Subject: DLM's and C code
Posted by Richard Tyc on Tue, 08 Jan 2002 22:48:09 GMT
View Forum Message <> Reply to Message

A somewhat IDL related question.
I am trying to link in some C code via a DLM. I use a wrapper routine to handle the call from IDL and manipulate the args and return data. Within the wrapper, I call C functions linked in through another DLL.

What is the best way to handle errors while deeply nested within layers of C functions.? The ANSI C code I am using essentially had exit(1) calls for major errors. Is there an IDL_ function (like say an exit handler) I can call to cleanly return to IDL rather than a trying to modify the call stack and get back to the IDL wrapper function to perform something like a return IDL_StrToSTRING("ERROR") ;

Thanks

--
Richard Tyc
Project Engineer
St. Boniface Hospital Research Center
351 Tache Ave
Winnipeg, MB
Canada
Tel: 204-237-2557
Fax: 204-231-0485
Email: richt@sbrc.umanitoba.ca

Subject: Re: DLM's and C code
Posted by Dominik[1] on Thu, 10 Jan 2002 08:44:27 GMT
View Forum Message <> Reply to Message

Hi Richard,

I do it a little bit easier way. If an error occurs in my C routine (in a DLL) I return with an error code, lets say:
#define ERROR_DIVISION_BY_ZERO -128
return ERROR_DIVISION_BY_ZERO;

In IDL I can check the return value

status = call_external()
if status EQ -128 then...

This works really good for me. The calculations which the C routine is doing for me, is written into a piece of memory and can then be seen by IDL.

Therefor I create a variable in IDL, pass it to my DLL by reference (I think it is the normal way to pass it by referenze), can manipulate the value and on returning to IDL, the variable will hold the new value.

Hope it helps you
Dom

"Richard Tyc" <Richard_Tyc@sbrc.umanitoba.ca> schrieb im Newsbeitrag news:a1fsur$78a$1@canopus.cc.umanitoba.ca...
> A somewhat IDL related question.
> I am trying to link in some C code via a DLM. I use a wrapper routine to
> handle the call from IDL and manipulate the args and return data. Within the
> wrapper, I call C functions linked in through another DLL.
>
> What is the best way to handle errors while deeply nested within layers of C
> functions.? The ANSI C code I am using essentially had exit(1) calls for
> major errors. Is there an IDL_ function (like say an exit handler) I can
> call to cleanly return to IDL rather than a trying to modify the call stack
> and get back to the IDL wrapper function to perform something like a return
> IDL_StrToSTRING("ERROR") ;
>
> Thanks
>
> --
> Richard Tyc
> Project Engineer
> St. Boniface Hospital Research Center
> 351 Tache Ave
> Winnipeg, MB
> Canada
> Tel: 204-237-2557
> Fax: 204-231-0485
> Email: richt@sbrc.umanitoba.ca
>
>

Subject: Re: DLM's and C code
Posted by Richard Tyc on Fri, 11 Jan 2002 19:26:58 GMT
View Forum Message <> Reply to Message

My problem was that the error did NOT occur in the C routine called by call_external but in other deeply nested routines with no simple means of

getting back to the routine called by CALL_EXTERNAL. Essentially, I adapted some existing ANSI C code to be called by IDL and added "IDL" specific features (like the testmodule example in docs for MAKE_DLL) so I could call various functions I needed. Unfortunately, major errors in this code were handled by simple calls to exit() which is not helpful to IDL and it would also have been a real pain to add returns throughout the many C functions to return the error back to IDL. Now I can use IDL_MESSAGE with the IDL_MSG_LONGJMP action.

Rich

Dominik Paul <dpaul@ukl.uni-freiburg.de> wrote in message news:a1jk1e$f8t$1@n.ruf.uni-freiburg.de...
> Hi Richard,
>
> I do it a little bit easier way. If an error occurs in my C routine (in a
> DLL) I return with an error code, lets say:
> #define ERROR_DIVISION_BY_ZERO -128
> return ERROR_DIVISION_BY_ZERO;
>
> In IDL I can check the return value
>
> status = call_external()
> if status EQ -128 then...
>
> This works really good for me. The calculations which the C routine is doing
> for me, is written into a piece of memory and can then be seen by IDL.
> Therefor I create a variable in IDL, pass it to my DLL by reference (I think
> it is the normal way to pass it by referenze), can manipulate the value and
> on returning to IDL, the variable will hold the new value.
>
> Hope it helps you
> Dom
>
>
> "Richard Tyc" <Richard_Tyc@sbrc.umanitoba.ca> schrieb im Newsbeitrag
> news:a1fsur$78a$1@canopus.cc.umanitoba.ca...
>> A somewhat IDL related question.
>> I am trying to link in some C code via a DLM. I use a wrapper routine to
>> handle the call from IDL and manipulate the args and return data. Within
> the
>> wrapper, I call C functions linked in through another DLL.
>>
>> What is the best way to handle errors while deeply nested within layers of

> C
>> functions.? The ANSI C code I am using essentially had exit(1) calls
for
>> major errors. Is there an IDL_ function (like say an exit handler) I
can
>> call to cleanly return to IDL rather than a trying to modify the call
> stack
>> and get back to the IDL wrapper function to perform something like a
> return
>> IDL_StrToSTRING("ERROR") ;
>>
>> Thanks
>>
>> --
>> Richard Tyc
>> Project Engineer
>> St. Boniface Hospital Research Center
>> 351 Tache Ave
>> Winnipeg, MB
>> Canada
>> Tel: 204-237-2557
>> Fax: 204-231-0485
>> Email: richt@sbrc.umanitoba.ca
>>
>>
>
>

Subject: Re: DLM's and C code
Posted by Dominik[1] on Mon, 14 Jan 2002 10:07:17 GMT
View Forum Message <> Reply to Message

Ok, now I see what you mean.

"Richard Tyc" <Richard_Tyc@sbrc.umanitoba.ca> schrieb im Newsbeitrag
news:a1ne9hock1@canopus.cc.umanitoba.ca...
> My problem was that the error did NOT occur in the C routine called by
> call_external but in other deeply nested routines with no simple means of
> getting back to the routine called by CALL_EXTERNAL. Essentially, I
adapted
> some existing ANSI C code to be called by IDL and added "IDL" specific
> features (like the testmodule example in docs for MAKE_DLL) so I could
call
> various functions I needed. Unfortunately, major errors in this code were
> handled by simple calls to exit() which is not helpful to IDL and it would
> also have been a real pain to add returns throughout the many C functions
to

> return the error back to IDL. Now I can use IDL_MESSAGE with the
> IDL_MSG_LONGJMP action.
>
> Rich
>
> Dominik Paul <dpaul@ukl.uni-freiburg.de> wrote in message
> news:a1jk1e$f8t$1@n.ruf.uni-freiburg.de...
>> Hi Richard,
>>
>> I do it a little bit easier way. If an error occurs in my C routine (in
a
>> DLL) I return with an error code, lets say:
>> #define ERROR_DIVISION_BY_ZERO -128
>> return ERROR_DIVISION_BY_ZERO;
>>
>> In IDL I can check the return value
>>
>> status = call_external()
>> if status EQ -128 then...
>>
>> This works really good for me. The calculations which the C routine is
> doing
>> for me, is written into a piece of memory and can then be seen by IDL.
>> Therefor I create a variable in IDL, pass it to my DLL by reference (I
> think
>> it is the normal way to pass it by referenze), can manipulate the value
> and
>> on returning to IDL, the variable will hold the new value.
>>
>> Hope it helps you
>> Dom
>>
>>
>> "Richard Tyc" <Richard_Tyc@sbrc.umanitoba.ca> schrieb im Newsbeitrag
>> news:a1fsur$78a$1@canopus.cc.umanitoba.ca...
>>> A somewhat IDL related question.
>>> I am trying to link in some C code via a DLM. I use a wrapper routine
to
>>> handle the call from IDL and manipulate the args and return data.
Within
>> the
>>> wrapper, I call C functions linked in through another DLL.
>>>
>>> What is the best way to handle errors while deeply nested within
layers
> of
>> C
>>> functions.? The ANSI C code I am using essentially had exit(1) calls

> for
>>> major errors. Is there an IDL_ function (like say an exit handler) I
> can
>>> call to cleanly return to IDL rather than a trying to modify the call
>> stack
>>> and get back to the IDL wrapper function to perform something like a
>> return
>>> IDL_StrToSTRING("ERROR") ;
>>>
>>> Thanks
>>>
>>> --
>>> Richard Tyc
>>> Project Engineer
>>> St. Boniface Hospital Research Center
>>> 351 Tache Ave
>>> Winnipeg, MB
>>> Canada
>>> Tel: 204-237-2557
>>> Fax: 204-231-0485
>>> Email: richt@sbrc.umanitoba.ca
>>>
>>>
>>
>>
>
>