Subject: CALL_EXTERNAL related problems
Posted by Chris Ditchman on Thu, 07 Sep 1995 07:00:00 GMT
View Forum Message <> Reply to Message

Hello,

I am a student who is new to UNIX and programming in general.
I am using IDL 4.0 on a sparc box running SunOs 5.3, which I
guess would be Solaris 2.4.1 if I remember correctly.
Anyhow, I have two questions.

1) I am trying to compile my shared object library with the GNU
C compiler version 2.6.3 for use with the call_external function.
What options should I be passing for the compiling and linking
phases? I currently supply -ansi -shared -symbolic -fPIC to gcc.
I create the .so with the cc compiler without complaint, but I get
a bus error when I switch to gcc and attempt to use what I've created.

2) I am writing a user interface that interacts with a number of C
functions. The C functions create a number of static structures
whose values I need to access. I currently create an array of
structures of parallel format and pass the array via the call_external
function. A co-worker has recently pointed out that I might have
alignment problems if I attempt to fill this IDL structure. Here is
a functional example of what I currently do:

In IDL, I do this:

..

```
vector = {VectorStruct, $
 text: '', $
 value: 0.0D }

array = replicate(vector, N)

ret = call_external('library.so', 'routine', array)
```

..

**note: N has been obtained in a previous call_external call.

I create a C header file called "IDL_types.h" with the following
definitions:

```
typedef struct {
 unsigned short slen;
 short stype;
```

```
 char *s;
 } IDLString;

typedef struct {
 IDLString text;
 double value;
 } IDLVectorStruct;
```

A typical function I would then create:

```
#include "IDL_types.h"
/* Include other necessary headers */

long fill_structure(int argc, void *argv[]) {

/* Miscellaneous variable declarations */

IDLVectorStruct *ptr_vector;

ptr_vector = &((IDLVectorStruct *)argv[0])[0];

..

/* From here I proceed to fill the array of structures */

return SUCCESS;
}
```

I should note that the array size N (from the IDL code)
is contained as one of the fields in the static structures
which I access.

The question is this: is there a way for the alignment
between the IDL array of structures which I pass in
and the structure I typecast the incoming argument as to
become askew as I change compilers and platforms? So long
as I keep my IDL and C structure definitions parallel,
and IDL is run on the same system that I create my
'so' on, what is the likelyhood of my running into
major problems? I would simply have to keep track of
what an IDL equivalent to an integer and the other types
are on the various systems in terms of byte size, right?

I guess another general question would be is if this is a
stable way of doing things and what a better approach might
be along these lines? The answer to this would either keep me
on track or have me redesign a fair bit of work.

I'm not a regular reader of news, so if anyone with any
help could forward a response to me, I would appreciate it.
Thanks in advance.

Christopher J. Ditchman
Jet Propulsion Laboratory
4800 Oak Grove Dr.
Pasadena, CA 91109
MS 238-600
(818) 354-0124
cjd@yosemite.jpl.nasa.gov
