

# Open CASCADE 5.1.1 Maintenance release

## Release Notes

### Overview

Open CASCADE 5.1.1 is a maintenance release, which includes bug fixes, new features and improvements over the previous minor release 5.1.

Due to certain API changes, Version 5.1.1 is binary incompatible with Version 5.1, so applications linked against the latter must be recompiled to run with 5.1.1

### Table of Contents

- **[New Features](#)**
  - ⇒ [Application Framework](#)
  
- **[Improvements](#)**
  - ⇒ [Data Exchange](#)
  - ⇒ [Foundation Classes](#)
  - ⇒ [Application Framework](#)
  - ⇒ [Visualization](#)
  - ⇒ [Test Harness](#)
  
- **[Changes](#)**
  
- **[Bug Fixes](#)**



## New Features

### Application Framework

- **Redirection of all messages from Persistence** has been implemented. This allows to use the so-called application Message Driver (if defined) for output of all warnings and any other relevant messages. If the Message Driver is defined, all warnings, additional messages will be directed to this Driver. If the Message Driver is not defined, the default `CDM_NullMessageDriver` will be used to process all messages In that case no messages are shown or passed to higher-level routines.

#### **Example:**

This is an example of the driver activation:

```
CDM_MessageDriver aDriver = new CDM_CoutMessageDriver();
myApp->SetMessageDriver (aDriver);
```

The following example (taken from `PCDM_ReadWriter_1.cxx`) shows how to use the already set driver to output custom messages:

```
TCollection_ExtendedString aMsg("warning: ");
aMsg.Cat("could not read the reference counter in
").Cat(aFileName).Cat("\0");
if(!theMsgDriver.IsNull())
theMsgDriver->Write(aMsg.ToExtString());
```

- A new `ExtStringArray` class has been added to the `TDataStd` package. This class is intended to provide the implementation of a new standard attribute to keep the array of Unicode strings within the Data Framework at a predefined label. For more advanced user information, please, consult OpenCASCADE Reference Documentation (chapter Standard Attributes).



## Improvements

### Data Exchange

- Translation of some STEP and IGES files has been sped up. The resulting average acceleration for STEP is about 22% and for IGES it is about 19% over Open CASCADE version 5.0.
- The reading of B-Splines weights during an IGES translation has been improved. Some systems write the values of weights incorrectly (zero or negative values). This is because these systems dump "memory garbage" in case of polynomial B-Splines. Now all such weights for this type of BSplines are set to 1.

### Foundation Classes

- A memory leak has been eliminated in the concatenation function (Cat) of the string types represented by classes AsciiString and ExtendedString from the TCollection package.

- **NCollection package**

The behavior of the Bind method of the DataMap class has been corrected so that it corresponds to its counterpart in the TCollection package. In particular, the method must bind the new value to a given key even if the same key is found in the map.

- Some improvements have been made in the NCollection package to make it more flexible:
  - Methods NCollection\_List::Append() and Prepend() have been modified to return the reference to a newly added item.
  - A new method NCollection\_IncAllocator::Reallocate() has been added. It allows to resize the most recently allocated memory block.
  - Method NCollection\_BaseMap::Iterator::Initialize() has been public. It allows a developer to reuse an already used iterator object once again with a different map object.
  - The compilation warning C4291 ("no matching operator delete found...") of MS Visual Studio has been eliminated.

- **Standard Memory Manager**

It has been optimized to work with memory-checking tools such as Rational Purify. Now the deactivation of the standard optimized mode (through environment variable MMGT\_OPT) makes the memory manager work directly through malloc and free and do not reserve extra space for the memory block header.



## ✓ Application Framework

- **TDocStd\_MultiTransactionManager class**

A possibility has been added to detect whether a just performed transaction changes something in a document and writes this changed information into the undo buffer.

- **LDOM package and XML persistence**

Now a string containing non-ASCII characters is correctly retrieved from an XML file. The operator of converting an object of type `LDOBasicString` to `TCollection_ExtendedString` takes care of such characters.

- **XML persistence**

The memory occupied by a temporary relocation table is freed at the end of the store/retrieve operation.

- The robustness of the **LDOM package** has been increased:

- Some memory leak has been eliminated;
- Reading of memory beyond the allocated space has been prevented, as well as reading of non-initialized memory.

- The Naming mechanism was extended to cover cases when several entities are considered as generators.

## ✓ Visualization

- **Visualization performance**

Now a developer can improve the performance of an application using a new optional boolean parameter of methods `Reset()` and `FitAll()` of the class `V3d_View` controlling whether an update is needed at the end of an operation.

## ✓ Test Harness Draw

Open CASCADE Test Harness has been improved. The improvement allows dynamic loading of a corresponding group of Draw commands using only one version of Draw executable - **DRAWEXE**. All numerous instances of Test Harnesses for Open CASCADE modules (such as TCAF for OCAF, XSDRAWEXE and XDEDRAWEXE for Data Exchange, etc) have been eliminated. This executable is now capable of handling arbitrary commands loaded as plug-ins at run-time.

For details, please refer to [Appendix 2](#).

## Changes

### System Requirements

- **Support of Linux RedHat 8.0 and gcc 3.2**

Open CASCADE has been ported to Linux RedHat 8.0 with support of gcc 3.2. Binaries delivered with Open CASCADE 5.1.1 have been built using this Linux distribution and the compiler.

- **Support of STL streams**

Open CASCADE has been modified to include support of STL (Standard Template Library) streams on Windows and Linux platforms (Sun Solaris keeps using older streams due to their missing support by cc4.2 compiler).

Binaries delivered with Open CASCADE have been built using STL implementation by the open source product STLPort 4.5.3 ([www.stlport.org](http://www.stlport.org)). Nevertheless the source code can be rebuilt using other STL implementations, for example provided by a compiler vendor.

To enable support of STL streams using STLPort when rebuilding Open CASCADE from sources the following should be done:

- First, download, install and build the STLPort.
- On Windows: when using MS Visual Studio set directories in the "Tools/Options" menu to the STLPort directory prior to the compiler directories.
- On Unix: when using GNU autoconf/automake tools specify options beginning with the `--with-stlport` prefix (e.g. `--with-stlport-include=/products/stlport-4.5.3/stlport,` `--with-stlport-library=/products/stlport-4.5.3/lib,` `--with-stlport-libname=stlport_gcc`).
- On Unix or Windows: when using WOK redefine the `%STLPortHome` and all its derivative variables in the `CSF.ed1` file (located in the wok directory).

#### **Notes:**

- For STL Installation Instructions see the Open CASCADE Reference Documentation (Chapter Building Modules).
- If you prefer not to use the STLPort in your project then you will have to rebuild Open CASCADE with the STL implementation you use.

- **Support of MS Visual Studio C++ 6.0 with Service Pack 5**

For most correct support of STL streams the MS Visual Studio compiler must be upgraded to SP5 (downloadable from [www.microsoft.com](http://www.microsoft.com)).



 **Installation**

- Starting from Open CASCADE 5.1.1:
  - debug versions of libraries (dll and lib) and executables (exe) for Windows are no longer delivered;
  - dynamic-link libraries (dll) and executables (exe) for Windows are located in ros/win32/bin (this will allow to shorten the Open CASCADE directory list added into the PATH system variable);

 **Building Tools**

- Dynamic-link libraries (dll) and executable (exe) files built by MS Visual Studio projects on the Windows platform, are now placed to the ros/win32/bin folder and replace their corresponding original libraries and executable files, respectively binaries for the debug mode are placed in the ros/win32/bind folder.
- New options for the GNU autoconf/automake tools to enable STLPort use (see "Support of STL streams" above).

 **Visualization**

- **V3d\_View class methods Reset and FitAll**

A new optional parameter of type `standard_Boolean` has been added

 **Test Harness**

- Using Draw Test Harness in the form of single executable (DRAWEXE) with extensible command libraries (dynamically loaded as plugins) has been implemented. Old executables (AISViewer, TTOPOLOGY and etc.) have been deleted.

See [Appendix 2](#) for details on Test Harness changes.

## **Bug Fixes**



- Open CASCADE 5.1.1 incorporates **42** modifications (bug fixes, enhancements and other corrections) over version 5.1. For details, refer to [Appendix 1](#).



## Appendix 1: Open CASCADE 5.1.1 Bug Fixes

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

<b>Data Exchange, 4 bug fixes</b>	
ID	Short Description
2821	IGES file is not imported correctly
3004	Regressions after improvement performance
3926	Exception during reading a file using XDEDRAWEXE
4082	Problems during translation of IGES files
<b>Foundation Classes, 4 bug fixes</b>	
ID	Short Description
3277	Memory leak in TCollection strings
4040	The method NCollection_DataMap::Bind must change the value anyway
4146	Various improvements in NCollection classes
4148	The work of Standard allocator when MGMT_OPT=0 has been improved
<b>Modeling Algorithms, 23 bug fixes</b>	
ID	Short Description
435	Exception Standard_ConstructionError is raised in the GeomConvert_CompCurveToBSplineCurve
479	GeomPlate_Surface::Uiso and GeomPlate_Surface::Viso return a NULL Geom_Curve
519	Draw hangs up in BRepAlgoAPI_Cut function during loading the e3i-file under pkv-GalleryTest workbench
523	Draw "hangs up" in BRepAlgoAPI_Cut function during loading the e3i-file under pkv-GalleryTest workbench
528	Result of bcut command is faulty although arguments are valid
583	Result of bopcut operation is wrong
584	Result of cut operation is wrong
600	Result of BOPCOMMON operation is a non-closed shape in spite of the fact that source solids are valid
605	No faces in result of the PIPE command
629	Exception when attempting to create a solid by command PIPE
763	Bad result of fuse operation between two cylinders
1077	A bug in boolean operations
1477	Problems in MakePipeShell
2500	BRepAlgoAPI_Section has a problem with calculation
2755	Boolean Bug in OCC5.0 with the shape having an ellipse curve





2991	BrepMesh_IncrementalMesh takes forever to mesh some faces
3403	A little bug in XCAFDoc_DocumentTool.cxx
3502	Incorrect work of command sprops
3565	Dynamic loading of Draw Commands
3643	Moving the tool during the cut operation fails
3644	Moving a component of the intersection operation fails
3645	Moving a component of the intersection operation fails
3918	Exception during retrieval
<b>Modeling Data, 1 bug fix</b>	
<b>ID</b>	<b>Short Description</b>
3721	When using Boolean Operations an exception is raised for some tolerance values
<b>Application Framework, 8 bug fixes</b>	
<b>ID</b>	<b>Short Description</b>
2933	In Std and XML plugins, and in PCDM messaging to cout has been removed using the CDM_MessageDriver
3427	New ExtStringArray attribute - improvement
3548	Multi Trans. Manager: need to know if just committed transaction had changes
3641	The result of moving during the cut operation fails
3646	Moving a component during the intersection operation fails
3985	XML persistent incorrect process (write/read) unicode strings
4149	Clear relocation table at the end of read/write a document using XML persistence
4150	Some improvements in the LDOM package
<b>Visualization, 2 bug fixes</b>	
<b>ID</b>	<b>Short Description</b>
230	Numeric Error occurs in V2d_View::WindowFit() , Magnify()
4147	A possibility of conditional update has been added in methods FitAll and Reset of V3d_View

## Appendix 2: Test Harness Draw (changes)

Since version 5.1.1 Open CASCADE introduces a single executable in the DRAW Test Harness that supersedes the several separate executables that existed before. Respectively the user does not need to have his own executables to activate his custom commands. All he needs to do is to implement the commands themselves, they will be activated in the common executable. This executable is now called **DRAWEXE**.

Commands grouped in toolkits can now be loaded at run-time thereby implementing dynamically loaded plug-ins. Thus, the user can work only with those commands that suit his needs adding these commands dynamically without leaving the Test Harness session.

Declaration of available plug-ins is done through the special resource file(s). The **pload** command loads the plug-in in accordance with the specified resource file and activates the commands implemented in the plug-in.

The whole process of using new advantages of the plug-in mechanism as well as instructions for extending Test Harness are described below.

### **Launching DRAW Test Harness**

Test Harness executable DRAWEXE is located in the `$CASROOT/<platform>/bin` directory (where `<platform>` is win32 for Windows, SunOS for Sun Solaris and Linux for Linux operating systems). Prior to launch it is important to make sure the environment is correctly set-up (usually this is done automatically after the installation process on Windows or after launching specific scripts on Unix/Linux) - refer to Technical Documentation for details.

### **Plug-in resource file**

Open CASCADE is shipped with the DrawPlugin resource file located in the `$CASROOT/src/DrawResources` directory.

The format of the file is compliant with standard Open CASCADE resource files (see the Resource\_Manager.cdl file for details).

Each key defines a sequence of either further (nested) keys or a name of the dynamic library. Keys can be nested down to an arbitrary level. However, cyclic dependencies between the keys are not checked.

**Example** (excerpt from DrawPlugin):

```
OCAF                : VISUALIZATION, OCAFKERNEL
VISUALIZATION       : AISV
OCAFKERNEL          : DCAF

DCAF                : TKDCAF
AISV                : TKViewerTest
```





**Activation of commands implemented in the plug-in**

To load a plug-in declared in the resource file and to activate the commands the following command must be used in Test Harness:

`pload [-PluginFileName] [[Key1] [Key2]...]`, where:

- `<-PluginFileName>` Defines the name of a plug-in resource file (prefix "-" is mandatory) described above. If this parameter is omitted then the default name DrawPlugin is used.
- `<Key>...` Defines the key(s) enumerating plug-ins to be loaded. If no keys are specified then the key named DEFAULT is used (if there is no such key in the file then no plug-ins are loaded).

According to the Open CASCADE resource file management rules, to access the resource file the environment variable `CSF_<PluginFileName>Defaults` (and optionally `CSF_<PluginFileName>UserDefaults`) must be set and point to the directory storing the resource file. If it is omitted then the plug-in resource file will be searched in the `$CASROOT/src/DrawResources` directory.

**Examples:**

`Draw[]>` `pload -DrawPlugin OCAF`  
 Will search the resource file DrawPlugin using variable `CSF_DrawPluginDefaults` (and `CSF_DrawPluginUserDefaults`) and will start with the OCAF key. Since the DrawPlugin is the file shipped with Open CASCADE it will be found in the `$CASROOT/src/DrawResources` directory (unless this location is redefined by user's variables). The OCAF key will be recursively extracted into two toolkits/plug-ins: TKDCAF and TKViewerTest (e.g. on Windows they correspond to TKDCAF.dll and TKViewerTest.dll). Thus, commands implemented for Visualization and OCAF will be loaded and activated in Test Harness.

`Draw[]>` `pload` (equivalent to `pload -DrawPlugin DEFAULT`).  
 Will find the default DrawPlugin file and the DEFAULT key. The latter finally maps to the TKTopTest toolkit which implements basic modeling commands.

**Mapping between former separate Test Harness executables and the new plug-ins**

Before version 5.1.1 Open CASCADE used to be shipped with several separate executables providing different sets of commands. The following table represents the mapping between former executables and new plug-ins.

Former executable	Current key
AISViewer	VISUALIZATION
TCAF	OCAF
TTOPOLOGY	MODELING
XDEDRAWEXE	DATAEXCHANGE
XSDRAWEXE	DATAEXCHANGEKERNEL



For instance, in order to activate commands available in the former AISViewer executable, now it is enough to use the command `pload VISUALIZATION`.

### ***Extending Test Harness with custom commands***

The following chapters explain how to extend Test Harness with custom commands and how to activate them using a plug-in mechanism.

#### ***Custom command implementation***

Custom command implementation has not undergone any changes since the introduction of the plug-in mechanism. The syntax of every command should still be like in the following example.

**Example:**

```
static Standard_Integer myadvcurve(Draw_Interpreter& di,
                                   Standard_Integer n,
                                   char** a)
{
  ...
}
```

For examples of existing commands refer to Open CASCADE (e.g. GeomliteTest.cxx).

#### ***Registration of commands in Test Harness***

To become available in the Test Harness the custom command must be registered in it. This should be done as follows.

**Example:**

```
void MyPack::CurveCommands(Draw_Interpreter& theCommands)
{
  ...
  char* g = "Advanced curves creation";

  theCommands.Add ("myadvcurve", "myadvcurve name p1 p2 p3 - Creates
                             my advanced curve from points",
                  __FILE__, myadvcurve, g);
  ...
}
```

#### ***Creating a toolkit (library) as a plug-in***

All custom commands are compiled and linked into a dynamic library (.dll on Windows, or .so on Unix/Linux). To make Test Harness recognize it as a plug-in it must respect certain conventions. Namely, it must export function `PLUGINFACTORY()` accepting the Test Harness interpreter





object (Draw\_Interpreter). This function will be called when the library is dynamically loaded during the Test Harness session.

This exported function `PLUGINFACTORY()` must be implemented only once per library. For convenience the `DPLUGIN` macro (defined in the `Draw_PluginMacro.hxx` file) has been provided. It implements the `PLUGINFACTORY()` function as a call to the `<Package>::Factory()` method and accepts `<Package>` as an argument. Respectively, this `<Package>::Factory()` method must be implemented in the library and activate all implemented commands.

**Example:**

```
#include <Draw_PluginMacro.hxx>

void MyPack::Factory(Draw_Interpreter& theDI)
{
    ...
    //
    MyPack::CurveCommands(theDI);
    ...
}

// Declare entry point PLUGINFACTORY
DPLUGIN(MyPack)
```

**Creation of the plug-in resource file**

As mentioned above, the plug-in resource file must be compliant with Open CASCADE requirements (see `Resource_Manager.cdl` file for details). In particular, it should contain keys separated from their values by a colon (":").

For every created plug-in there must be a key. For better readability and comprehension it is recommended to have some meaningful name.

Thus, the resource file must contain a line mapping this name (key) to the library name. The latter should be without file extension (.dll on Windows, .so on Unix/Linux) and without the "lib" prefix on Unix/Linux.

For several plug-ins one resource file can be created. In such case, keys denoting plug-ins can be combined into groups, these groups - into their groups and so on (thereby creating some hierarchy). Any new parent key must have its value as a sequence of child keys separated by spaces, tabs or commas. Keys should form a tree without cyclic dependencies.

**Examples** (file `MyDrawPlugin`):

```
! Hierarchy of plug-ins
ALL          : ADVMODELING, MESHING
DEFAULT     : MESHING
ADVMODELING : ADVSURF, ADVCURV

! Mapping from naming to toolkits (libraries)
ADVSURF     : TKMyAdvSurf
ADVCURV     : TKMyAdvCurv
MESHING     : TKMyMesh
```

For other examples of the plug-in resource file refer to the **"Plug-in resource file"** chapter above or to the `$CASROOT/src/DrawPlugin` file shipped with Open CASCADE.



**Dynamic loading and activation**

Loading a plug-in and activating its commands is described in the "**Activation of the commands implemented in the plug-in**" chapter.

The procedure consists in defining the system variables and using the pload commands in the Test Harness session.

**Example:**

```
Draw[]> set env(CSF_MyDrawPluginDefaults) /users/test  
Draw[]> pload -MyDrawPlugin ALL
```

