Subject: Re: IDL 8.0 compile_opt changes
Posted by JDS on Tue, 05 Jan 2010 22:41:09 GMT

While I long ago converted to [] for array subscripting for the reasons most succinctly expressed by Wayne, I have mixed feelings about converting the method invocation operator from '->' to '.' purely for cosmetic reasons. Problems I see with this:

1. Overloading structure dereference and method invocation breaks the ability to semantically parse a.b.

With this overloading, it is impossible to determine if 'a.b' is a method procedure call, or a structure field dereference, *except at runtime in the IDL interpreter*! For example, in IDLWAVE you can a->b [M-Tab] and have all "b..." procedure methods completed, or c=a->b[M-Tab] for function methods. Though I'm not sure, I suspect this would have a similar impact on the Workbench: loss of edit-time or shell-interaction-time differentiation among structure fields/object methods/etc through direct inspection of the source.

2. It will *not* be immediately obvious, or *ever* obvious, to a human observer that code which includes statements like a.b(c) must be compiled with IDL8.0 to run correctly. Consider a simple function found deeply buried on disk:

```
 function do_something, input
   return, input.something_else(1)
 end
```

This small function would compile equally in IDL 8 and IDL <8, but have a totally different meaning depending on what INPUT was passed in which version. You can make compile_opt idl2 the default in IDL 8, but this does little to relieve this issue, since older versions of IDL will compile this fine (and choke horribly if passed an object).

3. IDL is not Python. IDL enforces strict encapsulation of object data, i.e. all object data must be accessed through a method (except within the object's methods themselves). Python has no object data encapsulation. In Python it is natural to mix method invocation with data access. In IDL this only occurs only in an object's own methods. Which is clearer?

```
 self->limit, self.limit

 self.limit, self.limit
```

Just my $1D-2. (BTW, I think negative indexing sounds great!).

JD
