Subject: Re: IDL(rsi) + FORTRAN(digital) + DLL(windows nt)
Posted by Michel Kruglanski[1] on Fri, 21 May 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Steve Lionel wrote:
>
> I see that your Fortran DLL code has a PRINT * statement in it. This
> assumes that a console is present for the PRINT. While you can call
> the Win32 API AllocConsole to create the console, you might consider
> using the Win32 API MessageBox to put up a "Hello" box.
>

Effectively, when I remove the print statement, IDL doesn't crash anymore. So, the problem seems to be that the Windows version of IDL has no console. Is there a solution without using Windows specific statements in the Fortran code?
My point is: I have written several Fortran subroutines that I distribute as object library on several plateforms (OpenVMS, VAX/VMS, HP-UX, SunOS, DecOSF, Windows). Since most of the users of this library are using IDL, I have decided to write an IDL interface using the CALL_EXTERNAL function. Therefore I am writing a Fortran subroutine dedicated to be the interface between IDL and several subroutines of my library.
I succeeded already to apply this subroutine under OpenVMS HP-UX ans SunOS without including plateform-specific codes neither in the Fortran code nor in the IDL code (except the name of the sharable object/executable, i.e. test.exe, test.sl, test.so or test.dll, respectively)
Since some of the Fortran subroutines include WRITE(6,*) statement, is it a way to force IDL to accept a console without modifying the Fortran code (even through an option to DF)?

Michel K.
BIRA/IASB