Subject: Re: hooking external 'C' functions to IDL
Posted by rivers on Wed, 13 Mar 1996 08:00:00 GMT
View Forum Message <> Reply to Message

In article <4i6u40$ivf@peabody.colorado.edu>, sena@spot.Colorado.EDU (Michelle Sena) writes:
> I am currently working on a program that uses an IDL gui interface to access
> an external database (Sybase) that contains various data needed for analysis
> and display. I need to go through 'C' to access the database. I would be
> getting back an array or arrays of data. From what I've seen, CALL_EXTERNAL
> can only return discrete data values. Do I need to use the LINKIMAGE routine
> to get back arrays of data? Or am I missing an option for the CALL_EXTERNAL
> procedure that would allow me to do this?

There is no restriction on CALL_EXTERNAL returning arrays in its function arguments. I think it can probably only return a scalar as the function value. IDL essentially passes a pointer to your C routine. You need to make sure that IDL and C agree on what it is pointing to in terms of data type and number of elements.

Mark Rivers (312) 702-2279 (office)
CARS (312) 702-9951 (secretary)
Univ. of Chicago (312) 702-5454 (FAX)
5640 S. Ellis Ave. (708) 922-0499 (home)
Chicago, IL 60637 rivers@cars3.uchicago.edu (Internet)

Subject: Re: hooking external 'C' functions to IDL
Posted by Ken Knighton on Wed, 13 Mar 1996 08:00:00 GMT
View Forum Message <> Reply to Message

sena@spot.Colorado.EDU (Michelle Sena) wrote:
> I am currently working on a program that uses an IDL gui interface to access
> an external database (Sybase) that contains various data needed for analysis
> and display. I need to go through 'C' to access the database. I would be
> getting back an array or arrays of data. From what I've seen, CALL_EXTERNAL
> can only return discrete data values. Do I need to use the LINKIMAGE routine
> to get back arrays of data? Or am I missing an option for the CALL_EXTERNAL
> procedure that would allow me to do this?

CALL_EXTERNAL can pass arguments to the C function being called. These arguments are passed as argc (the number of arguments) and argv (an array of pointers to the arguments). The memory being addressed by these pointers can be manipulated at will (keeping in mind that you have to know how much memory is set aside for each IDL type). An argument could be a scalar or an array. It is not difficult to copy your Sybase data over into an array argument that you have passed into your C routine. When CALL_EXTERNAL returns, the new data is now in the IDL variable used as the parameter in

the CALL_EXTERNAL call. Returning values this way rather than as the return value of the function means that the return value can be used for something better, such as an error code.

In order to pass strings back to the calling program, create a byte array of the appropriate size and pass that into the C routine. The documentation warns against overwriting string areas that were passed to the C routine via CALL_EXTERNAL.

One of the drawbacks to this approach is that you have to figure out how much data is going to be passed back to your routine and allocate an array big enough to hold it all. If this presents a problem, you may want to set up some sort of a dual call system in which the first CALL_EXTERNAL returns the size of the data, you allocate an array that large in IDL, and then call the same routine (i.e. the routine uses static variables) or another routine that completes the process. I have never had a need for anything other than a single call with predefined data, but I don't see why it wouldn't work.

By the way, you should make sure that you thoroughly read all of the documentation on CALL_EXTERNAL in the User's Guide and in the Reference Manual as well as the documentation provided in text files with your IDL distribution. CALL_EXTERNAL can be a little tricky to use. If you make a mistake, you will probably crash IDL or corrupt it in some mysterious way.

Another suggestion: Since you are doing I/O from within a CALL_EXTERNAL routine, look in the IDL Advanced Development Guide for information about UNIX signals (if applicable). This can save you a lot of heartache.

I hope this helps.

Ken Knighton knighton@gav.gat.com knighton@cts.com
General Atomics
San Diego CA
