SALOME 3.2.6

Public release announcement

May 2007

General information

OPEN CASCADE is pleased to announce SALOME version 3.2.6. It is a public release that contains the results of planned major and minor improvements and bug fixes against SALOME version 3.2.2.

Table of Contents

New features, improvements and general changes

Prerequisites changes

Salome 3.2.6 is based on the new Open CASCADE version 6.2 public release. The most important features of the new OCCT version are:

- Implementation of multithread safety in OCCT Kernel
- Improvement of Exception mechanism on Linux and UNIX
- Implementation of new visualization approach based on OpenGL arrays for AIS shapes
- Numerous other improvements in Visualization
- Several fixes of bugs came from SALOME platform projects
- New keys for Open CASCADE 6.2 configuration have been added:
  - A new preprocessor macro _OCC64 has been added that allows to know, at compile time, whether the code is compiled for 64- or 32-bit platform. This knowledge is necessary for platform-specific code such as selection of value of UndefinedHandleAddress, memory block allocation granularity, etc. The –D_OCC64 key must be added to CFLAGS and CXXFLAGS when compiling on a 64-bit platform.
  - In order to support multithreading in OCCT and applications based thereon it/they must be linked with the `thread` library. To do so `–lthread` string must be added to LDFLAGS
  - New configuration keys for Open CasCade building have been implemented:
    - —disable-draw – allows Open CasCade building without Draw.
    - —disable-wok – allows Open CasCade building without WOK.
    - —disable-jcas – allows Open CasCade building without JCas (Wrappers).

  For example:

  ./configure LDFLAGS=-lpthread CFLAGS="-m64 -D_OCC64" CXXFLAGS="-m64 -D_OCC64" —disable-draw —disable-wok —disable-jcas --prefix=${INSTALL_DIR}

Important note: Due to some changes in OCT 6.2 NETGEN must be recompiled with the patch coming with Installation procedure in NETGENPlugin sources. Binary version of NETGEN in Install Wizard has been already recompiled for all platforms.

Installation wizard

SALOME Install Wizard modifications

Install Wizard has been modified to take into account the requirement to have a possibility to customize the buttons on the last "Finish Installation" page of the Installation Wizard.

OCC has implemented the support of an additional section for XML files (named <button>) which allows to add buttons and attach some script to these buttons. The script is responsible for the performance of certain actions (in addition this script is used to set the "enabled" state of the button - when no action can be done this script should give a signal about it).

Moreover, the existing "Launch SALOME" button is now also implemented with this feature. It means that this button can be completely removed from the Installation Wizard and replaced by another one (attribute <disable> of <button> section) .

Automatic building of SALOME modules

SALOME Installation Wizard now allows automatic building of SALOME modules from sources. The GUI provides an additional check box: "Build SALOME sources". If this check box is turned on, the selected SALOME module(s) will be built from sources during the installation. In this case the installation of the corresponding SALOME binaries is disabled (to avoid conflicts).
Note: this operation requires much more free disk space than for just unpacking the SALOME modules sources packages.
To enable this feature the build.sh script has been introduced. It implements the same functionality and supports the same set of options as build.csh script but it is targeted to the bash shell.
It is also possible now to install all the products from sources - special check box "Install all products from sources" has been added.
In batch mode this functionality is available via the --all-from-sources (-a) option.
This operation can be used to install SALOME on the Linux platform which is not officially supported.
Note: the installation of all products from sources is a very time consuming operation (it may require more than 24 hours depending on the computer).

The Installation Procedure documentation has been updated

General modifications

Mandriva 64 bit qualification

OCC has carried out a test campaign on Mandirva 2006-64 bit and made special debug for running SALOME on this 64 bit OS. A lot of problems were fixed and the version now runs without serious problems.

Export to STL format from GEOM and SMESH modules

Export to STL in GEOM module

Open CASCADE has implemented export to STL format from GEOM module as a new export function in menu item “export”. The user can now choose “*.stl” extension as well as brep, step or iges. The user can choose the save mode: STL binary or ascii. A new export interface has been created in accordance with current “export-import plugin” architecture in GEOM module, when interfaces are defined dynamically trough the resource file.

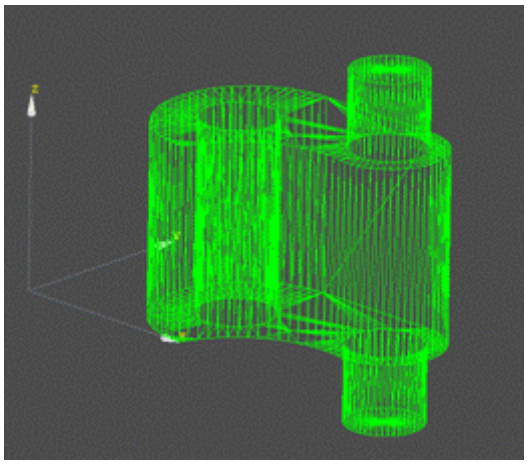


Fig 1. mesh_mechanic triangulation done with OCT internal meshing algorithm

Export to STL in SMESH module

Open CASCADE has implemented export to STL in SMESH module as a new menu item in the Export menu. In this case all triangles from the chosen mesh are exported into STL. If the mesh contains other kind of elements a “Warning” message is shown to the user.

If the mesh does not contain any triangles at all, a message box is shown to the user and no file is created.

Import/Export from/to SAT format in GEOM module
Open CASCADE has implemented import/export from/to SAT in GEOM module as a new menu item in the Import/Export menu. This functionality is not available in the Open Source. Please contact sales.contact@opencascade.com to purchase a corresponding license for this component activation.

GUI module

Update of the study using the SALOME session server

It is now possible to update the contents of the current study in the object browser outside of SALOME GUI by using the functions of SALOME session server `emitMessage('updateObjectBrowser')` or `emitMessageOneWay('updateObjectBrowser')`.

Changing the point of rotation in viewers

A new mode of rotation is available in SALOME now. The user can chose the center of gravity, or any existing vertex or node or just precise a point in the viewer. This selected point will be center of rotation.

A functionality to change the rotation point in GUI has been implemented for both, VTK and OCC viewers. The rotation point may be changed via the "Set Rotation Point" dialog which is accessible from viewers' toolbars.

Types of the rotation point:
1. The rotation point is the center of a bounding box (gravity center),
2. The rotation point is a point selected by the user (the user can type the points' coordinates in the dialog or select a point (node, vertex) in the viewer). The origin of the coordinate system belongs to this category.

X, Y and Z labels in dialogs

X, Y and Z labels have been aligned to the right in the following dialogs: "Move Node", "Add Node", "Rotation ", "Revolution", "Translation", "Symmetry" and "Extrusion Along a Path".

Significant improvement of SALOME PyQT interface

SALOME PyQT interface has been improved to provide more functionality for python modules.

1. Access to the log message output window from the Python modules has been implemented. The log messages output window is automatically shown by default when any Python module is activated. To disable this dockable window for some Python module, its window(...) callback function should be implemented in the Python GUI module.
   The API of the SalomePyQt module has been extended by the following methods:

   Print message to the log messages output window:
   message(<msg>, <add_sep>)
   where
   <msg> is the message to be printed in the embedded log output window;
   <add_sep> is a boolean flag: if it is True (by default) then the separator is printed in the log output window immediately after the message itself.

   Clear the contents of the log messages output window:
   clearMessages()

2. The API of the libSalomePy Python interface module has changed. All methods exported by this module: getRenderer(), getRenderWindow() and getRenderWindowInteractor() now accept boolean parameter "toCreate":
   a. - if this parameter is not 0, the new VTK window is always created;
   b. - if this parameter is 0 [default], a new VTK window is created only if no windows have been created by that moment; otherwise the existing window is activated and the corresponding VTK object is returned.

3. A new method has been implemented in libSalomePy Python module to show/hide the trihedron in the active VTK window: showTrihedon(<show>) where <show> is a boolean flag; if VTK is not opened there, nothing happens.

4. Method UpdateView() has been implemented in the libSALOME_Swig Python interface which redraws the contents of the current 2d/3d view:
```
import libSALOME_Swig
sg = libSALOME_Swig.SALOMEGUI_Swig()
sg.UpdateView()
```

5. Fixed bug: if custom VTK actors are displayed in the VTK viewer (e.g. via direct access to VTK renderer from the Python modules) which are not based on SALOME_Actor - the viewer operations like "FitAll" work incorrectly.

6. libSalomePy module : new methods have been implemented to operate with the VTK viewer

    a. fitAll() - to fit all contents of the current VTK viewer
    b. setView(<type>) - to set the Top, Bottom, Front, Back, Left or Right view position
    c. resetView() - to reset the view to the default state

New parameter of “runSalome” script

New command line parameter has been added for the runSalome.py script:
--test=[<hdf_file>|<python_script>][,<python_script>[,...]]

This parameter allows specifying the HDF file which should be automatically opened on the GUI desktop starting. In addition the user can specify one or more Python scripts to be imported in the opened study.

The files can appear in the arbitrary order. Only one HDF file can be specified.

If the HDF is not specified but Python script(s) appear with the –test parameter, a new empty study is created as alternative.

The Python scripts are imported in the order of their appearance in the option.
Caveats:

a. No checks for the Python scripts are made. The user should ensure that his Python scripts are available (PYTHONPATH environment variable is set correctly). If any error takes place during the Python script importing, the rest of the scripts are ignored.
b. If the extension is not given for some file - the ".py" is implied.

This option is available only in the GUI mode.
To import Python script(s) in the batch mode, use --terminal (-t) option.

Export of preferences from Python modules into standard “Preferences” widget of the SALOME Desktop

The possibility to export preferences from the Python modules to the common "Preferences" dialog box is available in SALOME.

To use it, implement in your MODULEGUI.py module method createSettings() and use SalomePyQt library to define the settings your module requires.

For example:

```
from qt import *
# The method createPreferences() is called automatically by SALOME GUI
# when the "Preferences" dialog box is invoked by the user.
# This method is called only once!
def createPreferences():
  # Import SalomePyQt module
```

```
  import SalomePyQt
  sg = SalomePyQt.SalomePyQt()
  # Create tab "Global" in the "Preferences" dialog box for your module's
page
  tid = sg.addPreference( "Global" )
  # Create a group of controls "General" on the page "Global"
  gid = sg.addPreference( "General", tid )
  # Let the group box "General" have two columns
  sgPyQt.setPreferenceProperty( gid, "columns", QVariant(2) )
  # Create "Display mode" combo box in the "General" group box.
  # Note that the name of the preference in the preferences file
  # is defined by the two last parameters.
  # You can always access this preference value by calling
  # sg.stringSetting("MODULE", "display_mode")
  # and
  # sg.addSetting("MODULE", "display_mode", "Shading")
  dispmode = sgPyQt.addPreference( "Display mode", gid,
                                   SalomePyQt.PT_Selector,
                                   "MODULE", "display_mode" )
  # Fill in the combo box with the values
  sl = QStringList(); sl.append("Wireframe"); sl.append("Shading")
  sgPyQt.setPreferenceProperty( dispmode, "strings", QVariant(sl) )
  # Add the font setting to the "General" group box
  sgPyQt.addPreference( "Font", gid,
                        SalomePyQt.PT_Font, "MODULE", "font" )
  # Add a new "Other" tab
  tid = sgPyQt.addPreference( "Other" )
  # Create group "Print settings" in the "Other" tab
  gid = sgPyQt.addPreference( "Sample", tid )
  # ...
  # etc
```

In general, preferences are added by the addPreference() method and different settings can be property set via the setPreferenceProperty() method.

SALOME GUI provides different controls which can be used in the "Preferences" dialog box: check box for boolean values; color, font and file selector tools; combo box for the list of possible choices; spin boxes for the integer and floating point values and other). Refer to the QtxListResourceEdit.* files to learn about available types of control and their properties (for example, spin box for the floating point value has "min", "max", "step", "suffix", "prefix" and "special" properties).

Please, note that since all these methods accept QVariant type as a value parameter, you need to construct it explicitly (as in the example above).

Unfortunately currently there is no support of QValueList<QVariant> values in the PyQt toolkit. Thus if the user needs to set the preference property which takes the list of values (except the list of strings - in this case you can use QVariant(QStringList())), you can use addPreferenceProperty() continuously instead of setPreferenceProperty().

For example, if you need to set custom values for the display mode items in the example above you can do the following:

```
# create string list
sl = QStringList()
# add first value
sl.append("Wireframe")
# add another value
sl.append("Shading")
# set strings to the combo box
sgPyQt.setPreferenceProperty( dispmode, "strings", QVariant(sl) )
```

```
# set value 10 for the first item (this value will be stored in the
# preferences file
sgPyQt.addPreferenceProperty( dispmode, "indexes", 0, QVariant(10) )
# set value 20 for the second item
sgPyQt.addPreferenceProperty( dispmode, "indexes", 1, QVariant(20) )
```

GEOM module

New partition algorithm

OCC has conducted a deep redesign of the partition algorithm. During this redesign the quality and performance of partition were significantly improved. API on new partition has also been changed. Only two parameters are necessary now: the object which is processed and the tool by which the algo makes section. Please refer to documentation.

New selection filter in OCC viewer

A filter for exclusive selection of objects of a certain type has been implemented in OCC Viewer. It can be activated through the menu "Select Only"-> ["Vertex", "Edge", "Wire", "Face", "Shell", "Solid", "Compound", "Select All"] from the 3D Viewer popup.

Inverse direction of the working plane

"Reverse the plane normal" check box has been added to the dialog. This control allows the user to inverse the direction of the working plane.

Value of reference object in object browser

The Object Browser now displays the proper value of the reference object instead of the value of the object it references to.

A new command `def geompy::KindOfShape (theShape)` allows obtaining information about a shape stored in GEOM_Object. For example, if the object is a sphere it returns its center and radius.

Generalization of the Gluing algorithm for non hexahedron solids

In Geom module an improvement has been implemented that allows the gluing algorithm to work on non hexahedral solids. From now it is more generic and can work with different solids.

The improvement deals with:

- ⇒ the inner parts of the main algorithm
- ⇒ existing auxiliary classes of the package GEOMAlgo
- ⇒ creating new auxiliary classes in the package GEOMAlgo

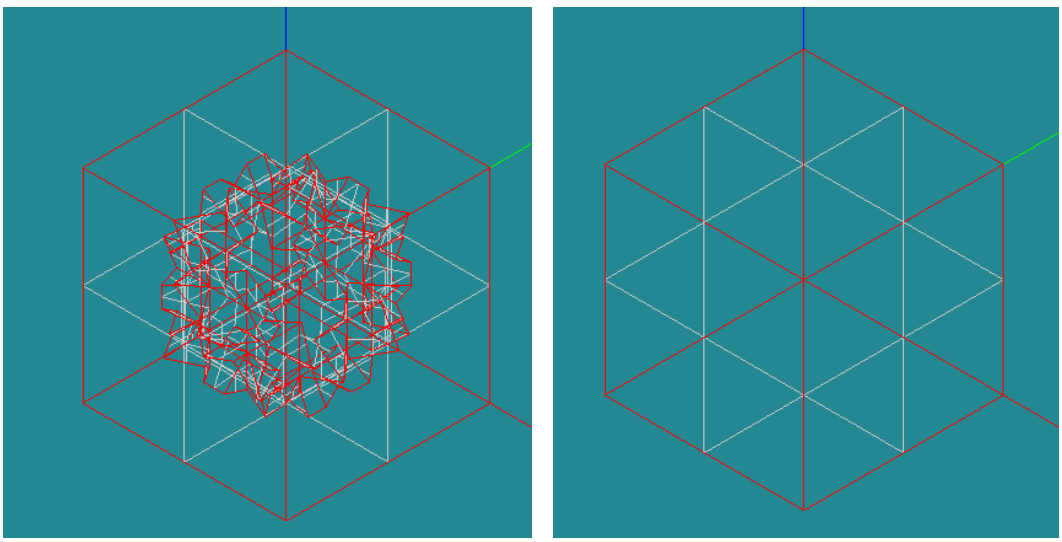


Figure 2a and 2b: Shape free faces before and after gluing

GetSame function which returns similar sub-shape from given arguments

A new function in Geom provides a way to find a similar shape in some argument. It can be used from the batch mode during the build of complex scripts. It allows finding the initial shape in some modified result to assign boundary condition properties.

New GetShapeOfBox function

New function if Geom allows finding a shape which is placed inside some box.

Rotation in Geometry module – new parameter

New mode has been added to Rotation functionality. In this mode, the object can be rotated and three points are given as arguments (the central one and two points to compute the rotation axis and rotation angle). This new mode is available in both Rotation dialog box and Geometry module TUI interface, `GEOM_ITransformOperations::RotateThreePoints()` and `RotateThreePointsCopy()` IDL methods, corresponding IDL and geompy.py functions.

New Geometry preferences

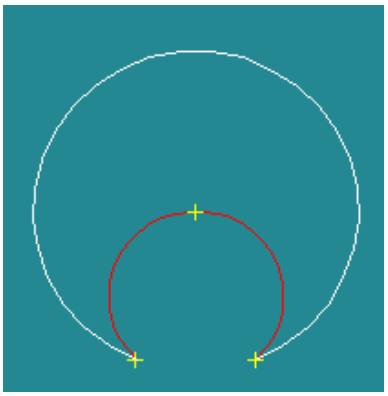
Two new fields have been added to the Geometry preference menu :

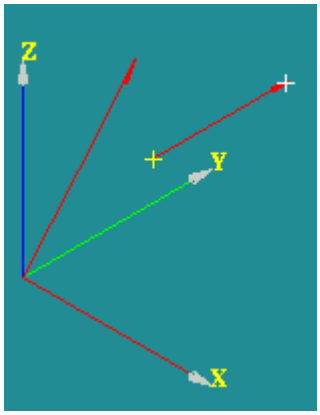
- type of the marker used to represent vertices.
- size of the marker.

New Arc Construction Algorithm

Now it is possible to build an arc by **Center, Start** and **End** points. The **Center** is the center of the circle of arc. The distance between the **Center** and the **Start** is the radius of the circle of arc. The **End** defines the angle of the arc of circle.

In the picture, the red arc is built using the old algorithm (three points lying on the curve), the white arc uses the new algorithm. The arc can be built even if the distance between the **Center** and the **End** is not equal to the radius.





Vectors with arrows

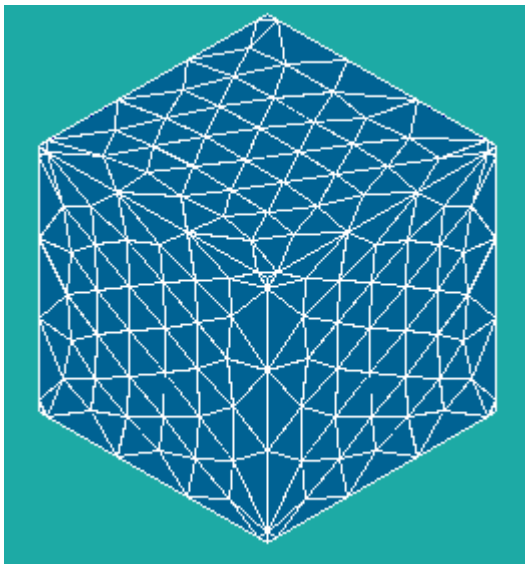
Now **Vectors** are displayed with arrows at their ends. The length of the arrow is equal to 1/10 of the length of the vector.

MESH module

New 0D Meshing Algorithm

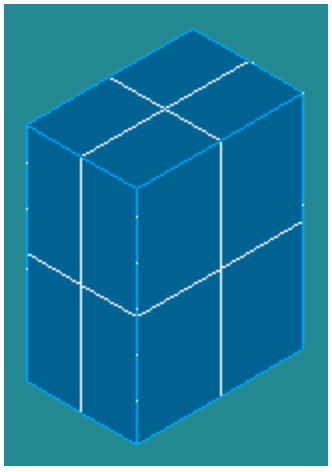
Segments around vertex meshing algorithm and the corresponding **Length near vertex** hypothesis now allow defining the local size of the elements in the neighborhood of a certain node. If applied to a whole geometric object, this algorithm meshes all its tops.

The picture shows a box with finely meshed corners.

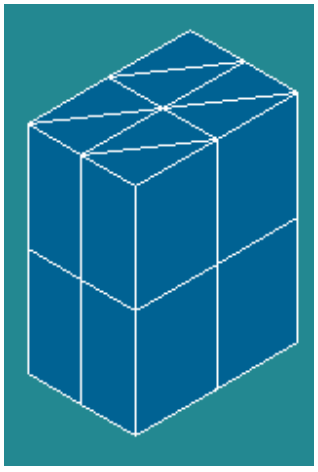


New TUI Commands for Splitting

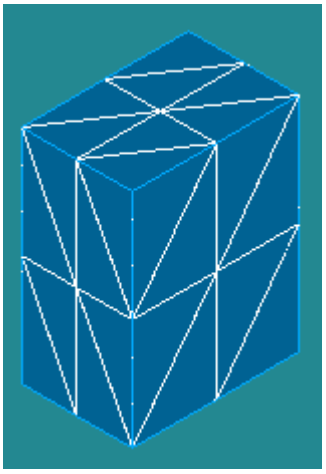
It is now possible to split quadrangle mesh faces into triangles, hexagonal volumes into tetrahedrons and hexahedral volumes into prisms using TUI commands `SplitQuadsNearTriangularFacets()`, `SplitHexaToTetras(theObject, theNode000, theNode001)` and `SplitHexaToPrisms(theObject, theNode000, theNode001)`.



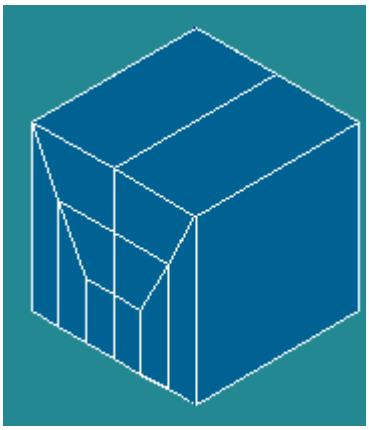
Hexahedrons



Prisms



Tetrahedrons

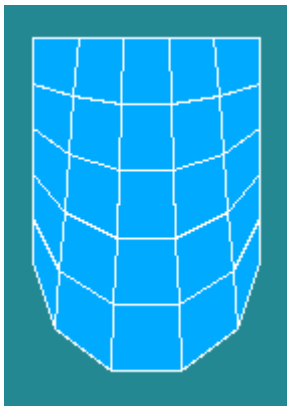


Hexahedron (i,j,k) algorithm improvement

Hexahedron (i,j,k) meshing algorithm is now able to mesh a rectangular shape even if the opposite edges of opposite faces have a different number of 1D mesh elements.

The picture shows a box meshed with the usage of Hexahedron (i, j, k) algorithm. The top and bottom edges of the front face have a different number of 1D mesh elements.

Note that it is required that the opposite faces have identical meshes.



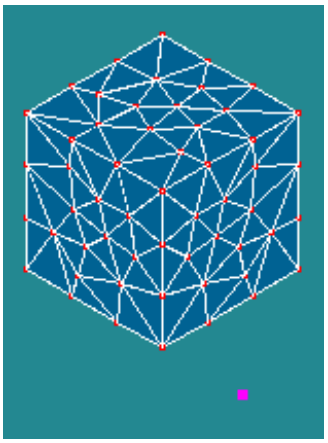
Meshing of faces with curvilinear edges

Quadrangle (Mapping) and **Hexahedron (i,j,k)** meshing algorithms are now able to work with quadrilateral geometrical faces bounded by more than 4 edges provided that all edges of a composite face side form C1 curve.

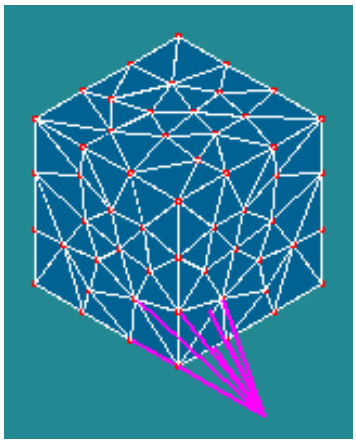
Composite side discretisation 1D meshing algorithm allows applying any 1D hypothesis to a whole side of a geometrical face even if it is composed of several edges provided that they form C1 curve, have the same hypotheses assigned and form one side in all faces of the main shape of a mesh.

Mesh to pass through point

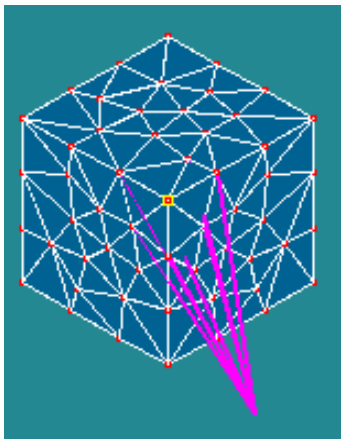
Mesh to pass through point mesh transformation functionality allows defining a node at a certain point either by creation of a new node, or by movement of the node closest to the point or by movement of the selected node to the point.



Create a new node



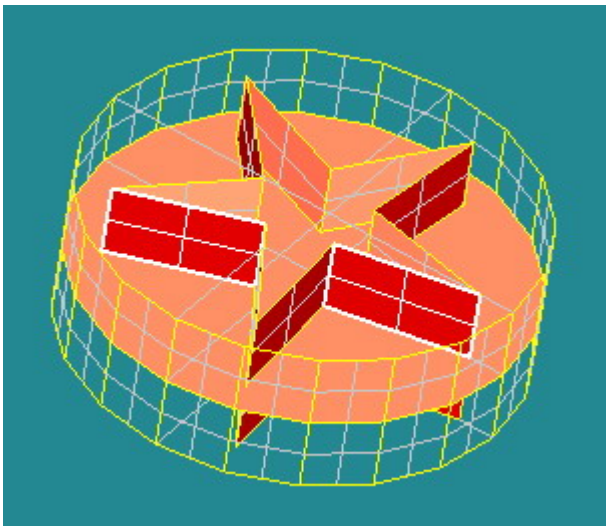
Move the closest node to the point



Move an arbitrary node to the point

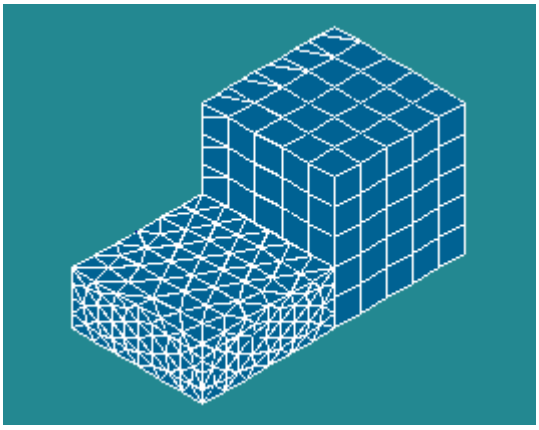
Selection of faces for gluing

This new functionality allows gluing only the selected faces. To perform this operation select the shape, specify the tolerance and press “Detect” button. Geometry module detects the faces where gluing can be performed and colors them in the screen with the red color. It is possible to select the faces for gluing in the 3D viewer. The selected faces will be marked in white.



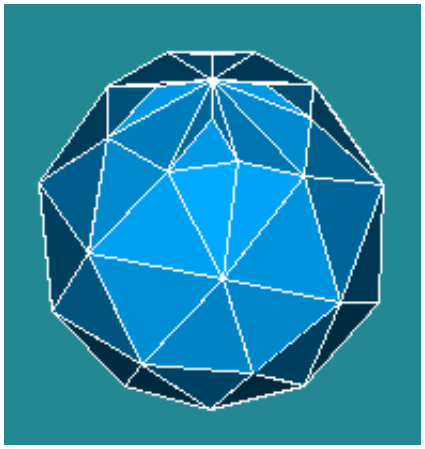
Creation of compound meshes

New functionality "Build compound mesh" allows uniting several meshes into one. The groups on the initial meshes that have the same names can be united or renamed. It is also possible to merge coincident nodes and elements.

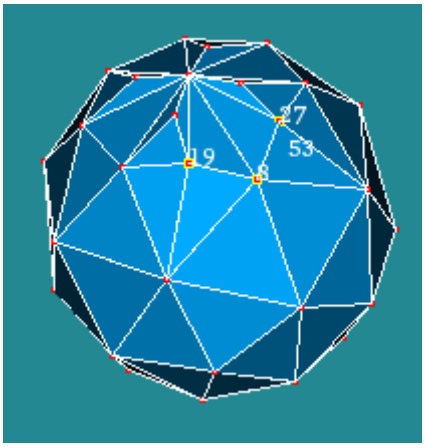


Redesign of Merge Nodes and Merge Elements

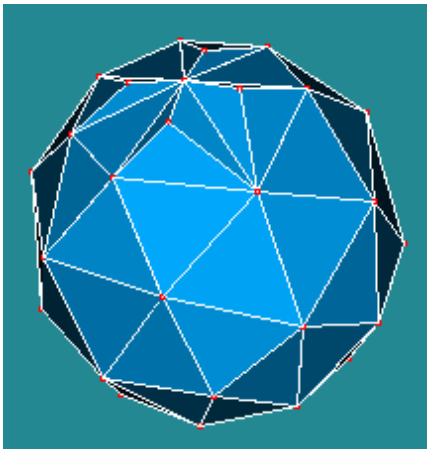
The functionalities "Merge Nodes" and "Merge Elements" have been extended with the possibility to edit the list of groups selected for merging adding or removing the groups and to edit each group in the list adding or removing its nodes or elements. It is also possible to give priority to a certain element or node, which means that all other elements or nodes will be merged into it. In the pictures below you can see the merging of only one group of elements: nodes 19, 27 and 53 are merged into node 8.



Mesh



Group of nodes on mesh



Result of merging

- The possibility to merge nodes only on a group or a sub-mesh has been implemented.
- New "Mesh Computation Succeeded/Failed" info box provides information on the mesh and a list of errors with the possibility of highlighting the subshapes causing errors.

New functionality to build a repetitive mesh

There is a new improvement which allows building a mesh in a specific way in the SMESH module. It projects a mesh existing on one shape to another one.

Five new meshing algorithms have been added.

Three algorithms, *"Projection 1D"*, *"Projection 2D"* and *"Projection 3D"*, copy mesh existing on one shape (edge, face and solid correspondingly) to the shape where the algorithm is assigned to. "Projection 3D" applies to blocks only.

"Prism 3D" algorithm can be used for meshing prisms, i.e. 3D Shapes defined by two opposing faces having the same number of vertices and edges and meshed with similar meshes. These two faces should be connected by quadrangle "side" faces.
The opposing faces can be meshed with either quadrangles or triangles, while the side faces should be meshed with quadrangles only.

"Radial Prism" algorithm applies to the meshing of a hollow 3D shape, i.e. such shape should be composed of two meshed shells: an outer shell and an internal shell without intersection with the outer shell. One of the shells should be a 2D Projection of the other shell. The meshes of the shells can consist of both triangles and quadrangles. The algorithm would fill the space between the two shells with prisms.

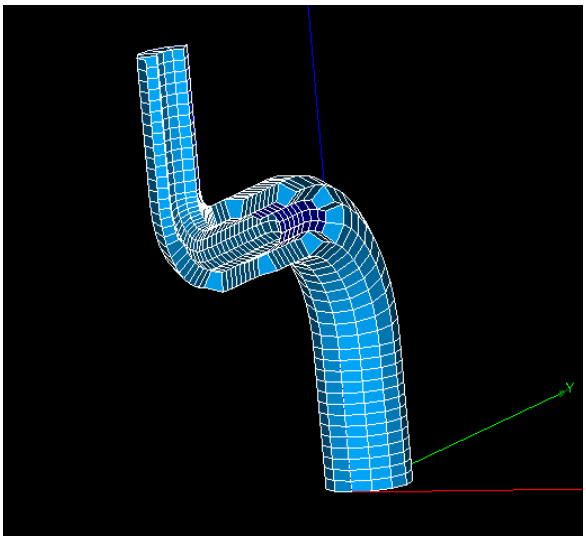


Figure 3a: Prism 3D

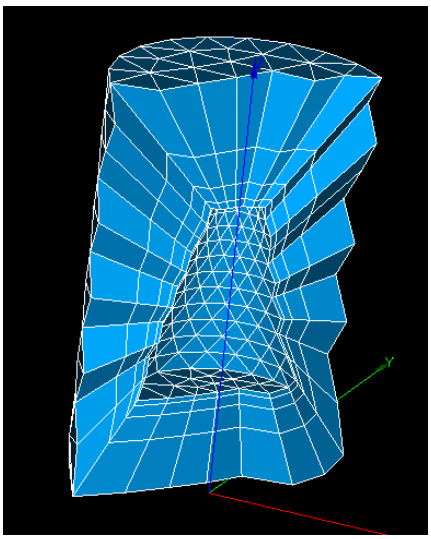


Figure 3b: Radial prism

New Mesh preferences

The "Automatic renumbering" option has been added to the MESH preferences menu. This option lets the user choose if the mesh has to be automatically renumbered before being exported (available for MED/DAT/UNV exports).

smesh.py interface

smesh.py public python API has been improved and now provides the same level of functionality like the IDL interface before.
During this task all SMESH_SWIG scripts have been rewritten for the new smesh.py interface.
Documentation examples were accordingly updated.

NETGEN 1-2D performance improvement

Special improvement of NETGEN 1-2D mesher has been performed. The mesher is now ~25 times faster for b-spline surfaces and ~twice faster for analytical ones. For example, computation of 2D mesh on “flight.brep” model takes half a minute instead of 15 minutes as before.

New filter predicate "BelongToGenSurface"

A new filter predicate "BelongToGenSurface" has been added to Mesh module. It allows finding nodes and elements lying on a geometrical surface of any kind within a given tolerance. It works in a way similar to "BelongToPlane" and "BelongToCylinder" predicates but without a limitation on the surface kind.

New Octree algorithm for detection of close nodes

The detection of close nodes was performed using an $O(n^2)$ algorithm, this algorithm has been substituted by a new Octree-like one. This Octree is an O(n) algorithm :

- With 28 586 nodes : 1min16sec vs 2sec for the Octree.
- With 141 245 nodes : 31min46sec vs 8sec for the Octree.
- With 1 476 528 nodes : 57h48min26sec vs 1min56 for the Octree.

VISU module

Building of presentations on groups

Now the user can build different presentations in VISU modules not only on the whole mesh, but as well as on different groups of this mesh.

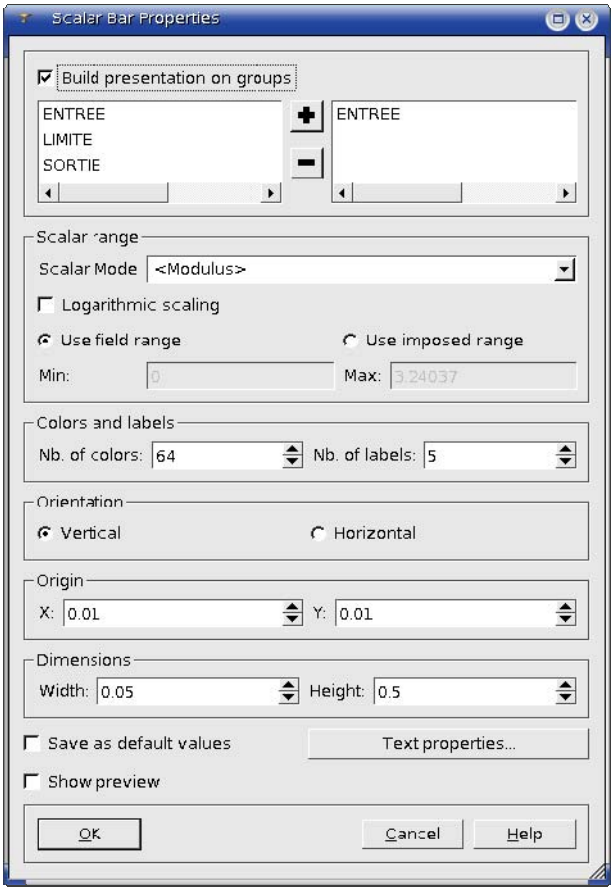


Figure 4: Dialogue box: group definition for a future presentation

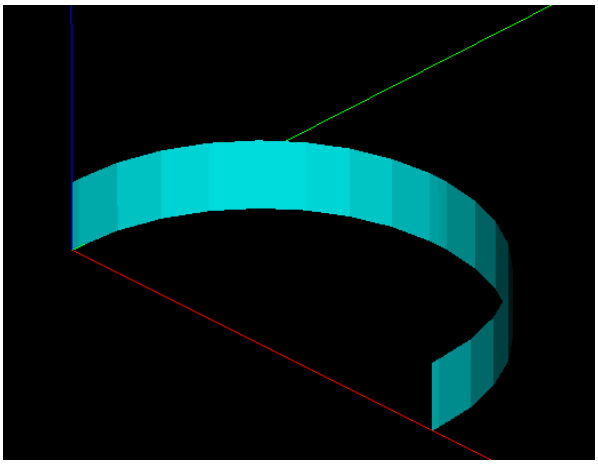


Figure 5a: A group

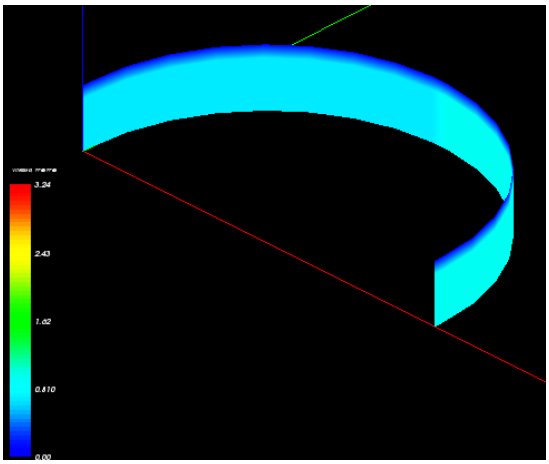
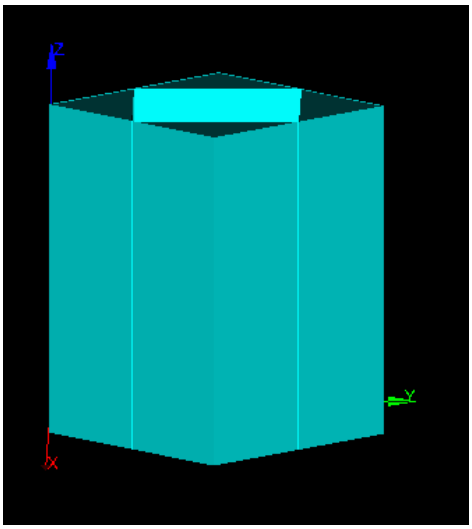


Figure 5b: Scalar map on a group



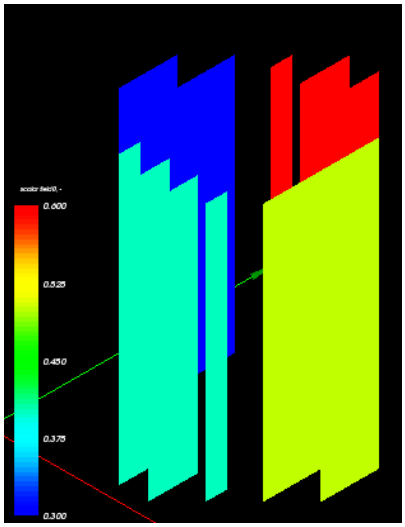


Figure 6a: Group presentation Figure 6b: ISO surfaces on a group

Support of structured mesh in VISU module

The VISU module now correctly supports internal mapping of elements during import of structured mesh. It also concerns clipping planes, selection info dialog.
The following new features are now available:

⇒ Clipping planes:
- o Work with unstructured meshes as before

⇒ Selection information must be used only on:
- o Families presentation
- o Group presentation

⇒ 3 Fields:
- o Scalar Map presentation
- o Deformed Shape presentation
- o Scalar Map On Deformed Shape presentation

The "Clipping planes" dialog now correctly works with structured meshes (i,j,k clipping).
The "Selection Info -> Data on elements" dialog has been modified and now more information is available to the user (for structured mesh only):

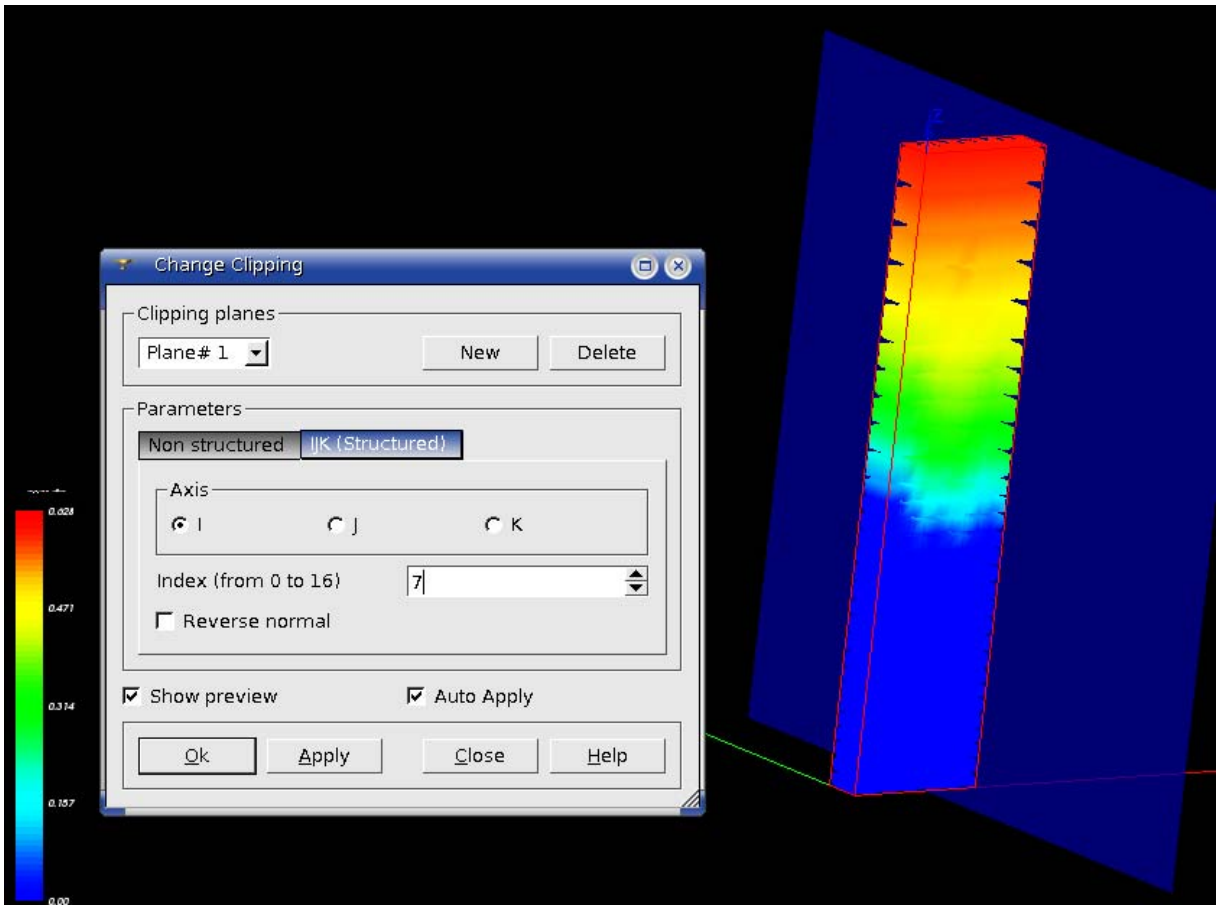


Figure 7: Support of structured mesh in VISU and its clipping

Successive animation mode

New animation mode is available now in VISU. If the user has different result files which contain computations of the same physical problem, and each file contains only a set of time stamps, now it is possible to animate all of them in one successive animation.
Parallel animation works as before.

Visu sweep animation improvement

"Sweep" Post-Pro operation can now be performed in three modes: linear (the only mode previously supported), cosinusoidal and sinusoidal.
In these new modes, the displayed values vary with time according to the formula F(t) = (1-cos(t))/2 (for cosinusoidal) or F(t) = sin(t-PI/2) (for sinusoidal). The sweeping mode is selected by the user through Post-Pro module preferences, along with the time interval (from 0 to PI or from -Pi to PI).

Visu – new kind of representation mode is available

"Surfaceframe" representation is now available for field data presentations (scalar map etc.). Similarly to the mesh presentation, this mode allows the user to see mesh edges on a shaded data presentation.

Visu preferences look-and-feel improvement

During the improvement of animation and some other improvement, preferences of VISU module were also improved. Now they have a more logical view, all preferences are logically organized into tabbed sections.

Visu – animation of bigger data is available now

New option of the animation dialog box allows cleaning of memory at each frame of animation. This allows animation of greater amount of data with the same memory limit as before.

Visu preferences have new option

New option "Display only on creation" has been added to preferences to tab "Representation". It is intended to erase any existing presentations in a view except the newly created objects, their scalar bar and meshes.

Redesign of SCALARMAP presentation

The SCALARMAP presentation on a MED field, built on a nodal profile, is now displayed with all mesh elements (points from profile subset). Scalar values are taken from the array of nodal profile values.

New mode during the creation of cut lines representation in Visu module

Possibility to switch between relative (0...1) and absolute cut line length has been added. Default value of this option is controlled through Post-Pro module preferences.

SUPERVISOR module

Edit Ports dialog improvement

"Edit ports" dialog has been improved to avoid duplication of contents of "Type" combo-box after changing workspace.

Improved handling of Supervisor Containers

The procedure of Container set-up in Supervision has been improved. The Container name for a new Factory node is now taken from the last node made by user for this component or from the last node of such component saved in an XML file if we modify imported dataflow.
Moreover some modifications have been done in the GUI part. Now the user can specify the Container name and other custom parameters during node creation. For this purpose the new button "Add Node and Customize Parameters" has been added into the "Add Node" dialog for Factory nodes. Parameters customization may be done with help of the "Set Custom Parameters" dialog.

XDATA module

Introduction

Since Salome 3.2.6 xdata python module has been introduced to the delivery.
This module is used to simplify the creation of python class with type verification and many other things like the automatic generation of a Graphical User Interface (GUI) or the plugin to Salome project toward corba...

For details see the README file.

Examples

Two new examples (Randomizer and Sierpinsky) are delivered with this Salome version:

Randomizer and Sierpinsky SALOME-based modules implement a simple interface to calculate Sierpinsky fields. For details see the README file in the SIERPINSKY_SRC_3.2.6 folder.

Supported Linux distributions and pre-requisites

SALOME 3.2.6 supports Mandrake 10.1, Debian Sarge, Mandriva 2006, RedHat 8.0, 9.0, RedHat Enterprise 4, Scientific Linux 4.2 , Scientific Linux 4.3 and Mandriva 64 bit.
SALOME 3.2.6 version has been mainly tested with the following pre-requisite list on Mandrake 10.1 platform.
SALOME 3.2.6 comes with different prerequisites versions on different platform (it try to reuse native products for Linux wherever it is possible), minimal supported version can be found in the second column of table below.
NOTE: For some platforms Salome uses prerequisites with patches like in RPM and defines specific keys. So if you compile products without Install Wizard we strongly recommend you to check compilation keys using shell files located in config_files folder of the Installation Procedure.

	Minimal required version	Mandriva 2006 32bit	Mandriva 2006 64bit	Debian Sarge	Mandrake 10.1	RedHat Enterprise 4 32bit	RedHat 8	RedHat 9	RedHat Scientific 4.2 32bit	RedHat Scientific 4.3 32bit
Gcc	3.2	4.0.1	4.0.1	3.3.5	3.4.1	3.4.1	3.2	3.2	3.4.4	3.4.4
Automake	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9
Autoconf	2.59	2.59	2.59	2.59	2.59	2.59	2.59	2.59	2.59	2.59
libtool	1.5.6	1.5.6	1.5.6	1.5.6	1.5.6	1.5.6	1.5.6	1.5.6	1.5.6	1.5.6
Tcltk	8.0	8.4.5	8.4.5	8.4.5	8.4.5	8.4.5	8.0	8.0	8.4.5	8.4.7
Python	2.3.4	2.4.1	2.4.1	2.3.5	2.3.4	2.3.4	2.3.4	2.3.4	2.3.4	2.3.4
Qt&msg2qm	3.3.3	3.3.4	3.3.4	3.3.4	3.3.3	3.3.3	3.3.3	3.3.3	3.3.3	3.3.3
Sip	4.1	4.2.1	4.2.1	4.1	4.1	4.1	4.1	4.1	4.1	4.1
PyQt	3.13	3.14.1	3.14.1	3.13	3.13	3.13	3.13	3.13	3.13	3.13
Boost	1.31.0	1.32.0	1.32.0	1.32.0	1.31.0	1.31.0	1.31.0	1.31.0	1.31.0	1.31.0
Swig	1.3.24	1.3.24	1.3.24	1.3.24	1.3.24	1.3.24	1.3.24	1.3.24	1.3.24	1.3.24
OpenCASCADE Technology	6.2	6.2	6.2	6.2	6.2	6.2	6.2	6.2	6.2	6.2
Qwt	0.4.1	4.2.0	4.2.0	4.2.0	4.2.0	4.2.0	0.4.1	0.4.1	4.2.0	4.2.0
OmniORB	4.0.5	4.0.7	4.0.7	4.0.7	4.0.5	4.0.5	4.0.5	4.0.5	4.0.7	4.0.7
Hdf	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4	5-1.6.4
Med	2.2.3	2.2.3	2.2.3	2.2.3	2.2.3	2.2.3	2.2.3	2.2.3	2.2.3	2.2.3
Vtk	4.2.2	4.2.6	4.2.6	4.2.6	4.2.6	4.2.6	4.2.2	4.2.2	4.2.6	4.2.6
Numeric	22.0	23.7	23.7	23.7	23.7	23.7	22.0	22.0	23.7	23.7
Graphviz	1.9	2.2.1	2.2.1	2.2.1	2.2.1	2.2.1	1.9	1.9	2.2.1	2.2.1
Doxygen	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6	1.4.6
NETGEN	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5	4.5
docutils	0.3.7	0.3.7	0.3.7	0.3.7	0.3.7	0.3.7	0.3.7	0.3.7	0.3.7	0.3.7

- Permanently tested platforms

- Platforms we do not test but know that Salome can be run on

Other versions of pre-requisites may also work.

SALOME 3.2.6 depends of some product for run time execution, some of other are necessary only for compilation or generation of development documentation (like doxygen for example). Below there is a list of mandatory and optional products.

Software Requirements

	Compilation and Development		Execution		Remarks
	Mandatory	Optional	Mandatory	Optional	
gcc	X		X		
Automake	X				
Autoconf	X				
libtool	X				
GNU make	X				
Tcltk					for OCC compilation from source files only
Python	X		X		
Qt	X		X		
msg2qm	X				
PyQt	X		X		
Boost	X		X		
Swig	X				
OpenCASCADE Technology	X		X		
Qwt	X		X		
OmniORB	X		X		
Hdf	X		X		
Med	X		X		
Vtk	X		X		
Numeric		X			
Graphviz		X			
Doxygen		X			
NETGEN	X		X		for NETGENPLUGIN mesh plug-in only
docutils		X			
cppunit		X			
mpi		X			required only if used at compilation step
openpbs		X			required only if used at compilation step
lsf		X			required only if used at compilation step
Ghs3d			X		for GHS3DPLUGIN mesh plug-in only

How to install and build SALOME

- Please follow README file from Installation Wizard for processing correctly installation of SALOME and all prerequisites
- If you would like to compile SALOME from scratch, please use build.csh script delivered with Installation Wizard. Call “build.csh –h” to see all parameters of this script.
    - *Important remark*: on RedHat 8 with the native automake-autoconf tools, sources of KERNEL from CVS can not be compiled. As a workaround there is specially prepared sources of KERNEL in Installation Wizard (after “make dist” step from Mandrake 10.1). They can be compiled with old prerequisites, but user must not call “build_configure” step. To compile he must call “configure”, “make”, “make install” as usual. Because of this please don’t use “build.csh” with “–b” option for KERNEL, because this option forces build_configure step. Call of “build.csh –i –o” process compilation and installation on RedHat8 correctly.

How to get the version and pre-requisites

The SALOME 3.2.6 pre-compiled binaries for Mandrake 10.1, Debian Sarge, Mandriva 2006, Mandriva64 and RedHat 8.0 can be retrieved from the www.salome-platform.org/download section.
There are sources of modules inside, and user can build sources from scratch using “build.sh” script coming with installation procedure.

There are two patches on NETGEN which are placed inside NETGENPLUGIN sources. One path for all 32 bit platforms, other one is addition to first and should be applied only for Mandriva 64.
During the compilation on NETGEN from sources by Install Wizard, the patches are applied automatically to the standard NETGEN installation. You can download NETGEN 4.5 from CVS of their official site http://www.hpfem.jku.at/netgen/

All other pre-requisites shall be obtained either from your Linux distribution (*please be sure to use a compatible version*) or from the distributors of these pre-requisites (*www.trolltech.com for QT for example*).

Known problems and limitations

- Compilation on 64 bit when the video card driver is not installed in standard location may fail and as a solution the prerequisites have to be recompiled.
- Netgen 1D-2D and 1D-2D-3D algorithm does not need definition of 2D and 1D algorithms and hypotheses both for mesh and sub-mesh. If you have defined 2D and 1D algorithms and hypotheses together with Netgen 1D-2D or 1D-2D-3D algorithm they will be ignored during calculation.
- SALOME supports reading of documents from the previous version, but documents created in the new version may not be opened in older ones.
- During the compilation of OCT 6.1.x by makefiles on a station with NVIDIA video card you may experience problems because the installation procedure of NVIDIA video driver removes library libGL.so included in package libMesaGL from directory /usr/X11R6/lib and places this library libGL.so in directory /usr/lib. However, libtool expects to find the library in directory /usr/X11R6/lib, which causes compilation crash (See /usr/X11R6/lib/libGLU.la). We suggest making links:

  "ln -s /usr/lib/libGL.so /usr/X11R6/lib/libGL.so ln -s /usr/lib/libGL.la /usr/X11R6/lib/libGL.la"
- VISU module does not support timestamps defined on the same field but on different meshes
- In the current implementation of "Save VISU" state operation the parameters of Gauss view Partition mode are not stored. If a window has been partitioned and saved, it will be restored as non-partitioned. The same concerns the background color.
- Fails of display of some presentation on quadratic elements in VISU (cannot create animation for IsoLines, CutPlanes etc.) is inside of the VTK. Currently used version of the VTK library (4.2.6) can not properly process the quadratic mesh elements (only ScalarMap and DeformedShape can be created only) that is presented in the MED file. Unfortunately it is impossible to replace or overload the VTK functionality outside of the library. This problem will be fixed automatically when we port the SALOME platform on the VTK 5.0 or higher version). This concerns Gauss viewer on quadratic elements. On some files with quadratic elements it is impossible to build gauss presentation.
- Step-by-step execution in SUPERVISOR on some graphs fails. This functionality is only a prototype and has not been finished completely
- Due to VTK 4.4 limitation, display of numbers of nodes in SMESH module has problems (some numbers disappear from the viewer)
- VTK presentation in GEOM was not completely finished and has problems with performance and memory usage. It desirable to use OCT viewer in GEOM module.
- MEFISTO algorithm sometimes produces different results on different platforms
- On some cases the number of triangles generated by MEFISTO may change from each attempt of building the mesh
- End user documentation for Supervisor module has been updated only in part of screenshots.