Subject: Re: a plea for more reliable mathematical routines
Posted by davidf on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Arno (fruncan@zedat.fu-berlin.de) writes:

> I definitely disagree. It is inferior to Java, Python, C/C++ (if You're able to
> program a little bit of OpenGL and Motif yourself) to name only some, far too
> expensive, introducing new bugs with every release (maybe a merger with Micro$
> would be adequate), lacking hooks for any reasonable development environment (or
> have You ever managed to get it to work with Rose or SNiFF+ to name only a few).

Of course it is inferior to Java, Python, C/C++. These languages (with the exception of C, which is what IDL is written in) didn't even exist when IDL was written. You would hope that new languages would be better than old ones. But do you re-write *your* programs every time a new, better language comes around? I sure don't.

I don't even have a clue what Rose or SNiFF+ are, and I barely know anything at all about Java and Python. And that is part of my point. I spend a lot of time with people struggling to learn IDL. I'm sure they (like me) look at your alphabet soup of new languages to learn and think "Right. All that just to get a line plot!?"

Yeah, OK, if you know Java and Perl and can throw in a little Motif programming so you could get some simple graphic on the display, maybe you can do something better than IDL. (Although heaven help you if your boss suddenly decides the whole mess should be ported to the Mac.) If so, I'm all for it. Go for it.

But my point is that even some no-account programmer like me can take IDL and figure it out well enough in a short amount of time to make a handsome living. I'm pretty darn sure that wouldn't have happened if I would have chosen Python or C++ as my language of choice.

And I've noticed that anyone who can mention five programming languages in the same sentence rarely likes IDL. Too simple, too high level, too "non-programmer" orientated. Too true. But that is *exactly* why it appeals to me and my friends. :-)

> Secondly, I definitely did not characterize objects as childish but the way
> they're used and implemented in IDL (look folks, now we're object oriented !).

No, I suspect you are all for objects, as any thinking person would be. :-) What you object to (pun intended) is that IDL doesn't look like C++ or Java. It's a valid point. Or

at least it *would* be if we were talking about a language
that had been written recently. But we are talking about
a language that is 16 years old!

I mean, honestly, that fact that IDL is still selling as well
as it does is not a testament to what a great language it is.
It is a testament to how hard it is to write something like
it that can beat it in the marketplace. Software like IDL
is not expected to live for 16 years! The life span of almost
any software program is surely limited to single digits,
just *because* new programming languages come along that
offer new, more powerful features.

I think the fact that something remotely *like* objects can be
grafted onto IDL in such as way as to greatly extend the
power of the language is remarkable. I wouldn't have
expected it, and I'm grateful to have it, even if it
isn't implemented perfectly.

I've no beef with the people who want accurate numerical
functions and software that works like the documentation
says it should work. I think this, rather than new features,
should be the primary focus at RSI, as I've told them
many, many times. But I have little patience with people
who complain that IDL isn't like this or that. No, it's not.
And it is not ever going to be like this or that. Not until
somebody in a garage somewhere decides that they are going
to take the very latest, most powerful language and build
the whole damn thing over again from the ground up.

Somebody has to be looking at the ol' man and thinking
"I can do better than that." Perhaps that somebody is
you, Arno. If so, sign me up for the first shipment.
But in the meantime, I'm going to forego the alphabet
fog and write myself an IDL program.

Best Regards,

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