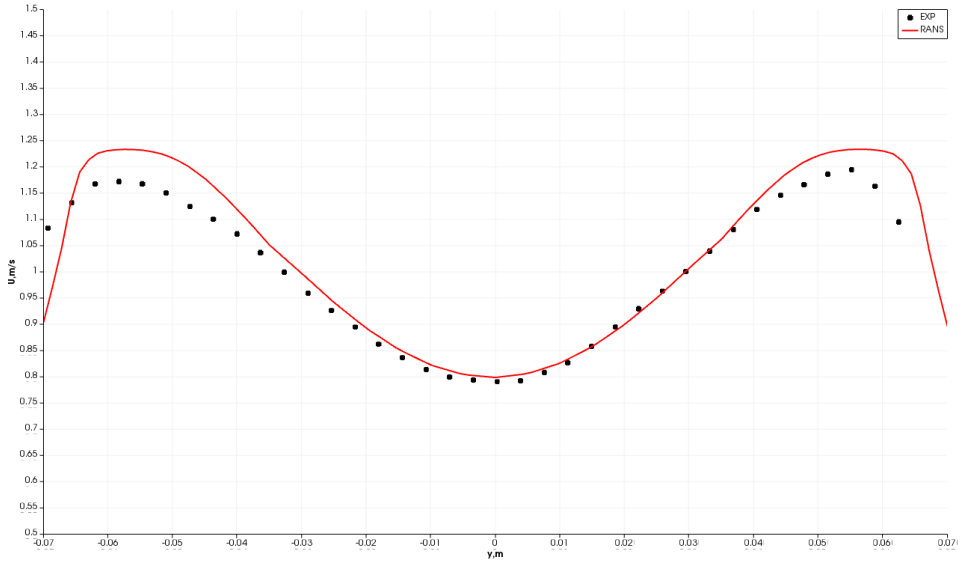
1 Comparison of the axial velocity with the experimental data.

You can find in Figure V.1a to Figure V.2d the computational results compared to the experimental for the axial velocity along vertical and horizontal lines.

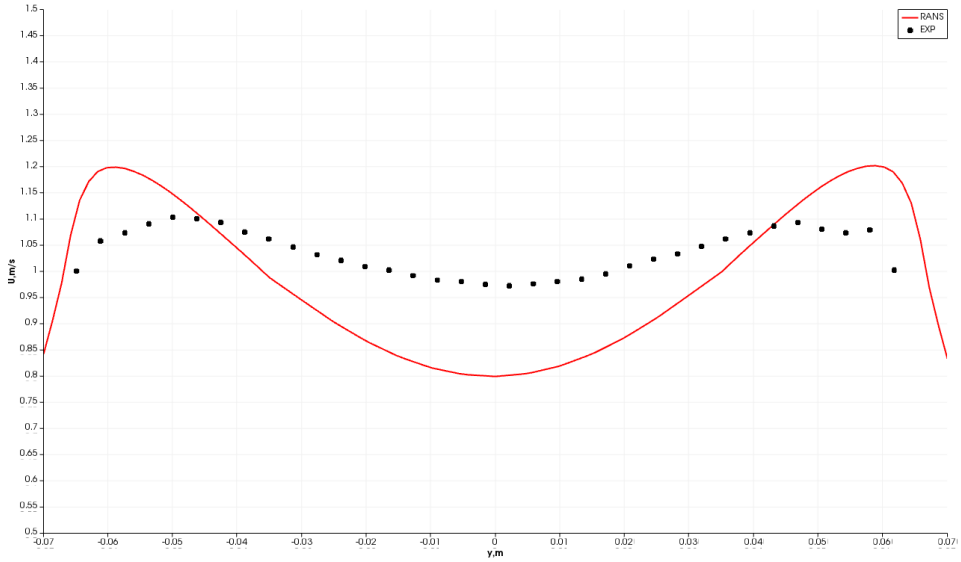
1.1 Comparison along horizontal lines.



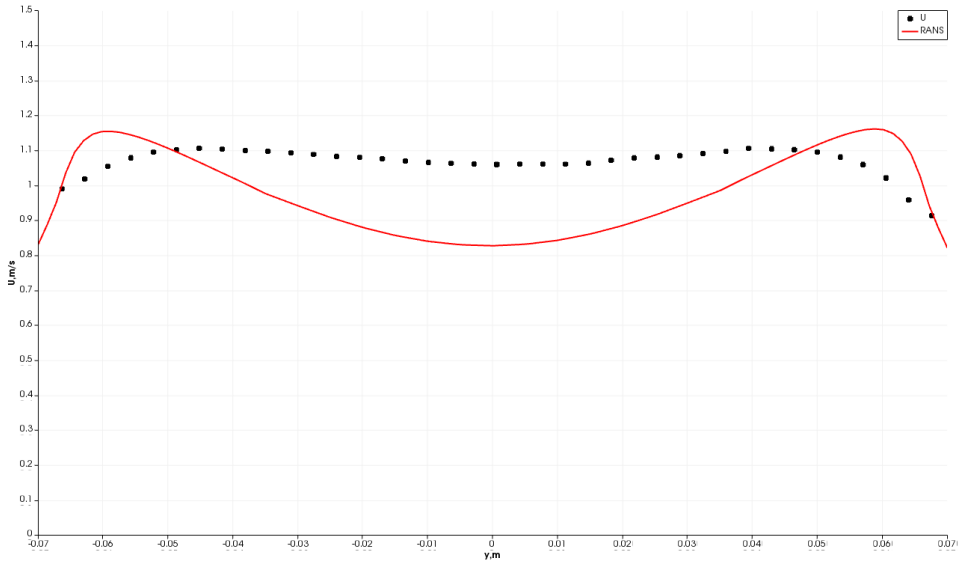
(a) at x=1.6D



(b) at x=2.6D



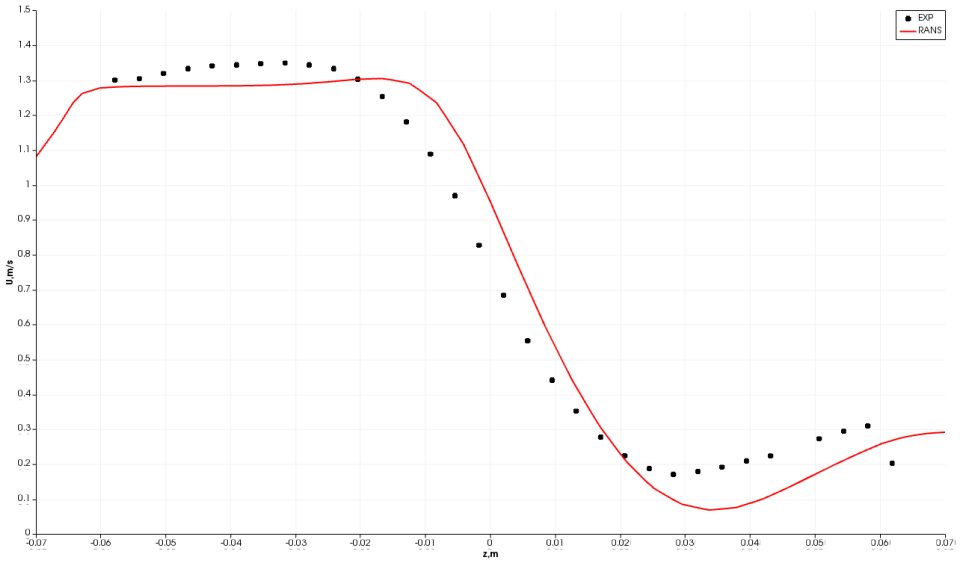
(c) at x=3.6D



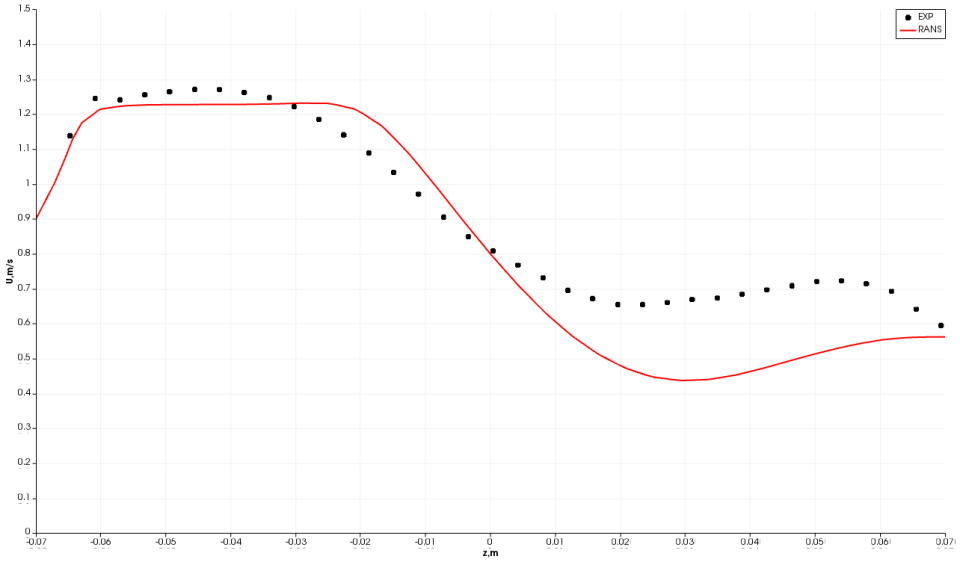
(d) at x=4.6D

Figure V.1: x-component of the velocity for the RANS simulation along horizontal lines at x = 1.6D, 2.6D, 3.6D and 4.6D.

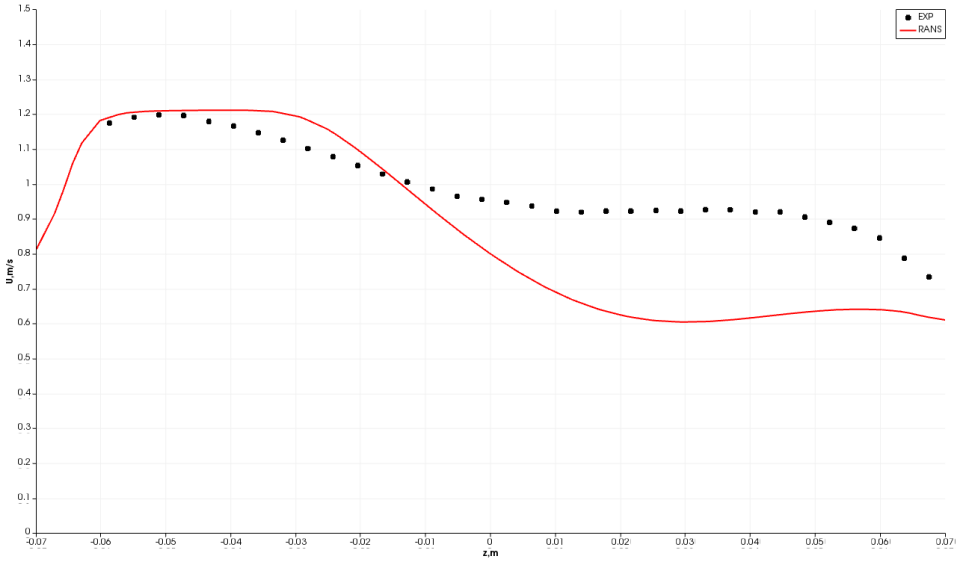
1.2 Comparison along vertical lines



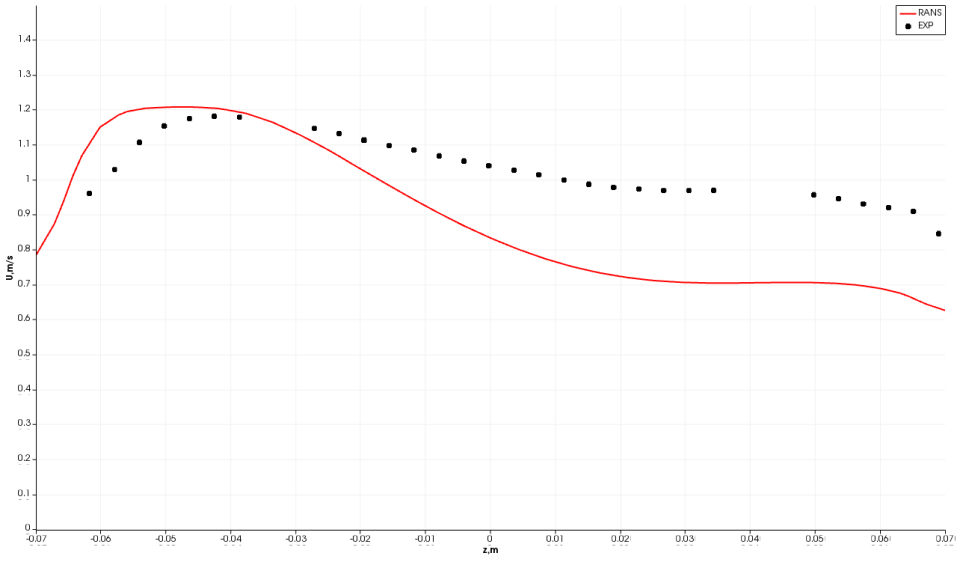
(a) at x=1.6D



(b) at x=2.6D



(c) at x=3.6D



(d) at x=4.6D

Figure V.2: x-component of the velocity for the RANS simulation along vertical lines at x = 1.6D, 2.6D, 3.6D and 4.6D.

2 Comparison the dimentionless temperature with the experimental data

To plot the dimensionless temperature create the variable "$< T >^*$" in a calculator. This temperature is given by :

$$< T >^* = \frac{< T > - T_{cold}}{T_{hot} - T_{cold}} \tag{V.1}$$

The dimensionless temperature in Figure V.3 is plotted along the domain at y=0 and z=0.07. Figure V.4 is plotted along the domain at y =-0.07 and z=0.

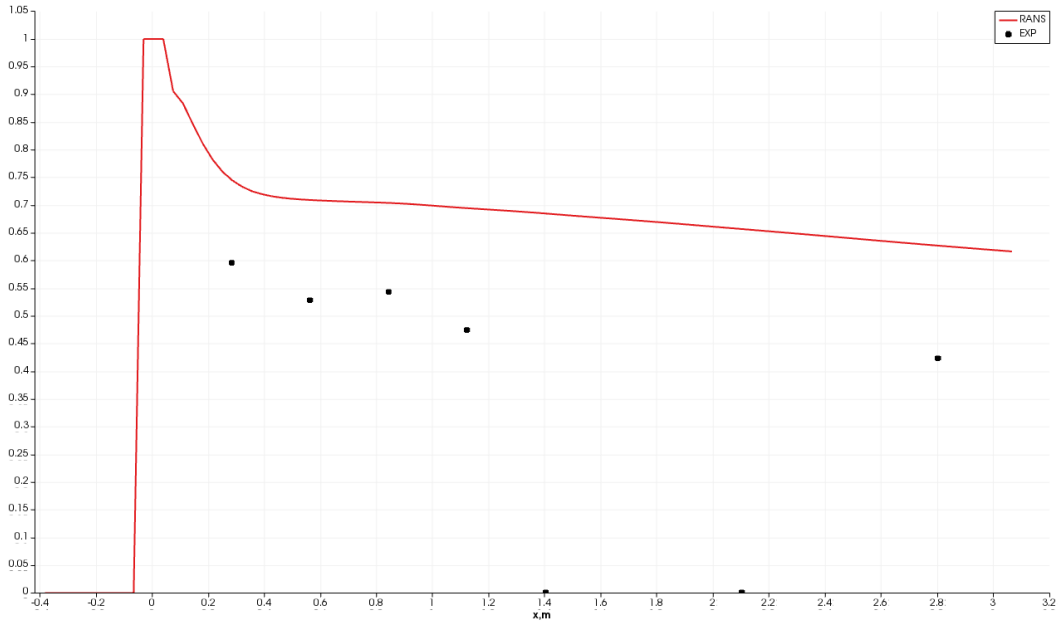


Figure V.3: Dimensionless temperature alond the lines defined in the main pipe at y=0 and z=0.07.

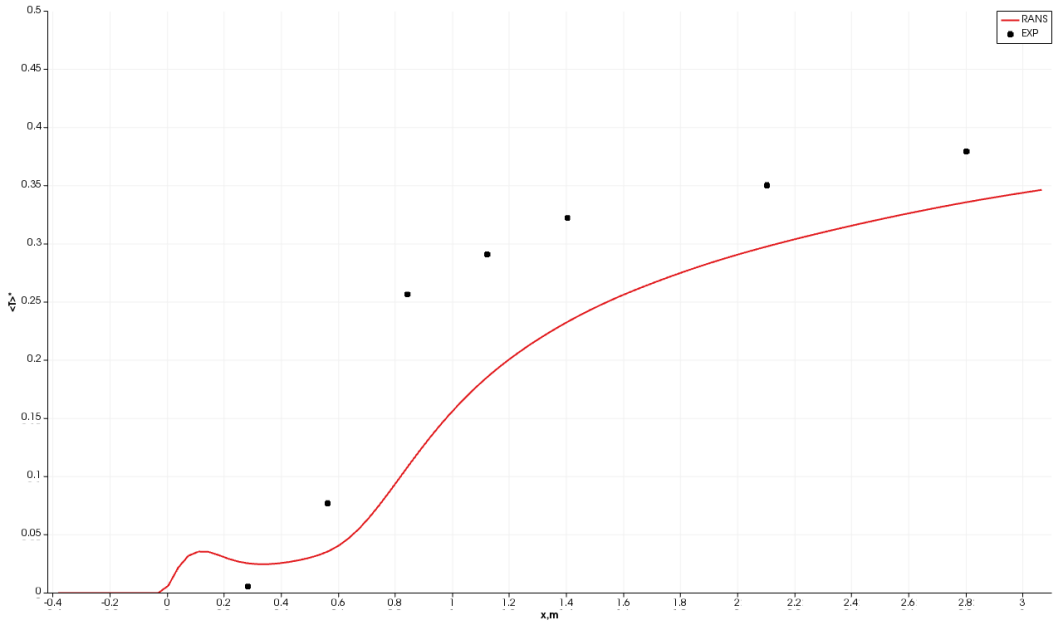


Figure V.4: Dimensionless temperature alond the lines defined in the main pipe at y=-0.07 and z=0.

Chapter VI

Part 5 - LES Computation

1 What you will learn

In this fourth part of this tutorial, you will learn how to set-up an LES simulation of the flow in the T-Junction. The set-up procedure involves user coding for the Synthetic Eddy Method for estimating the turbulent profiles at the hot and cold flow inlets to the computational domain.

1.1 Setting up the CFD simulation

In this section, only the modifications required in order to set-up the LES calculation are presented here.

The *Code_Saturne* case is set-up and run from the CFDStudy module. Move to the 'LES' case directory and start the GUI. Create a 'New File' and verify that the GUI has correctly recognised the case directory structure and save the file as 'LES.xml'.

You can now proceed with setting up the case, in the top down order of the folders in the left-hand column of the GUI.

- In the 'Meshes' panel of the 'Meshes selection' sub-folder, add the volume mesh 'Mesh_LES.med'
- In the 'Calculation features' folder, maintain an 'unsteady flow' in the drop down menu at the top and, as before, leave all the other default values unchanged: multiphase flow, atmospheric flows, combustion and the electrical and compressible models are all inactive
- Next, change 'Turbulence models' to '(LES Smagorinsky)'
- Do not specify any boundary conditions in the 'Boundary Conditions' panel. These will be specified in the user subroutines (see 2)

Having set up the mesh and physics of the problem, the numerical parameters may be specified

- In the 'Numerical Parameters' folder, leave all the default settings in the 'Global parameters' sub-folder and choose the 'SIMPLEC' as 'Velocity-Pressure algorithm'. Leave all default parameters in the 'Equation parameters' sub-folder
- In the 'Time step' sub-folder choose the 'constant' option for the time step and specify a reference time step of 0.00015s as shown in the Figure VI.1. This time step is adapted to have a CFL inferior to 1 for the mesh 'Mesh_LES.med'

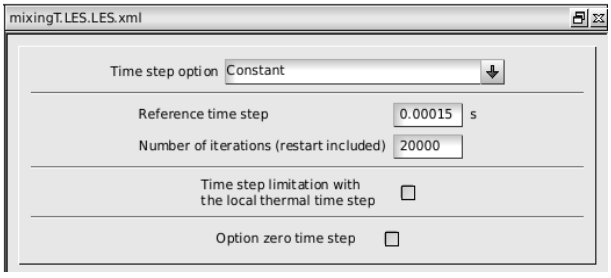


Figure VI.1: Time step menu.

The setting up of the LES simulation using the GUI has now finished and the 'LES.xml' file should be saved at this point. However, before you can run the simulation, user coding for the subroutines 'cs_user_boundary_conditions.f90', cs_user_physical_properties.f90 and 'cs_user_les_inflow.f90' subroutines must be coded.

2 Programming the physical properties with user coding

2.1 User Boundary Conditions

Copy the sample file 'cs_user_boundary_conditions.f90' from the tutorial's /../MixingTee/LES/SRC/EXAMPLES directory to your SRC directory in order to create a local copy. This local copy can be customised accordingly and will be automatically compiled and linked to the 'cs_solver' executable at run time. Once copied, open your local version of this file using the text editor of your choice.

The specification of the different boundary conditions is done in the 'cs_user_boundary_conditions' subroutine. Scrolling through this subroutine, you can see that several examples for different boundary conditions are available. In this tutorial, you are going to implement your own boundary conditions in the routine 'getfbr'. For clarity, you may remove all other examples from the file. The customised code available for the LES tutorial is already commented. Here we describe the main parts and the logic behind them.

1. Use the subroutine 'getfbr' to select a particular boundary condition.
2. Cycling through the boundary faces.
   (a) For the boundaries 'inlet_1' and 'inlet_2' apply the type 'ientre' and the temperature to all boundary faces
   (b) For the boundary 'outlet' apply the type 'isolib' to all boundary faces
   (c) For the wall boundaries apply the type 'iparoi' to all boundary faces

2.2 User physical properties

To begin with, copy the sample file 'cs_user_physical_properties.f90' from the tutorial /../MixingTee/RANS_k_eps_LP/SRC/REFERENCE directory to your SRC directory. This is done in order to create a local copy which you will be able to customise and which will be automatically recompiled and linked to the 'cs_solver' executable at run time.

Once copied, open your local version of the file using the text editor of your choice. The file contains a number of subroutines and the specification of physical properties is done in subroutine 'usphyv'. Scrolling through this subroutine, you can see that there are several coded examples for the different physical properties. In this tutorial, you are going to modify 'Example 1' with your own implementation of the density, viscosity, specific heat and thermal conductivity as a function of temperature (IV.1 to IV.4). For clarity, you should remove all the other examples from the file. The customised code available with the tutorial is already commented. Here we describe the main parts of this user coding and the logic behind them.

1. Declare your own local variables at the top of the subroutine, either as double precision real values or as integer values
2. Activate the example (replace '.false.' with '.true.')
3. Initialise the integers ipcrom, ipcvis, ipccp, ipcvls for the rank of density, viscosity, specific heat and thermal conductivity respectively
4. Compute the coefficient of laws for the physical properties: density, viscosity, specific heat, (thermal conductivity)/(specific heat)
5. Cycling through the internal cells of the computational domain:
   (a) Compute the density as a function of temperature (Eq. IV.4)

(b) Compute the viscosity as a function of temperature (Eq. IV.2)

(c) Compute the specific heat as a function of temperature (Eq. IV.1)

(d) Compute the (thermal conductivity)/(specific heat) as a function of temperature (Eq. IV.3)

(e) Compute the thermal conductivity as the multiplication of d. and c.

2.3 LES inflow boundary conditions

The specification of the LES inflow boundary conditions is done in the 'cs_user_les_inflow-base.f90' subroutine. Copy the sample subroutine 'cs_user_les_inflow-base.f90' from the tutorial's /../MixingTee/LES /SRC/EXAMPLES directory to your SRC directory in order to create a local copy named 'cs_user_les_inflow.f90'. As before, you will be able to customise this subroutine which will be automatically compiled and linked to the 'cs_solver' executable at run time. Then, open your local version of this file using the text editor of your choice.

This file contains three subroutines 'cs_user_les_inflow_init', 'cs_user_les_inflow_define' and 'cs_user_les_inflow_advanced'. The first routine is for specifying the global characteristics of synthetic turbulence at the domain inlets. The second routine is for specifying the characteristics of specific inlets. The last routine is to specify the accurate specification of target statistics at all inlets.

The customised code available with the tutorial is already commented. Here we describe the main parts and the logic behind them.

1. In the subroutine 'cs_user_les_inflow_init', specify the number of synthetic turbulent inlets, here 2 inlets are used.
2. In the subroutine 'cs_user_les_inflow_define':
   (a) Initialise the velocity, the turbulent kinetic energy and the dissipation scales to zero (these values will be specified in the subroutine 'cs_user_les_inflow_advanced'
   (b) For the cold inlet specify that:
      i. The Synthetic Eddy Method is used
      ii. 300 synthetic eddies contribute to the turbulent fluctuations
      iii. This inlet is applied to the boundary condition 'inlet_1' through the 'getfbr' subroutine
   (c) For the hot inlet specify that:
      i. The Synthetic Eddy Method is used
      ii. 300 synthetic eddies contribute to the turbulent fluctuations
      iii. This inlet is applied to the boundary condition 'inlet_2' through the 'getfbr' subroutine
3. In the subroutine 'cs_user_les_inflow_advanced':
   (a) Initialise the variables concerning the inlet velocities, hydraulic diameters and other variables
   (b) Compute the mean velocity vector, Reynolds stresses and the dissipation rate

3 Running and analysing the simulation

In the folder 'Calculation management', select the 'Prepare batch calculation' sub-folder. In this subfolder's panel, the default selections of 'runcase' for the 'Script file', 'standard' (straight calculations without mesh or partitions import or processing) and '1' processor for the 'Calculation script parameters' are set. Change the number of processor to those required for this type of simulation.

Press the 'Start calculation' button to run *Code_Saturne*. The pop-up panel for the run opens, listing in real time the different stages of the calculation, from user-subroutines compilation to saving the results.

The calculation is initially run for 3s of real time in order to remove the initial field from the averaging that will follow after 3s. When the 3s of physical time is reached, you can proceed with time-averaging process.

4 Time averages

In this tutorial, you will want to compute the average of the temperature, the velocity components and the Reynolds stresses components. In the folder 'Calculation control', click on the 'Time averages' sub-folder. For the temperature time average, enter 'the label of time average' 'average_T' in the panel, leave the other parameters at their default values and select the variable 'TempC' from the drop-down menu. Then, click on the icon in order to obtain the final set-up shown in Figure VI.2 and click on 'Add'.

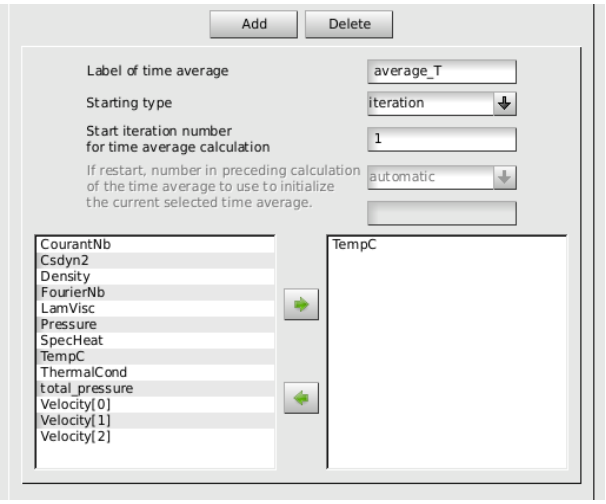


Figure VI.2: Temperature time average.

Repeat the process for the velocity components, as shown in the Figure VI.3.

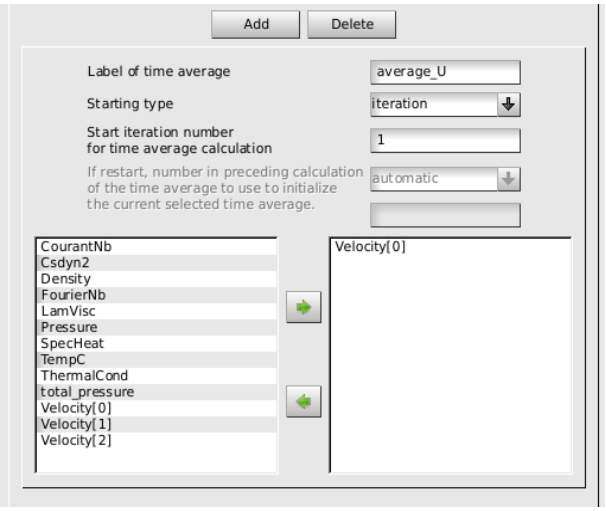


Figure VI.3: Velocity time average.

This functionality allows the calculation of time averages of the type $< f1 * f2... * fn >$. For the

component u'u' of the Reynolds stress, specify a 'label of time average' 'average_U_U' and insert twice the 'VelocityX' from the drop-down menu (Figure VI.4).

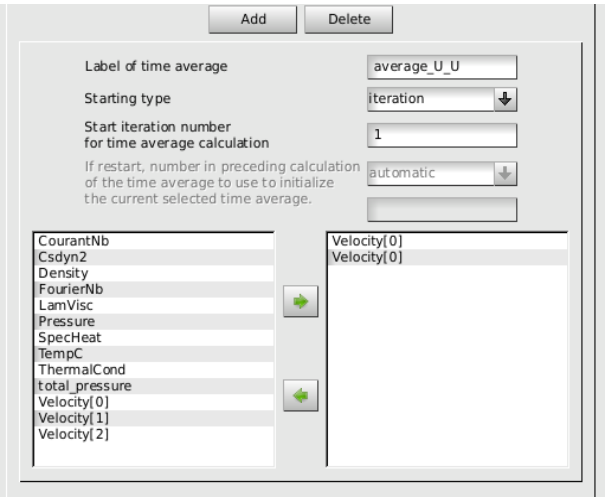


Figure VI.4: u'u' time average.

It is possible to modify a time average of a variable. First, you select the average to be modified by clicking on it. Then, you can change that which is necessary and confirm the change by clicking on 'Modify'. The list of the time averages of variables necessary for the comparison with experimental data is listed in the Table VI.1.

Number	Average name	Variables
1	average_U	< VelocityX >
2	average_V	<VelocityY >
3	average_W	<VelocityZ >
4	average_T	<TempC >
5	average_U_U	<VelocityX * VelocityX >
6	average_V_V	<VelocityY * VelocityY >
7	average_W_W	<VelocityZ * VelocityZ >
8	average_U_W	<VelocityX * VelocityZ >

Table VI.1: List of all time averages.

Having set-up the averages to be calculated, return to the 'Numerical parameters' ⇒ 'Time step' sub-folder. Keep the time step constant at 0.00015s and choose 53334 iterations in order to simulate 8s of physical time as shown in Figure VI.5.

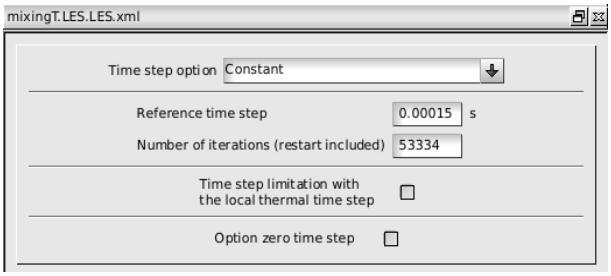


Figure VI.5: Time step menu.

Go to the 'Calculation management' folder and select the 'Start/Restart' sub-folder. In the 'Start/Restart' menu, click on the 'on' icon for the 'Calculation restart', then click on the icon in order to select the checkpoint folder of the previous calculation. The directory structure of the 'checkpoint' folder should be MixingTee/LES/RESU/DateOfRunTimeOfLastRun/checkpoint.

Now, press the 'Start calculation' button to run *Code_Saturne* from the sub-folder 'Prepare batch calculation'.

5 Results Analysis

For the post-processing of the results, follow the instructions given in 4.4.

Chapter VII

References

1 References

[1] www.salome-platform.org

[2] F. Archambeau, N. Méchitoua, M. Sakiz,
Code_Saturne: a Finite Volume Code for the Computation of Turbulent Incompressible Flows - Industrial Applications,
International Journal on Finite Volumes, Vol. 1, 2004.

[3] www.code-saturne.org

[4] -
Experimental data of T-Junction benchmark,
VATTENFALL R&D, 2009.

[5] EDF,
Tutorial 1: Shear Driven Cavity Flow,
Code_Saturne Tutorial Series

[6] http://docs.salome-platform.org/salome_6_6_0/gui/HEXABLOCK/index.html

[7] Incropera and DeWitt,
Heat and Mass transfer,
Fourth edition, 2000.

[8] N. Jarrin, S. Benhamadouche, D. Laurence, R. Prosser,
A synthetic-eddy-method for generating inflow conditions for large-eddy simulations,
International Journal of Heat and Fluid Flow, Volume 27, Issue 4, August 2006, Pages 585-593

[9] EDF,
Tutorial 3: Heated Square Cavity Flow,
Code_Saturne Tutorial Series

Chapter VIII

Appendix

1 Appendix A – Experimental Data from [4]

Experimental data from [4] are listed in the Table VIII.1 to Table VIII.9. The Table VIII.1 presents the dimensionless temperature along the top of the main pipe at 0deg and along the bottom of the main pipe at 180deg. The Table VIII.2 to Table VIII.5 are the U and V components of the velocity and the Reynolds stresses on horizontal lines at x=1.6D, 2.6D, 3.6D and 4.6D respectively. The Table VIII.6 to Table VIII.9 are the U and V components of the velocity and the Reynolds stresses on vertical lines at x=1.6D, 2.6D, 3.6D and 4.6D respectively.

x	0deg (y=0.07)	180deg (y=-0.07)
0.28	0.5950	0.0051
0.56	0.5274	0.0769
0.84	0.5423	0.2561
1.12	0.4737	0.2906
1.4	NaN?	0.3219
2.1	NaN?	0.3502
2.8	0.4224	0.3791

Table VIII.1: Experimental dimensionless temperature along the lines defined in the main pipe at (y=0.07; z=0) and (y=-0.07; z=0) [4] [4].

y	U	V	uu	vv	uv
-0.068803501	1.220961266	0.0080508	0.016806403	0.003684748	-0.000320576
-0.065143373	1.272156037	0.036178944	0.013349062	0.005739933	0.000132636
-0.061483245	1.287670348	0.007311955	0.014539606	0.007853589	0.000463675
-0.057823117	1.300325409	-0.002032071	0.017086371	0.010877009	0.000619996
-0.054162989	1.305862451	-0.009434521	0.02108371	0.016523966	0.001532841
-0.050502861	1.30103421	-0.012886409	0.028432521	0.02160839	0.00509258
-0.046842733	1.290144564	-0.006630974	0.037175548	0.026161018	0.007402013
-0.043182604	1.263691319	-0.002114936	0.048615376	0.032273092	0.009368141
-0.039522476	1.22199861	0.003949942	0.064236241	0.041531123	0.011279648
-0.035862348	1.168973395	0.005407237	0.082154374	0.049030202	0.013744189
-0.03220222	1.112672368	0.004366237	0.102669408	0.056323366	0.017127909
-0.028542092	1.047415842	0.006996903	0.122138707	0.064871654	0.01660063
-0.024881964	0.988924841	0.012906095	0.129131566	0.074142514	0.013766377
-0.021221836	0.927699906	0.01423193	0.142070022	0.081606706	0.014728216
-0.017561708	0.860383309	0.016585659	0.154439636	0.086292057	0.014205319
-0.01390158	0.808679486	0.022463178	0.159971359	0.08678276	0.012338107
-0.010241451	0.769015946	0.01556741	0.158728468	0.091342135	0.011248339
-0.006581323	0.740261152	0.010405492	0.158311994	0.091807149	0.010732402
-0.002921195	0.723447658	0.010103166	0.15879933	0.094939579	0.008482735
0.000738933	0.710321741	0.014230932	0.159189175	0.095000866	0.006181472
0.004399061	0.717625555	0.014685395	0.157553876	0.093895669	0.003943113
0.008059189	0.746772417	0.010404808	0.15904384	0.09196193	0.003450575
0.011719317	0.782873507	-0.003508939	0.162104348	0.088615309	0.002228167
0.015379445	0.841142639	-0.003193653	0.162902251	0.086685592	-0.004351654
0.019039574	0.901711036	-0.002126663	0.163693601	0.083417332	-0.005159765
0.022699702	0.978435006	-0.007529482	0.14988788	0.077438934	-0.010079361
0.02635983	1.048777782	-0.005956403	0.130931978	0.068254219	-0.014573342
0.030019958	1.118789385	-0.006130006	0.104269032	0.060250078	-0.013488905
0.033680086	1.182609871	-0.01020512	0.082532171	0.047382859	-0.008778693
0.037340214	1.236606974	-0.001066346	0.057044378	0.03904077	-0.009019601
0.041000342	1.279031712	0.006137361	0.041672372	0.033292919	-0.008095095
0.04466047	1.308363214	-0.003099608	0.028704703	0.024498138	-0.00419303
0.048320598	1.319369484	-0.001702236	0.022576626	0.018235141	-0.00356895
0.051980727	1.321133913	-0.000665381	0.016679991	0.012408706	-0.001077844
0.055640855	1.31053268	-0.007085745	0.01391511	0.008643992	-0.000300852
0.059300983	1.286586848	-0.016616003	0.012555618	0.005567511	-9.98E-05
0.062961111	1.236213676	-0.025118743	0.017471484	0.002323452	0.000660312

Table VIII.2: Experimental data along a horizontal line defined at x = 1.6D and z=0 [4].

y	U	V	uu	vv	uv
-0.069368535	1.082202739	0.026302991	0.038695911	0.007840121	0.002295221
-0.065708407	1.131046578	0.035433371	0.028031243	0.009267074	0.003859765
-0.062048279	1.166985976	0.037071193	0.025640287	0.013675077	0.00540781
-0.058388151	1.171782666	0.038566529	0.031786283	0.016835954	0.006296806
-0.054728023	1.167348604	0.036674632	0.032538149	0.020614559	0.00786736
-0.051067895	1.149949236	0.033123758	0.037408521	0.026404955	0.009072984
-0.047407766	1.123811353	0.031810201	0.045428415	0.033126556	0.011185124
-0.043747638	1.099509644	0.028389849	0.05226019	0.037951129	0.014134049
-0.04008751	1.071998363	0.029369624	0.060218483	0.042570819	0.016360686
-0.036427382	1.036329117	0.032669894	0.065899146	0.048751158	0.015959809
-0.032767254	0.998930657	0.034466616	0.073064337	0.052231505	0.016730107
-0.029107126	0.95867343	0.027886514	0.079520063	0.055805322	0.016930019
-0.025446998	0.925338541	0.029814858	0.080342487	0.057292124	0.013045066
-0.02178687	0.893781719	0.030764208	0.083688938	0.058288418	0.011910962
-0.018126741	0.861278943	0.032316221	0.083220585	0.059901962	0.013117824
-0.014466613	0.835528676	0.033209271	0.082663928	0.060811253	0.010500281
-0.010806485	0.813123652	0.034404578	0.080569423	0.061493656	0.008102665
-0.007146357	0.798648807	0.031215134	0.078576249	0.063455177	0.007179793
-0.003486229	0.793086657	0.028231063	0.07558212	0.063773323	0.004646465
0.000173899	0.789686558	0.028559546	0.073580275	0.063062361	0.001877638
0.003834027	0.791760175	0.025706662	0.076202974	0.064225946	-6.80E-05
0.007494155	0.807564044	0.024254404	0.080692485	0.063358101	-0.002722524
0.011154283	0.826577421	0.027537324	0.083442747	0.062115561	-0.003618879
0.014814412	0.857432698	0.030286232	0.082439545	0.065210008	-0.006605145
0.01847454	0.893976145	0.026540454	0.079182009	0.064305459	-0.009463321
0.022134668	0.929206961	0.023501275	0.077320256	0.060805833	-0.012073691
0.025794796	0.962237057	0.013935585	0.077674665	0.056827215	-0.012094994
0.029454924	1.000052663	0.003634836	0.076472514	0.051245221	-0.010797203
0.033115052	1.039073637	-0.003050727	0.071488631	0.046273464	-0.011956342
0.03677518	1.079924784	-0.00253535	0.065033301	0.041336503	-0.013450124
0.040435308	1.118071692	-0.004411852	0.055176102	0.036497281	-0.012149277
0.044095437	1.145131275	-0.004150723	0.045473452	0.03269635	-0.01196506
0.047755565	1.165485328	-0.004920335	0.036857727	0.028393736	-0.010486433
0.051415693	1.185875494	-0.008496849	0.027216309	0.022528539	-0.008366044
0.055075821	1.194083949	-0.013409531	0.025368073	0.016344231	-0.006755233
0.058735949	1.162226174	0.001728245	0.029157959	0.01099537	-0.004571036
0.062396077	1.094684941	0.006786894	0.038650217	0.010413713	-0.00192275

Table VIII.3: Experimental data along a horizontal line defined at x = 2.6D and z=0 [4].

y	U	V	uu	vv	uv
-0.064942165	0.999567818	0.009954602	0.029016388	0.008540657	9.47E-05
-0.061218133	1.057757626	0.004135579	0.028858794	0.0103888	0.000910212
-0.057494101	1.072924419	0.008392337	0.030050541	0.013413533	0.00196188
-0.053770069	1.090139419	0.006314843	0.031057516	0.016118005	0.003488428
-0.050046037	1.102540116	0.005657113	0.031159201	0.018934966	0.004500034
-0.046322005	1.099709414	0.0040022	0.031816112	0.021012532	0.00586983
-0.042597973	1.093006917	0.008850913	0.030656428	0.018450632	0.005894702
-0.038873941	1.074345773	0.01021927	0.029318075	0.020392342	0.005700237
-0.035149909	1.060961484	0.005879035	0.030851	0.027083946	0.006987888
-0.031425877	1.045948346	0.003944355	0.031528389	0.02896152	0.007662895
-0.027701845	1.030735964	0.00285597	0.032496777	0.030275366	0.008154697
-0.023977813	1.020657541	0.003178069	0.034416098	0.03226989	0.005764448
-0.020253782	1.008273501	0.000930945	0.035054195	0.030847171	0.005836718
-0.01652975	1.002084348	0.00083683	0.034728355	0.031412221	0.005682902
-0.012805718	0.990839278	0.004780686	0.034036449	0.032887815	0.004158329
-0.009081686	0.982275657	0.007802901	0.033325668	0.034269964	0.002942319
-0.005357654	0.979729863	0.008560281	0.032865986	0.035012801	0.001258851
-0.001633622	0.974561526	0.005312477	0.033022983	0.034468453	0.000776853
0.00209041	0.97212795	0.00338701	0.034373612	0.034706914	0.00010455
0.005814442	0.975179785	0.002752661	0.034370339	0.03470714	-0.002198185
0.009538474	0.979582738	-0.001098453	0.034075185	0.033589896	-0.002365536
0.013262506	0.984652046	0.000404763	0.03476567	0.033805428	-0.003794216
0.016986538	0.994663571	0.000487668	0.036310626	0.033211698	-0.005896862
0.02071057	1.009783672	-0.004768269	0.037126843	0.031658488	-0.005443132
0.024434602	1.02254729	-0.006339431	0.03466642	0.031340528	-0.006791351
0.028158633	1.032503474	-0.009624849	0.034369533	0.0288003	-0.007398747
0.031882665	1.046962452	-0.023746891	0.033970426	0.020544993	-0.006235022
0.035606697	1.061100946	-0.029393032	0.033802898	0.015907466	-0.00537082
0.039330729	1.073241912	-0.013872118	0.033774269	0.018664718	-0.00645483
0.043054761	1.08612208	-0.038007553	0.034490763	0.012213388	-0.004446262
0.046778793	1.092421819	-0.062375854	0.035369231	0.007174623	-0.002727415
0.050502825	1.079983638	-0.041756481	0.032887519	0.009172826	-0.003020898
0.054226857	1.072927223	-0.010031112	0.03253559	0.012690408	-0.001670477
0.057950889	1.078399857	-0.006157044	0.036098903	0.012445449	-0.000642243
0.061674921	1.001946727	-0.012585638	0.036291396	0.008511331	0.000312972

Table VIII.4: Experimental data along a horizontal line defined at x = 3.6D and z=0 [4].

y	U	V	uu	vv	uv
-0.066407476	0.988708927	-0.015767472	0.027101743	0.00907339	-0.000416653
-0.062882707	1.017737232	-0.011512028	0.024849209	0.009340111	-8.32E-05
-0.059357939	1.05340401	-0.008319259	0.027504711	0.011023551	0.000955366
-0.055833171	1.078246118	-0.006477538	0.029499445	0.012934425	0.001404052
-0.052308403	1.093848403	-0.005790252	0.031196558	0.014276329	0.001699225
-0.048783635	1.100732808	-0.005465027	0.032311178	0.016288499	0.002408012
-0.045258867	1.105119368	-0.003335601	0.03087782	0.01786313	0.004085187
-0.041734099	1.102725302	-0.003186139	0.03085378	0.019670465	0.004391908
-0.038209331	1.098852833	-0.001409321	0.030553301	0.021950381	0.004748426
-0.034684563	1.096729814	-0.002413396	0.029890226	0.022742538	0.004907242
-0.031159794	1.091875875	-0.00573699	0.028576069	0.023714839	0.004788496
-0.027635026	1.087205502	-0.006791523	0.029298409	0.024855936	0.00448303
-0.024110258	1.082276254	-0.006576663	0.028929775	0.026695223	0.003880063
-0.02058549	1.079079469	-0.010419375	0.027801067	0.027087746	0.00375633
-0.017060722	1.075195002	-0.008924121	0.027596131	0.027613351	0.003183658
-0.013535954	1.06859212	-0.00753008	0.028256989	0.027858687	0.002847416
-0.010011186	1.065172938	-0.007426849	0.029327659	0.028392772	0.00294188
-0.006486418	1.061969967	-0.006351563	0.028493244	0.02795662	0.001799254
-0.00296165	1.060396701	-0.002560392	0.027950139	0.027596715	0.000867593
0.000563119	1.059204217	0.000471791	0.02820745	0.026398808	-0.000173785
0.004087887	1.05965793	-0.002139525	0.028077979	0.025757034	-0.000887411
0.007612655	1.059894843	-0.004035707	0.027265874	0.024594476	-0.00210305
0.011137423	1.059361401	-0.002693754	0.027511492	0.024751051	-0.00290675
0.014662191	1.062996836	-0.002728297	0.027582325	0.02491474	-0.003878176
0.018186959	1.071019346	-0.004423416	0.027974569	0.025718191	-0.004579473
0.021711727	1.077721766	-0.00673771	0.029100437	0.024998186	-0.00429708
0.025236495	1.080524357	-0.008494971	0.030396043	0.023228166	-0.00440375
0.028761263	1.08413835	-0.008220301	0.030732884	0.022283598	-0.00409482
0.032286032	1.090107392	-0.007397497	0.031180065	0.022010863	-0.004104643
0.0358108	1.096126295	-0.007186596	0.029732098	0.020785226	-0.004306261
0.039335568	1.104678601	-0.008159286	0.029849885	0.018718804	-0.004006554
0.042860336	1.103894488	-0.009366057	0.029981621	0.016448562	-0.003226551
0.046385104	1.101549312	-0.01124995	0.029582293	0.015731062	-0.002816598
0.049909872	1.093950886	-0.012199547	0.028284045	0.014719867	-0.002630838
0.05343464	1.080434602	-0.010639156	0.027282761	0.013268299	-0.00234641
0.056959408	1.058969242	-0.009022812	0.025724699	0.011044458	-0.002042546
0.060484177	1.02069144	-0.005366073	0.022281618	0.008210158	-0.000836225
0.064008945	0.957525994	0.00211208	0.019278774	0.005120943	2.72E-05
0.067533713	0.912716686	0.005829673	0.021250605	0.004642221	0.000363108

Table VIII.5: Experimental data along a horizontal line defined at x = 4.6D and z=0 [4].

z	U	W	uu	ww	uw
-0.057941698	1.299230276	0.005484151	0.007248031	0.000738478	-0.000484864
-0.054202146	1.303399456	0.006565643	0.005160965	0.001167607	-0.000466642
-0.050462594	1.31825248	0.001318495	0.005294526	0.001927864	-0.000392073
-0.046723042	1.332934409	-0.000144944	0.005115544	0.003070585	-0.000321822
-0.04298349	1.340205895	-0.002834759	0.005075901	0.004466527	-0.000417118
-0.039243938	1.343765663	-0.006040463	0.00600479	0.005314762	-0.000481472
-0.035504386	1.346458191	-0.005978749	0.008057735	0.009069329	-0.0002282
-0.031764834	1.348203934	-0.005398402	0.010800535	0.01551128	0.000595873
-0.028025282	1.343766348	-0.004069757	0.0141533	0.020495686	0.002345698
-0.02428573	1.332978414	-0.003903595	0.021049004	0.027576721	0.006448853
-0.020546178	1.302675383	-0.00201396	0.033014636	0.037547422	0.013313782
-0.016806626	1.252098796	0.009208601	0.054063247	0.045872856	0.01745912
-0.013067074	1.180285599	0.021407309	0.075175701	0.051505651	0.024939144
-0.009327522	1.08714842	0.032207453	0.101273666	0.061625824	0.035954727
-0.00558797	0.969093869	0.04376598	0.126493337	0.072589039	0.046151297
-0.001848418	0.825969274	0.056140321	0.14680867	0.080277523	0.055018431
0.001891134	0.683011373	0.055059503	0.155675174	0.085950986	0.059698685
0.005630686	0.553105388	0.045778981	0.150463047	0.097490404	0.061725571
0.009370238	0.440625106	0.037230749	0.134530036	0.099694123	0.058191066
0.01310979	0.350977732	0.027165431	0.11415845	0.096573072	0.053667382
0.016849342	0.277133281	0.008756674	0.096044309	0.098374744	0.043618259
0.020588894	0.223709003	-0.014293709	0.08318708	0.100735911	0.033467123
0.024328446	0.187033073	-0.045226737	0.0722462	0.094233767	0.024249567
0.028067998	0.170423448	-0.074620019	0.069468475	0.090845561	0.016793216
0.03180755	0.178828428	-0.083358738	0.071252722	0.089303059	0.007780511
0.035547102	0.190396673	-0.074239935	0.080617551	0.094041342	0.004708013
0.039286654	0.208925107	-0.07843283	0.082723581	0.092887227	0.004345919
0.043026206	0.222960679	-0.056427728	0.082688577	0.058345413	0.000952234
0.05050531	0.271781481	-0.083948916	0.091173672	0.091876628	0.001153354
0.054244862	0.293809755	-0.079245639	0.090618306	0.09108081	0.000808216
0.057984414	0.308942333	-0.049319135	0.089258737	0.074694788	0.004600543
0.061723966	0.202378828	-0.041475049	0.083635529	0.045666678	-0.000930679
0.065463518	-0.32674463	-0.070635862	0.074941505	0.009692831	-0.007211437
0.06920307	-0.403207308	0.005703933	0.06141292	0.036366361	0.006862185

Table VIII.6: Experimental data along a vertical line defined at x = 1.6D and z=0 [4].

z	U	W	uu	ww	uw
-0.064849161	1.13795496	-0.023221025	0.014044541	0.003286613	0.001093659
-0.061016841	1.244474561	0.028931712	0.00779616	0.001854919	0.000750882
-0.057184521	1.240358061	0.012574452	0.007214413	0.002341483	0.000596236
-0.053352201	1.254985653	0.013146271	0.008490619	0.004348931	0.000733625
-0.049519881	1.263694338	0.013630143	0.00979187	0.007217198	0.000623434
-0.045687561	1.270525964	0.013788982	0.011720505	0.011283437	0.000812327
-0.041855241	1.269726042	0.017565204	0.013170967	0.016514827	0.002780935
-0.038022921	1.261676105	0.021009896	0.017154458	0.020606149	0.006509786
-0.034190601	1.247073416	0.024704287	0.021783436	0.025394146	0.008417038
-0.030358281	1.220727662	0.025552301	0.029600595	0.032008161	0.010696956
-0.026525961	1.183815442	0.027288459	0.038755819	0.038058227	0.014842882
-0.022693641	1.1402731	0.028565516	0.048711293	0.04312766	0.017530434
-0.018861321	1.0880853	0.022380214	0.062010625	0.049957166	0.020923879
-0.015029001	1.032472881	0.019067763	0.073162662	0.055515221	0.025730123
-0.011196681	0.970021553	0.014991577	0.079651538	0.060572618	0.029245822
-0.007364361	0.904516672	0.016115594	0.082356461	0.060930905	0.029971296
-0.003532041	0.848562561	0.018027047	0.081702251	0.064283012	0.029269519
0.000300279	0.807688424	0.015572434	0.07903627	0.06521071	0.027782595
0.004132599	0.766144846	0.007777763	0.074096724	0.064418937	0.026076668
0.007964919	0.730135993	-0.000884641	0.068779029	0.065383609	0.022949421
0.011797239	0.695068572	-0.0100499	0.064745564	0.064528809	0.02056488
0.015629559	0.671260793	-0.017179767	0.062468198	0.065160748	0.014531556
0.019461879	0.654174038	-0.027391057	0.06039031	0.066627987	0.008751327
0.023294199	0.65406777	-0.037502789	0.054817566	0.063877172	0.009278693
0.027126519	0.659236079	-0.042084778	0.054661524	0.064704597	0.004968933
0.030958839	0.668067714	-0.047481811	0.053006221	0.066044694	0.004046325
0.034791158	0.672566687	-0.060068824	0.051855638	0.060156955	-0.00072256
0.038623478	0.683179932	-0.061806909	0.052315479	0.060237999	-0.001485697
0.042455798	0.695455638	-0.04692092	0.05188848	0.066967154	-0.003842083
0.046288118	0.708079991	-0.045423895	0.050782677	0.060418211	-0.001198328
0.050120438	0.719294332	-0.045061921	0.045189539	0.052917085	-0.001165429
0.053952758	0.721841866	-0.034038865	0.053317474	0.045428407	0.000582388
0.057785078	0.713718248	-0.022072876	0.048511619	0.037776053	0.00365118
0.061617398	0.691262241	-0.010542674	0.040146677	0.029209274	0.002803962
0.065449718	0.639828205	-0.006282634	0.043543137	0.019438015	0.004646402
0.069282038	0.594443458	0.000531982	0.052336831	0.016732245	0.00517291

Table VIII.7: Experimental data along a vertical line defined at x = 2.6D and z=0 [4].

z	U	W	uu	ww	uw
-0.058819134	1.174140906	-0.024960367	0.02015271	0.007043262	-0.001326338
-0.054993342	1.189838388	-0.004860482	0.018008582	0.008696553	0.002206541
-0.05116755	1.197005732	0.024448882	0.01938581	0.012812955	0.004351747
-0.047341757	1.195991102	0.02920367	0.018781665	0.016230602	0.003842177
-0.043515965	1.178911667	0.030656873	0.019342832	0.018075243	0.00467684
-0.039690173	1.165475078	0.031574053	0.020595003	0.019148968	0.006288433
-0.035864381	1.146652721	0.033572284	0.023691164	0.021235466	0.006827855
-0.032038589	1.12487831	0.031554095	0.030465578	0.025109897	0.011265029
-0.028212797	1.100225152	0.027143254	0.034733989	0.028936209	0.013276452
-0.024387005	1.077919913	0.033796561	0.035263865	0.027494393	0.011470188
-0.020561213	1.052692514	0.03302582	0.038390266	0.029364447	0.012493781
-0.016735421	1.027339256	0.030935498	0.040564122	0.031932192	0.012034631
-0.012909629	1.00562469	0.02854156	0.038748064	0.034085697	0.012439064
-0.009083837	0.984463646	0.024639788	0.038563565	0.033372301	0.01359141
-0.005258045	0.963881079	0.024531336	0.04124035	0.036975615	0.014218442
-0.001432253	0.95511166	0.026902799	0.042644081	0.037582754	0.01362075
0.00239354	0.946271224	0.026752756	0.037589378	0.03989348	0.010220166
0.006219332	0.935926102	0.022182644	0.039412686	0.037353247	0.010185531
0.010045124	0.921577914	0.019839273	0.037250313	0.03560414	0.006926185
0.013870916	0.918082949	0.014442475	0.040207378	0.036639478	0.00888558
0.017696708	0.922123174	0.007440122	0.034728224	0.035793072	0.005979122
0.0215225	0.922122291	-0.002295966	0.03033082	0.033744594	0.004045389
0.025348292	0.923916393	-0.009900135	0.030529553	0.035210127	0.004948066
0.029174084	0.922077112	-0.004179909	0.031311266	0.036528363	0.002788234
0.032999876	0.925965744	0.021667536	0.041482848	0.042270421	0.000266185
0.036825668	0.925626562	0.030731423	0.040937297	0.037228858	0.001922627
0.04065146	0.91885609	1.99E-05	0.032916456	0.034077924	0.001892577
0.044477252	0.919770565	-0.005403114	0.028090257	0.028815386	0.00305018
0.048303044	0.904924752	-0.004589232	0.031778399	0.02301714	0.002204115
0.052128836	0.88943742	-0.006831871	0.031342636	0.019446519	0.00275302
0.055954629	0.87238195	-0.008021649	0.025046944	0.016300146	0.003471058
0.059780421	0.845117415	-0.00918554	0.027477551	0.011847637	0.00371154
0.063606213	0.787023619	-0.012831916	0.024701696	0.007277638	0.001211547
0.067432005	0.733229562	-0.025083887	0.03598198	0.008001107	0.001761846

Table VIII.8: Experimental data along a vertical line defined at x = 3.6D and z=0 [4].

z	U	W	uu	ww	uw
-0.061874809	0.959484147	-0.00803782	0.017808776	0.003799387	-0.001768154
-0.058029305	1.027774842	-0.002837923	0.014724706	0.002785074	-0.000241262
-0.0541838	1.105972039	0.004710245	0.014093984	0.004418722	0.00055353
-0.050338296	1.15261044	0.013032161	0.014477012	0.007530744	0.002085263
-0.046492792	1.173580614	0.017857926	0.015053429	0.008470849	0.003066095
-0.042647288	1.18030621	0.022515622	0.016384305	0.011010336	0.003572255
-0.038801784	1.177434184	0.017314513	0.017625569	0.011834828	0.004653462
-0.027265272	1.14541713	-0.004437967	0.021681239	0.018039813	0.006292412
-0.023419768	1.130980222	0.028418203	0.023028971	0.019095729	0.007555947
-0.019574264	1.112369546	0.037890307	0.023974323	0.02019204	0.007945708
-0.01572876	1.09608106	0.036452997	0.026173637	0.020646666	0.007540846
-0.011883256	1.08397857	0.038556822	0.025435552	0.021035328	0.00746682
-0.008037752	1.067283783	0.039650554	0.02581529	0.022161261	0.00880889
-0.004192248	1.05247221	0.038677725	0.027094948	0.022775288	0.008364076
-0.000346744	1.038556782	0.038038705	0.02862014	0.024613821	0.00845454
0.00349876	1.025708079	0.038537671	0.028215624	0.024290695	0.007877187
0.007344264	1.012486929	0.037074351	0.027543184	0.024125407	0.006198288
0.011189768	0.997577598	0.035321958	0.026326366	0.022751142	0.006809415
0.015035272	0.986295982	0.032245093	0.025199213	0.021961937	0.005710094
0.018880776	0.97682535	0.031515042	0.024631428	0.021822003	0.004751203
0.02272628	0.971966011	0.02796159	0.0252851	0.023665226	0.004191861
0.026571784	0.967849132	0.025452988	0.024542438	0.023711228	0.003596782
0.030417289	0.967755984	0.011373205	0.022727799	0.02425152	0.002176046
0.034262793	0.968542002	0.019260048	0.022228277	0.02254682	0.001481359
0.049644809	0.955932563	0.010731949	0.018758393	0.017270213	0.001404984
0.053490313	0.944472245	0.010614739	0.02119731	0.016755693	0.003961212
0.057335817	0.928999302	0.007730796	0.020028418	0.013481031	0.002399826
0.061181321	0.919018673	-0.000799028	0.019286587	0.011085464	0.002200096
0.065026825	0.90848081	0.001957227	0.019911026	0.009522656	0.002499238
0.068872329	0.845005453	-0.025735644	0.021397975	0.006245586	0.001501816

Table VIII.9: Experimental data along a vertical line defined at x = 4.6D and z=0 [4].

2 Appendix B – Script SALOME.

```
# -*- coding: utf-8 -*-

###
### This file is generated automatically by SALOME v7.6.0 with dump
    python functionality
###

import sys
import salome

salome.salome_init()
theStudy = salome.myStudy

import salome_notebook
notebook = salome_notebook.NoteBook(theStudy)
sys.path.insert( 0, r'/home/brunows1/Documents/2016/EDF/Tutorial/
    Tutorial5-Mixing_Tee/new decomposition')
```

```
###
### GEOM component
###

import GEOM
from salome.geom import geomBuilder
import math
import SALOMEDS


geompy = geomBuilder.New(theStudy)

O = geompy.MakeVertex(0, 0, 0)
OX = geompy.MakeVectorDXDYDZ(1, 0, 0)
OY = geompy.MakeVectorDXDYDZ(0, 1, 0)
OZ = geompy.MakeVectorDXDYDZ(0, 0, 1)
Vertex_1 = geompy.MakeVertex(0, 0.05, 0.14)
Vertex_2 = geompy.MakeVertex(0, 0.025, 0.14)
Line_1 = geompy.MakeLineTwoPnt(Vertex_1, Vertex_2)
Rotation_1 = geompy.MakeRotation(Line_1, OZ, 45*math.pi/180.0)
Rotation_2 = geompy.MakeRotation(Rotation_1, OZ, 90*math.pi/180.0)
Rotation_3 = geompy.MakeRotation(Rotation_2, OZ, 45*math.pi/180.0)
Vertex_3 = geompy.MakeVertex(0, 0, 0.14)
Rotation_3_vertex_2 = geompy.GetSubShape(Rotation_3, [2])
Rotation_2_vertex_2 = geompy.GetSubShape(Rotation_2, [2])
Arc_1 = geompy.MakeArcCenter(Vertex_3, Rotation_3_vertex_2,
    Rotation_2_vertex_2,False)
Arc_1_vertex_3 = geompy.GetSubShape(Arc_1, [3])
Rotation_1_vertex_2 = geompy.GetSubShape(Rotation_1, [2])
Arc_2 = geompy.MakeArcCenter(Vertex_3, Arc_1_vertex_3,
    Rotation_1_vertex_2,False)
Arc_2_vertex_3 = geompy.GetSubShape(Arc_2, [3])
Line_1_vertex_2 = geompy.GetSubShape(Line_1, [2])
Arc_3 = geompy.MakeArcCenter(Vertex_3, Arc_2_vertex_3, Line_1_vertex_2,
    False)
Rotation_3_vertex_3 = geompy.GetSubShape(Rotation_3, [3])
Rotation_2_vertex_3 = geompy.GetSubShape(Rotation_2, [3])
Line_2 = geompy.MakeLineTwoPnt(Rotation_3_vertex_3, Rotation_2_vertex_3)
Rotation_1_vertex_3 = geompy.GetSubShape(Rotation_1, [3])
Line_3 = geompy.MakeLineTwoPnt(Rotation_2_vertex_3, Rotation_1_vertex_3)
Line_3_vertex_3 = geompy.GetSubShape(Line_3, [3])
Line_1_vertex_3 = geompy.GetSubShape(Line_1, [3])
Line_4 = geompy.MakeLineTwoPnt(Line_3_vertex_3, Line_1_vertex_3)
Line_4_vertex_3 = geompy.GetSubShape(Line_4, [3])
Line_2_vertex_2 = geompy.GetSubShape(Line_2, [2])
Line_5 = geompy.MakeLineTwoPnt(Line_4_vertex_3, Line_2_vertex_2)
Face_1 = geompy.MakeFaceWires([Line_2, Line_3, Line_4, Line_5], 1)
Face_2 = geompy.MakeFaceWires([Rotation_2, Rotation_3, Arc_1, Line_2], 1)
Face_3 = geompy.MakeFaceWires([Rotation_1, Rotation_2, Arc_2, Line_3], 1)
Face_4 = geompy.MakeFaceWires([Line_1, Rotation_1, Arc_3, Line_4], 1)
plan_yz = geompy.MakeFaceHW(0.05, 0.05, 2)
plan_xy = geompy.MakeFaceHW(0.05, 0.05, 1)
plan_zx = geompy.MakeFaceHW(0.05, 0.05, 3)
Mirror_1_1 = geompy.MakeMirrorByPlane(Face_1, plan_yz)
```

```
Mirror_1_2 = geompy.MakeMirrorByPlane(Face_2, plan_yz)
Mirror_1_3 = geompy.MakeMirrorByPlane(Face_3, plan_yz)
Mirror_1_4 = geompy.MakeMirrorByPlane(Face_4, plan_yz)
Face_supp_small_pipe = geompy.MakeCompound([Face_1, Face_2, Face_3,
    Face_4, Mirror_1_1, Mirror_1_2, Mirror_1_3, Mirror_1_4])
Scale_1 = geompy.MakeScaleTransform(Face_supp_small_pipe, Vertex_3, 1.4)
Translation_1 = geompy.MakeTranslation(Scale_1, 0, 0, -0.14)
Rotation_4 = geompy.MakeRotation(Translation_1, OY, 90*math.pi/180.0)
[Face_5,Face_6,Face_7,Face_8,Face_9,Face_10,Face_11,Face_12] = geompy.
    ExtractShapes(Rotation_4, geompy.ShapeType["FACE"], True)
Extrusion_1 = geompy.MakePrismVecH2Ways(Rotation_4, OX, 0.14)
Box_1 = geompy.MakeBoxDXDYDZ(0.21, 0.21, 0.21)
Translation_2 = geompy.MakeTranslation(Box_1, 0, -0.105, 0)
Rotation_5 = geompy.MakeRotation(Translation_2, OY, -45*math.pi/180.0)
Disk_1 = geompy.MakeDiskR(0.07000000000000001, 2)
Extrusion_2 = geompy.MakePrismVecH2Ways(Disk_1, OX, 0.42)
Rotation_6 = geompy.MakeRotation(Extrusion_2, OX, 45*math.pi/180.0)
Common_1 = geompy.MakeCommonList([Rotation_5, Rotation_6], True)
Extrusion_3 = geompy.MakePrismVecH(Face_supp_small_pipe, OZ, -0.14)
[Solid_1,Solid_2,Solid_3,Solid_4,Solid_5,Solid_6,Solid_7,Solid_8] =
    geompy.ExtractShapes(Extrusion_3, geompy.ShapeType["SOLID"], True)
Compound_1 = geompy.MakeCompound([Solid_1, Solid_2, Solid_3, Solid_4])
Cut_1 = geompy.MakeCutList(Solid_1, [Rotation_6], True)
[Edge_1,Edge_2,Edge_3,Edge_4,Edge_5,Edge_6,Edge_7,Edge_8,Edge_9,Edge_10,
    Edge_11,Edge_12,Edge_13] = geompy.ExtractShapes(Cut_1, geompy.
    ShapeType["EDGE"], True)
Cut_3 = geompy.MakeCutList(Solid_3, [Rotation_6], True)
Cut_4 = geompy.MakeCutList(Solid_4, [Rotation_6], True)
Cut_2 = geompy.MakeCutList(Solid_2, [Rotation_6], True)
[Edge_26,Edge_27,Edge_28,Edge_29,Edge_30,Edge_31,Edge_32,Edge_33,Edge_34,
    Edge_35,Edge_36,Edge_37] = geompy.ExtractShapes(Cut_2, geompy.
    ShapeType["EDGE"], True)
Extrusion_4 = geompy.MakePrismVecH(Face_1, OZ, -0.098)
[Edge_14,Edge_15,Edge_16,Edge_17,Edge_18,Edge_19,Edge_20,Edge_21,Edge_22,
    Edge_23,Edge_24,Edge_25] = geompy.ExtractShapes(Extrusion_4, geompy.
    ShapeType["EDGE"], True)
[Face_17,Face_18,Face_19,Face_20,Face_21,Face_22] = geompy.ExtractShapes(
    Extrusion_4, geompy.ShapeType["FACE"], True)
Edge_15_vertex_2 = geompy.GetSubShape(Edge_15, [2])
Edge_1_vertex_2 = geompy.GetSubShape(Edge_1, [2])
Line_6 = geompy.MakeLineTwoPnt(Edge_15_vertex_2, Edge_1_vertex_2)
Edge_15_vertex_3 = geompy.GetSubShape(Edge_15, [3])
Edge_3_vertex_3 = geompy.GetSubShape(Edge_3, [3])
Line_7 = geompy.MakeLineTwoPnt(Edge_15_vertex_3, Edge_3_vertex_3)
Fuse_1 = geompy.MakeFuseList([Edge_1, Edge_3], True, True)
geomObj_1 = geompy.MakeGlueEdges(Fuse_1, 0.001)
FuseEdges_1 = geompy.FuseCollinearEdgesWithinWire(Fuse_1, [])
Face_13 = geompy.MakeFaceWires([Line_6, Line_7, FuseEdges_1, Edge_15], 1)
Face_14 = geompy.MakeFaceWires([FuseEdges_1, Edge_2, Edge_4, Edge_5], 1)
Face_15 = geompy.MakeFaceWires([Line_6, Edge_4, Edge_7, Edge_14], 1)
Face_16 = geompy.MakeFaceWires([Line_7, Edge_5, Edge_9, Edge_17], 1)
Shell_1 = geompy.MakeShell([Face_3, Face_13, Face_14, Face_15, Face_16,
    Face_17])
Solid_9 = geompy.MakeSolid([Shell_1])
Edge_18_vertex_2 = geompy.GetSubShape(Edge_18, [2])
```

```
Edge_29_vertex_2 = geompy.GetSubShape(Edge_29, [2])
Line_8 = geompy.MakeLineTwoPnt(Edge_18_vertex_2, Edge_29_vertex_2)
Face_23 = geompy.MakeFaceWires([Line_8, Edge_34, Edge_36, Edge_22], 1)
Face_24 = geompy.MakeFaceWires([Edge_26, Edge_29, Edge_30, Edge_34], 1)
Line_6_vertex_2 = geompy.GetSubShape(Line_6, [2])
Face_25 = geompy.MakeFaceWires([Line_6, Line_8, Edge_29, Edge_18], 0)
Shell_2 = geompy.MakeShell([Face_2, Face_15, Face_23, Face_24, Face_25,
    Face_18])
Solid_10 = geompy.MakeSolid([Shell_2])
Mirror_1 = geompy.MakeMirrorByPlane(Solid_10, plan_zx)
verti = geompy.MakeCompound([Extrusion_4, Solid_9, Solid_10, Mirror_1])
Cut_5 = geompy.MakeCutList(Common_1, [Extrusion_4], True)
Cut_6 = geompy.MakeCutList(Cut_5, [Solid_9], True)
Cut_7 = geompy.MakeCutList(Cut_6, [Solid_10], True)
Cut_8 = geompy.MakeCutList(Cut_7, [Mirror_1], True)
Cylinder_1 = geompy.MakeCylinderRHA(0.1, 0.1, 45*math.pi/180.)
Rotation_7 = geompy.MakeRotation(Cylinder_1, OZ, 90*math.pi/180.0)
Common_2 = geompy.MakeCommonList([Cut_8, Rotation_7], True)
Rotation_8 = geompy.MakeRotation(Rotation_7, OZ, 45*math.pi/180.0)
Fuse_2 = geompy.MakeFuseList([Rotation_7, Rotation_8], True, True)
Rotation_9 = geompy.MakeRotation(Fuse_2, OZ, 45*math.pi/180.0)
Common_3 = geompy.MakeCommonList([Cut_8, Rotation_9], True)
Rotation_10 = geompy.MakeRotation(Rotation_8, OZ, 90*math.pi/180.0)
Common_4 = geompy.MakeCommonList([Cut_8, Rotation_10], True)
Extrusion_5 = geompy.MakePrismVecH(Rotation_4, OX, -0.14)
[Solid_11,Solid_12,Solid_13,Solid_14,Solid_15,Solid_16,Solid_17,Solid_18]
    = geompy.ExtractShapes(Extrusion_5, geompy.ShapeType["SOLID"], True)
Cut_12 = geompy.MakeCutList(Solid_15, [Rotation_5], True)
[Edge_38,Edge_39,Edge_40,Edge_41,Edge_42,Edge_43,Edge_44,Edge_45,Edge_46,
    Edge_47,Edge_48,Edge_49] = geompy.ExtractShapes(Cut_12, geompy.
    ShapeType["EDGE"], True)
[Face_30,Face_31,Face_32,Face_33,Face_34,Face_35] = geompy.ExtractShapes(
    Cut_12, geompy.ShapeType["FACE"], True)
geomObj_2 = geompy.GetSubShape(Cut_12, [6])
Extrusion_4_vertex_10 = geompy.GetSubShape(Extrusion_4, [10])
Line_9 = geompy.MakeLineTwoPnt(geomObj_2, Extrusion_4_vertex_10)
Cut_12_vertex_16 = geompy.GetSubShape(Cut_12, [16])
Extrusion_4_vertex_17 = geompy.GetSubShape(Extrusion_4, [17])
Line_10 = geompy.MakeLineTwoPnt(Cut_12_vertex_16, Extrusion_4_vertex_17)
Extrusion_4_vertex_7 = geompy.GetSubShape(Extrusion_4, [7])
Cut_12_vertex_7 = geompy.GetSubShape(Cut_12, [7])
Line_11 = geompy.MakeLineTwoPnt(Extrusion_4_vertex_7, Cut_12_vertex_7)
Cut_12_vertex_18 = geompy.GetSubShape(Cut_12, [18])
Extrusion_4_vertex_24 = geompy.GetSubShape(Extrusion_4, [24])
Line_12 = geompy.MakeLineTwoPnt(Cut_12_vertex_18, Extrusion_4_vertex_24)
Face_26 = geompy.MakeFaceWires([Line_10, Line_12, Edge_20, Edge_48], 1)
Face_27 = geompy.MakeFaceWires([Line_9, Line_10, Edge_15, Edge_46], 1)
Face_28 = geompy.MakeFaceWires([Line_9, Line_11, Edge_18, Edge_47], 1)
Face_29 = geompy.MakeFaceWires([Line_11, Line_12, Edge_23, Edge_49], 1)
Shell_3 = geompy.MakeShell([Face_26, Face_27, Face_28, Face_29, Face_19,
    Face_35])
Solid_19 = geompy.MakeSolid([Shell_3])
Cut_9 = geompy.MakeCutList(Common_2, [Solid_19], True)
Cut_10 = geompy.MakeCutList(Common_3, [Solid_19], True)
Cut_11 = geompy.MakeCutList(Common_4, [Solid_19], True)
```

```
mid = geompy.MakeCompound([Solid_19, Cut_9, Cut_10, Cut_11])
Cut_13 = geompy.MakeCutList(Solid_12, [Common_1], True)
Cut_14 = geompy.MakeCutList(Solid_16, [Common_1], True)
Cut_15 = geompy.MakeCutList(Solid_18, [Common_1], True)
hori = geompy.MakeCompound([Cut_12, Cut_13, Cut_14, Cut_15, Solid_11,
    Solid_13, Solid_14, Solid_17])
Mirror_2_1 = geompy.MakeMirrorByPlane(verti, plan_yz)
Mirror_2_2 = geompy.MakeMirrorByPlane(mid, plan_yz)
Mirror_2_3 = geompy.MakeMirrorByPlane(hori, plan_yz)
Compound_2 = geompy.MakeCompound([verti, mid, hori, Mirror_2_1,
    Mirror_2_2, Mirror_2_3])
Glue_1 = geompy.MakeGlueEdges(Compound_2, 0.0001)
Glue_2 = geompy.MakeGlueFaces(Glue_1, 0.0001)
[Face_1_1,Face_2_1,Face_3_1,Face_4_1,Face_5_1,Face_6_1,Face_7_1,Face_8_1,
    Face_9_1,Face_10_1,Face_11_1,Face_12_1,Face_13_1,Face_14_1,Face_15_1,
    Face_16_1,Face_17_1,Face_18_1,Face_19_1,Face_20_1,Face_21_1,Face_22_1,
    Face_23_1,Face_24_1,Face_25_1,Face_26_1,Face_27_1,Face_28_1,Face_29_1,
    Face_30_1,Face_31_1,Face_32_1,Face_33_1,Face_34_1,Face_35_1,Face_36,
    Face_37,Face_38,Face_39,Face_40,Face_41,Face_42,Face_43,Face_44,
    Face_45,Face_46,Face_47,Face_48,Face_49,Face_50,Face_51,Face_52,
    Face_53,Face_54,Face_55,Face_56,Face_57,Face_58,Face_59,Face_60,
    Face_61,Face_62,Face_63,Face_64,Face_65,Face_66,Face_67,Face_68,
    Face_69,Face_70,Face_71,Face_72,Face_73,Face_74,Face_75,Face_76,
    Face_77,Face_78,Face_79,Face_80,Face_81,Face_82,Face_83,Face_84,
    Face_85,Face_86,Face_87,Face_88,Face_89,Face_90,Face_91,Face_92,
    Face_93,Face_94,Face_95,Face_96,Face_97,Face_98,Face_99,Face_100,
    Face_101,Face_102,Face_103,Face_104,Face_105,Face_106,Face_107,
    Face_108,Face_109,Face_110,Face_111,Face_112,Face_113,Face_114,
    Face_115,Face_116,Face_117,Face_118,Face_119,Face_120] = geompy.
    ExtractShapes(Glue_2, geompy.ShapeType["FACE"], True)
Extrusion_6 = geompy.MakePrismVecH(Face_1_1, OX, -0.28)
Extrusion_7 = geompy.MakePrismVecH(Face_2_1, OX, -0.28)
Extrusion_8 = geompy.MakePrismVecH(Face_3_1, OX, -0.28)
Extrusion_9 = geompy.MakePrismVecH(Face_4_1, OX, -0.28)
Extrusion_10 = geompy.MakePrismVecH(Face_5_1, OX, -0.28)
Extrusion_11 = geompy.MakePrismVecH(Face_6_1, OX, -0.28)
Extrusion_12 = geompy.MakePrismVecH(Face_7_1, OX, -0.28)
Extrusion_13 = geompy.MakePrismVecH(Face_8_1, OX, -0.28)
Extrusion_14 = geompy.MakePrismVecH(Face_113, OX, 2.94)
Extrusion_15 = geompy.MakePrismVecH(Face_114, OX, 2.94)
Extrusion_16 = geompy.MakePrismVecH(Face_115, OX, 2.94)
Extrusion_17 = geompy.MakePrismVecH(Face_116, OX, 2.94)
Extrusion_18 = geompy.MakePrismVecH(Face_117, OX, 2.94)
Extrusion_19 = geompy.MakePrismVecH(Face_118, OX, 2.94)
Extrusion_20 = geompy.MakePrismVecH(Face_119, OX, 2.94)
Extrusion_21 = geompy.MakePrismVecH(Face_120, OX, 2.94)
Extrusion_22 = geompy.MakePrismVecH(Face_33_1, OZ, 0.17)
Extrusion_23 = geompy.MakePrismVecH(Face_42, OZ, 0.17)
Extrusion_24 = geompy.MakePrismVecH(Face_51, OZ, 0.17)
Extrusion_25 = geompy.MakePrismVecH(Face_53, OZ, 0.17)
Extrusion_26 = geompy.MakePrismVecH(Face_69, OZ, 0.17)
Extrusion_27 = geompy.MakePrismVecH(Face_70, OZ, 0.17)
Extrusion_28 = geompy.MakePrismVecH(Face_79, OZ, 0.17)
Extrusion_29 = geompy.MakePrismVecH(Face_89, OZ, 0.17)
```

```
Compound_3 = geompy.MakeCompound([Glue_2, Extrusion_6, Extrusion_7,
    Extrusion_8, Extrusion_9, Extrusion_10, Extrusion_11, Extrusion_12,
    Extrusion_13, Extrusion_14, Extrusion_15, Extrusion_16, Extrusion_17,
    Extrusion_18, Extrusion_19, Extrusion_20, Extrusion_21, Extrusion_22,
    Extrusion_23, Extrusion_24, Extrusion_25, Extrusion_26, Extrusion_27,
    Extrusion_28, Extrusion_29])
Glue_3 = geompy.MakeGlueEdges(Compound_3, 0.0001)
Tjunction = geompy.MakeGlueFaces(Glue_3, 0.0001)
proff_small_int = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(proff_small_int, [578, 602, 590, 539, 544, 561])
proff_small_ext = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(proff_small_ext, [556, 626, 614, 534, 573, 532])
proff_mid_hori_int = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE
    "])
geompy.UnionIDs(proff_mid_hori_int, [239, 423, 506, 436, 426, 489, 184,
    263, 167, 174, 171, 419])
proff_mid_hori_ext = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE
    "])
geompy.UnionIDs(proff_mid_hori_ext, [501, 484, 256, 447, 232, 191, 443,
    195, 208, 472, 220, 460])
proff_long_ext = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(proff_long_ext, [645, 647, 739, 669, 727, 686])
proff_long_int = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(proff_long_int, [715, 674, 652, 703, 657, 691])
proff_mid_verti_int = geompy.CreateGroup(Tjunction, geompy.ShapeType["
    EDGE"])
geompy.UnionIDs(proff_mid_verti_int, [25, 11, 301, 308, 8, 18])
proff_mid_verti_ext = geompy.CreateGroup(Tjunction, geompy.ShapeType["
    EDGE"])
geompy.UnionIDs(proff_mid_verti_ext, [90, 336, 73, 53, 332, 49])
proff_verti_int = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(proff_verti_int, [765, 787, 770, 823, 804, 828])
proff_verti_ext = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(proff_verti_ext, [799, 840, 758, 760, 852, 782])
grand_arc = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(grand_arc, [503, 21, 762, 181, 708, 816, 595, 32, 616,
    433, 717, 864, 260, 772, 583, 176, 830, 51, 462, 311, 271, 511, 604,
    428, 210, 696, 729, 688, 575, 334, 118, 270, 312, 33, 144, 259, 325,
    377, 42, 110, 401, 22])
petit_arc = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(petit_arc, [842, 349, 93, 359, 372, 28, 222, 784, 705,
    806, 105, 246, 801, 83, 66, 649, 566, 741, 290, 76, 315, 316, 494,
    247, 525, 411, 474, 607, 389, 115, 854, 280, 679, 558, 789, 486, 380,
    305, 431, 546, 592, 304, 179, 518, 421, 127, 193, 628, 638, 29, 659,
    169, 156, 281, 671, 291, 14, 352, 235, 751, 445, 536, 720, 15, 362,
    825, 833, 236])
cote = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(cote, [242, 88, 619, 201, 154, 809, 399, 243, 46, 45,
    142, 198, 792, 508, 266, 775, 662, 71, 693, 267, 857, 56, 631, 477,
    125, 491, 580, 387, 767, 85, 213, 549, 129, 563, 225, 68, 732, 329,
    845, 339, 541, 465, 654, 340, 57, 328, 676, 450, 744])
h_mid = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE"])
geompy.UnionIDs(h_mid, [147, 113, 132, 135, 103, 404, 370, 375, 392, 101,
    159, 108])
```

```
geompy.DifferenceIDs(cote, [242, 88, 619, 201, 154, 809, 399, 243, 46,
    45, 142, 198, 792, 508, 266, 775, 662, 71, 693, 267, 857, 56, 631,
    477, 125, 491, 580, 387, 767, 85, 213, 549, 129, 563, 225, 68, 732,
    329, 845, 339, 541, 465, 654, 340, 57, 328, 676, 450, 744])
geompy.UnionIDs(cote, [242, 88, 619, 201, 154, 809, 399, 243, 46, 45,
    142, 198, 792, 508, 266, 775, 662, 71, 693, 267, 857, 56, 631, 477,
    125, 491, 580, 387, 767, 85, 213, 549, 129, 563, 225, 68, 732, 329,
    845, 339, 541, 465, 654, 340, 57, 328, 676, 450, 744, 453])
inlet_1 = geompy.CreateGroup(Tjunction, geompy.ShapeType["FACE"])
geompy.UnionIDs(inlet_1, [620, 550, 567, 639, 584, 596, 632, 608])
inlet_2 = geompy.CreateGroup(Tjunction, geompy.ShapeType["FACE"])
geompy.UnionIDs(inlet_2, [834, 793, 865, 846, 776, 817, 858, 810])
wall = geompy.CreateGroup(Tjunction, geompy.ShapeType["FACE"])
geompy.UnionIDs(wall, [458, 218, 530, 523, 393, 554, 482, 624, 133, 148,
    230, 74, 470, 571, 612, 441, 189, 636, 254, 412, 499, 206, 160, 288,
    405, 756, 350, 47, 330, 838, 360, 780, 850, 91, 797, 862, 643, 737,
    725, 667, 684, 749])
outlet = geompy.CreateGroup(Tjunction, geompy.ShapeType["FACE"])
geompy.UnionIDs(outlet, [697, 721, 680, 733, 709, 663, 752, 745])
geompy.DifferenceIDs(proff_long_ext, [645, 647, 739, 669, 727, 686])
geompy.UnionIDs(proff_long_ext, [645, 647, 739, 686])
proff_long_ext_top = geompy.CreateGroup(Tjunction, geompy.ShapeType["EDGE
    "])
geompy.UnionIDs(proff_long_ext_top, [727, 669])
proff_small_ext_top = geompy.CreateGroup(Tjunction, geompy.ShapeType["
    EDGE"])
geompy.UnionIDs(proff_small_ext_top, [556, 614])
geompy.DifferenceIDs(proff_small_ext, [556, 626, 614, 534, 573, 532])
geompy.UnionIDs(proff_small_ext, [626, 534, 573, 532])
geomObj_3 = geompy.MakeVertex(0, 0, 0)
geomObj_4 = geompy.MakeVectorDXDYDZ(1, 0, 0)
geomObj_5 = geompy.MakeVectorDXDYDZ(0, 1, 0)
geomObj_6 = geompy.MakeVectorDXDYDZ(0, 0, 1)
geompy.addToStudy( O, 'O' )
geompy.addToStudy( OX, 'OX' )
geompy.addToStudy( OY, 'OY' )
geompy.addToStudy( OZ, 'OZ' )
geompy.addToStudy( Tjunction, 'Tjunction' )
geompy.addToStudyInFather( Tjunction, proff_small_int, 'proff_small_int'
    )
geompy.addToStudyInFather( Tjunction, proff_small_ext, 'proff_small_ext'
    )
geompy.addToStudyInFather( Tjunction, proff_mid_hori_int, '
    proff_mid_hori_int' )
geompy.addToStudyInFather( Tjunction, proff_mid_hori_ext, '
    proff_mid_hori_ext' )
geompy.addToStudyInFather( Tjunction, proff_long_ext, 'proff_long_ext' )
geompy.addToStudyInFather( Tjunction, proff_long_int, 'proff_long_int' )
geompy.addToStudyInFather( Tjunction, proff_mid_verti_int, '
    proff_mid_verti_int' )
geompy.addToStudyInFather( Tjunction, proff_mid_verti_ext, '
    proff_mid_verti_ext' )
geompy.addToStudyInFather( Tjunction, proff_verti_int, 'proff_verti_int'
    )
```

```
geompy.addToStudyInFather( Tjunction, proff_verti_ext, 'proff_verti_ext'
    )
geompy.addToStudyInFather( Tjunction, grand_arc, 'grand_arc' )
geompy.addToStudyInFather( Tjunction, petit_arc, 'petit_arc' )
geompy.addToStudyInFather( Tjunction, cote, 'cote' )
geompy.addToStudyInFather( Tjunction, h_mid, 'h_mid' )
geompy.addToStudyInFather( Tjunction, inlet_1, 'inlet_1' )
geompy.addToStudyInFather( Tjunction, inlet_2, 'inlet_2' )
geompy.addToStudyInFather( Tjunction, wall, 'wall' )
geompy.addToStudyInFather( Tjunction, outlet, 'outlet' )
geompy.addToStudyInFather( Tjunction, proff_long_ext_top, '
    proff_long_ext_top' )
geompy.addToStudyInFather( Tjunction, proff_small_ext_top, '
    proff_small_ext_top' )

###
### SMESH component
###

import  SMESH, SALOMEDS
from salome.smesh import smeshBuilder

smesh = smeshBuilder.New(theStudy)
Mesh_1 = smesh.Mesh(Tjunction)
Regular_1D = Mesh_1.Segment()
Nb_Segments_1 = Regular_1D.NumberOfSegments(15)
Nb_Segments_1.SetDistrType( 0 )
Quadrangle_2D = Mesh_1.Quadrangle(algo=smeshBuilder.QUADRANGLE)
Hexa_3D = Mesh_1.Hexahedron(algo=smeshBuilder.Hexa)
proff_small_int_1 = smesh.CreateHypothesis('NumberOfSegments')
proff_small_int_1.SetNumberOfSegments( 56 )
proff_small_int_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,proff_small_int)
status = Mesh_1.AddHypothesis(proff_small_int_1,proff_small_int)
status = Mesh_1.AddHypothesis(Regular_1D,proff_small_ext)
status = Mesh_1.AddHypothesis(proff_small_int_1,proff_small_ext)
proff_mid_hori_int_1 = smesh.CreateHypothesis('NumberOfSegments')
proff_mid_hori_int_1.SetNumberOfSegments( 28 )
proff_mid_hori_int_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,proff_mid_hori_int)
status = Mesh_1.AddHypothesis(proff_mid_hori_int_1,proff_mid_hori_int)
status = Mesh_1.AddHypothesis(Regular_1D,proff_mid_hori_ext)
status = Mesh_1.AddHypothesis(proff_mid_hori_int_1,proff_mid_hori_ext)
proff_long_ext_1 = smesh.CreateHypothesis('NumberOfSegments')
proff_long_ext_1.SetNumberOfSegments( 588 )
proff_long_ext_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,proff_long_ext)
status = Mesh_1.AddHypothesis(proff_long_ext_1,proff_long_ext)
status = Mesh_1.AddHypothesis(Regular_1D,proff_long_int)
status = Mesh_1.AddHypothesis(proff_long_ext_1,proff_long_int)
proff_mid_verti_int_1 = smesh.CreateHypothesis('NumberOfSegments')
proff_mid_verti_int_1.SetNumberOfSegments( 20 )
proff_mid_verti_int_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,proff_mid_verti_int)
status = Mesh_1.AddHypothesis(proff_mid_verti_int_1,proff_mid_verti_int)
```

```
status = Mesh_1.AddHypothesis(Regular_1D,proff_mid_verti_ext)
status = Mesh_1.AddHypothesis(proff_mid_verti_int_1,proff_mid_verti_ext)
proff_verti_int_1 = smesh.CreateHypothesis('NumberOfSegments')
proff_verti_int_1.SetNumberOfSegments( 34 )
proff_verti_int_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,proff_verti_int)
status = Mesh_1.AddHypothesis(proff_verti_int_1,proff_verti_int)
status = Mesh_1.AddHypothesis(Regular_1D,proff_verti_ext)
status = Mesh_1.AddHypothesis(proff_verti_int_1,proff_verti_ext)
grand_arc_1 = smesh.CreateHypothesis('NumberOfSegments')
grand_arc_1.SetNumberOfSegments( 14 )
grand_arc_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,grand_arc)
status = Mesh_1.AddHypothesis(grand_arc_1,grand_arc)
petit_arc_1 = smesh.CreateHypothesis('NumberOfSegments')
petit_arc_1.SetNumberOfSegments( 6 )
petit_arc_1.SetDistrType( 0 )
status = Mesh_1.AddHypothesis(Regular_1D,petit_arc)
status = Mesh_1.AddHypothesis(petit_arc_1,petit_arc)
h_mid_1 = smesh.CreateHypothesis('NumberOfSegments')
status = Mesh_1.AddHypothesis(Regular_1D,h_mid)
status = Mesh_1.AddHypothesis(h_mid_1,h_mid)
cote_1 = smesh.CreateHypothesis('NumberOfSegments')
h_mid_1.SetNumberOfSegments( 7 )
h_mid_1.SetDistrType( 0 )
cote_1.SetNumberOfSegments( 12 )
cote_1.SetScaleFactor( 1.4 )
cote_1.SetReversedEdges( [ 154, 399, 142, 125, 329, 129, 387, 46, 71, 88
    ] )
cote_1.SetObjectEntry( "0:1:1:132" )
status = Mesh_1.AddHypothesis(Regular_1D,cote)
status = Mesh_1.AddHypothesis(cote_1,cote)
Fixed_Points_1D_1 = smesh.CreateHypothesis('FixedPoints1D')
Fixed_Points_1D_1.SetPoints( [ 0.001204, 0.0023809 ] )
Fixed_Points_1D_1.SetNbSegments( [ 1, 1, 586 ] )
Fixed_Points_1D_1.SetReversedEdges( [] )
Fixed_Points_1D_1.SetObjectEntry( "0:1:1:132" )
status = Mesh_1.AddHypothesis(Regular_1D,proff_long_ext_top)
status = Mesh_1.AddHypothesis(Fixed_Points_1D_1,proff_long_ext_top)
Fixed_Points_1D_2 = smesh.CreateHypothesis('FixedPoints1D')
Fixed_Points_1D_2.SetPoints( [ 0.010714, 0.025 ] )
Fixed_Points_1D_2.SetNbSegments( [ 1, 1, 54 ] )
Fixed_Points_1D_2.SetReversedEdges( [] )
Fixed_Points_1D_2.SetObjectEntry( "0:1:1:132" )
status = Mesh_1.AddHypothesis(Regular_1D,proff_small_ext_top)
status = Mesh_1.AddHypothesis(Fixed_Points_1D_2,proff_small_ext_top)
isDone = Mesh_1.Compute()
[ Sub_mesh_1, Sub_mesh_2, Sub_mesh_3, Sub_mesh_4, Sub_mesh_5, Sub_mesh_6,
     Sub_mesh_7, Sub_mesh_8, Sub_mesh_9, Sub_mesh_10, Sub_mesh_11,
    Sub_mesh_12, Sub_mesh_13, Sub_mesh_14, Sub_mesh_15, Sub_mesh_16 ] =
    Mesh_1.GetMesh().GetSubMeshes()
Sub_mesh_1 = Mesh_1.GetSubMesh( proff_small_int, 'Sub-mesh_1' )
Sub_mesh_2 = Mesh_1.GetSubMesh( proff_small_ext, 'Sub-mesh_2' )
Sub_mesh_3 = Mesh_1.GetSubMesh( proff_mid_hori_int, 'Sub-mesh_3' )
Sub_mesh_4 = Mesh_1.GetSubMesh( proff_mid_hori_ext, 'Sub-mesh_4' )
```

```
Sub_mesh_5 = Mesh_1.GetSubMesh( proff_long_ext, 'Sub-mesh_5' )
Sub_mesh_6 = Mesh_1.GetSubMesh( proff_long_int, 'Sub-mesh_6' )
Sub_mesh_7 = Mesh_1.GetSubMesh( proff_mid_verti_int, 'Sub-mesh_7' )
Sub_mesh_8 = Mesh_1.GetSubMesh( proff_mid_verti_ext, 'Sub-mesh_8' )
Sub_mesh_9 = Mesh_1.GetSubMesh( proff_verti_int, 'Sub-mesh_9' )
Sub_mesh_10 = Mesh_1.GetSubMesh( proff_verti_ext, 'Sub-mesh_10' )
Sub_mesh_11 = Mesh_1.GetSubMesh( grand_arc, 'Sub-mesh_11' )
Sub_mesh_12 = Mesh_1.GetSubMesh( petit_arc, 'Sub-mesh_12' )
Sub_mesh_13 = Mesh_1.GetSubMesh( h_mid, 'Sub-mesh_13' )
Sub_mesh_14 = Mesh_1.GetSubMesh( cote, 'Sub-mesh_14' )
Sub_mesh_15 = Mesh_1.GetSubMesh( proff_long_ext_top, 'Sub-mesh_15' )
Sub_mesh_16 = Mesh_1.GetSubMesh( proff_small_ext_top, 'Sub-mesh_16' )


## Set names of Mesh objects
smesh.SetName(Regular_1D.GetAlgorithm(), 'Regular_1D')
smesh.SetName(Hexa_3D.GetAlgorithm(), 'Hexa_3D')
smesh.SetName(Quadrangle_2D.GetAlgorithm(), 'Quadrangle_2D')
smesh.SetName(proff_small_int_1, 'proff_small_int')
smesh.SetName(proff_mid_hori_int_1, 'proff_mid_hori_int')
smesh.SetName(Nb_Segments_1, 'Nb. Segments_1')
smesh.SetName(proff_verti_int_1, 'proff_verti_int')
smesh.SetName(grand_arc_1, 'grand_arc')
smesh.SetName(proff_long_ext_1, 'proff_long_ext')
smesh.SetName(proff_mid_verti_int_1, 'proff_mid_verti_int')
smesh.SetName(petit_arc_1, 'petit_arc')
smesh.SetName(h_mid_1, 'h_mid')
smesh.SetName(Mesh_1.GetMesh(), 'Mesh_1')
smesh.SetName(Sub_mesh_9, 'Sub-mesh_9')
smesh.SetName(Sub_mesh_8, 'Sub-mesh_8')
smesh.SetName(Sub_mesh_3, 'Sub-mesh_3')
smesh.SetName(Sub_mesh_2, 'Sub-mesh_2')
smesh.SetName(Sub_mesh_1, 'Sub-mesh_1')
smesh.SetName(Sub_mesh_7, 'Sub-mesh_7')
smesh.SetName(Sub_mesh_6, 'Sub-mesh_6')
smesh.SetName(Sub_mesh_5, 'Sub-mesh_5')
smesh.SetName(Sub_mesh_4, 'Sub-mesh_4')
smesh.SetName(Fixed_Points_1D_2, 'Fixed Points 1D_2')
smesh.SetName(Fixed_Points_1D_1, 'Fixed Points 1D_1')
smesh.SetName(cote_1, 'cote')
smesh.SetName(Sub_mesh_12, 'Sub-mesh_12')
smesh.SetName(Sub_mesh_13, 'Sub-mesh_13')
smesh.SetName(Sub_mesh_10, 'Sub-mesh_10')
smesh.SetName(Sub_mesh_11, 'Sub-mesh_11')
smesh.SetName(Sub_mesh_16, 'Sub-mesh_16')
smesh.SetName(Sub_mesh_14, 'Sub-mesh_14')
smesh.SetName(Sub_mesh_15, 'Sub-mesh_15')


if salome.sg.hasDesktop():
  salome.sg.updateObjBrowser(1)
```