Subject: Re: Structures:
Posted by Mark Hadfield on Wed, 21 Feb 2001 20:52:42 GMT
View Forum Message <> Reply to Message

"Chris Bull" <cjbull@another.com> wrote in message
news:owUk6.10166$5n4.162849@news6-win.server.ntlworld.com...
> A question regarding IDL structures, I have an application that I am
> writing that I am using a very complex
> structure as a Uvalue to hold my data in, as this application evolves I am
> wondering whether it
> is possible to add 'branches' to a structure after it is defined (and set
as
> a uvalue), from within a seperate subroutine?

Well, yes and no. The IDL built-in CREATE_STRUCT allows you to add tags to a structure, but what it actually does is erase the original value and generate a new one, i.e. a full copy of the structure data in memory. This is fine for small structures, but inefficient on large ones. It is a consequence of the decisions IDL made when they designed the structure data type: keep it contiguous (except for padding) in memory, simple and fast.

> I'm Thinking along the lines of describing an application that can hold
many
> data sets, of different types in memory
> and load them up seperatly (much like a imaging program or multiple
> documents in word) except the data sets are
> very complex and of differing structures

You could use a data container base on pointers, but accessing and modifying this will get complicated. Me, I get confused by all those dereferncing symbols, e.g. *(*c(*a.b)))

So my reccommendation is to use an object-based data container. There are no adequate ones supplied with IDL (more's the pity) so you will have to write your own or adapt someone else's. For a start you could look at a few on my WWW page: go to http://katipo.niwa.cri.nz/~hadfield/gust/software/idl/ and look at:

mgh_vector__define
mgh_queue__define
mgh_stack__define

The vector is the one that might be of use. It is a 1-D vector that can hold data elements of any type, indexed by number. The size is adjusted dynamically & automatically as elements are added.

A few months ago I tried writing a similar "dictionary" object, i.e. one that holds a series of name:value pairs (rather like an IDL structure.) I

found that accessing the data elements was too slow. Such data structures are common and provide reasonable performance in other languages (e.g. Python, Matlab) so it should be possible in principle to create one in IDL, but you've got to get clever in the way you look up the names.

--

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research
