Subject: Re: Call_External and referenced symbol not found
Posted by Malcolm Walters on Thu, 09 May 2002 14:41:58 GMT
View Forum Message <> Reply to Message

"Todd David halter" <thalter@beta.tricity.wsu.edu> wrote in message
news:abbhks$21ov$1@murrow.it.wsu.edu...
> I do have the -lZeb and it still does not work. If you would post you
> long re: that would be great.
>
> Todd

Todd, I'm not 100% sure exactly what you are doing, I mainly work on Linux
but this should transfer ok.

What I have done before on several occasions is;
1) Start with an archive file. (libA.a)
2) Write functions calling this library (B.c) and compile using something of
the form
gcc -shared -o libB.so B.c -lA
3) Call the wrapper functions in libB from within IDL.

Note that all the functions you are going to call in 'libA.a' from IDL must
be referenced in 'B.c'. If you miss any out then the resulting shared object
does not contain the functions and they cannot be called. This can be done
in a dummy routine that you will never call if necessary. All call_externals
should call 'libB.so'.

If you have the source for libA then recompile it to give a shared object.
This way B.c only needs to make calls that actually do something.
Then if you wish to call functions in libA then you must call these directly
(call_external,'libA.so',...)
You will need to have both libraries on your 'LD_LIBRARY_PATH' environmental
variable.

Note that either of these should give you only one instance of each shared
library. If you have an archive do not try something like
gcc -shared -o libOpenB.so OpenB.c -lA
gcc -shared -o libCloseB.so CloseB.c -lA
if you need internal data within libA to be preseved between the open and
close calls.

Malcolm