Subject: Re: IDL 6.0
Posted by Randall Skelton on Thu, 27 Mar 2003 23:01:50 GMT
View Forum Message <> Reply to Message

>> My *hope* is that IDL 6 will extend the current object code base and
>> support (1) operator overloading, (2) proper inheritance mechanisms (i.e.
>> avoiding name space clashes that occur in structures), (3) public /
>> private methods and (4) a C-API interface to object/heap variables. I'm
>> also keen to see an object interface to map drawing...
>
> Oh, dear. Maybe in 6.1. :-(

OK, I realize that I am being a tad greedy here but I really do think that IDL objects are too limited in their current scope and it is beginning to become a real problem for me. I do a fair amount of multivariate analysis where a given vector or matrix is really a composite of various different data sets. I solve large systems of this sort and have a variety of routines in c/fortran that work on these composite objects efficiently. In many cases it is as simple as defining an object that holds a vector or array, with a hash table that describes what various parts of the vector or array correspond to, and some ancillary stuff about the measurements. I quite like operator overloading as it makes working with these composite objects (i.e. solving them as linear/non-linear systems) easy code and read. I do this all the time in c++, fortran and (recently) Matlab. I have some great object graphics routines written in IDL to visualize my data but I really have no way of actually operating on my data without resorting to writing very cryptic code. Try re-writing:

```
xh = xh + invert( (1+gamma) * SaI + Kts # k) # $
    ( Kts # (y_obs-y_f) - SaI # (x_h-x_a) )
```

when each addition is 'obj->add(a,b)', scalar multiplication is 'obj->shift(1)', and matrix multiplication 'obj->mm(A,B)'. IMHO, data encapsulation should not prevent people from actually working with their data in a logical way!

My message to RSI is that I don't really need any more GUI tools, spread-sheet views or slow pointy-clicky interfaces. What I would like is a more complete set of core object programming interfaces so I can program more logically and manipulate/visualize my data more efficiently. I hate to admit it, but I have been using Matlab a lot recently as it's object model is more suited to how I code using data objects. Now if only it had pointers :(

I promise this will be my last 'IDL objects' rant for a while.

Cheers,
Randall