Subject: AAARGH-2 !!! CALL_EXTERNAL and IDL5.2 (the solution)
Posted by StefanoM on Wed, 02 May 2001 08:42:03 GMT
View Forum Message <> Reply to Message

Some days ago a I posted a request of help because my IDL5.2 crashed every time I made a CALL_EXTERNAL to a DLL (based on Fortran subroutines). I want to thanks Frank Randall for having advised me that the IDL_TOOLS work only with IDL5.3 or higher (in fact I tried a workaround to my problem by using the IDL_TOOLS ...).

Well, I would like to share my experience. I use IDL 5.2 and the Microsoft Develop. Studio (ver. 4.0) to compile some Fortran subroutines into a DLL. I compiled the example given with the IDL distribution (in the call_external subdirectory) but it doesn't work. After many tries I found the modifications needed; in the following there is a subroutine ILAT that calls and passes the variables to the ILAT1 subroutine which do the calculations:

```
 ------------------------------------------------------------ ----------------
------------------------------

      SUBROUTINE ILAT(argc, argv)                    !Called by IDL
      !MS$ ATTRIBUTES C, DLLEXPORT :: ILAT

      INTEGER*4 argc, argv(*)                        !Argc and
Argv are integers

      j = LOC(argc)
!Obtains the number of arguments (argc)

!Because argc is passed by VALUE.

c Call subroutine ILAT1, converting the IDL parameters
c to standard FORTRAN, passed by reference arguments:

      CALL ILAT1(%VAL(argv(1)), %VAL(argv(2)))
      RETURN
      END

c
     SUBROUTINE ILAT1(glat,i_lat)

     !MS$ ATTRIBUTES C, DLLEXPORT :: ILAT1
     !MS$ ATTRIBUTES REFERENCE :: glat,i_lat

     DOUBLE PRECISION galt,glat,glong,xtm,xerr,BF,LS,i_lat
 ....
 ....
 ....
```

---------- ----------

The fundamental command (meta-command) is:

!MS$ ATTRIBUTES REFERENCE :: glat,i_lat

which says to the linker that the two variables, glat and i_lat, have to be passed by reference. Without this line IDL crashes when performing the CALL_EXTERNAL command. The other fundamental one is:

!MS$ ATTRIBUTES C, DLLEXPORT :: ILAT1

whitout this the linker gives a "undefined symbol _ILAT1". The other similar command in the first subroutine ILAT (together with the "C" option) is also useful since it allow to call the subroutine directly with his name without the need to add extra sybmols like "_ILAT@8" or similar.

I'm sorry if these considerations are to "stupid" for many of you, but maybe useful for someone ... "dummy" like me.

have a nice day

Stefano