Subject: Returning A Variable Length struct to IDL from C
Posted by K Banerjee on Tue, 06 Nov 2001 00:10:54 GMT
View Forum Message <> Reply to Message

Folks,

I wrote a DLM, following the examples in "Calling C From IDL"
by Mr. Ronn Kling. Here's the simplified layout of what I
am doing:

In the file idl_vbio.h, I have the following struct:

```
typedef struct
{
  IDL_STRING vers;
 IDL_STRING *userHeader;
 IDL_LONG byteOffSet;
} vbHeader;
```

I need to read the text header of a data file and then populate
the vbHeader struct. However, there will be a varying number of
text lines that comprise the userHeader, i.e., 2 data files
may not have the same number of "user header" text lines.

In the function that reads the text header, called
idlvbio_get_cube_header(), I have:

```
int userHeaderArrayLength = some_function_that_returns_int();
```

Later on in this function, I have:

```
IDL_STRUCT_TAG_DEF vbHeaderTags[] =
{
   {"VERS", 0, (void *) IDL_TYP_STRING},
  {"USERHEADER", dims_user_header, (void *) IDL_TYP_STRING},
  {"BYTEOFFSET", 0, (void *) IDL_TYP_LONG},
  {0}
};
```

where

```
static IDL_LONG dims_user_header[] = {1, userHeaderArrayLength};
```

Continuing in this function, I have:

```
  typedef struct
  {
    IDL_STRING vers;
    IDL_STRING userHeader[userHeaderArrayLength];
    IDL_LONG byteOffSet;
  } vbHeaderActual;
```

The difference between the above struct and the first struct
(found in the C include file) is that the second field in the
first struct is a pointer to IDL_STRING while in the
above struct, the second field is an array of type IDL_STRING of
length userHeaderArrayLength.

An instance of the vbHeader struct is created, called theHeader,
and then the following line is executed:

```
theHeader->userHeader = new IDL_STRING[userHeaderArrayLength];
```

The function then continutes to populate the fields of theHeader.

The function then instantiates a struct of type vbHeaderActual,
called theHeaderActual.

The function then copies the fields from theHeader
to theHeaderActual. So far, so good.

The next two lines are used to create the return value to IDL:

```
void *psDef = IDL_MakeStruct(NULL, vbHeaderTags);
IDL_VPTR ivReturn = IDL_ImportArray(1, ilDims, IDL_TYP_STRUCT,
(UCHAR *) theHeaderActual, releaseMemory, psDef);
```

where releaseMemory is the function:

```
extern "C" void releaseMemory(UCHAR *ptr)
{
  deleteMem(ptr);
} // extern "C" void releaseMemory(UCHAR *ptr)
```

and ilDims is:

```
ilDims[0] = 1;
```

The return line is:

```
return ivReturn;
```

Now assume that I need to read the header of 2 data files,
file1 and file2. Further assume that file1 has 10 user header
lines and file2 has 15 user header lines. So from IDL,
here's what it looks like:

```
IDL> h = idlvbio_get_cube_header('file1')
IDL> h = idlvbio_get_cube_header('file2')
```

The second IDL line causes a core dump due to a segmentation
fault occurring in IDL_MemFree():

```
#0  0x4008cf58 in IDL_MemFree () at ../../gcc-2.95.2/gcc/cp/exception.cc:343
343     ../../gcc-2.95.2/gcc/cp/exception.cc: No such file or directory.
```

What I think happens is that with the first call to the function
idlvbio_get_cube_header(), a certain amount of memory is allocated
to the IDL variable h. Now the second call to idlvbio_get_cube_header()
does not cause IDL to delete the memory already allocated to h
and then reallocate memory to h. More memory is now needed by h
since there are 15 user header lines in file2.

However, using a second IDL variable, h2, overcomes this
problem:

```
IDL> h = idlvbio_get_cube_header('file1')
IDL> h2 = idlvbio_get_cube_header('file2')
```

What I'd like to be able to do is reuse the IDL variable h.

I can not resort to using the IDL procedure delvar since
delvar is only available at the IDL prompt and can not be used
from IDL functions and procedures (I need to wrap
idlvbio_get_cube_header() inside IDL functions and procedures).

I will need to return a struct with a variable length array as
its second field (the userHeader[] array).

Any suggestions on how I can do what I need to do?

My environment is:
Redhat 6.2 Linux
Kernel 2.2.19
gcc 2.95.2
IDL 5.3

Thanks.

K. Banerjee

--

"One World, One Web, One Program" -- Microsoft Promotional Ad

"Ein Reich, Ein Volk, Ein Fuhrer" -- Adolf Hitler