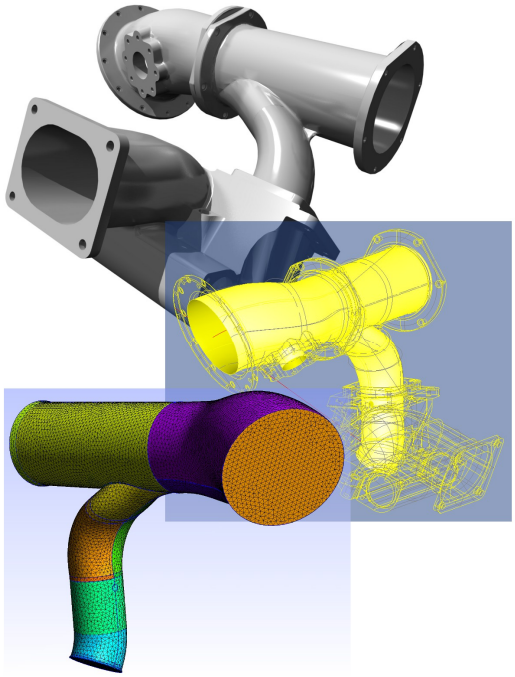
RESEARCH REPORT

VTT-R-02440-09



Meshing Tools for Open Source CFD – A Practical Point of View

Authors: Juha Kortelainen

Confidentiality: Public

Report's title	
Meshing Tools for Open Source CFD – A Practical Point of View	
Customer, contact person, address CSC – IT Center for Science Ltd. P.O. Box 405 FI-02101 Espoo, Finland	Order reference VTT-V-27856-08
Project name Avoimen lähdekoodin CFD-esikäsittelytyökalut teollisessa käytössä	Project number/Short name 27856/LSCFD
Author(s) Juha Kortelainen	Pages 24/0
Keywords CFD, geometry, gmsh, grid, OpenFOAM, pre-processing, mesh, meshing, SALOME	Report identification code VTT-R-02440-09

Summary

This document is part of the project *Tools for Large Scale Computational Fluid Dynamics, LSCFD*. The project is managed by CSC – IT Center for Science Ltd. (CSC) and partially funded by Tekes (Finnish Funding Agency for Technology and Innovation). The main objective of the project is to promote the use of large scale CFD in Finland. The project will support, enhance and document the use of selected open source CFD solvers. Mesh generation is an essential part of the solution procedure which often consumes the most of the human resources.

In this report three different tools or utilities for creating a computational mesh for OpenFOAM CFD code are presented: SALOME, gmsh and OpenFOAM snappyHexMesh. All these software are open source. The point of view of this study is on practical side, i.e. usability, software ergonomics and ability to produce high quality mesh are emphasised instead of meshing algorithms. The same study and test procedure has been applied for all three tools. Software functionality is studied based on soft-ware manuals and by using the software. One common, industrial meshing test case was selected. With this test case the usability of these tools could be tested into some extend and compared. In the last section there is presented some comments about these tools and in general about the development of open source mesh generation tools.

A typical pre-processing (mesh generation) process for CFD or FEM computation is described in general. In this process description the domain geometry is taken as is from the design system (typically CAD system) and all the modifications are done during the pre-processing process.

The study indicates that there are good open source software components already available for CFD pre-processing, but at least the tested implementations haven't been completely successful in using them and implementing a really user friendly, intuitive and industrial strength pre-processing tool. None of the tested tools or methods can be recommended for general industrial use as such. More development needs to be done for these meshing tools to reach a level where the whole process is efficient and produces high quality results. But the development seems to be on right path and all the elements for good progress are already available.

Confidentiality	Public

Espoo 15.4.2009

Written by	Reviewed by	Accepted by
Juha Kortelainen, Senior Research Scientist	Tero Kiviniemi, Sales Manager	Pekka Koskinen, Research Manager

VTT's contact address
VTT Technical Research Centre of Finland, P.O. Box 1000, FI-02044 VTT, Finland

Distribution (customer and VTT)
CSC: 2 pieces
VTT: 1 piece

Preface

Computational methods are becoming ever more important methods and tools for both industrial product development and scientific research. These methods enable fast and cost effective studies of systems and phenomena that often are very difficult and expensive or dangerous or even impossible to execute. These methods can provide designers and researchers information that helps to understand complex phenomena and thus further develop new and better products.

In this report one narrow area of applying computation methods has been shortly studied. Spatially discretised methods like finite element method or control volume method are widely used both in research and product development. To apply these methods to some specific problems require that the pre-processing tools are available and have the functionality needed for the specific purpose. This pre-phase of the problem solving is a kind of a mandatory pain in the process but still it has remarkable influence on the reliability of the results as well as to fluency of the whole process.

I want to thank CSC – IT Center for Science Ltd. (CSC) and LSCFD project for the opportunity to do this study and test all these software. I find these tools both very interesting in the area of computational methods and very important in the whole chain of work phases in computation. I hope this work will turn the focus of interest of open source tool developers to the whole pre-processing chain so that all those great open source computational solvers and other tools would reach the same level in easy-of-use than commercial tools already have.

Espoo 15.4.2009

Juha Kortelainen

Contents

Preface 2

Abbreviations and acronyms 4

1 Background 5

2 Overview of mesh generators 5

2.1 Mesh generation process 6
2.2 Test case 7

3 SALOME 8

3.1 Introduction 8
3.2 Import/Export capabilities 9
3.3 Geometry manipulation capabilities 9
3.4 Meshing capabilities 9
3.5 Meshing efficiency 10
3.6 Mesh generator in action 10
3.6.1 Creating the flow channel geometry 10
3.6.2 Meshing 13
3.7 Short introduction SALOME version 4.1.4 17
3.8 Using SALOME with OpenFOAM 17

4 Gmsh 18

4.1 Introduction 18
4.2 Geometry capabilities 18
4.3 Meshing capabilities 18
4.4 Import/Export capabilities 19
4.5 Meshing efficiency 19
4.6 Mesh generator in action 19
4.6.1 Creating the flow channel geometry 19
4.6.2 Meshing 20
4.6.3 Using gmsh with OpenFOAM 20

5 Using OpenFOAM with external meshing tools 20

6 OpenFOAM snappyHexMesh 21

6.1 Introduction 21
6.2 Meshing capabilities 21
6.3 Import/Export capabilities 21
6.4 Meshing efficiency 22
6.5 Mesh generator in action 22

7 Discussion and Conclusions 22

References 24

Abbreviations and acronyms

1/2/3D	One/two/three dimensional.
ACIS	3D solid geometry file format developed by Spatial Corporation.
AMR	Adaptive mesh refinement; a method to refine mesh based of local solution, e.g. flow velocity gradient.
ASCII	American standard code for information interchange.
BREP	Boundary representation; often used method for solid geometry modelling.
CAD/CAE	Computation aided design/engineering.
CFD	Computational fluid dynamics.
CSG	Constructive solid geometry, a method to create complex solid geometries from primitive geometries using geometric Boolean operations.
CVM	Control volume method.
DAT	Abbreviation to data, often used in file name extensions.
DXF	Drawing Exchange Format, a 3D geometry file format from Autodesk Inc.
FEM	Finite element method.
FOAM	Open field operation and manipulation, software library for control volume based computation; OpenFOAM is an open source version of the software.
GHS3D	A meshing algorithm developed by team Gamma in INRIA, France.
GNU	Gnu's not Unix, GNU open source project's symbol.
GPL	GNU general public license.
HDF5	Hierarchical data format, version 5; a file format, application programming interface and routine library for especially scientific data storage.
HTML	Hyper text markup language, HTML is an SGML application conforming to ISO 8879 standard of Standard Generalized Markup Language (SGML).
IGES	Initial graphics exchange specification, 3D geometry exchange format and specification.
LGPL	GNU lesser/library general public license, open source license.
LSCFD	Large scale computational fluid dynamics, research project's short name.
NASA	National aeronautics and space administration.
PC	Personal computer.
Plot3D	File format for writing CFD structured grids and solutions; format is developed in NASA.
STEP	Standard for the Exchange of Product model data, ISO 10303 standard for data exchange.
STL	Stereolitography, a simple file format to store triangulated surface geometry data.
VRML	Virtual reality modelling language.
VTK	Visualization toolkit, open source scientific visualisation library developed by Kitware Inc.
UNV	Universal file format, a file format originally developed by Structural Dynamics Research Corporation (SDRC) for file transfer between CAD and CAE systems.

1 Background

This document is part of the project *Tools for Large Scale Computational Fluid Dynamics, LSCFD*. The project is managed by CSC – IT Center for Science Ltd. (CSC) and partially funded by Tekes (Finnish Funding Agency for Technology and Innovation). The main objective of the project is to promote the use of large scale CFD in Finland. The project will support, enhance and document the use of selected open source CFD solvers. Mesh generation is an essential part of the solution procedure which often consumes the most of the human resources.

This survey is not comprehensible as it will study only a few alternatives in more detail. The point of view of this study is on practical side, i.e. usability, software ergonomics and ability to produce high quality mesh are emphasised instead of meshing algorithms. In selecting the software open source has not been a necessary condition since also some proprietary mesh generators are often used with open source solvers. However, a rather inexpensive license pricing is desirable since typically the users of open source software have limited monetary resources.

For large scale simulations the mesh generation tools must be efficient enough to be able to handle large meshes interactively on the workstation. For complex geometries the a posteriori techniques of mesh refinement are not satisfactory as the geometry description is different on the finest level. Therefore it is imperative that large meshes can be created with the chosen tools.

2 Overview of mesh generators

Mesh generation is a mandatory process phase in typical CFD, structural and acoustic simulations and analyses. Although being just requirement for performing calculation it has an important impact on efficiency and accuracy of computation; element or cell shape and size does matter on both computation speed and numerical accuracy. For CVM and in most cases FEM, hexahedron elements are more favourable to numerical efficiency, but there isn't any general automatic and robust mesh generation algorithm available for such a mesh type. Due to increase in computational power, the number of elements doesn't have that much significance any more. This has lead to use of automated tetrahedral mesh generators both in CVM and FEM. These meshing tools can produce relatively high quality mesh in just tens of seconds or couple of minutes. The mesh quality is often good enough for structural analysis but in CFD results are more sensitive on mesh type and quality and the use of tetrahedral mesh can lead to a high number of cells to achieve the same computational accuracy than when using hexahedral cells. One solution for the element or cell type issue is the use of a hybrid mesh, i.e. a mesh containing both hexahedral and tetrahedral elements or cells. This allows e.g. creation of hexahedral mesh is critical areas manually and mesh the rest of the volume using automatic mesh generation and tetrahedral elements or cells. There are also methods to automatically create hybrid mesh in arbitrary geometry. This area is not covered in this document.

In this report three different tools or utilities for creating a computational mesh for OpenFOAM CFD code are presented: SALOME, gmsh and OpenFOAM snappyHexMesh. All these software are open source, which mean that the source code of these software is freely available. The same study and test procedure has been applied for all three tools. Software functionality is studied based on software manuals and by using the software. One common, industrial meshing test case was selected. With this test case the usability of these software could be tested into some extend and compared. In the last section there is presented some comments about these tools and in general about the development of open source mesh generation tools.

2.1 Mesh generation process

Computational meshes are needed in spatially discretised methods like finite element method (FEM), and control volume method (CVM), mostly used for computation fluid dynamics (CFD). In these methods the domain of interest, e.g. a structure or a flow domain, is divided into small computational volumes, called elements or cells, which cover the domain so that they fill the domain completely but do not overlap on another. Different numerical methods require some specific features from the mesh, e.g. for some methods the mesh has to be structured, meaning the mesh has fixed topological structure through the whole mesh, or the mesh elements or cells have to have a specific form like hexahedron. All the special limitations combined with the shape of the domain itself can set difficult requirements for the meshing process and methods used to perform it.

In Figure 1 there is illustrated one typical overall meshing process for an industrial application. In this process description the domain geometry is taken as is from the design system (typically CAD system) and all the modifications are done during the pre-processing process. It is also possible, and often more efficient, to modify the geometry already in the design system and start the meshing process using more suitable domain geometry.

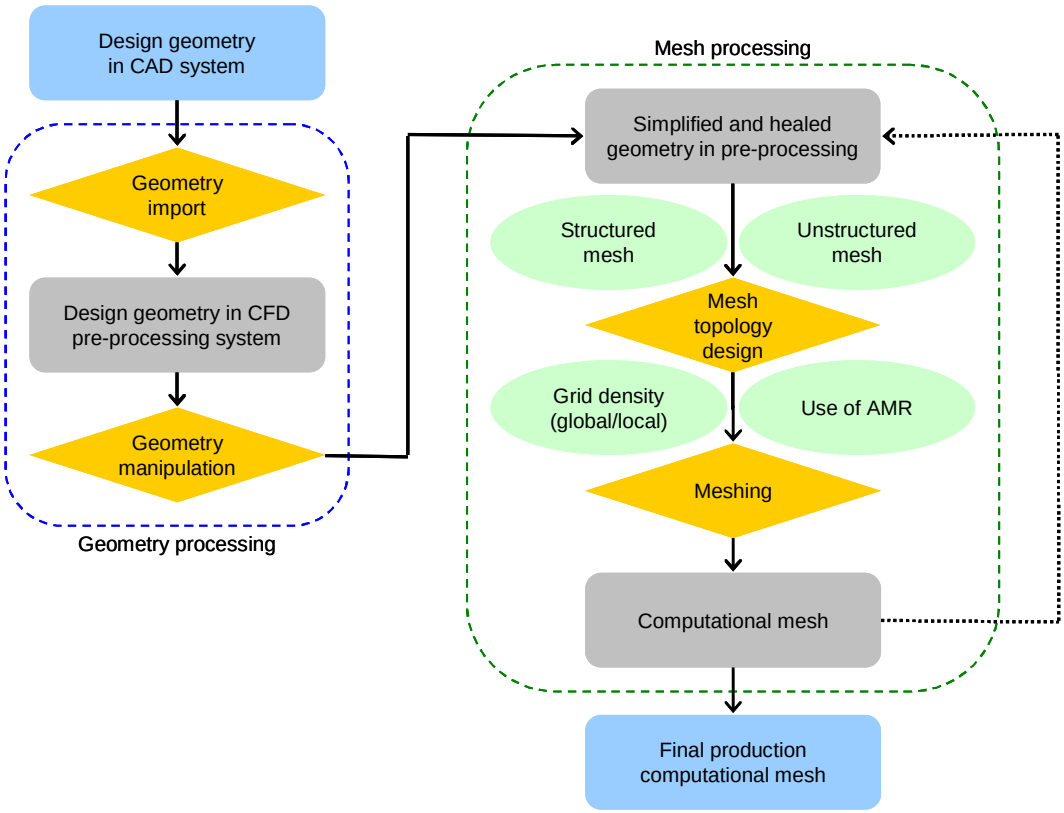


Figure 1: Ideal work flow for meshing starting from CAD geometry.

The process starts from geometry import to the pre-processing tool. In practice this means importing the geometry in a file format that the pre-processing tool supports. There are couple of neutral formats like STEP (ISO standard) and IGES that most of the design environments, CAD systems and pre-processing tools support. Also some proprietary formats have become defacto standards in geometry exchange; an example of these is DXF from Autodesk Inc. The difficulty in geometry import is that there are many file formats available and tools support these formats in differing level. To find the most usable and robust format between two software usually requires some testing. Another difficulty is the geometry itself. The same geometrical shape can be created using different procedures resulting different geometrical de-

tails. Although the shape may be the same, the geometry may contain small edge and surface elements, depending on how the final shape has been processed.

The geometry manipulation phase in the process includes typically removal of small irrelevant details, like chamfers, roundings and small holes. These details may require large amount of computational elements or cells and thus unnecessarily increase the needed computational effort. This phase should also include the checking of geometry quality and fixing or removal of geometry representation anomalies, like divided edges with small edge pieces or open seams in surfaces. Depending on the selected meshing algorithm, these may ruin the meshing process or lead to poor mesh quality.

In the mesh topology design phase the meshing strategy is decided. Depending on the computational task and used software, the mesh type (structure and unstructured), element or cell type (e.g. tetrahedral or hexahedral) and mesh density (global and local) need to be designed. Also the computational domain splitting into several blocks is often required to have better control over the mesh quality and to decrease the memory allocation during individual meshing phase. The meshing process is typically iterative in nature; the mesh quality is seen only after the mesh has been created and if some parameters need to be adjusted, the process starts often from the beginning in the mesh processing phase. In case of using several computational mesh blocks, there may be procedure iterations inside different mesh blocks.

The meshing phase differs substantially depending on the mesh type. Unstructured tetrahedral mesh is usually generated automatically into the given geometry. The user needs to define meshing parameters like desired element size (mesh density) and optional local mesh adjustment parameters and mesh quality optimisation attributes. The actual meshing is done by the routine. For structured mesh there are usually several manual process phases. The user defines the blocks for the mesh; typically there has to be 12 block edges and six faces that topologically form a block shape (see Figure 2). Creating a structured mesh for industrial geometry requires more time and may be a difficult task for complex geometries with many geometrical details. Especially for CFD this is often a decision and compromise between computational efficiency, accuracy and the time required for preprocessing.

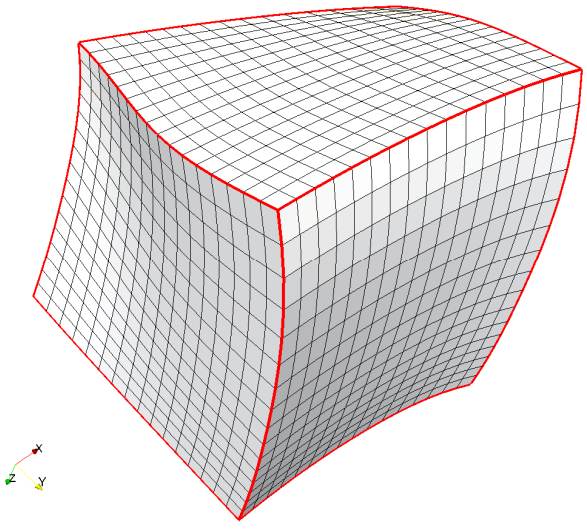


Figure 2: A block topology for structured mesh.

After the mesh has been successfully created the process continues with computation definition, including among others setting boundary conditions and initial conditions for the flow case. This phase is out of the scope of this report.

2.2 Test case

The import functionality from a CAD system was tested with realistic design geometry of a Wärtsilä W20V34SG exhaust channel model. The geometry was exported form a commercial I-deas CAD software in IGES format. The case geometry is a typical CFD case with complex

geometric features and for CFD use unnecessary details. The original test case geometry is shown in Figure 3.

The basic assumption with the test case was that the person responsible of mesh generation gets the geometry from the designer as it is and does to necessary manipulation to the geometry before meshing. In this case the original geometry is relatively complex related to the requirements of the CFD meshing. The case could have been chosen so that the geometry is already manipulated for meshing in the original CAD system (unnecessary features removed and the flow channel modelled as a solid geometry).

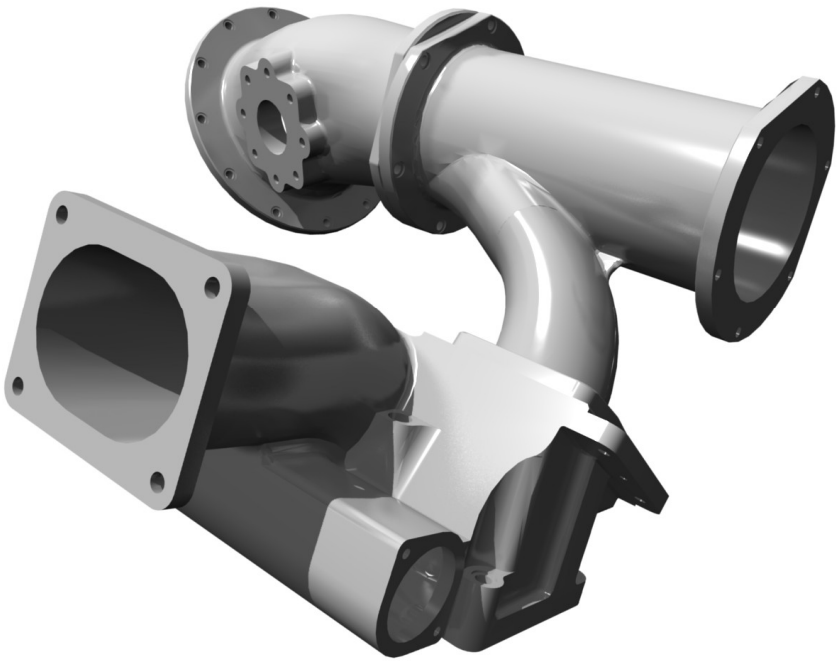


Figure 3: Original CAD model.

All the tests were done in a PC workstation running Fedora Linux version 8 and kernel version 2.6.26.6. The workstation was Dell WS 540 with two Intel Xeon III processors (1.8 GHz) and 1.0 GB of physical memory. The graphics processor was nVidia Quadro4 700 XGL with only 64 MB of physical memory. The used workstation does not satisfy present requirements for a workstation used for CFD or FEM pre-processing in industrial or demanding research work. But it is still usable for comparing the differences of the studied pre-processing tools and running tests with the selected test case.

3 SALOME

3.1 Introduction

SALOME is a general graphical environment for numerical computing using spatially discretised, mesh-based methods like finite element or control volume method. The environment contains separate working modes for geometry creation or manipulation (Geometry), meshing (Mesh), solver management (Supervisor), post-processor (Post-Pro), and communication module (MED). The software is open source software under GNU Lesser General Public License (LGPL). The geometry management in the environment is based on Open Cascade geometry kernel and Visualization Toolkit (VTK) visualisation library. The software is available as binary form for several Linux distributions and in source code. At the time of testing the

software the latest public version was 3.2.6. At the time of writing this report a new version of 4.1.4 was published. This section describes the older version 3.2.6 and it's use for CFD mesh generation. In section *Short introduction SALOME version 4.1.4* the impressions of the newer version of SALOME are presented.

The documentation for end user is somewhat limited and sometimes offers useless information. All the basic functions are shortly described in a HTML-based document system, but extensive examples, detailed parameter information and description of what operations are meant to be used for is usually missing. This is quite typical for open source software as the developers know how the software works and thus don't need the documentation and others are either not capable of producing documentation or are not motivated to do so.

3.2 Import/Export capabilities

SALOME supports via Open Cascade the following solid geometry file formats: ACIS, SALOME native BREP, IGES, and STEP. Solid geometry can be exported in ACIS, BREP, IGES (version 5.1 and 5.3), and STEP formats. Also STL (both ASCII and binary) can be used as the model export format.

Created mesh can be exported from SALOME in DAT, MED, I-deas UNV, and STL format (ASCII or binary). DAT format is general ASCII table which first lists all node point coordinates and then elements based on point order. This format enables writing simple converter utilities so that meshes can be used in different applications. MED format is based HDF5 data format for mesh and field specific data. Using files with this format have to be written and read using specific I/O library routines. I-deas UNV format is an ASCII format that has an open format specification. It is design especially for FEM data and is suitable for all mesh data storage. There is a utility for data conversion from UNV format to OpenFOAM internal mesh representation. For more information see section *Using OpenFOAM with external meshing tools*. The STL format is used for surface mesh export and is not usable for CFD mesh export. At the moment only mesh in UNV format can be used with OpenFOAM.

3.3 Geometry manipulation capabilities

SALOME has quite extensive capabilities in creation and manipulation of solid geometry. New geometries can be created either using CSG functionality or BREP operations. New solids can be constructed from manipulated existing surface components by adding new surface so that a closed volume is produced. SALOME also has good shape healing operations to e.g. find and remove small edge and surface components and to combine surface components.

3.4 Meshing capabilities

SALOME is capable of producing tetrahedral, hexahedral and prism meshes. For tetrahedral mesh generation there are several options that are described in more detail later in this report. Hexahedral mesh generation requires hierarchical modelling from volume edges and volume faces to mesh blocks. This can be done also for CAD geometries but is easier for geometries that are created in SALOME. In this report hexahedral mesh generation is not presented. Prism meshes are generated by extruding an existing mesh surface to a volume. This method can be useful for generating a boundary layer mesh for CFD. Prism mesh generation is also discarded in this report.

In SALOME there are quite good control over the mesh topology in general. The mesh can be composed from sub-meshes which may simplify the meshing and allows larger number of cells to be produced; this is because the meshing and optimisation algorithms don't have to

manage the whole set of cells at once. If the generated mesh contains some local anomalies, the user can manually fix this even in cell vertex level. This means the user can define individual cells by defining corner points (vertices), edges (lines connecting vertices) and finally cells.

The generated mesh can be checked for different measures with the *controls* tools:

- Free borders, for edges geometry that belong to one geometry face only
- Length, for geometry edge length checking
- Borders at multi-connection, showing how many geometry faces share the same geometry edge
- Free edges, for cell edges that belong to one cell face only
- Length 2D, for cell edge length checking
- Borders at multi-connection 2D, showing how many cell faces share the same cell edge
- Area, for checking cell face areas
- Taper, for checking the squarness of a face with four corners
- Aspect ratio, for checking how close cell faces are to the optimal shape of their type (how equal are the lengths of the cell edges)
- Minimum angle, for checking the minimum angle between to adjacent cell faces
- Warping angle, for checking the straightness of a quad face (actually constructed of two triangles)
- Skew, for checking the shape of cell faces
- Aspect ratio 3D, same as Aspect ratio above but for cell volume
- Volume, for checking cell volumes in the mesh

These measures can be visualised for quickly checking the different quality measures.

3.5 Meshing efficiency

Automatic mesh generation using tetrahedron elements/cells is relatively fast with both basic methods (2D surface meshing with either Netgen or Mefisto); the 3D meshing is done using Netgen meshing routine. Depending on method selection, the meshing operation may require quite a lot of computer memory. In case the physical memory limit is exceeded in the workstation, the operation becomes extremely slow. To avoid this, the meshing should be started from very coarse mesh and the mesh density should be increased gradually to reach the desired mesh density.

3.6 Mesh generator in action

3.6.1 Creating the flow channel geometry

In the SALOME database the solid geometry was represented as one part (see Figure 4). To be able to simplify the geometry and to create a negative model (the volume of the flow channel) the manifold inner surface was separated from the original model. To do this, first the solid geometry was exploded into a solid component and further to geometric faces. Open Cascade routines, used in SALOME, use boundary representation (BREP) method for solid geometry, thus the solid can be exploded into primitive face components and new or modified solid geometries can be built from these. The operation produced 1049 separate face components for exhaust manifold.

In the test computer the exploded model was very difficult to manage. Even rotating and panning the geometry was difficult and after couple of view operations both zooming and pan-

ning hanged. To simplify the manipulation of the geometry, all unnecessary faces were removed so that only 33 inner surface components of the original flow channel were left (Figure 5). At this phase the geometry was not manifold, but all the channel ends were still open. The model contained also small surface components which make it difficult to mesh the final solid volume. To force SALOME not to use the small faces, they were left out from the next phase, which was building a shell of the face components. The problematic face components are highlighted in Figure 6.

Building a solid continued first by sewing the gaps in the shell using the *sewing* function (Figure 7) and then automatically closing the shell using the *suppress holes* function (Figure 8) to produce a volume for solid. The tolerance parameter for sewing had to be figured out by try and error to close all the gaps. The suppress holes function produced flat surfaces if possible. To ensure that the volume was closed and all the primitive geometric components were valid a *shape processing* function was run. This included additional removal of small edge and face components (Figure 9).

The geometry manipulation for CFD meshing in SALOME seems quite straight forward, but the lack of detailed documentation and difficulties to notice and find small edge and surface components made the process very time consuming. After learning both the software and the geometry of the test case the whole process from the original CAD geometry to computational tetrahedral mesh could be done in just couple of hours. The difficulties in graphics processing and representing made it sometimes frustrating to work with SALOME. But after the learning period this tool was found to be very useful, at least for research use. The final flow domain solid and the original CAD geometry are shown in Figure 10.

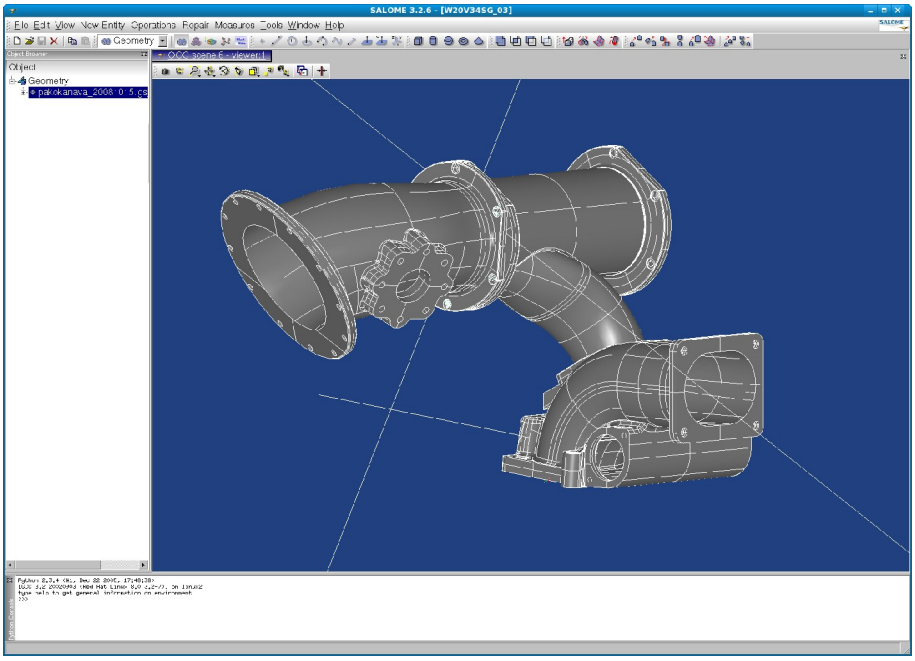


Figure 4: The original CAD geometry imported into SALOME in IGES format.

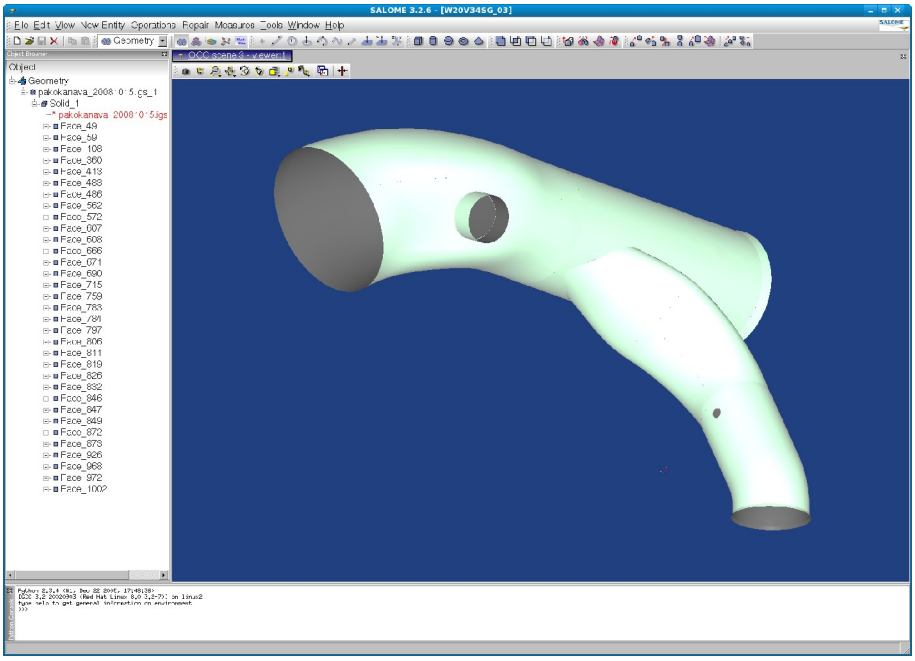


Figure 5: Original geometry exploded into a solid and further to faces. Unnecessary face components removed so that only flow channel inner face components are left.

In GEOM module the geometry modify functionality seems to be missing. New model geometry is constructed in hierarchical manner starting from points (3D co-ordinates) which define edges (e.g. lines, arcs, and splines) and further volumes. There are some primitive volume components, like block and sphere, available. If lower preference element, like point, is modified the dependent geometry does not follow. That means the geometry has to be constructed right from the very beginning. This means, manual healing and modification of surfaces and further volumes can't be done by just modifying the existing primitive components but these components have to be replaced with new ones that are defined from points, edges and then faces.

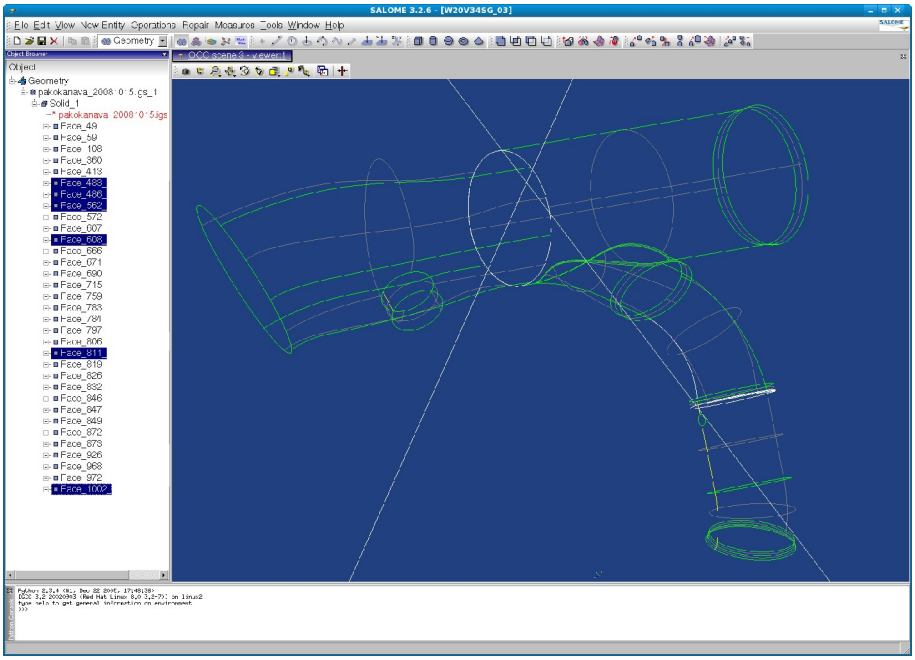


Figure 6: Problematic small face components highlighted. These face components were left out of the shell model.

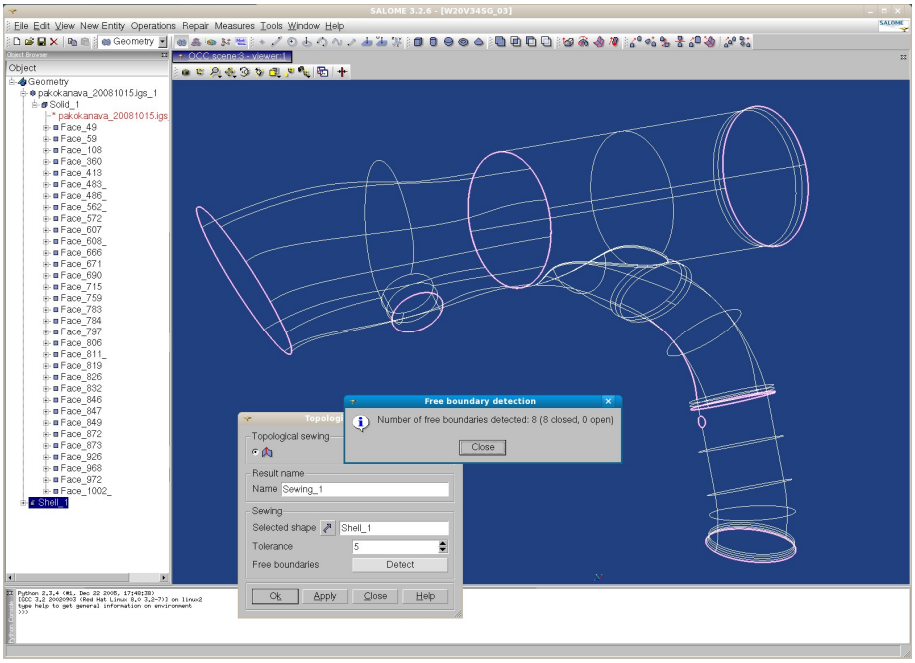


Figure 7: Sewing tool to automatically close small gaps that were introduced when small surface components were removed manually.

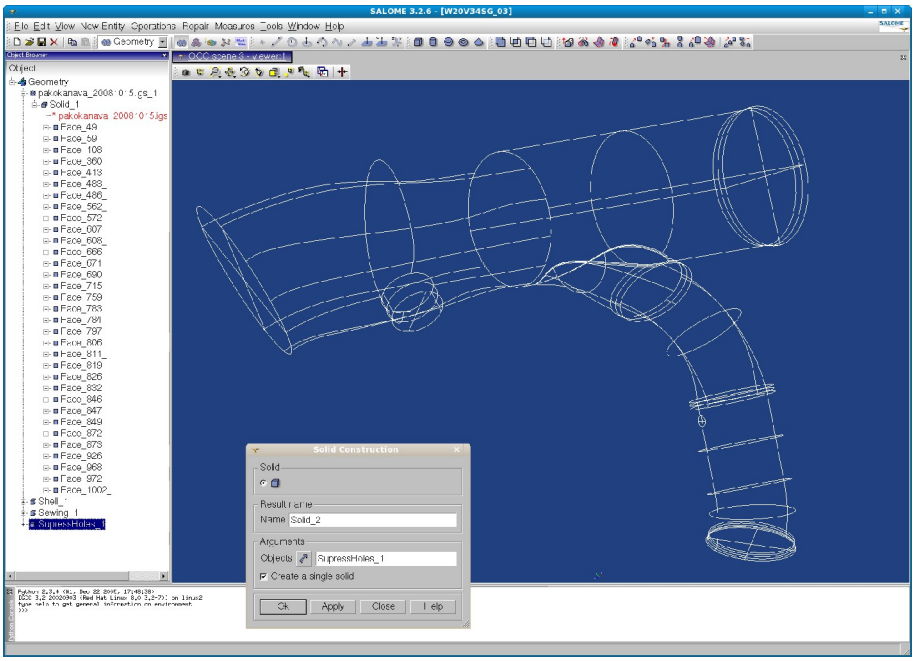


Figure 8: The suppress holes function was used to automatically close the computational volume for solid creation.

3.6.2 Meshing

Meshing in SALOME is done in MESH module (see Figure 11). In principle the meshing procedure is quite straight forward. First, the solid geometry for meshing is selected from the tree view and then *create mesh* tool from the tool palette or menu is selected. The create mesh panel contains four tabs: 3D, 2D, 1D and 0D. In each of these tabs first an *algorithm* for meshing is selected and then based on this a *hypothesis* for the algorithm. For general geometry there is an option of *assign a set of hypotheses* available which automatically set the hypotheses for e.g. tetrahedron meshing.

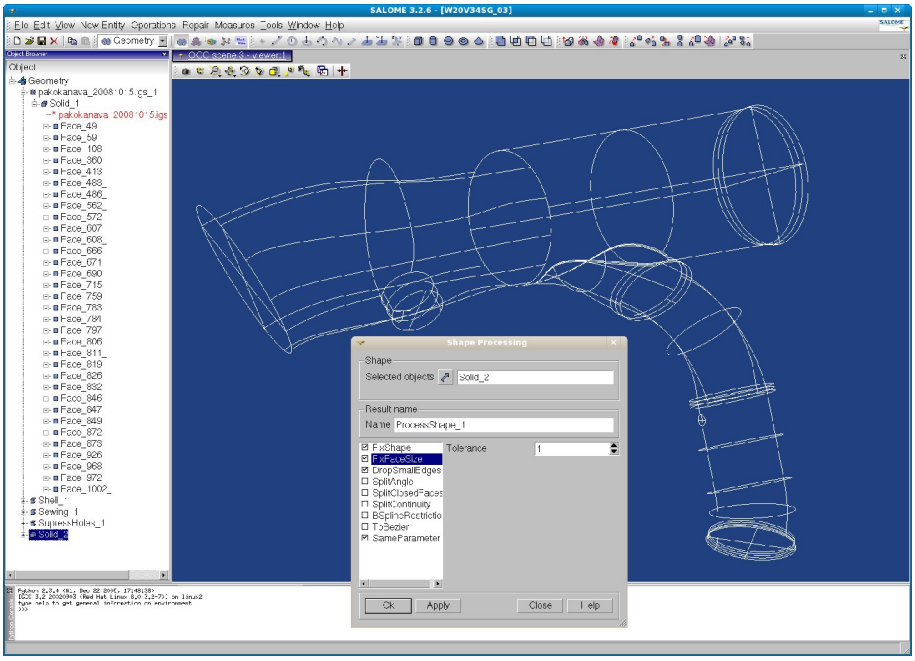


Figure 9: The final geometry check and healing using shape processing function.

For unstructured, practically tetrahedral, mesh generation SALOME offers two routines: GHS3D and Netgen. The GHS3D is a commercial meshing routine and thus is not considered here. It is also possible to extrude 2D surface mesh as prisms to create e.g. boundary layer mesh for CFD. For structured mesh the meshing procedure begins from geometry primitives like edges, faces and finally volumes. This procedure is difficult to apply for existing CAD geometries.

There are two procedure paths for meshing with Netgen routine: *Netgen 1D-2D-3D* which uses Netgen for all meshing phases, and *Tetrahedron (Netgen)* which allows using either Netgen for 1D and 2D meshing or e.g. Mefisto routine for 2D meshing and primitive algorithms for 1D meshing.

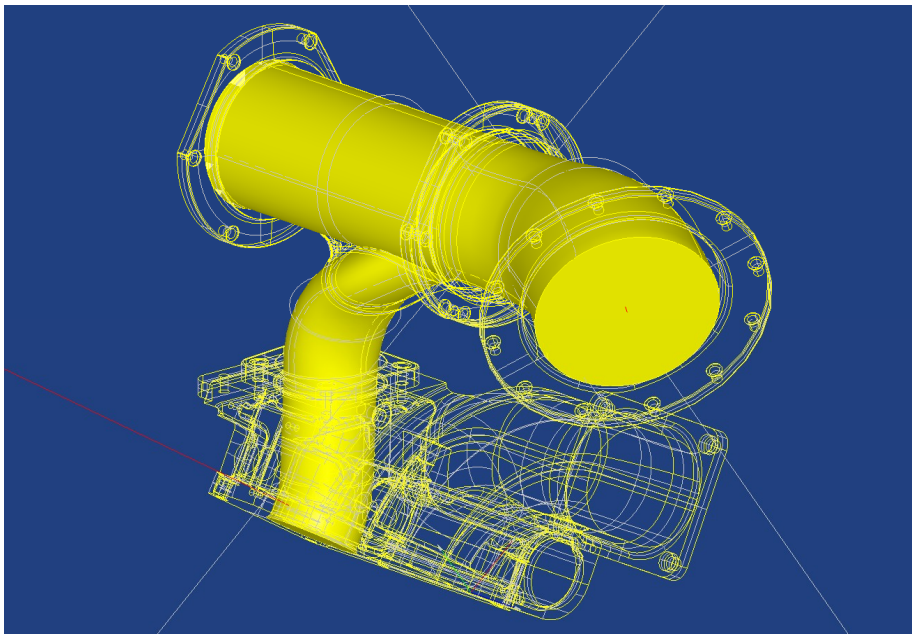


Figure 10: The used test geometry for one exhaust channel of Wärtsilä W20V34SG diesel engine. The original geometry is shown as wireframe graphics and the created flow domain in solid.

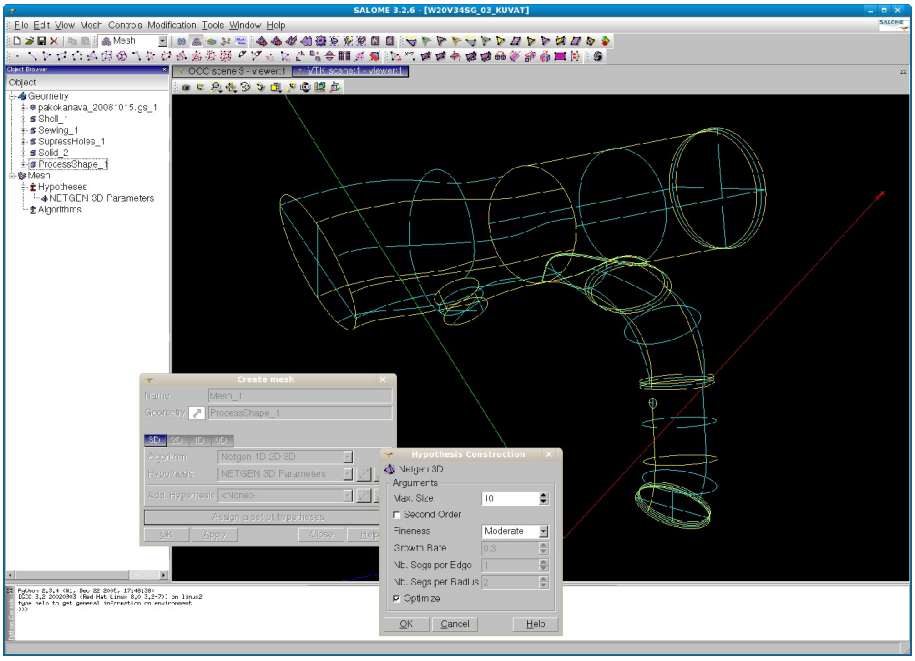


Figure 11: Meshing in MESH module using Netgen meshing routine.

For initial meshing of CFD computation Netgen provided simple and robust routine. It was easier to create a mesh with relatively equal sized elements with Netgen than using manually defined meshing algorithms and parameters for 2D and 1D meshing. Meshing with manual 2D and 1D routine definition often ended up in low quality mesh that was not suitable for CFD use (Figure 12). Below is described meshing procedure with Netgen routine in more detail.

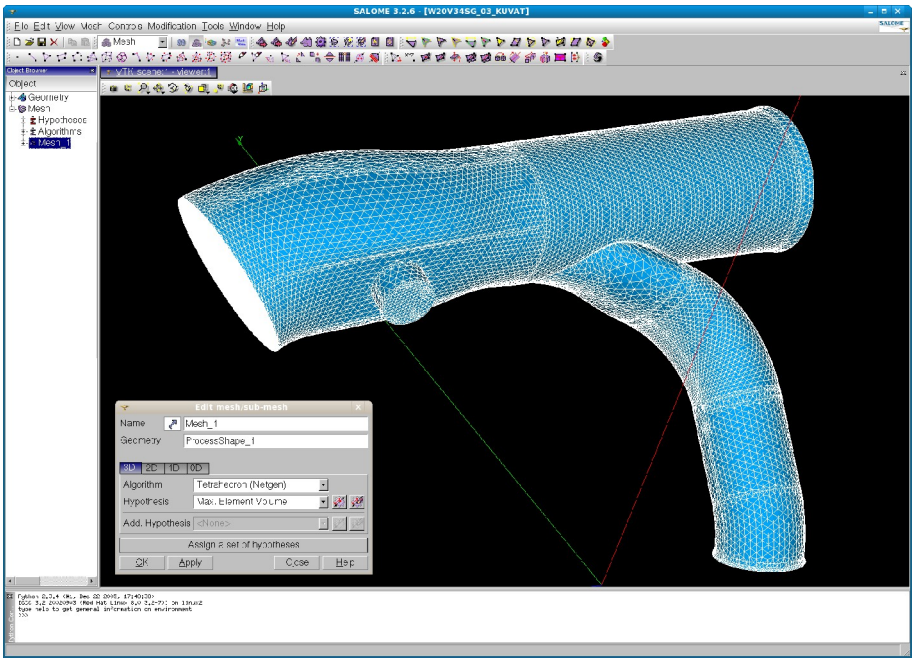


Figure 12: Meshing with manual 2D (Mefisto in this case) and 1D routine definition often ended up with low quality mesh. In this case there are higher mesh densities in surface component boundaries and the overall mesh is not relatively equal sized.

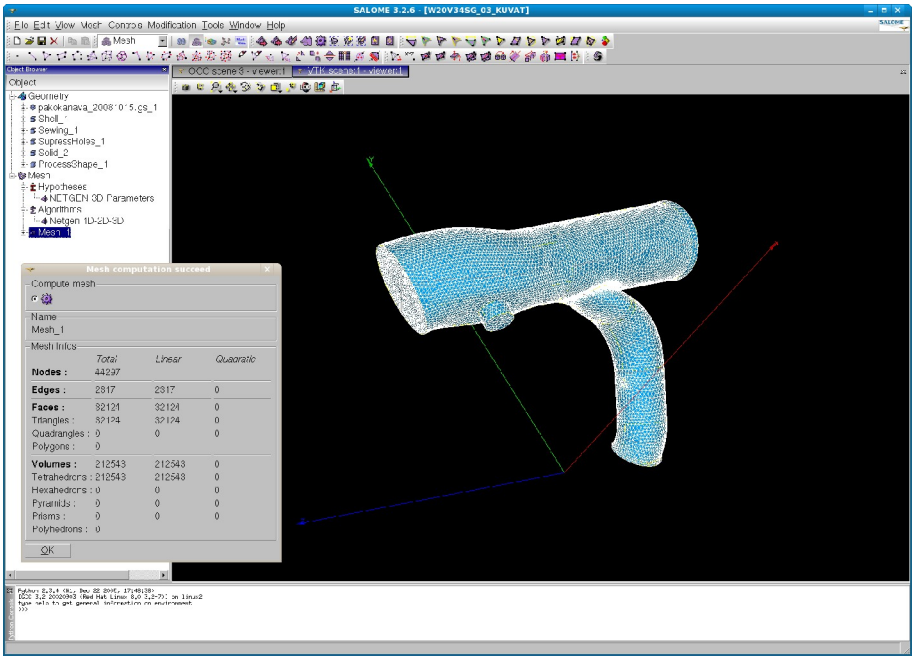


Figure 13: Mesh generation result. The whole meshing operation took only couple of minutes the test system. There were only few parameters that could be adjusted for meshing. The mesh is probably more optimized for structural analysis than for CFD; the mesh is relatively dense in channel joint corners.

Meshing using Netgen algorithm was slower but didn't require that much of physical memory. There were only few parameters that could be passed to Netgen routine. E.g. minimum size of an element couldn't be defined. Netgen routine seemed to produce small elements in areas of small geometric details, like the joint of the channels. The algorithm could well follow small fillets in the joint but the result was a mesh with locally relatively small elements (see Figure 13 and Figure 14). The mesh is most likely more suitable for structural analysis than for computational fluid dynamics. Now the small elements in joint area are not in the area of high flow transients. Still these small elements often define the maximum time step for the CFD computation and thereby ruin the efficiency of the computation.

Creating structural mesh with hexahedral elements is practically impossible for such a complex geometry as the one used in this study. Structured mesh can't be created easily from a solid geometry so that all the faces and edges are well discretised. Automatic hexahedralisation would have required lower level geometry components to exist. Also the geometry should have had suitable topology for meshing.

With large geometry models the updating of the visual is very slow causing e.g. the change of representation from wireframe to shaded take several minutes. This makes it difficult to work fluently when trying to simplify the original CAD geometry to more suitable form for meshing.

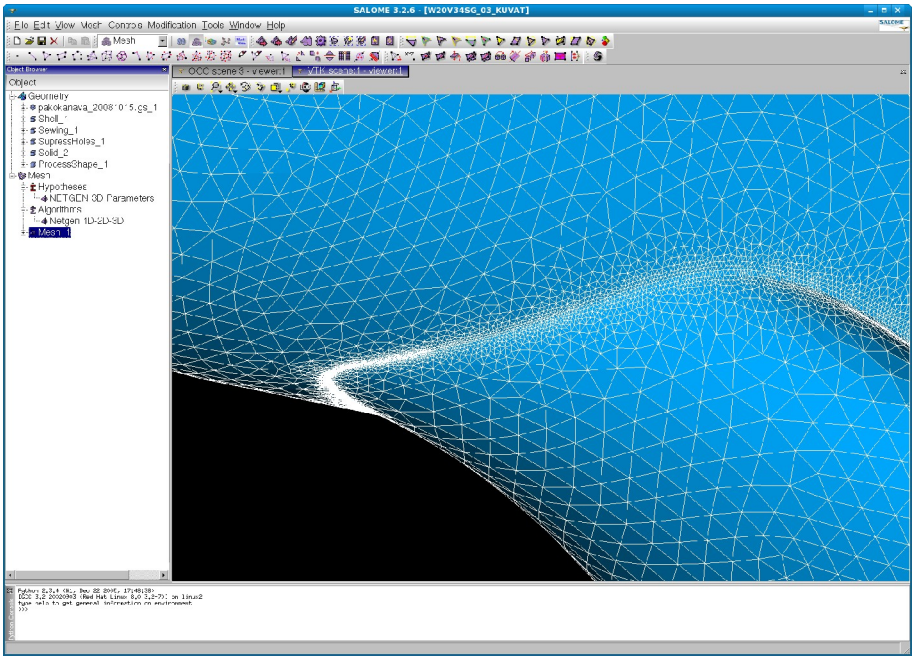


Figure 14: Small elements in exhaust manifold channel joint area. These small elements set the maximum time step for flow computation.

Robustness of the code is in some places low. E.g. manipulating the view can lead into situation where interactive mouse selection is off-set from visual representation. This makes it difficult to work with small geometric components. In some situations the left side database tree representations gets corrupted and lots of additional components are presented in the tree. Selections between solids and surfaces are not clear and displaying (selecting something to display) may take tens of seconds to update the view. Naturally this is a matter of both graphics routines and graphics hardware. Working with a mesh with hundred thousands or even millions elements requires quite powerful graphics hardware to be fluent.

3.7 Short introduction SALOME version 4.1.4

At the writing of this report a newer version of open source SALOME software was released. This version 4.1.4 contains numerous bug fixes and enhancements related to the tested version 3.2.6, including e.g. improvements to slow graphics with large models. There is also some new functionality but due to limited resources these have not been tested for this report. [3]

3.8 Using SALOME with OpenFOAM

SALOME can be used for mesh generation for OpenFOAM CFD software. OpenFOAM is very flexible what comes to element/cell shape and topology. Practically any shape of volume is suitable for OpenFOAM, thus tetrahedral mesh can be used. In general the most suitable element shape for control volume method would be hexahedron due to its optimal surface and volume ratio. Also, structural mesh can be more easily created so that the mesh conforms to flow directions which more favourable especially for CVM formulations. But because no general robust algorithms for automatic hexahedral mesh have been found, and because the overall process time of computation is critical, tetrahedral mesh is a good compromise.

4 Gmsh

4.1 Introduction

Gmsh is an open source modeller and meshing tool for simple geometries. At the website it is said “its design goal is to provide a simple meshing tool for academic problems with parametric input and advanced visualization capabilities”. In the most resent versions of gmsh support for CAD geometries is included using Open Cascade geometry kernel.

4.2 Geometry capabilities

Gmsh has quite limited geometry creation and modification capabilities. New geometry is created topologically from primitive components: points, edges, faces, and finally volumes. Open Cascade extension enables use of CAD geometries. Imported CAD geometries can’t be modified in gmsh. Gmsh can be used together with SALOME via Open Cascade connection. Geometries created with SALOME can be imported into gmsh and further meshed. This is how gmsh was tested with the test case geometry in this project.

4.3 Meshing capabilities

Like in SALOME also in gmsh there are different operation modes for different modelling phases. Geometry creation and manipulation is done in *Geometry* mode and meshing in *Meshing* mode. There is also *Solver* and *Post-processing* modes available for solver interfacing and results visualisation. The operation mode is selected from the main menu window’s select list (Figure 15). To use an existing CAD geometry, first a model needs to be created using *file new* operation. This creates a .geo file to the selected directory. CAD geometry is imported using *merge* operation from file menu. Parameters for different operations, like geometry merge and meshing, are set in *option* panel.

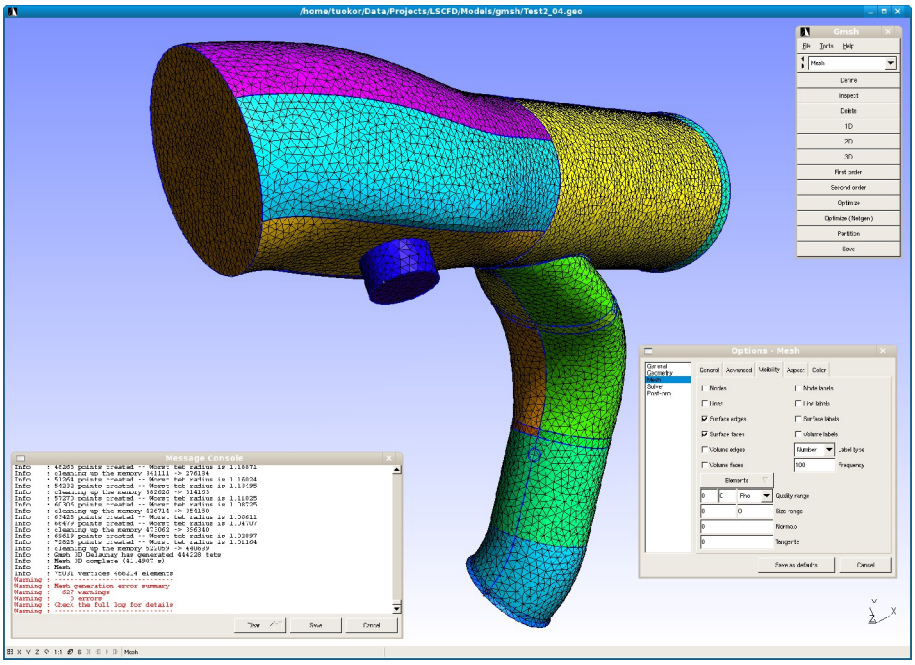


Figure 15: Gmsh user interface with main menu, options and console windows open.

There are couple of choices for 3D meshing in gmsh, all for tetrahedral mesh. For 2D surface meshing options are: *Frontal*, *Delaunay* or *MeshAdapt+Delaunay*. In the tested version of gmsh Frontal and Delaunay routines are still experimental. Visually comparing Frontal rou-

tine seemed to produce more structured surface mesh than Delaunay routine. In Figure 16 are shown surface meshes with the three different options; other parameters for meshing have been the same. For 3D meshing options are: *Tetgen+Delaunay* or *Netgen*. Surface meshing with different routines was in general the same. If the geometry was well formed the meshing parameters were set in *option* panel and *3D* option in *mesh* mode main menu was selected. Meshing progress and additional information could be seen in *console* window. The generated mesh is saved using *Save as* option from File menu and selecting the appropriate mesh file format.

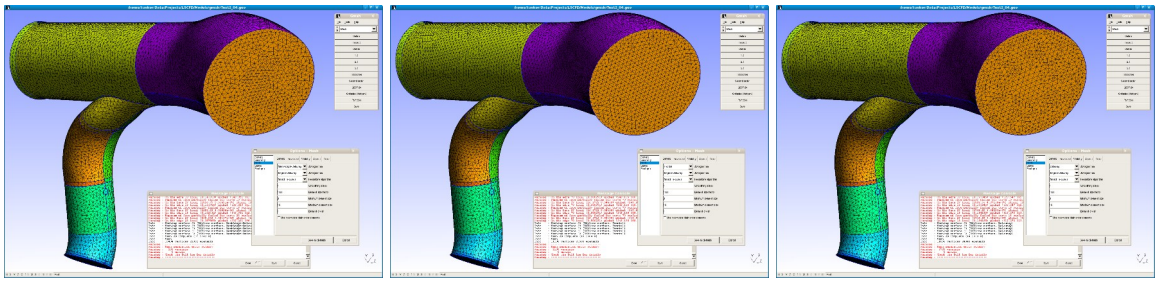


Figure 16: Surface mesh with different meshing routines in gmsh. From left to right: Frontal, Delaunay, and MeshAdapt+Delaunay.

4.4 Import/Export capabilities

Although the software has quite limited geometrical capabilities it can read CAD geometries in the following format: IGES, STEP, BREP, and naturally gmsh own GEO format. For the CAD geometry information management the system uses Open CASCADE geometry kernel.

Existing meshes can be imported in several formats: I-deas UNV, VTK mesh, MED formats, Medit mesh, Nastran bulk data, Plot3D structured mesh, STL surface mesh, and VRML surface mesh. Exporting the mesh is possible in the same formats as import and also in DIFFPACK 3D mesh format.

4.5 Meshing efficiency

Netgen meshing routine uses Delaunay meshing algorithm for tetrahedral mesh generation. The meshing algorithm implementation requires only modest amount of physical memory and is relatively fast; generation of an unstructured mesh of over 800 000 elements takes only couple of minutes. Gmsh has two options for mesh optimisation. This function optimises the mesh inner topology so that unnecessary elements are removed. Depending on parameter values and selected method, optimisation can take longer time than initial generation of the mesh. Still this operation takes just couple of minutes and is worth using the decrease the size of computational model, and thus increasing execution efficiency of the final computation.

4.6 Mesh generator in action

4.6.1 Creating the flow channel geometry

Gmsh has a strange logic for model creation starting from CAD geometry. First, a new model needs to be created and then an existing CAD geometry is merged to the newly created model. Here, the problem is basically in naming the procedures; it would be more intuitive if the merge operation had been named import.

Gmsh does not have any geometry manipulation functionality for imported CAD geometries. The only thing that can be done is sewing surface component caps and removing small edge

components using Open Cascade routines. If the geometry is imported in STL format, first the geometry solid needs to be defined. This operation defines that the given closed volume is the domain the user wants to mesh. The gmsh meshing routine goes from edges to surfaces and finally to volume. In case of STL geometry this means the edge and surface meshing is already done (STL is already triangulated representation of the volume surface) and can't be modified anymore. Due to this limitation it is practical to import the geometry in some other format, e.g. Open Cascade BREP format. This allows more freedom to meshing fine tuning. BREP was used when testing gmsh with the selected test geometry. After the test case geometry had been imported there wasn't much to do for it.

4.6.2 Meshing

The meshing process started from defining meshing parameters. Default values were used except for maximum element size. Also the option for *Use incomplete high order elements* was deselected. After this the meshing was done in phases by selecting in sequence 1D, 2D, and 3D from the gmsh main panel.

Different gmsh meshing routines were tested with the same input parameter values. Tetgen routine produced 397 348 elements with given parameters. For the same model and same meshing parameters Netgen produced only 110 636 elements. Both methods produced few elements with very high aspect ration (Tetgen: > 3 000, Netgen: > 1.0E+06).

The test geometry was meshed with gmsh several times. The meshing algorithm used in gmsh does the meshing by first discretising surface component edges, then surfaces starting from edges, and finally volumes starting from surfaces. This approach is sensitive to primitive geometries and the overall quality of the geometry. If surface edges are composed from small pieces, the meshing can produce extremely small elements in the middle of a simple surface (Figure 17).

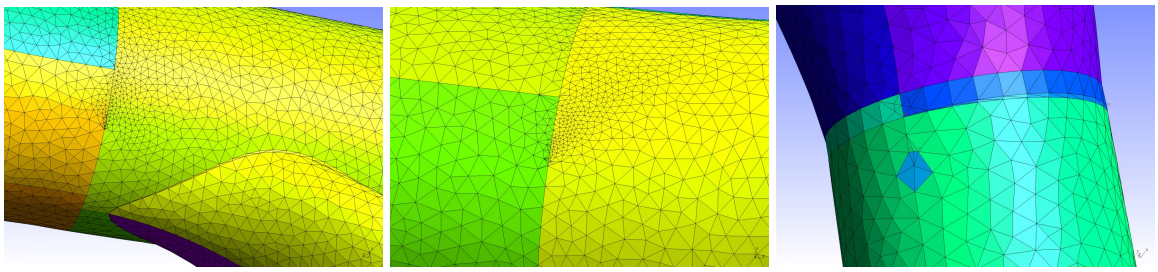


Figure 17: Example of small edge component influence on mesh density. Unhealed geometry imported from SALOME to gmsh and meshed using Netgen routine.

4.6.3 Using gmsh with OpenFOAM

There are two ways to use mesh generated with gmsh, either in native gmsh mesh format and using OpenFOAM gmshToFoam utility or using I-deas UNV format and OpenFOAM IdeasUnvToFoam utility.

5 Using OpenFOAM with external meshing tools

To convert UNV files, generated e.g. with SALOME or gmsh, to OpenFOAM native mesh the ideasUnvToFoam utility is used. Before using this utility the minimum OpenFOAM case structure has to be created including the *case* directory, the *system* sub-directory and *controlDict* input file. The syntax to run mesh conversion is:

```
ideasUnvToFoam . Test_01 geometry/test_01.unv
```

Here the "." refers to the OpenFOAM case root directory, the next parameter (Test_01) to the case sub-directory and the last parameter to the input UNV file (in this case under another sub-directory).

Fot the use of native gmsh mesh files there is a utility in OpenFOAM:

```
gmshToFoam . Test_01 geometry/test_01.msh
```

After converting the mesh the of OpenFOAM is in principle as with native OpenFOAM mesh.

To automatically find and separate patches in the mesh the autoPatch utility is used. This routine finds edges of the mesh that are sharper than given angle value. The syntax is:

```
autoPatch . Test_01 60
```

Here again the first parameter is the OpenFOAM case root directory, the second parameter to the case sub-directory and the last parameter to patch edge angle.

In OpenFOAM version 1.5 case root directory and case directory can be left out if the utility is run in a sub-directory level. More information how to use OpenFOAM and its utilities can be found from the user documentation.

6 OpenFOAM snappyHexMesh

6.1 Introduction

OpenFOAM snappyHexMesh method differs from the traditional way of doing pre-processing for CFD. This method uses automatic procedure to create orthogonal hexahedral mesh either around or inside given geometric surface. The surface is given is STL format and can be generated with a CAD or some other pre-processing tools like SALOME or gmsh. In principle this method enables fast and robust meshing of complex geometries and can be even used for automated computation procedures. The meshing procedure is described in detail in OpenFOAM User Guide [5].

6.2 Meshing capabilities

There are several parameters for mesh creation adjustments. The initial mesh density is set by creating a block mesh that covers the whole domain. This is the mesh refinement level 0. The refinement level for the surface area can be set either for whole model or for different surface components separately. The surface adaptation layer thickness and the maximum allowed non-orthogonality of the snapped surface cells can be set among many other parameters. The overall meshing process is sequentially iterative and also the mesh computation parameters need to be set.

The quality of the mesh can be increased in some cases by using the mesh layer functionality. This offsets the selected surfaces and creates a mesh layer that follows the surface. This is especially useful for computing boundary layer effects like aerodynamic drag or flow separation.

6.3 Import/Export capabilities

The geometry for OpenFOAM snappyHexMesh is described either in OpenFOAM's own tools like blockMesh or by using surface model of the geometry in STL format. Almost all CAD tools can produce STL formatted geometry models. STL is triangulated surface ap-

proximation of the geometry and it contains only information about triangle corner coordinates. This makes it very simple to represent any geometry in STL but also looses the information of the original mathematical surface (e.g. sphere or cylinder).

6.4 Meshing efficiency

Depending on the model size and complexity and if e.g. separate surface mesh is defined, the meshing with snappyHexMesh tools takes from couple of minutes to tens of minutes. After the meshing parameters have been successfully defined, the meshing process is automatic. This is especially practical if different mesh densities are to be tested or used. Because the meshing is not interactive and no graphical tools are used, the creation of large meshes is efficient. This enables large computational meshes to be created in a normal workstation PC.

6.5 Mesh generator in action

In Figure 18 is presented a mesh created for the test case. On left (blue mesh) the core mesh in mesh density level 0 is presented. Clear jagged surface can be seen, thus the shape of individual cells is near optimal (a cube). In the centre (green mesh) the finer refinement level is presented. The mesh is still jagged but can follow the outer surface relatively well. On the right (red mesh) the outer mesh surface is presented. In this layer the outer cells are not anymore block shaped but one or several cell surfaces are snapped to geometry surface.

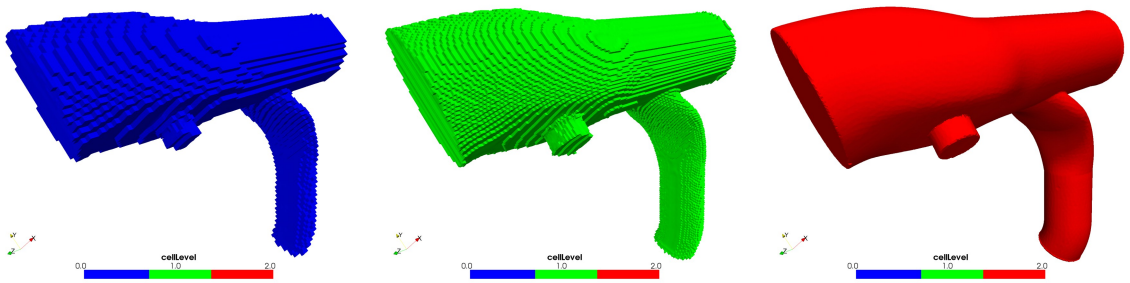


Figure 18: Different mesh refinement levels in the tested case. The coarsest mesh on the left is the initial mesh defined using the OpenFOAM's blockMesh utility, the intermediate mesh is in the centre and the finest surface mesh on the right.

In the present version the recognition of surface edges and corners is still limited and the generated mesh can't always follow these geometrical features. In some cases this can cause the flow case to be ruined. In Figure 19 can be seen how the meshing algorithm can't follow sharp details the test case geometry. The left side (red surface) is the original STL geometry and the surface on right (in green) is the snappyHexMesh mesh. Edges of the original geometry are partially chamfered in the mesh.

7 Discussion and Conclusions

In this work three different open source tools or methods to create computational meshes for CFD, and especially for OpenFOAM software, were studied and tested. Two of these, SALOME and gmsh, were independent mesh generation software and the third one was a utility in OpenFOAM software package. The purpose was to mimic an industrial meshing process starting from a real-world geometry model in IGES format. With this procedure the process included also all the difficulties that a detailed geometry can produce for geometry manipulation and meshing. The selected case was found to be very useful for the purpose; the geometry was apparently simple but still included small geometry details and from meshing point of view some difficult geometric features.

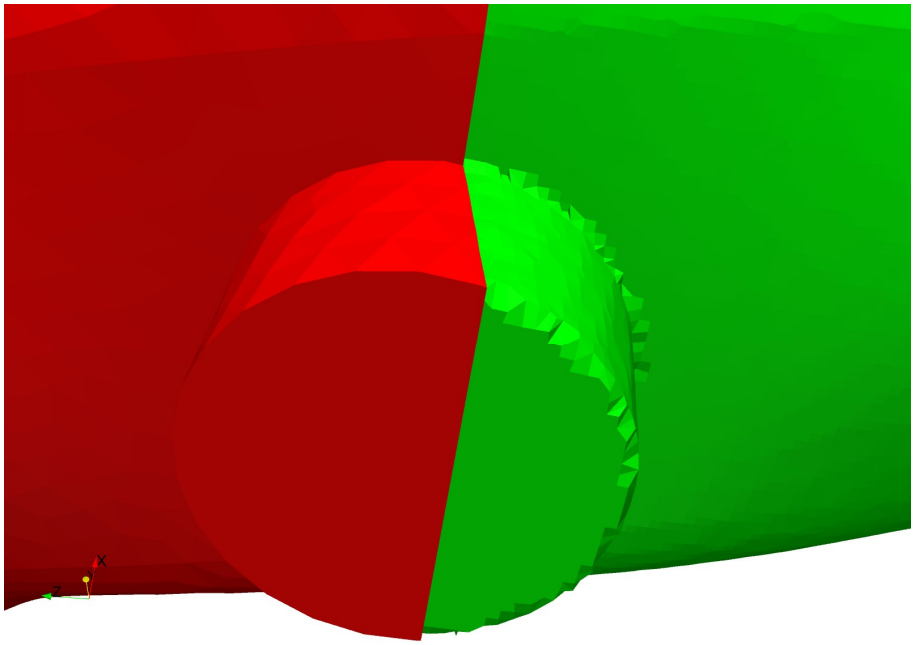


Figure 19: An example how OpenFOAM snappyHexMesh in the tested version 1.5 can't always follow sharp edges and corners. On the left in red is the STL geometry and on the right in green is the meshed geometry.

The SALOME pre-processing software was the most complete package for this purpose. It includes quite good functionality for geometry model creation and modifications. The meshing is made relatively simple and in principle it is possible to generate structured hexahedral mesh. In practice the meshing process requires concentration on documentation for both to understand the used terminology and all the process phases. And in addition the control over the meshing of the geometry is not simple. It looked like the software was optimised for FEM meshing so that geometry edges and small roundings produced really small computational cells.

Gmsh was very compact package it was quite easy to create computationally good quality mesh for CFD. The big disadvantage of gmsh is that it does not have geometry manipulation functionality for imported CAD geometries, but it can only mesh the imported geometry as is. At the software's website it is clearly said that to software is focused for academic use. Still the software is quite intuitive to use and it can handle also complex geometries.

The snappyHexMesh utility in OpenFOAM software package introduces a different but exciting method to efficiently create meshes for complex geometries. The learning step with this tool is relatively high, but the step is worth taking. This utility enables also large computational meshes to be created in a normal workstation. The batch kind of working process is especially efficient if the same geometry is needed to remesh with different mesh density or modified details. The big disadvantage is obviously that the process is not interactive and many of the definitions need to be learned by try-and-error. Also the geometry model in STL format has to be generated with some third party programs. Also, the present version of the snappyHexMesh utility can't handle well sharp edges and corners. This can be a serious problem in some CFD cases and may prevent using this method for meshing.

For fast meshing of complex industrial geometries the best way is either to mix SALOME and gmsh for unstructured tetrahedral mesh or to use OpenFOAM snappyHexMesh, if the geometry and computation case allows it.

With the tested two interactive pre-processing tools, SALOME and gmsh, one surprising defect was found: defining named boundary regions for further use is either not possible or it is difficult and doesn't work properly. Named boundary regions mean that the user can define bounded mesh surface areas (sets of cell external surfaces) and give these regions names. E.g. in case the mesh is used with OpenFOAM, named boundary regions remarkably simplifies the definition of the flow computation case and allows the case definition without using a visualisation tools just to check the boundaries and naming them with some descriptive names.

An industrial meshing process is usually a compromise of mesh quality and the time used for meshing. Regarding this, none of the tested tools or methods can be recommended for general industrial use as such. More development needs to be done for these meshing tools to reach a level where the whole process is efficient and produces high quality results. But the development seems to be on right path and all the elements for good progress are already available.

The study indicates that there are good open source software components already available for CFD pre-processing, but at least the tested implementations haven't been completely successful in using them and implementing a really user friendly, intuitive and industrial strength pre-processing tool. The trend in commercial tools seems to be towards integrated modelling and simulation environments. The advantage in this approach is that the dataflow between different tools in the process is fluent. The disadvantage is that some generality is lost because often this approach just ignores the possibility of using just some tools of the software and the change to a totally different software product. The other approach, in which separate tools concentrate only on few tasks but do those tasks well, is usually supported by expert users. The problem especially for beginners and temporary users is the complexity of different tools interaction and compatibility of data in different process phases.

References

[1] Gmsh website http://www.geuz.org/gmsh/ (December 19, 2008).

[2] SALOME online documentation for version 3.2.6 (December 19, 2008).

[3] SALOME online documentation for version 4.1.4 (December 19, 2008).

[4] SALOME website forum pages http://www.salome-platform.org/forum/ (December 19, 2008).

[5] *User Guide*. OpenFOAM, version 1.5. 9th July 2008. OpenCFD Limited.

[6] *Programmer's Guide*. OpenFOAM, version 1.5. 9th July 2008. OpenCFD Limited.