**2009-03-18**

**[Distributing your software](http://opencascade.blogspot.com/2009/03/distributing-your-software.html)**

Many software developers often prefer to focus on code design and development. They find writing technical specifications, fixing bugs, composing documentation or making installers too boring for their creative minds. Yes, developing with the Open Source free CAD/CAM/CAE kernel can be fascinating, but productization is a mandatory part of any successful software product.

Let me share with you a relatively easy way how to streamline your installation of your Open CASCADE based application. I will focus on defining your redistributables and setting environment variables.

**Dynamic libraries**
Here is how you can check which dynamic libraries you need (on Windows):
1. Run Depends.exe which is part of MS Visual Studio installation (e.g. c:\Program Files\Microsoft Visual Studio 8\Common7\Tools\Bin\Depends.exe) and load your executable (or a toolkit). The application will build a dependency tree of dynamic libraries. Pick up those which belong to Open CASCADE (and to other toolkits you might be using).
2. Add TKOpenGl.dll as it’s not directly linked and is loaded at run-time.
3. If you use OCAF, add FWOSPlugin.dll and other dlls corresponding to your formats. For instance, if you use XDE and Xml persistence and do not add your own attributes you need XmlXCAFPlugin.dll and other libraries it depends on. Again, use Depends.exe to unwind dependencies.

**Resource files**
The number of resource files you will need to ship depends on which modules you use. Here are ones that can be most frequently used:
1. Data Exchange.
Include %CASROOT%\src\SHMessage\SHAPE.<>, %CASROOT%\src\XSMessage\IGES.<>, XSTEP.<> where <> is either us or fr depending on which you intend to use. <> is controlled by the CSF\_LANGUAGE variable and defaults to us.
Add %CASROOT%\src\XSTEPResource\IGES, STEP. You will need to set system variables CSF\_SHMessage, CSF\_XSMessage, CSF\_IGESDefaults, CSF\_STEPDefaults to refer to directory(ies) where the files are installed.
2. OCAF
You will need a resource file named <> and a variable CSF\_ <> Defaults with contents describing a format, file extensions, and identifiers (GUIDs) of storage and retrieval document drivers. XDE applications may directly use the XCAF file from the StdResource directory.
Then you will need a file named Plugin and a CSF\_PluginDefaults variable. The Plugin file must contain a line to refer to FWOSPlugin.dll and other drivers if your application maintains store/retrieve operations.
If you use Xml storage format, you will need %CASROOT%\src\XmlOcafResource and the CSF\_XmlOcafResource variable.

3. Textures
If you plan to use prebuilt textures you will need to ship files from %CASROOT%\src\Textures and to set the CSF\_MDTVTexturesDirectory variable.

There are also fonts and units-related files but I don’t remember any use case where they might be needed. If anyone does please reply with the comments.

**System variables**
You will need to set the following variables:
- CSF\_GraphicShr (set to TKOpenGl.dll or a full path where it is located)
- Those mentioned above (if you use respective resource files)
- Optionally MMGT\_\* that control memory management (e.g. MMGT\_OPT=1 activates Open CASCADE memory manager while MMGT\_OPT=0 – standard malloc/free).

Setting environment variables can be done via a bat file which can be created to use some single variable that is set according to the user environment, and all other variables are set relative to it. I am now using another way – setting all variables inside the application itself. This makes it more robust. The following code is executed before main():

//set environment vars to not depend on pre-set environment
OSD\_Environment ("CSF\_LANGUAGE", "us").Build();
OSD\_Environment ("CSF\_GraphicShr", "TKOpenGl.dll").Build();

const int aMaxPath = 256;
TCHAR szPath[aMaxPath];
char aPath [aMaxPath];
DWORD aRes = GetModuleFileName (NULL, szPath, aMaxPath);
QOLib\_ASSERT (aRes);
char \*p = aPath, \*q = (char\*)szPath;
while (\*p++ = \*q++) {
q++;
}
TCollection\_AsciiString aCasRootDir (aPath);
aCasRootDir.Trunc (aCasRootDir.SearchFromEnd ("\\") - 1);
aCasRootDir.Trunc (aCasRootDir.SearchFromEnd ("\\") - 1);
OSD\_Environment ("CASROOT", aCasRootDir).Build();

TCollection\_AsciiString aResDir = aCasRootDir + "/resources";
OSD\_Environment ("CSF\_SHMessage", aResDir).Build();
OSD\_Environment ("CSF\_XSMessage", aResDir).Build();
...

The only limitation is that MMGT\_\* cannot be set this way, as they are also verified before the main() function. This does not bother me as I’m using default values, likely you won’t be affected either.

The approach above allows you to launch your executable from any location (Start menu, its working directory, etc) without a headache of prior tweaking a user’s environment.

After including required libraries and resource files, and setting environment variables, you should be ready to go. Good luck!

Posted by Roman Lygin at [22:52](http://opencascade.blogspot.com/2009/03/distributing-your-software.html) 

|  |  |
| --- | --- |
| Rate this post:  |  |

**3 comments:**



QbProg said...

I did compile OCC into a series of static lib files, and it works really nice, since the compiler deletes unused code!
Also, once I tryed to compile the whole OCC into a single static LIB file, but ... don't do that :)

[March 23, 2009 10:33 AM](http://opencascade.blogspot.com/2009/03/distributing-your-software.html?showComment=1237793580000" \l "c8570772639279945456" \o "comment permalink) 



[Timo](http://www.blogger.com/profile/03921721165516072908) said...

About units-related files:

If you want to use AIS\_Trihedron or AIS\_Axis, you have to set the following environment variables:
set CSF\_UnitsLexicon=%CASROOT%\src\UnitsAPI\Lexi\_Expr.dat
set CSF\_UnitsDefinition=%CASROOT%\src\UnitsAPI\Units.dat

[October 5, 2009 1:59 PM](http://opencascade.blogspot.com/2009/03/distributing-your-software.html?showComment=1254736772712" \l "c3350992222416728525" \o "comment permalink)