Exercise 8 Language Integration

This exercise demonstrates how to throw exception objects from a C++ server to a C++ client. You will write a server class that has a method that may throw exceptions and a client that can decode exceptions.

Step 1 - Getting Started

Fire up VC6 and create a new ATL COM Wizard project in the directory:

```
COM Programming\Exercises\Language Integration
```

Choose a DLL project with a name of ***Errors***.

Step 2 - Creating the Server

Create a new COM object using the wizard. Call the object ***Server*** and make sure you check the ***ISupportErrorInfo*** box.

Add the following method to the ***IServer*** interface:

```
HRESULT SquareRoot([in] double number, [out, retval] double* pResult);
```

Step 3 - Implementing the Server

You only have one method to implement so coding should be quite simple. Use the library routine ***sqrt*** from ***math.h*** to calculate the square root. Obviously, you need to check that the input number is not negative before you take the square root. If it is negative, use the ***CComCoClass::Error*** function to report the error. Don't forget that your COM class derives from ***CComCoClass*** so you can call ***Error*** directly.

If you look in the MSDN you will find a number of error functions to choose from. Select the function with prototype:

```
static HRESULT Error( LPCOLESTR lpszDesc, const IID& iid = GUID_NULL,
                      HRESULT hRes = 0 );
```

You could just supply a description to this function, but to get more insight into what you can pass to the client, provide information in parameters two and three. We suggest you use the GUID of the ***IServer*** interface and an HRESULT of ***E_FAIL***.

Build the server.

Step 4 - Writing the Client

Add a console-based application to your workspace called ***Test***. Choose ***Simple Application*** as the project type, as this will provide a ***main*** function for you.

In the main program, import the type library of the server using the ***#import*** directive. In the function ***main*** call ***CoInitialize*** and then create an instance of the server. If you are feeling adventurous you can use a smart pointer to create the server object:

```
IServerPtr sp(__uuidof(Server));
```

Use the ***__uuid*** syntax to determine all the GUIDs you require. Now loop round calculating the square roots of the numbers 100.0, 95.0, 90.0 and so on, decreasing by 5.0 on each iteration. Eventually you will attempt to take the square root of -5.0 and this will throw a ***_com_error*.**

Trap the error and decode it. You should be able to extract the ***Description***, ***Source***, ***GUID*** and the ***ErrorMessage***. Build a string containing all of this information and display it in a *MessageBox*.

Hint: To display a GUID as a string use ***StringFromCLSID***.

Build and test.

Step 5 - Testing with Visual Basic

Now try the Visual Basic test harness provided and check out how errors get propagated to VB. Note that you get a lot less error information in this case.