

Open CASCADE 5.1.2 Maintenance release

Release Notes

Overview

Open CASCADE Maintenance release 5.1.2 includes bug fixes, new features and improvements over the previous maintenance release 5.1.1.

Due to API changes, this Version 5.1.2 is binary incompatible with the previous Version 5.1.1, so applications linked against the latter must be recompiled to run with 5.1.2.

Highlights

- ▶ **OPEN CASCADE NOW SUPPORTS WINDOWS XP!**
- ▶ **STL STREAMS SUPPORT HAS BEEN SIGNIFICANTLY IMPROVED IN ORDER TO ELIMINATE THE PROBLEM OF COMPILATION WARNINGS. OPEN CASCADE NOW USES COMPILER'S IMPLEMENTATION AS PREFERRED STL IMPLEMENTATION.**
- ▶ **DOCUMENTATION ON THE PIPE AND OFFSET ALGORITHMS HAS BEEN EXTENDED. JAVA EXTRACTOR DOCUMENTATION HAS BEEN ADDED.**
- ▶ **AN IMPORTANT IMPROVEMENT IN VISUALIZATION: NOW A LARGE NUMBER OF OBJECTS (UP TO THOUSANDS) CAN BE HIGHLIGHTED IN THE 3D VIEWER.**



Table of Contents

- **[New Features](#)**
 - ⇒ [Data Exchange](#)
 - ⇒ [Visualization](#)

- **[Improvements](#)**
 - ⇒ [Technical Documentation](#)
 - ⇒ [Application Framework](#)
 - ⇒ [Data Exchange](#)
 - ⇒ [Foundation Classes](#)
 - ⇒ [Visualization](#)
 - ⇒ [WOK](#)

- **[Changes](#)**
 - ⇒ [General Changes](#)
 - ⇒ [Application Framework](#)
 - ⇒ [Modeling Algorithms](#)
 - ⇒ [Visualization](#)

- **[Bug Fixes](#)**



New Features

Data exchange

- Timers have been implemented that are used for measuring the performance of a current operation or any part of code, and provide the necessary API. Timers are used for debugging and performance optimizing purposes. See Open CASCADE Reference Documentation (MoniTool Package) and Shape Healing User's Guide for more details.
- A new method: `ShapeFix_Face::FixIntersectingWires()` has been created. It detects and fixes the specific case when a face has more than one wire and these wires have an intersection point. In this case a common vertex is assigned (either new or an existing one) for the intersection of wires in the point of intersection.

Visualization

- In order to increase performance when working with large selections the previous approach of highlighting selected entity owners has been extended to support a smarter behavior. New methods for highlighting selected and detected entity owners in the 3D viewer have been introduced:
 - `AIS_LocalContext::UpdateSelected(const Handle(AIS_InteractiveObject)&, Standard_Boolean)`
 - `SelectMgr_EntityOwner::IsAutoHighlight()`

`SelectMgr_SelectableObject` class.

Methods:

- `HighlightSelected()`,
- `ClearSelected()`,
- `HighlightOwnerWithColor()`,
- `IsAutoHighlight()`,
- `SetAutoHighlight()` have been added.

An optional boolean argument has been added to the `ClearSelections()` method.

Please, see Reference Documentation for full details.

Note:

The traditional way of highlighting selected entity owners adopted by the Open CASCADE library assumes that each entity owner highlights itself on its own. This approach has two drawbacks:

- each entity owner has to maintain its own `Prs3d_Presentation` object, that results in large memory overhead for thousands of owners;
- drawing selected owners one by one is not efficient from the OpenGL usage viewpoint.



That is why a different method has been introduced. On the basis of `SelectMgr_EntityOwner::IsAutoHighlight()` return value an `AIS_LocalContext` object either uses the traditional way of highlighting (`IsAutoHighlight()` returned true) or groups such owners according to their `Selectable Objects` and finally calls `SelectMgr_SelectableObject::HighlightSelected()` or `ClearSelected()`, passing a group of owners as an argument.

Hence, an application can derive its own interactive object and redefine `HighlightSelected()`, `ClearSelected()` and `HighlightOwnerwithColor()` virtual methods to take advantage of such OpenGL technique as arrays of primitives. In any case, these methods should at least have empty implementation.

The `AIS_LocalContext::UpdateSelected(const Handle(AIS_InteractiveObject)&, Standard_Boolean)` method can be used for efficient redrawing a selection presentation for a given interactive object from an application code.

Additionally, the `SelectMgr_SelectableObject::ClearSelections()` method now accepts an optional boolean argument. This parameter defines whether all object selections should be flagged for further update or not. This improved method can be used to re-compute an object selection (without redisplaying the object completely) when some selection mode is activated not for the first time.



Improvements

✓ Technical documentation

- WOK User's Guide has been extended to describe the use of Java extractor. Refer to the User's Guide for full details.
- Reference Documentation has been updated, including, the description of Pipes and Offsets (BRepOffsetAPI_MakeOffsetShape and BRepOffsetAPI_MakeThickSolid).

✓ OCAF

- An improvement has been made which is internal to package TDF and class TDF_AttributeIterator. The type of the myValue field of the TDF_AttributeIterator class has been changed from a smart pointer Handle(TDF_Attribute) to a standard pointer TDF_Attribute*. This improvement allows to gain about 25% performance increase especially when working with extensive OCAF documents.

1. Method TDF_AttributeIterator::Value now returns (TDF_Attribute *), previously it returned Handle(TDF_Attribute). This is done to avoid creation/copying of Handle by this method. Generally this change does not require modification of the existing application code (pointer is implicitly casted to Handle).
2. The AttributeIterator object no longer holds the current OCAF attribute as a possessor of a reference to it (via the mechanism of Handle). This means that the attribute may be destroyed when the Iterator is still active, and the call of the method Next() would cause unpredictable results.

In practice, this can only happen when the method TDF_Label::ForgetAttribute is called inside the loop managed by a corresponding AttributeIterator instance. To avoid such problem you should always call Next() BEFORE any possible calling of TDF_Label::ForgetAttribute, like in method TDF_Label::ForgetAllAttributes:

```
TDF_AttributeIterator itr1 (myLabelNode);
// OCC5031: iterator must be incremented before removal of the
attribute
while (itr1.More()) {
    const Handle(TDF_Attribute) anAttr = itr1.Value();
    itr1.Next();
    ForgetFromNode (myLabelNode, anAttr);
}
// Previous code (dangerous):
// while (itr1.More()) {
//     ForgetFromNode(myLabelNode,itr1.Value());
//     itr1.Next();
// }
```



- **TDataStd_IntegerArray and TDataStd_RealArray TDataStd_ExtStrinArrayArray classes**

Extra backups for an attribute have been removed if Value is not really changed. Backup is performed only if:

- dimension of <NewArray> differs from that of the old Array (<myValue>).
- at least one element of <newArray> differs from a correspondent element of <myValue>.

If the Backup is performed, a new instance of HArray1Of... (HArray1OfInteger or HArray1OfReal) is created and this instance consequently handles <myValue> if current and backed up attributes are different.

Data Exchange

- A fix has been added for faces having wires composed of several closed loops of edges. This fix splits such wires into closed loops, then wires are created from such loops. The algorithm tries to build a correct Open CASCADE face consisting of one external and several internal wires (holes). Otherwise a set of separate faces is created. The resulting shapes have a simpler topology and are easier to process by Open CASCADE algorithms and algorithms of other systems.

Foundation Classes

- **OSD package**
A problem has been fixed when due to a different system API on Win9x, method SystemTimeToTzSpecificLocalTime did not perform any time conversion, it just left all fields of the output parameter structure non-initialized, leading to application crash.

Visualization

- Performance of selection highlighting in 3D has been improved by introducing a standard sorting algorithm SortTools_QuickSort instead of a very rough algorithm used previously. The effect should be visible when there are many (up to thousands) entities under the mouse cursor.
- **AIS package**
Some methods now erase all object presentations for display modes different from the required one (passed as an argument to SetDisplayMode() or contained in the <myDisplayMode> field of an interactive object, in case of Display ()) and then make sure that the object presentation for the given mode is displayed and may be highlighted (according to the current highlighting state of the object). Previously it was impossible to switch to a required display mode in the field of an interactive object before calling SetDisplayMode(), because other presentations remained visible that led to a mess in the 3D viewer. So an application could not use the <myDisplayMode> field of an object to pass the mode information to some custom display mode manipulation logic. Now SetDisplayMode() does not rely on the display mode coming from the object field, so this field can be used safely by applications. Display() can now be applied to an object already displayed (formerly, it did nothing in such situation), to make sure that it is shown in a proper display mode (a convenient shortcut to combination of several method calls). It is necessary just to



call `AIS_InteractiveObject::SetDisplayMode(mode)` and `SetToUpdate()` methods before calling `Display()` to make it work. It should be noted that calling `Display()` for an already displayed object activates the default selection mode for this object (usually 0), as it does for objects not yet displayed.



- **Java extractor**

WOK Java extractor wraps Open CASCADE C++ classes into Java classes using JNI (Java Native Interface) calls. It has undergone several corrections and improvements:

- Processing of the `Standard_Transient` subclasses (i.e. those manipulated by handle) have been corrected. This has been achieved with introducing a new base Java class `jas.Transient` (in the `jas.nocdpack`), which extends `jas.Object`. The latter remains a base class for all classes manipulated by value as well as for all `Standard_Storable` subclasses.
 - Redundant methods `IsKind()` and `DynamicType()` of former `jas.Object` have been removed. This allows correct extracting of classes of the `Standard` package. However, `Standard_Persistent` subclasses still may not be correctly wrapped but this does not limit the extractor use (since these classes are not supposed to be used directly in Java).
 - The extractor now correctly extracts classes without a default constructor. This constructor is no longer required for the classes to be wrapped to Java. If a null C++ object is passed to Java (what is normally impossible) then an exception is thrown (instead of the previous attempt to call a default constructor).
 - Processing of large interface units declared in the `jni` unit has been improved. A command line with the list of java files passed to the java compiler now is verified against overflow and is split if necessary.
 - Dependency chain between generated `.java`, `.class`, `.h` and `.cxx` files has been created. Thus, if some classes disappear from the extraction scope (e.g. excluded from the interface unit) their respective generated files will be removed.
 - Generated header `.h` files are now put into the `drv` directory of the unit instead of common `inc` directory of the workbench. This facilitates packaging and keeps the include directory cleaner.
- Various minor Windows-specific improvements:
 - The `WOKSteps_Remove_File` step now actually removes orphan files (e.g. generated from other source files, which have been removed).
 - The `%CMPLRS_CXX_DBMSOpt` parameter is now always used in compilation.
 - Correct start of the WOK session when the `%WOK_ROOTADMDIR%` variable points to a drive different from Open CASCADE installation drive.



Changes

✓ System Requirements

- Starting from Open CASCADE 5.1.2, Windows XP has been included on the list of supported platforms.

✓ Support of STL streams

- Starting from Open CASCADE 5.1.2 binaries delivered with Open CASCADE are built using STL implementation by the supported compilers. The source code can be rebuilt using other implementations (for example STLPort). However Open CASCADE S.A. did not undertake any certification actions on such implementations and therefore cannot guarantee a reliable work with other implementations.

✓ Standard MFC samples

- The OCAF document browser has been integrated to the OCAF MFC sample. By default the Qt-based browser is linked with the sample. The OCAF document browser is a GUI library used for navigation over the document created with Open CASCADE Application Framework (OCAF). The library can be loaded to explore the document at run-time thereby significantly simplifying the debugging process to ensure correctness of the data model and validity of internal document state. The browser is supplied with realization on Fltk, Qt and TclTk graphic toolkits.

✓ OCAF

- Exception handling has been corrected in class `TDocStd_Application` that provides methods to store/retrieve documents.
- Implementation of methods: `Init`, `SetValue`, `Paste` and `ChangeArray` of classes `TDataStd_IntegerArray` `TDataStd_RealArray` `TDataStd_ExtStrinArrayArray` has been changed in order to eliminate extra backups for an attribute in case when `Value` is not really changed. (see the corresponding improvement).

Warning:

Earlier the `ChangeArray()` method had the following non-documented behavior: a handle given to the method `ChangeArray()` was set as an actual attribute data. Therefore it was possible to modify attribute data via this handle after setting a new value by means of the `ChangeArray()` method.

For instance, the following code could have been used to change an array attribute and fill it with new data:

```
Handle(TDataStd_IntegerArray) anAttr = ...;
Handle(TColStd_HArray1ofInteger) anArr = ...;

anAttr->ChangeArray ( anArr );

for ( int i=anArr->Lower(); i <= anArr->Upper(); i++ )
    anArr->SetValue ( /* some code */ );
```



Now this code may fail to work because there is no guarantee that anArr given as an argument to ChangeArray() will be the same handle as stored in the attribute.

Therefore all code of this kind in the user applications should be identified and corrected in order to ensure proper work. Either:

- method SetValue() of an attribute should be used,
- or method ChangeArray() should be called after array data are completely prepared

Modeling Algorithms

- **BRepBndLib** package
Method BRepBndLib::Add() has been changed. See Reference Documentation for more details.
- **IntWalk_Pwalking** class
The intersection algorithm has been modified to extend an intersection line in the tangent zone.
A new sub-algorithm has been added to the Boolean operations algorithm. In order to correctly process some complex cases of source shapes mutual disposition, this new sub-algorithm sews intersection lines and parts of split edges (splits) using their intersection points.

Visualization

- TriangleMeshAdd(TSM_ELEM_DATA d, Tint n, cmn_key *k) function in the OpenGL_tmesh.c file has been corrected: special precautions are now taken if the number of faces to be displayed (data->num_facets value) is 1. This case is treated separately, so as not to run out of the array of face normals while computing vertex normals.
- The Draw_array() function (OpenGL_PrimitiveArray.c) has been improved, so that now uniformly shaded polygons can be safely drawn after the Graphic3d_ArrayOfPolygons containing colored vertices has been drawn, since proper material attributes are restored due to a call to TelResetMaterial() immediately after the GL_COLOR_MATERIAL mode is disabled.



Bug Fixes



- Open CASCADE 5.1.2 incorporates **51** modifications (bug fixes, enhancements and other corrections) over the previous maintenance version 5.1.1. For details, refer to [Appendix 1](#).



Appendix 1: Open CASCADE 5.1.2 Bug Fixes

- [Data Exchange](#)
- [Foundation Classes](#)
- [Modeling Algorithms](#)
- [Application Framework](#)
- [Visualization](#)
- [Shape Healing](#)
- [WOK](#)

Data Exchange, 18 bug fixes	
ID	Short Description
210	Improved FixShape to correct the case of touching wires
1080	bug in step writer
3142	Case igs/005/A2 can not always finish. Regress.
3147	Problems of reading an IGES file.
3397	Incorrect work of ShapeFix_ComposeShell
3430	Incorrect creation of pcurves
3925	Exception during reading a file using XDEDRAWEXE
3962	Cannot read an stp file
4142	Regression on reading e3i files
4503	Integration of perf meter
4504	Integration of perf meter
4562	Problems with compilation the SMDS package
4648	Problems with writing to STEP
4792	Shape becomes invalid after Canonical Recognition (no shading, no face)
4819	Integration changes for optimization of XDE reading
4822	Translation surfaces of revolution and extrusion instead of canonical
4968	ShapeHealing can not correct invalid self-intersecting wire
5027	Incorrect result translating a wire with loops from STEP
Foundation Classes, 3 bug fixes	
ID	Short Description
4417	OSD_Error incorrectly treats some Windows error codes
5091	NCollection classes should raise exceptions according to OCC standard
5092	OSD_FileNode: methods AccessMoment and CreationMoment fail on Win9x
Modeling Algorithms, 13 bug fixes	
ID	Short Description
311	Hung up and incorrect result in Segment (2d and 3d)
496	Exception occurs during fuse operation
620	No possibility to cut the shapes.
1764	Unstable work of checkshape for detection of cases of a self-intersection wire
3565	Dynamic loading of Draw Commands
3727	Problem of Geom2dAPI_InterCurveCurve
4315	Invalid result of intersection of 2d curves
4315	Invalid result of intersection of 2d curves
4426	Incorrect result of intersection in 2D between a circle and line





4455	Invalid result or exception in offset algorithm
4993	Problem in boolean fusion
4994	Problem in boolean fusion
5067	Regressions on windows XP. Pipes
5101	Regression on STL strings in MOA.
Application Framework, 8 bug fixes	
ID	Short Description
3713	MultiTransactionManager: bug in treatment of undo limit
3960	Undo does not work after copying an array attribute into another one
4327	Incorrect exception handling in TDocStd_Application
4800	Incorrect processing of Plugin file absence
5023	Performance regression in opening an OCAF file
5031	Improved performance of TDF_AttributeIterator
5052	Saving document to a file: dot sign is attached to a file name.
5053	Implement methods to access document saved time.
5100	Regression on STL strings in OCAF.
Visualization, 5 bug fixes	
ID	Short Description
4201	Performance problem with SelectMgr
4298	Memory beyond array bound read in TriangleMeshAdd() (OpenGL_tmesh.c)
4373	Improper analysis of current display mode in AIS_InteractiveContext methods
4723	Material attributes are wrong after drawing an array of polygons+vertex colors.
5090	Advanced mechanism for highlighting selected/detected entity owners
Shape Healing, 1 bug fix	
ID	Short Description
2851	Regression on performance in dev:ros
WOK, 3 bug fixes	
ID	Short Description
4732	Avoiding of compilation errors (jcas package).
5082	Some corrections and improvements in WOK Java extractor.
5083	Some Windows-specific corrections and improvements in WOK