



Open CASCADE 6.1.1 Maintenance Release

Release Notes

Overview

Open CASCADE Technology 6.1.1 is a maintenance release, which includes new features, improvements and bug fixes, over minor release 6.1.



Version **6.1.1** is binary incompatible with the previous versions of Open CASCADE Technology, so applications linked against a previous version must be recompiled to run with this Version 6.1.1.



Table of Contents

- **[New Features](#)**
 - ✚ [Foundation Classes](#)
 - ✚ [Modeling Algorithms](#)
 - ✚ [Visualization](#)
 - ✚ [Products](#)

- **[Improvements](#)**
 - ✚ [Foundation Classes](#)
 - ✚ [Modeling Algorithms](#)
 - ✚ [Visualization](#)
 - ✚ [Data Exchange](#)
 - ✚ [Draw Test Harness](#)
 - ✚ [WOK](#)
 - ✚ [Documentation](#)
 - ✚ [Products](#)

- **[Changes](#)**
 - ✚ [Foundation Classes](#)
 - ✚ [Visualization](#)
 - ✚ [Data Exchange](#)
 - ✚ [Products](#)

- **[Appendix 1: Bug Fixes](#)**





Highlights

- **Open CASCADE**
 - **Implementation of multithread safety in OCCT Kernel**
 - **Improvement of Exception mechanism on Linux and UNIX**
 - **Numerous improvements in Visualization**
 - **Porting of OCCT to Microsoft Visual Studio 2005**
- **Products**
 - **Parasolid reader now supports schema numbers up to 17**
 - **Implementation of Surface Modification and Surface Curvature Analysis algorithms to SSP product;**
 - **Numerous improvements in OMF**
 - **Redesign of Products deliveries including new installation procedures**



New Features

Foundation Classes

- Basic support for multithreading has been provided at the level of TKernel services of OCCT:
 - New classes `Standard_Mutex` and `OSD_Thread` provide encapsulation of functions provided by the operating system to manipulate mutexes and manage threads in the uniform way, adapted for usage with Open CASCADE.

- Open CASCADE optimized memory manager has been protected for safe work within multithread applications where different threads can access it simultaneously. Note that this feature is not activated by default with the purpose of optimal performance. Environment variable `MMGT_REENTRANT` must be set to 1 to activate this feature. Note that setting variable `MMGT_OPT` to 0 (i.e. using CRT memory heap directly) is also thread safe.

Special note: for applications that heavily use OCCT memory manager from more than one thread, on multiprocessor hardware, execution with option `MMGT_OPT=0` can be more productive than with option `MMGT_REENTRANT=1`.

- OCCT exceptions and signal handling, type system (RTTI), and some other low-level services are protected to be thread-safe.

Note that these improvements provide only basic features for using OCCT in multithreaded applications; the application itself is responsible for ensuring safe access to both its own data and not-yet-protected data of OCCT.

- Four new classes have been added to the package `Bnd`, to be used in performance-critical algorithms
 - `Bnd_B2d`: 2-dimensional box using double-precision floating point
 - `Bnd_B3d`: 3-dimensional box using double-precision floating point
 - `Bnd_B2f`: 2-dimensional box using single-precision floating point
 - `Bnd_B3f`: 3-dimensional box using single-precision floating point

These boxes have the following differences from `Bnd_Box` and `Bnd_Box2d`:

- New boxes are always limited, which improves the performance of some algorithms.
- All new types consume less memory, particularly when single-precision boxes are used.
- Line-3dbox intersection (`Bnd_B3x::IsOut(theLine)`) now has the 2nd parameter telling if the line is treated as a ray; this improvement is aimed primarily at aiding 3D visual selection. The new box types are compatible with `NCollection_UBTree`, they can be used to instantiate AABB-trees in the same way as `Bnd_Box` and `Bnd_Box2d`.
- New macro definition has been added to `Standard_Version.hxx`: `OCC_VERSION_HEX`, which defines complete version number of Open CASCADE Technology as a hex number (with two positions per each major, minor, and maintenance version numbers). This facilitates checks for precise version number of OCCT. For example, to check if the current version is OCCT 6.1.1 or above, input: `#if OCC_VERSION_HEX >= 0x060101`

Modeling algorithms

- The new method `RotationPart()` has been added to the class `gp_Trsf2d` to obtain a rotation angle corresponding to rotational component of the transformation.



Visualization

- Interface of MeshVS component has been extended to allow interpolation of color by single element in accordance with arbitrary color scale (in addition to existing RGB interpolation).
- The possibility to activate dynamic highlighting of already selected objects during mouse movement has been implemented. A new method `SetToHighlightSelected()` has been added for this purpose in the class `AIS_InteractiveContext`. `SetToHighlightSelected(Standard_True)` method call activates dynamic highlighting of selected objects.
- New visualization approach based on OpenGL arrays for AIS shapes has been implemented both for Wireframe and Shading display modes. This approach gives an advantage in performance for visualization of AIS shapes.

Products

DXF

- New global Boolean parameter `read.dxf.anonymous.blocks` has been added in DXF reader. This parameter specifies whether anonymous blocks in a DXF file should be translated or not. Anonymous blocks contain geometrical representation of DIMENSION, HATCH, and other application-defined constructs not belonging to the geometry of the drawing itself, so their translation may be undesirable. By default, this parameter is enabled, i.e. such blocks are translated.

OMF

- Smooth shading visualization mode has been added in OMF. In this mode, the normal vector is taken at each node of each mesh element and passed to OpenGL to achieve a realistic image of the presentable mesh object. Old flat color mode is also available.
- The possibility to select interactively only visible mesh nodes or elements has been added. It is implemented through the standard mechanism of selection filters of OCCT visualization. For that purpose new filter classes `SMDSelect_VisibleFilter` and `SMDMeshVS_VisibleFilter` have been developed as direct descendants of `SelectMgr_Filter`. Support of this feature has been added in OMF Sample and can be evaluated there.
- Now OMF Sample uses MeshVS presentation instead of `SMDS_Interactive`. `SMDMeshVS` has been modified to support 3D elements. The order of nodes should be correct. There is still a possibility to use `SMDS_Interactive` by commenting `"#define USE_MESHVS"` in `"StdAfx.h"` file.

Surfaces from Scattered Points

- New Surface Modification algorithm defines surface point displacement set by the user with the help of the appropriate GUI. On algorithmic level the requested displacement is defined by U,V coordinates on the initial surface S_0 and the final point P that must have the same U,V coordinates on the modified surface S.

Two method of surface modification have been implemented.

- The first method uses only one point displacement directly requested by the user and some additional constraints to fix boundaries, if necessary. Any surface boundary is fixed, if the U,V space distance between the modified point and the boundary exceeds the given value (algorithm parameter).



- The second method uses the number of displacements (algorithm parameter) to calculate surface point displacement basing on the main displacement with the help of the user defined function.
- New Surface Curvature Analysis algorithm facilitates the analysis of the curvature of a surface at its boundaries (edges) and interior.

At input the algorithm takes a TopoDS_Shape (single face, shell, solid or compound of faces). At output it computes curvature values (Gaussian, Mean, Max or Min). and a normal curvature vector at the set of points on the relevant faces, corresponding to the vertices of triangulation on these faces.

Basing on that algorithm, the SSP sample application provides visual means for surface curvature analysis. The interior curvature of the surface is visualized by mapping curvature values to surface color according to Color Scale. The curvature at boundaries is displayed by straight segments directed at the center of the curvature at each point.



Improvements

Foundation Classes

- Declaration of basic OCCT classes such as `Standard_Transient` and WOK templates for generation of Handle classes have been improved to optimize the creation and destruction of Handle objects and other typical low-level operations with classes manipulated by such Handles as `IsKind()` and `DownCast()`.

Note that method `Standard_Transient::IsKind()` is not virtual anymore, and it is not redefined by every descendant class.

- The size of bucket arrays in the implementation of hash maps in both `TCollection` and `NCollection` packages has been optimized to increase performance, especially for big numbers of items (100K - 1M and more).
- The percent sign (%) has been added to the list of measurement units supported by `UnitsAPI` package.

Modeling Algorithms

- The memory leak in the class `TopExp_Explorer` has been eliminated. The code removing previously created objects of type `TopoDS_Iterator` has been added in the methods `Init()` and `Clear()`.
- The algorithm for Extrema calculation has been improved to be more precise if one of the arguments is an infinite face or an infinite edge.
- The method `BRepExtrema_DistShapeShape` has been improved to correctly process edges based on curves with C0 continuity (non-continuous first derivative).

The algorithm of 2d point on face classification used in Boolean operations has been improved to ensure correct processing of the cases when the test ray passes through vertex on the face boundary

- The sewing algorithm has been improved to provide correct `Same Parameter` flag on seam edges for cylinders.
 - A lot of uninitialized fields have been cleaned in packages `Extrema`, `math`, `ProjLib`, `Blend`, etc. T
- Earlier the realization of `SameParameter` raised memory exception on BSplines with the number of knots exceeding 1024. Now a protection has been added, and BSplines with more than 1024 knots are simply ignored and the function returns failed status.
- Shape Healing classes have been modified to avoid exceptions on internal vertices and edges present in the shape and to keep them in the resulting shape after Shape Healing operations.
- The bug in `ShapeFix_IntersectingTool` class has been fixed. Earlier the contour was self-intersecting if areas of parts of the contour were approximately equal to each other.

Visualization

- Two types of mesh visualization in shading mode have been added in `MeshVS`:

To enable the default shading mode you should set `MeshVS_DA_Reflection` flag of `MeshVS_Drawer` to `Standard_True` and `MeshVS_DA_SmoothShading` to `Standard_False`.



To enable smooth shading mode you should set *MeshVS_DA_Reflection* flag of *MeshVS_Drawer* to *Standard_True* and *MeshVS_DA_SmoothShading* to *Standard_True*.

To enable flat shading mode you should set *MeshVS_DA_Reflection* flag of *MeshVS_Drawer* to *Standard_False* and *MeshVS_DA_SmoothShading* to *Standard_False*.

- Highlighting of selected objects has been fixed. Now objects can not be highlighted if they have already been selected.
- Implementation of a new visualization approach based on OpenGL arrays for AIS shapes necessitated adding new arguments in the following methods:

In **Prs3d** package:

- *Prs3d_WFRestrictedFace::Add()* – an out argument retrieving a sequence of face curves.
- *Prs3d_WFDeflectionRestrictedFace::Add()* - an out argument retrieving a sequence of face curves.

In **StdPrs** package:

- *StdPrs_Curve::Add()* - an out sequence of curve points and Boolean which is set to *Standard_True* if it is necessary to display the curve using the old visualization approach.
 - *StdPrs_DeflectionCurve::Add()* - an out sequence of curve points and Boolean which is set to *Standard_True* if it necessary to display the curve using old visualization approach.
 - *StdPrs_WFDeflectionRestrictedFace::Add()* - an out argument which retrieves a sequence of face curves.
- WNT package compatibility with ANSI C++ has been improved to avoid problems with compilation under Visual Studio 8.0.
 - Bug causing exception in MeshVS component (when displaying mesh with nodes having big values of ID) has been fixed
 - The bug in *BRepMesh_FastDiscret* class has been fixed. Earlier faces in shading mode disappeared when an end vertex of some edge erroneously had a point on the curve representation.
 - The bug in wireframe presentation of a shape has been fixed. Earlier isolines were built incorrectly on large shapes (size $\sim 10^6$ and larger): sometimes they could be drawn at a distance from the source faces.

Data Exchange

- The management of warning and fail messages raised during parsing of STEP files has been improved. Earlier non critical fails or warnings were registered once per application life time, after this fix, such messages are registered at each file reading.
- Protection and correct handling of import of toruses with negative radiuses from STEP have been implemented. Such toruses can be transferred from some systems; however, negative radiuses are prohibited by STEP standard.
- The problem with cutting strings with the length exceeding 72 symbols has been corrected.



Draw Test Harness

- DRAW Tcl interpreter has been fixed to correctly process strings containing symbols in extended part of ASCII symbol table.
- The sweeping algorithm has been fixed to avoid creation of invalid sweep for circular paths.

WOK

- Automatic generation of methods `new()` and `delete()` in the classes that inherit from others is now avoided.

Documentation

- Foundation Classes User's Guide has been updated.

Products

OMF

- A lot of methods have been introduced and modified in OMF.

Package SMDS:

- `SMDS_Direction`, implements an equivalent of `gp_Dir`, but twice smaller in size. The components of direction are stored as integers.
- `SMDS_MeshElement`: packs normals to integers and stores them as `SMDS_Direction` values. `gp_Dir` type is used only in the interface (for compatibility). This allows defining normals for some element nodes only, not all at once. Two new methods have appeared:
 - `ClearNormal (Standard_Integer)`
 - `GetNDirection (Standard_Integer)` - takes `SMDS_Direction` without conversion of integers to doubles.
- `SMDS_MeshNode::SetPnt` receives an additional Boolean parameter informing if the normals relevant to that node should be nullified.
- `SMDS_Mesh::RemoveSubMesh()` nullifies the `Parent` field of the removed submesh.
- `SMDS_Mesh` uses `TColStd_PackedMapOfInteger` for the map of nodes (faster than the previous `MapOfInteger` class).
- `SMDS_Mesh::Clear()`, is now called from the destructor. It has an optional parameter which removes all nodes that become free after the removal of mesh elements.
- `SMDS_Mesh::GetParent()` returns the parent mesh or a null handle if none.
- `SMDS_Mesh::HasInverseConnections` finds out if inverse connections exist, without rebuilding them.
- `SMDSMesh::GetElementFromTree()` replaces `FindElement` finding a node or an element in the mesh+children+parents; it is faster than `FindElement()`.

Package SMDSControl :

- `SMDSControl_BoundaryEdges` can create a free boundary in a separate mesh, no more requirements to use the same mesh or a submesh for the result.

Package SMDSAI go:



- SMDSAI go_BoxBndTreeSelector and SMDSAI go_IntersectOnTree unify the architecture of intersection algorithms (Mesh + some entity), using AABB-tree (NCollection_UBTree) and the pre-selection of intersection zone (as a bounding box defined by the caller).
- SMDSAI go_LineIntersect - New Mesh-Line/Ray intersection class
- Distances in SMDSAI go::ProjectPointOnMesh() are now calculated correctly.

Package SMDSEdit:

- Creation of non-convex quadrangles is now avoided.

Package SMDSMeshVS:

- SMDSMeshVS_VisibleFilter, implements SelectMgr_Filter interface for selection of visible mesh entities.
 - SMDSMeshVS_DataSource supports Smooth and Flat shading modes. The smooth shading is maintained by the array of internal normal directions (class SMDSMeshVS_DirectionsBuffer). The predefined normals on mesh faces are passed to the smooth shading drawer; otherwise (if indefinite) these normals are locally calculated and stored in the DataSource.
 - SMDSMeshVS_Mesh adapts to the mesh hierarchical structure: method GetActiveMeshId permits to determine if the current mesh (0) or one of its submeshes (>0) or the current mesh with all its submeshes (-1) are displayed by this interactive object.
- It has become possible to create custom element types without subclass SMDS_Mesh. For example, now we can create a custom Edge element:

```
class MyEdge : public SMDS_MeshEdge { ... /* must redefine Copy() */
... }
Handle(MyEdge) newEdge = new MyEdge (idNode1, idNode2);
myMesh->AddElement (newEdge);
```

Earlier such added custom element would have been incomplete, having a void internal back reference to myMesh.

Parasolid

- Handling of Parasolid XT blended edges has been improved. The resulting algorithm performs computation of blended edges close to Parasolid original.
- Reading of colors from Parasolid XT format has been fixed (it failed to work when *color* was not the first attribute attached to an entity).
- Handling of Parasolid XT schema 17 has been completed. Reading of ESC sequences and entity 99 have been implemented.

Express Mesh

- The effect of AutoSize option has been extended to faces (see QMShape_Tessellator class). Now the algorithm always reproduces small features of the shape in the output mesh to the extent defined by the option AutoSize. Earlier, the form of a cylinder was lost degenerating to plane if the radius of the cylinder was less than the *MinSize* parameter.

Generation of numerous excess triangles along the borders between surfaces of high and low curvature is now avoided.



Changes

Foundation Classes

- Implementation of exceptions and signals handling on Linux and UNIX platforms has been changed. The existing code using try {} statements should be updated (see below).

Now exceptions are raised and handled as normal C++ exceptions, not emulated by C longjumps as before. This provides the following advantages over previous versions:

- Usage of OCC library in the program that uses C++ exceptions does not require special workarounds to provide consistent exception handling
- Generic catch(. . .) with period can be used without conflicting with OCC code (as it was with some implementations of STL)
- Exception raised after catch() {} in the same code block will not be erroneously caught by this catch. Two consecutive try{} catch{} blocks are possible within the same code block.
- C++ stack unwinds with appropriate destruction of objects allocated in the stack.

However, handling of signals is still performed with longjump functions (on UNIX and Linux systems). The macro OCC_CATCH_SIGNALS should be inserted in the code to be able to catch a signal as an exception. This macro instantiates a signal handling object; in case of a software signal the execution will return to this point and an appropriate exception will be raised from this macro. It is recommended to insert OCC_CATCH_SIGNALS macro as the first statement in every try {} block containing code which may generate signal (for instance, generic try {} blocks catching Standard_Failure).

See *Foundation Classes User's Guide* for more details.

Note that this change is controlled by the new compiler option OCC_CONVERT_SIGNALS. The old option NO_CXX_EXCEPTIONS still remains for compatibility; it still can be used to provide the same behavior as in previous versions of OCCT.

- Static global map of types supported previously by Standard_Type class is eliminated. In applications that load and unload some OCCT libraries dynamically, this allows to prevent crashes caused by type objects created in these libraries and remaining in the static map even after unloading of the relevant library.

As a consequence of this change, static methods of that class providing access to that map (Find(), Exists(), Add(), NumberOfKnownTypes(), GiveKnownTypeNumber(), Display_Types()) have been removed.

To allow checking of object type by its name rather than by Handle(Standard_Type) object, new methods Standard_Type::SubType() and Standard_Transient::IsKind() have been provided. These methods replicate the existing methods accepting Handle(Standard_Type) but accept the type name as Standard_CString.

Visualization

- Interface of MeshVS_NodalColorPrsBuilder has been extended with new methods for building presentations using color scale interpolation. It is built by means of texture mapping. It is necessary to perform the following steps to specify parameters of interpolation:
 - Set the required type of interpolation with UseTexture method call (RGB interpolation is used by default for compatibility with the previous version)
 - Specify colors for texture generation with SetColorMap method call



- Specify correspondence between node identifiers and corresponding texture coordinates (range [0, 1]) with `SetTextureCoords` method call
- The following problems have been corrected in the framework of industrialization of texture mapping (in particular, environment mapping) in OCCT 3D viewer:
 - disappearance of environment texture after opening a new document;
 - exception raised at application closing;
 - incorrect display of texture-mapped fonts;
 - ineffective memory usage (static variables defined in `AI_S_InteractiveContext_1.cxx`);
 - reflection of the environment texture from polylines with environment mapping turned on.
- The algorithm of the method `Prs3d_WFShape::Add` has been modified to avoid putting unnecessary line contexts in the group by calling `SetPrimitivesAspect()` when no primitives are actually added to the group. These redundant line attributes might affect the performance of rendering.
- The type of `myPrimitives` field of `Graphic2d` object has been changed from `Graphic2d_SequenceOfPrimitives` to `TColStd_IndexedMapOfTransient` in order to improve the performance of `SetIndex` method. The performance of `Pick` operation has been improved.
- `Graphic3d_AspectText3d::Values()` method has been improved to set `<ATextureMappedFont>` output argument correctly.

Data Exchange

- STEP translator now reads additional shapes attached to the main shape of the PRODUCT with `SHAPE_REPRESENTATION_RELATIONSHIP` entities regardless of the contents of the main shape (previously they were read only when the main shape was empty).

Though it was correct according to AP203 and AP209 (mixed model representation), this could lead to reading unwanted entities (such as auxiliary lines written in this way by some CAD systems). Use parameter `read.step.shape.relationship` to manage this situation.

Products

OMF

- The classes `MeshTools_NASParser` and `MeshTools_NASBaseReader` have been renamed to `OMFTools_NASParser` and `OMFTools_NASBaseReader` correspondingly.

Bug Fixes



- Since last minor release (version 6.1) Open CASCADE 6.1.1 incorporates **84** modifications (bug fixes, enhancements and other corrections). For details, refer to [Appendix](#).



Appendix: Open CASCADE 6.1.1 Bug Fixes

- [Foundation Classes](#)
- [Modeling Algorithms](#)
- [Visualization](#)
- [Data Exchange](#)
- [Draw Test Harness](#)
- [WOK](#)
- [Documentation](#)

Products

- [DXF](#)
- [OMF](#)
- [Parasolid](#)
- [Surfaces from Scattered Points](#)
- [Express Mesh](#)

Foundation Classes, 9 bug fixes	
ID	Short Description
9755	Incorrect behavior of try - catch macro
10010	Test behavior of Units package and add support of percent (%) as unit
11664	New optimized bounding box types
12131	Improvement of Exception mechanism on Unix and Linux platforms
12186	Basic multithread safety in OCCT Kernel
13051	Using nonexistent objects in Standard_ForMapOfTypes class
13131	Optimize code of OCC classes generated by WOK
13151	Provide complete version number of OCCT as hex
13189	Optimize OCC maps for work with big amounts of data
Modeling Algorithms, 33 bug fixes	
ID	Short Description
10086	Query SAMT-05-002
10392	Extrema_ExtPC class does not initialize its fields in constructor.
11081	BRepExtrema_DistShapeShape misses one of two solutions.
11789	There is a severe memory leak in TopExp_Explorer methods Init and Clear.
12203	Non manifold sewing generates invalid shape.
12257	Fuse operation fails.
12487	Boolean operations fail on some compounds.
12522	Extrema problems. Case: one of the Extrema arguments is an infinite face or an infinite edge.
12572	Extrema problem on edges based on curves with continuity C0.
12627	Classification of a point comparing to a face is incorrect.



12635	Bugs in projector algorithm
12661	Wrong calculation of bnd box for edge if edge has polygon of triangulation.
12805	It is necessary to add the new method IntTools_FClass2d::IsHole()
12831	Extrema_ExtPC has uninitialized fields.
12832	math_NewtonFunctionRoot has uninitialized fields (X for example)
12833	Uses uninitialized fields in ProjLib_Fuction class
12836	Method to get rotation angle from gp_Trsf2d
12840	Uninitialized fields in IntImp_ZerCSParFunc
12849	Blend_Extremity has uninitialized fields.
12850	class TopOpeBRepDS_Curve has uninitialized fields
12851	class Geom2dHatch_Intersection has uninitialized fields
12884	Wrong result of projection curve on surface
12888	Wrong result of projection curve on surface
12918	Boolean Operations failed
12919	Add new methods BOPTools_SSInterference::SetSharedEdges()/SharedEdges()
13081	It is necessary to have some modifications in IntTools_FClass2d.cxx
13115	Modification of the Shape Healing operations to keep non-manifold topology.
13116	Boolean Operations produce faulty shape.
13149	Following of exception improvement
13186	Problem with Boolean operation
13209	Exception is raised while performing Boolean operations
13267	Summary: FlxShape works incorrectly on small contours
13354	Exception in Same Parameter
Visualization, 16 bug fixes	
ID	Short Description
9833	Modify MeshVS to support smooth and flat shading
9915	Provide means in MeshVS to interpolate color along element according to color scale.
11770	Selected sub-objects can be highlighted by mouse in 3D Viewer only one time.
11804	Regression concerning to incorrect definition of a bounding box (minimal and maximal values) for markers in 3D view.
11904	Problems with texture mapping.
12146	Package WNT: Compilation problems under Visual Studio 8.0.
12297	Migration to new visualization approach based on OpenGL arrays
12327	Prs3d_WFShape::Add creates unnecessary line contexts in the group
12476	Improving highlighting performance of large scale model
12531	V2d_View::ScreenPostScriptOutput method produce incorrect .ps file.
12844	Regression in Trihedron
12977	Creating a second 3D view after closing the first one crashes an application on ATI Radeon cards
13100	Crash in MeshVS with large meshes
13142	Face is not displayed in Shading mode after gluing Operation
13144	Incorrect isolines building after Cut operation
13439	The text in ColorScale is not displayed on Linux.
Data Exchange, 5 bug fixes	
ID	Short Description



11856	Error in reading of step file.
11857	Step toruses with negative radiuses did not translated
12995	Avoid using cout in STEP and IGES translators
13105	Revising of parameters of STEP translator
13200	Impossible of correct import/export sequence of operations.
Draw Test Harness, 2 bug fixes	
ID	Short Description
63	It's impossible to open files containing localization characters in the name.
12213	Invalid sweep was created by command "buildsweep" in DRAW in case circular path.
WOK, 2 bug fixes	
ID	Short Description
10033	WOK: Avoid defining new() and delete() for classes derived from other.
12619	The extraction on Windows fails.
Documentation, 1 bug fix	
ID	Short Description
12994	Undocumented parameters in STEP and IGES translators.

Product Bug Fixes

The following bug fixes have been performed for Open CASCADE specific development customers.

DXF, 1 bug fix	
ID	Short Description
12980	Option to avoid reading data from anonymous blocks (HATCH, DIMENSION, and other generated data).
OMF, 9 bug fixes	
ID	Short Description
7638	Nastran files are imported incorrectly (see attached picture).
9115	Introduce smooth shading mode.
9665	Add possibility to select only visible mesh entities.
9832	Modify OMF sample so as to use MeshVS_Mesh presentation instead of SMDSInter and modify SMDS and SMDSMeshVS packages.
12251	Various improvements in OMF.
12481	Addition of elements with same ID destroys the integrity of SMDS_Mesh.
13102	SmoothShading mode crashes when the element IDs form a sparse array.
13214	Back pointer to mesh is not set in SMDS_Mesh::AddElementWithID.
13252	Rename package MeshTools to OMFTools.
Parasolid, 3 bug fixes	



ID	Short Description
12389	Problems with translation of BlendedEdge
13042	Colors are not read from XT file
13353	Fails in reading of Parasolid XT files
Surfaces from Scattered Points, 2 bug fixes	
ID	Short Description
13129	Modification to keep non-manifold topology.
13440	Implementation of Surface Modification and Surface Curvature Analysis algorithms.
Express Mesh, 1 bug fix	
ID	Short Description
13193	Implement AutoSize feature for discretizing interior of face by analogy with edge.

