Subject: Re: CALL_EXTERNAL puzzle (still) ?
Posted by davidf on Fri, 04 Sep 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Rose (rmlongfield@my-dejanews.com) writes:

> However, why can't I pass a pointer? And if I want to pass a pointer, and
> print the value of another pointer just before the CALL_EXTERNAL, why is the
> wrong one passed?

I'm certainly not going to improve on Stein Vidar's excellent
answer, but I did want to give you the short version:
you can't pass pointers and (really) variables because
those pointers and variables are not really what you
think they are. In other words, even though we call
these things "pointers" in IDL, they are not the same thing
as the "pointers" over in your C program. To believe otherwise
is to invite strange behavior in your program, as you have
discovered.

The same is mostly true of variables as well. A variable
in IDL is a fairly complicated structure. What Call_External
does is strip out the *data* portion of this structure
and pass it to your C program. Since it does this invisibly,
it is easy to think that you are "passing your variable".
You are doing no such thing. (This is, by the way, why
you MUST create the variable or allocate memory for it on
the IDL side and not on the C side. Variable creation makes
the whole big thing that "describes" the real variable.
However, as Stein Vidar points out, this can be done in
your C program if you were using LinkImage.) When the data
comes back from the C program, IDL puts it back into its
larger variable structure. This is why it is essential
that your C program doesn't change the size or type of
the variable. If it did, the real IDL variable would have
bogus information about itself and all hell would break
loose. :-)

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438, Toll-Free Book Orders: 1-888-461-0155
Coyote's Guide to IDL Programming: http://www.dfanning.com/