On Testing C++ DLLs

by

Edward M. Rosen
EMR Technology Group

Introduction

It is generally recommended that C++ DLLs be tested in the C++ environment before being made available for calling from VBA (Visual Basic for Applications). The procedures used for preparing and calling C++ DLLs from VBA have been described by Rosen (1). The C++ environment is Visual C++ (6.0).

The procedures for developing the C++ DLLs and testing them in the IDE (Integrated Development Environment) using the Wizard differ from those used in preparing and calling the DLLs from VBA.

Preparing C++ DLLs for Use in C++

1. After invoking Visual C++ select "File" and "New"

2. In the project window, give the project a name. Here we have chosen "zzz" and chosen to place it in c:\ctest\

3. Select a "Win32 Dynamic-Link Library"

4. Select "A DLL that exports some symbols"

The Wizard then informs you that the following programs and headers have been prepared:

DllMain
zzz.cpp
zzz.h
Stdafx.h
Stdafx.cpp

5. Figure 1 is a listing of the file zzz.cpp after the function Vari has been added. The "Rebuild All" command is invoked and the compiler informs you that the files zzz.dll and zzz.lib have been created. The zzz.dll file is in c:\ctest\zzz\debug\. Note the differences between Figure 1 and Figures 1a and 1b of Reference (1).

6. After the zzz.cpp compiles successfully, under “Build” invoke the “Set Active Configuration” to “Win32 Release” and then “Rebuild All”.

7. Copy the zzz.dll in c:\ctest\zzz\release\ and place it in Windows\System folder so it can be found by other programs.

Visual C++ Test Program

The general testing procedures are described in by McMahon (2) and Horton (3) though they differ somewhat in that detailed here.

1. Invoke Visual C++ 6.0 and select “File and “New”. Here we have chosen the testzzz for the project name and c:\ctest as the location.

2. Select a “Win32 Console Application”

3. Select “A simple application”

   The Wizard will inform you that the following have been prepared:

   Main: testzzz.cpp
   Stdafx.h
   Stdafx.cpp

4. Figure 2 is a listing of the file testzzz.cpp , the test program that calls the Vari function.

   It is important that locations of the zzz.dll and the zzz.lib be specified during the “Rebuild All”.

   Note that we have placed the zzz.dll file into the Windows/System folder. We must also add the zzz.lib, in c:\ctest\zzz\release\zzz.lib, in the libraries searched under Project Settings – Link.

5. After the testzzz program has been successfully compiled and linked, reset the Active Configuration to “Win32 Release” and the “Rebuild All”.

When the execute testzzz.exe item is invoked, a DOS window appears and the result of the calculations are indicated. Figure 3 is the output.

Summary

The procedures for utilizing C++ DLLs in a C++ program and in VBA differ both in the preparation of the DLLs and their use.
Procedures other than those described here and in Reference (1) may exist. It is noted that the DUMPBIN utility (see the Internet for a description) may be useful in solving linker problems.

References

1. Rosen, E. M. "Use of C++ DLLs in Visual Basic for Applications With Excel 2000", *CACHE News* Fall, 2000.

2. McMahon, *Rapid Application Development with Visual C++*, McGraw-Hill, New York (2000).

3. Horton, Ivor, *Beginning Visual C++ 6*, WROX Press, Chicago (1998).

```
// zzz.cpp : Defines the entry point for the DLL application.

    #include "stdafx.h"
    #include "zzz.h"

    BOOL APIENTRY DllMain( HANDLE hModule,
                           DWORD  ul_reason_for_call,
                           LPVOID lpReserved
                                             )
    {
       switch (ul_reason_for_call)
         {
                   case DLL_PROCESS_ATTACH:
                   case DLL_THREAD_ATTACH:
                   case DLL_THREAD_DETACH:
                   case DLL_PROCESS_DETACH:
                           break;
       }
       return TRUE;
    }

    // This is an example of an exported function.
    ZZZ_API double Vari( double in_data[], long NumItems)

    {
        //Function : Vari
        //Description : Returns the variance for array in_data

        long j;
        double sumx=0, sumsquared=0;
        double vari=0;

        for (j=0; j< NumItems; j++)
        {
                        sumx += in_data[j];
                        sumsquared += in_data[j]*in_data[j];
        }

        vari =(sumsquared - (sumx*sumx/(double)NumItems))/((double)NumItems -1.0);

        return(vari);
    }
```

Figure 1. The zzz.cpp File

```
// testzzz.cpp : Defines the entry point for the console application.
#include <stdafx.h>
#include <iostream>
using namespace std;

extern double Vari(double in_data[] , long  NumItems);

int main(int argc, char* argv[])
{
    long NumItems1;
    long NumItems2;
    long NumItems3;

    double data1[] =  {5.,6.,8.,9.,12.,33.,19.,22.};
    double data2[] =  {8.,9.,20.,33.,56.};
    double data3[] =  {1345.,1301.,1368.,1322.,1310.,1370.,1318.,1350.,1303.,1299.};
    double var;

   NumItems1 = (sizeof data1)/(sizeof data1 [0]);
   NumItems2 = (sizeof data2)/(sizeof data2 [0]);
   NumItems3 = (sizeof data3)/(sizeof data3 [0]);

   cout <<endl;
   cout <<"\nNumItems1 =   "  <<NumItems1;
   cout <<endl;

  cout <<endl;
  cout <<"\nNumItems2 =   "  <<NumItems2;
  cout <<endl;

  cout <<endl;
  cout <<"\nNumItems3 =   "  <<NumItems3;
  cout <<endl;

  var = Vari(data1, NumItems1);

  cout <<endl;
  cout <<"\nvar =   "  <<var;
  cout <<endl;

    var = Vari(data2, NumItems2);

   cout <<endl;
   cout <<"\nvar =   "  <<var;
   cout <<endl;

    var = Vari(data3, NumItems3);

   cout <<endl;
   cout <<"\nvar =   "  <<var;
   cout <<endl;

   return 0;
}
```

Figure 2.   testzzz.cpp File

```
NumItems1 =   8

NumItems2 =   5

NumItems3 =   10

var =   94.2143

var =   398.7

var =   754.267
```

Figure 3. Output from testzzz