Subject: Re: Does this make sense? (scalar objects)
Posted by JD Smith on Mon, 08 Dec 2003 18:34:13 GMT

View Forum Message <> Reply to Message

On Sat, 06 Dec 2003 01:33:07 -0700, Marc Schellens wrote:

> JD Smith wrote:
>> For objects, it's quite clear why you can't apply methods across a
>> vector of object variables:
>>
>> IDL> objs=[obj_new('IDL_Container'), obj_new('MyFooObj')] IDL>
>> objs->DoSomeMethod ; WRONG
>>
>> Since objects are generic pointers, and a vectors of objects can
>> contain any combination of object classes, it's clear why you can't use
>> this notation. The same is true of pointer arrays, for nearly the same
>> reasons:
>>
>> IDL> ptrs=[ptr_new('string'),ptr_new(indgen(5))] IDL> print,*ptrs+5
>> ;WRONG
>
> With the pointers it would be messy indeed (if your data is that uniform
> that such an expression would really make sense, use an array). Another
> thing is of course that there is no reason to not allow your example for
> a single element pointer array.
>
>
> With the objects though there would be no problem: Just let IDL call the
> appropriate method for each individual object. I even would think that
> this is more along the IDL array oriented way.
>
>

And what if all of the objects in the array do not implement the same
method, and what if the values they return cannot be concatenated into an
array (e.g. one returns a string, another a floating vector). I
originally was of your opinion, but have come to see how painful things
could get.

>> Single element vectors are different than scalars in several ways: they
>> can be transposed, reformed, and rebinned, whereas scalars cannot, and
>> they can have matrix multiplications applied to them, etc. A better way
>> of asking the question is "What can't you do with scalars that you can
>> do with vectors?". The answer to this consists of the long list of IDL
>> vector operations discussed here daily. There may not be any *useful*
>> distinctions between scalars and single-element vectors, but there are
>> certainly plenty of programmatic distinctions, which would break
>> backward compatibility if ignored --- hence, we are stuck with both.

>
> As I said, I agree that the cannot be abolished, but if from now on
> scalars could be transposed, rebined, etc. (and reffering to my OP:
> method called on single object arrays), This would not break any
> existing code, would it?

It's tough to say... probably none of my code, but I'm sure there are examples where the very inability to treat a scalar like a vector is capatalized upon.

JD