Subject: Re: object suggestion for future IDL versions
Posted by davidf on Mon, 21 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

John Persing (persing@frii.com) writes:

> I don't know if anybody else has had this problem, but it occurs with
> objects that take advantage of inheritance. The object may take advantage
> of the inherited methods by using the notation
>
> self->method_name, arg1, arg2, etc
>
> But if the new object redefines any of the inherited methods (for example
> "assign"), it cannot make reference to the inherited method.

Not so. If a method is called on an object, IDL searches the
current class for a method of that name. Failing to find it,
IDL search all the inherited classes in the order in which they
are inherited for the method.

Providing a method of the same name of a superclass method
is called method overriding and is supported by the above
mechanism. But you can always call a specific superclass
method by directly specifying the superclass name in the
method call.

> PRO dat_new__define
> struct = {dat_new, time_stamp:'', INHERITS dat_old}
> END
>
> can define our new object. Ideally, I think, we would want to extend the
> assign method like this
>
> PRO dat_new::assign, field_string, value
> self->assign, field_string, value
> self.time_string = STRING(SYSTIME())
> END

You have it almost right. But the ASSIGN method should be
called like this:

```
self->DAT_OLD::assign, field_string, value
```

Cheers,

David
--
David Fanning, Ph.D.

Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

[Note: This follow-up was e-mailed to the cited author.]
