Subject: Re: unloading a dlm...
Posted by Stein Vidar Hagfors H[1] on Mon, 17 Sep 2001 18:44:41 GMT
View Forum Message <> Reply to Message

Richard Younger <younger@ll.mit.edu> writes:

> Randall Skelton wrote:
>>
>> I suspect this is impossible, but does anyone know of a way to force IDL
>> unload a DLM (without doing a reset_session).
>>
>> My problem is that I've quickly bolted a large Fortran model onto IDL and
>> nearly every variable is in a common block (i.e. a global C structure).
>> It amounts to me consuming an extra 50MB of RAM after this particular DLM
>> is loaded :(It would be nice to reclaim this memory when I am done with
>> the model...
>
> I'm afraid you're just giving ammunition to the common block snobs
> (myself among them). I can't help you unload a dlm, but I can think of
> a few poor alternatives. :-)
>
> If you happen to be on an Intel platform and not developing for anyone
> else, RAM is dirt cheap compared with six months ago. Well less than
> US$100 will get you an extra 128 Megs of memory and you can let Moore's
> law absorb the extra 50 MB hit.
>
> If that's not feasible, you can use the aforementioned
> .full_reset_session from the main level in a script (@-file).

Guess he doesn't really want that: He needs to keep the data in IDL!

> Compiling as an executable and running with spawn, communicating with
> pipes or sockets and the like, would probably at least a couple steps
> backwards.

I think this may be a good candidate for Remote Procedure Calls, i.e. idlrpc,
with two front-ends: One accepting keyboard input, like the idlrpc example
program, and the other (the model) stuffing IDL with the data after doing the
calculations, then simply dying!

You'd at least avoid the writing/reading of data to disk... But if you're
talking about very large volumes, I'm not sure exactly how the RPC protocol
buffers things... (they might go via the disk anyway?). You should at least
use the IDL_RPCImportArray for creating variables to send over large stuff,
but there *will* be some overhead in the sending...

> There might be an obscure option on your F** compiler to change the way
> it compiles common blocks, but since I haven't really used Fortran much,

> this is pure speculation on my part.
>
> All of those alternatives are limited and clunky, if they exist at all.
> I don't know any better way than to do the obvious (time consuming)
> thing and gain some quality time with your favorite Fortran compiler and
> search-and-replace tool.

I would guess there are no ideal solutions to this, short of rewriting the original code to avoid the static memory allocation (which I understand is not an option).

Come to think of it: If you are passing along the *majority* of the statically allocated data from F77 to IDL, then you might be better off with using Callable IDL - again using the IDL_ImportArray (note the lack of RPC in the routine name).

This way, your F77 program is linked together with IDL, you provide a main program (in C, I would hope) that calls IDL and F77 routines in some alternating fashion.. I guess that technically it's possible to "share memory" in arrays between the two processes (but be *very* careful with writing to those variables in IDL!!).. There's not automatic "reset" of the F77 routine, though, as a .full_reset_session would cause.

> Good luck,

Yes, indeed!

--

-- --------------
Stein Vidar Hagfors Haugan
ESA SOHO SOC/European Space Agency Science Operations Coordinator for SOHO

NASA Goddard Space Flight Center,   Email: shaugan@esa.nascom.nasa.gov
Mail Code 682.3, Bld. 26, Room G-1,   Tel.: 1-301-286-9028/240-354-6066
Greenbelt, Maryland 20771, USA.   Fax: 1-301-286-0264
-- --------------