Subject: Re: IDL alternatives?
Posted by patterso on Wed, 31 May 1995 07:00:00 GMT
View Forum Message <> Reply to Message

Charles Cavanaugh (cavanaug@uars1.acd.ucar.edu) wrote:

: I will grant you that lack of an executable creator tool is a major deficit, but
: it is my opinion that of all the other deficits you mentioned, none are problems
: for competent, thorough programmers and engineers.

I agree. But not everybody using IDL is a trained programmer.
Tools like these would help.

: Maintaining poorly written
: code is a problem in every language, and nothing (again in my opinion) specific
: to IDL makes this problem worse.

Agree. But some of the most powerful feautures of IDL
also allow you to write bad code very easily if
you are not careful. I wasn't trying to single out IDL in
this respect. I know it is easy to write bad code in C or
whatever language.

: And as far as a lack of an integrated environment,
: this is almost a given in the UNIX world.

I love these sorts of comments :) I moved into Unix from VMS and
when I complain that I miss some of the integrated tools that
VMS had, I always get "that's the way it is in Unix" answers.
Just because they don't exist, doesn't mean there isn't a need
for them. Your Fortran 90 and Lisp examples show that :)
(But I don't want to start any "my OS is better than
your OS" flame wars here)

: IDL is also the best tool I've ever used for fast data-visualization prototyping. I have
: used Fortran and NCAR Graphics, but what took me a day with those languages takes me
: only a few hours with IDL.

Agreed. But because it is so easy to quickly develop applications,
it also means that many "quick hacks" become part of bigger and
bigger programs. I would guess that most IDL applications
are not developed under strict software control, etc, but just
"grow".

: Sorry about my long diatribe, but I'm still not convinced that IDL is the black
: sheep of the computer language family.

I don't think IDL is the black sheep either. But I don't necessarily
agree that because it has the same short-comings as other languages,
that these short-comings should be accepted.

At the very least, I think IDL needs a more useful debugging
environment.

Tim Patterson

Subject: Re: IDL alternatives?
Posted by cavanaug on Wed, 31 May 1995 07:00:00 GMT
View Forum Message <> Reply to Message

In responding to an earlier post of mine, Tim Patterson wrote :

> I wonder if he was referring not to the actual language itself, but to the
> lack of development tools for IDL? There is no debugger for example. No
> integrated environemnt. No way to compile stand alone code. Perhaps
> this is what he meant by the programmer interface? (It's how I
> interpreted it anyway).
>
> As somebody who has often had to adapt and update other people's IDL/PV-WAVE
> code, I know how difficult it is to write easily maintainable code in these
> languages. And there are no tools for this purpose. Even the emacs IDL
> mode is not officially supported. This can be a real probelm when
> trying to develop and debug large systems written in IDL/PV-WAVE.

I will grant you that lack of an executable creator tool is a major deficit, but
it is my opinion that of all the other deficits you mentioned, none are problems
for competent, thorough programmers and engineers. Maintaining poorly written
code is a problem in every language, and nothing (again in my opinion) specific
to IDL makes this problem worse. And as far as a lack of an integrated environment,
this is almost a given in the UNIX world. Also, have you ever tried debugging
Fortran 90 code? I know of no robust UNIX Fortran 90 debugger. My co-workers and I
have no way of looking inside our structures, following our pointers, etc., yet we
still have developed tons of good scientific research programs. And what about
debugging LISP code? Granted, I have not programmed in LISP for about 3 years, but
back then we did not have a LISP debugger available to us. We learned that a
good design is the best debugger. And not knowing PV-WAVE (though I wish I could
try it out for a while), there is no better way (again in my opinion) than using IDL
to create event-driven, point-and-click interfaces to our large data pools. IDL
is also the best tool I've ever used for fast data-visualization prototyping. I have
used Fortran and NCAR Graphics, but what took me a day with those languages takes me
only a few hours with IDL.

One last point : as far as all the "How come IDL wont work with X machine, running Y system, when I try to do Z" questions (which I do also post), the volume of these posts is in my opinion comparable with the other comp.lang groups that I read. And until these language and compiler companies stop with the develop-and-screw mentality, these sorts of posts will continue, ad nauseum. (I would not include GNU, NCSA or most other freeware providers in the above group)

Sorry about my long diatribe, but I'm still not convinced that IDL is the black sheep of the computer language family.

Charles
--
Charles Cavanaugh | "Words are very unnecessary, they can only do harm"
cavanaug@ncar.ucar.edu | - Depeche Mode
NCAR Boulder, CO, USA | "Facts all come with points of view"
My opinions | - Talking Heads

Subject: Re: IDL alternatives?
Posted by patterso on Wed, 31 May 1995 07:00:00 GMT
View Forum Message <> Reply to Message

Charles Cavanaugh (cavanaug@uars1.acd.ucar.edu) wrote:
: In article <3qh748$kc2@nntp.Stanford.EDU>, zowie@banneker.stanford.edu (Craig DeForest) writes:

: >Well, I got tired of buggy behavior from my old copy of IDL (3.5.0 for
: >Ultrix) (it's dumping core again), and called RSI to get a price on
: >the update to 4.0. I've been using IDL for about six months, enough
: >time to be excited by the functionality, and horrified at the 1970s
: >programmer interface.

: I am probably opening myself up for a resounding flaming, but I just
: could not help myself . . .

: I read in various places (Mr. Deforest's above posting being one) about
: how IDL's API is so-o-o-o horrible. I do not understand this. To me,
: IDL seems like a Fortran 90 - Pascal morph, with dynamic typing,
: automatic variables, automatic garbage collection and a useful event-
: driven paradigm all thrown in the mix.

I wonder if he was referring not to the actual language itself, but to the lack of development tools for IDL? There is no debugger for example. No integrated environemnt. No way to compile stand alone code. Perhaps this is what he meant by the programmer interface? (It's how I interpreted it anyway).

As somebody who has often had to adapt and update other people's IDL/PV-WAVE

code, I know how difficult it is to write easily maintainable code in these languages. And there are no tools for this purpose. Even the emacs IDL mode is not officially supported. This can be a real probelm when trying to develop and debug large systems written in IDL/PV-WAVE.

Tim

Subject: Re: IDL alternatives?
Posted by cavanaug on Wed, 31 May 1995 07:00:00 GMT
View Forum Message <> Reply to Message

In article <3qh748$kc2@nntp.Stanford.EDU>, zowie@banneker.stanford.edu (Craig DeForest) writes:

> Well, I got tired of buggy behavior from my old copy of IDL (3.5.0 for
> Ultrix) (it's dumping core again), and called RSI to get a price on
> the update to 4.0. I've been using IDL for about six months, enough
> time to be excited by the functionality, and horrified at the 1970s
> programmer interface.

I am probably opening myself up for a resounding flaming, but I just could not help myself . . .

I read in various places (Mr. Deforest's above posting being one) about how IDL's API is so-o-o-o horrible. I do not understand this. To me, IDL seems like a Fortran 90 - Pascal morph, with dynamic typing, automatic variables, automatic garbage collection and a useful event-driven paradigm all thrown in the mix.

OK, so maybe you have to specify a continuation character, but in C you have to suffix lines with a ';'. To be honest, I would rather put a '$' at the end of the few lines I continue than put a ';' at the end of nearly every line. But here I am digressing. (Let's not get started on RSI's business practices.)

My point being : it aint LISP, it aint object-oriented, but it does well (mostly) what it was designed to do.

But maybe I am in the dark about this whole 1970's interface (I have only been programming since 1989), and I am always open to change. So if you (or another API slammer) could show tangible evidence that IDL's programming interface is a lava lamp or mood ring compared to C's video-conferencing or big-house-with-no-backyard, I will take back all that I have said, and jump on [insert language X here]'s bandwagon.

Charles
--

Charles Cavanaugh | "Words are very unnecessary, they can only do harm"
cavanaug@ncar.ucar.edu | - Depeche Mode
NCAR Boulder, CO, USA | "Facts all come with points of view"
My opinions | - Talking Heads