Subject: Calling IDL from Fortran called by IDL
Posted by Kevin A. Park on Mon, 26 Nov 2001 22:28:37 GMT
View Forum Message <> Reply to Message

Hi,

I have an IDL GUI which sits on top of a calculation engine which is written in Fortran 90. Currently IDL accesses the calculation engine by calling C wrapper functions via CALL_EXTERNAL. These C wrappers then call Fortran routines. The system runs both on Solaris and Windows platforms. I am currently using IDL 5.4, but will upgrade to IDL 5.5 soon.

Some of the calculations in the Fortran take a long time, so what I would like to do is have IDL create a progress bar which can be updated from the Fortran. Having waded through the IDL External Development Guide, I have a few questions.

1. The only way of calling IDL common to both UNIX and Windows platforms is callable IDL, yes?

2. According to the External Development Guide, callable IDL can not be used from code that is called from IDL via CALL_EXTERNAL or LINKIMAGE. But can callable IDL be used in a dynamically loaded module (DLM)?

3. Does callable IDL always start a new IDL process (i.e. a new "virtual machine" to borrow Java terminology) or can it be used to access data and functions in an existing IDL process?

4. Is there any way of passing function pointers to IDL methods to a native compiled language such as C so that these routines could be called from the compiled language (e.g. C)? (In Java there is a way to use the Java-Native Interface (JNI) to accomplish just this sort of thing.)

Thanks for any help you can provide.

Kevin Park

--

Kevin A. Park
Prism Computational Sciences, Inc.
16 N. Carroll St., Suite 950
Madison, WI 53703
Phone: (608) 287-1042
Fax: (608) 280-9390

mailto:kpark@prism-cs.com

Subject: Re: Calling IDL from Fortran called by IDL
Posted by Craig Markwardt on Wed, 28 Nov 2001 02:34:03 GMT
View Forum Message <> Reply to Message

"Kevin A. Park" <kpark@prism-cs.com> writes:
> Hi,
>
> I have an IDL GUI which sits on top of a calculation engine which
> is written in Fortran 90. Currently IDL accesses the calculation engine
> by calling C wrapper functions via CALL_EXTERNAL. These C wrappers then
> call Fortran routines. The system runs both on Solaris and Windows
> platforms. I am currently using IDL 5.4, but will upgrade to IDL 5.5
> soon.
>
> Some of the calculations in the Fortran take a long time, so what I
> would like to do is have IDL create a progress bar which can be updated
> from the Fortran. Having waded through the IDL External Development
> Guide, I have a few questions.

Kevin! Good to see you alive and kicking. [In case you don't remember me I was Ben's roommate.]

You've gotten some good suggestions from other replies to your request, and they are worth considering.

Personally I would say that you may be asking too much, to make an IDL-to-FORTRAN-to-IDL call chain. It must be possible, because several routines do allow such an operation. Consider CONSTRAINED_MIN.

You are probably looking into using the internal C functions IDL_Execute() or IDL_ExecuteStr(), which may be exactly what you are looking for. What you would do is call these functions from within your FORTRAN subroutine, with the name of an IDL procedure that would update the GUI. While these functions are documented under the "Callable IDL" section, I do not see why these couldn't be used in your case.

Another possibility is to use IDL to manage a separate process via the SPAWN command. Under Unix at least, it is possible for the IDL process to intercept all I/O to a subprocess, so you could have your FORTRAN routine print status messages which could be used to update the GUI.

Good luck!

Craig

--
---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Calling IDL from Fortran called by IDL
Posted by Mark Rivers on Wed, 28 Nov 2001 14:45:01 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@cow.physics.wisc.edu> wrote in message news:onherfoc0q.fsf@cow.physics.wisc.edu...
>
> Oooh, but be careful, because if you allow your IDL variable to be
> released, say by reassigning it, then your external C routine will
> probably end up overwriting some part of memory you didn't want it to.

Yes, this technique should only be used for carefully written IDL "applications", and not for interactive use. Any IDL variable whose address has been passed to the external C or FORTRAN code can never appear on the left hand side of an assignment.

As I think about it I actually use the technique I described all the time now, without really being aware of it. I am using it in our real-time control system, which is a client/server system using TCP/IP networking. IDL interacts with this by doing the following:

- IDL wants to do a "put" or "get" of a real-time system parameter
- IDL calls external C code which issues a TCP/IP request and returns immediately.
- TCP/IP response comes back to the C code
- IDL polls the C code to see if the response has come back
- IDL reads the response from the C code

This idea can be directly applied to the problem of interacting with compute-bound FORTRAN code.

- IDL calls C wrapper which spawns a new FORTRAN process. It does not need to be a thread, but can be a new process because it is not going to share memory with IDL.

- FORTRAN process sends progress messages to C wrapper interface via sockets, pipes (or any other inter-process communication mechanism)

- IDL polls progress via C wrapper interface

> Also, one always needs to worry that things like malloc() and printf()
> may not be threadsafe on one's platform. If your work thread and the
> IDL thread clash, then *pow*, that will hurt.

This eliminates that problem because the FORTRAN is a new process, not a thread.

On a related note, I have recently added a GUI interface to my IDL tomography processing routines, which were formerly accessible only via the command line. The IDL routines that do the processing are compute-bound for 10 minutes or longer. It was trivial to make these routines "GUI-friendly" by making them accept 2 optional keywords, the IDs of a "status widget" and an "abort widget". If these keywords are defined, and point to a valid widget then on each time through a loop (every couple of seconds) progress information is written to the status widget, so status information is visible in the GUI. "widget_event" is called for the abort widget, and the uvalue of it is read. If it is 1 then the computation is aborted:

```
if (widget_info(abort_widget, /valid_id)) then begin
   event = widget_event(/nowait, abort_widget)
   widget_control, abort_widget, get_uvalue=abort
   if (abort) then return
endif
```

A very nice side-effect of this technique is that, because widget_event is being called every couple of seconds, both my GUI screen and the IDLDE window respond to mouse events, so they are refreshed and can be moved around on the screen. This was not possible previously, they were totally unresponsive for 10 minutes when the calculation was in progress.

Mark Rivers

Subject: Re: Calling IDL from Fortran called by IDL
Posted by Craig Markwardt on Wed, 28 Nov 2001 22:38:49 GMT
View Forum Message <> Reply to Message

"Mark Rivers" <rivers@cars.uchicago.edu> writes:

> Craig Markwardt <craigmnet@cow.physics.wisc.edu> wrote in message
> news:onherfoc0q.fsf@cow.physics.wisc.edu...
>>
>> Oooh, but be careful, because if you allow your IDL variable to be
>> released, say by reassigning it, then your external C routine will
>> probably end up overwriting some part of memory you didn't want it to.

>
> Yes, this technique should only be used for carefully written IDL
> "applications", and not for interactive use. Any IDL variable whose address
> has been passed to the external C or FORTRAN code can never appear on the
> left hand side of an assignment.
...
> This idea can be directly applied to the problem of interacting with
> compute-bound FORTRAN code.
...
> This eliminates that problem because the FORTRAN is a new process, not a
> thread.

These are all really great ideas! Thanks. The only thing I worry about is Kevin's initial mention of the dreaded word, "Windows" :-)

I really like your ideas on polling a widget for an abort event. I've been thinking about how to do that for something like MPFIT, but didn't want to figure out all the details. Looks like you did it for me!

Craig

--

---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: Calling IDL from Fortran called by IDL
Posted by Stein Vidar Hagfors H[1] on Fri, 30 Nov 2001 21:12:15 GMT
View Forum Message <> Reply to Message

Craig Markwardt <craigmnet@cow.physics.wisc.edu> writes:

> "Kevin A. Park" <kpark@prism-cs.com> writes:
>> Hi,
>>
>> I have an IDL GUI which sits on top of a calculation engine which
>> is written in Fortran 90. Currently IDL accesses the calculation engine
>> by calling C wrapper functions via CALL_EXTERNAL. These C wrappers then
>> call Fortran routines. The system runs both on Solaris and Windows
>> platforms. I am currently using IDL 5.4, but will upgrade to IDL 5.5
>> soon.
>>
>> Some of the calculations in the Fortran take a long time, so what I
>> would like to do is have IDL create a progress bar which can be updated
>> from the Fortran. Having waded through the IDL External Development

>> Guide, I have a few questions.
>
> Kevin! Good to see you alive and kicking. [In case you don't
> remember me I was Ben's roommate.]
>
> You've gotten some good suggestions from other replies to your
> request, and they are worth considering.
>
> Personally I would say that you may be asking too much, to make an
> IDL-to-FORTRAN-to-IDL call chain. It must be possible, because
> several routines do allow such an operation. Consider
> CONSTRAINED_MIN.
>
> You are probably looking into using the internal C functions
> IDL_Execute() or IDL_ExecuteStr(), which may be exactly what you are
> looking for. What you would do is call these functions from within
> your FORTRAN subroutine, with the name of an IDL procedure that would
> update the GUI. While these functions are documented under the
> "Callable IDL" section, I do not see why these couldn't be used in
> your case.

I seem to remember that this *is* possible, but one problem is in
passing parameters in a clean way.. If you create variables inside the
C/Fortran routines, you've got either $MAIN$ scope or the scope of the
IDL routine calling the DLM (or using CALL_EXTERNAL).. Have to be
careful not to overwrite existing variable names when you're about to
construct a call string like "update_progress,n_percent"... As far as
I remember, anyway..

If you're doing your own "dirty work", however, you'll have the
knowledge you need to avoid those problems.

I think also that someone (from RSI?) said one couldn't expect this to
be functional, because the IDL_Execute() and IDL_ExecuteStr()
functions are meant to be used *only* in callable idl scenarios, not
by DLMs/call_external code..

But I do wish they'd release the documentation to do it cleanly, like
CONSTRAINED_MIN..

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

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