Subject: Re: IDL's handling of LOGICAL quantities (WHERE)
Posted by Mark Fardal on Tue, 12 Oct 1999 07:00:00 GMT
View Forum Message <> Reply to Message

James Tappin <sjt@star.sr.bham.ac.uk> writes:

> \begin{rant}
> I've finally decided to have a public moan about one of the weaknesses of=
> IDL's
> handling of logical operations: to boot -- that the WHERE function follo=
> ws
> a C-like interpretation while most other things are Fortran-like.
>
...
> IDL> print, where(a eq 0)
> 0 2
> IDL> print, where(not (a ne 0))
> 0 1 2 3
>
> I guess the proper answer isto have aproper logical or boolean type and
> functions like FINITE and logical operations should return it, and of cou=
> rse
> WHERE should accept it.

Actually, it would be fine if IDL were more C-like in this case. C doesn't have a boolean type either. But it does maintain a consistent definition of "true", which helps a lot. In C, 2.0 and 2 are both true. In IDL, 2.0 is true but 2 is false. Even worse, IDL conflates the boolean with the bitwise operators. AND, OR, XOR, and NOT are boolean operators for floating types and bitwise operators for the integral types. In C you don't have this problem, because there are different symbols for the boolean and bitwise operators. (Did I get that explanation right? It only took me eight years with IDL to figure out how it works...)

This illogical logic has been around for so long that I think we're stuck with it. But if anyone out there is writing an IDL replacement (hey Arno, you can handle this right?), this is my nomination for the second thing to fix.

Mark Fardal
UMass