Subject: Re: idl 5.3 runtime on idl 5.2 system work?
Posted by Liam Gumley on Thu, 23 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

wbiagiot@suffolk.lib.ny.us wrote:
> I wouldn't have expected that. While that may be periodically necessary
> for RSI, it is bad news for customers. We are currently in the process
> of accelerating sales of our IDL application to military customers (with
> IDLRT 5.2). Maintenance is currently not included because it would make
> for a configuration nightmare having to bi-yearly upgrade our in-field
> platforms. This SAV file format change now precludes us from taking
> advantage of any new 5.3 development environment enhancements and new
> bug fixes for older problems.
>
> I would submit to RSI that major 6 month upgrades are *NOT* always a
> good thing and that backwards compatibility be maintained AT LEAST in
> the major revision class (i.e. IDL4.x, IDL5.x, IDL6.x).
>
> Liam, thanks for the info - I stand corrected. I assumed that since
> 5.2.1 apps run on 5.2.0 RT that some backwards compatibility existed.
> My boss is going to be real happy with this tidbit of information.

Bill,

The following is extracted from the IDL 5.2 online help for SAVE:

"The SAVE procedure saves variables, system variables, and IDL routines in a file using the XDR (eXternal Data Representation) format for later recovery by RESTORE. Note that variables and routines cannot be saved in the same file. Note also that save files containing routines may not be compatible between different versions of IDL, but that files containing data are always backwards-compatible."

I distribute my SHARP application via SAVE files:
http://cimss.ssec.wisc.edu/~gumley/sharp/sharp.html

I've tried to make sure that SHARP supports IDL 5.0, 5.1, 5.2 (and now 5.3), and it means that I need to create a SAVE file for each version. This implies that I need to have 4 different licensed versions of IDL available, which is not always easy. I think it can probably be done without too much trouble in Windows (I had IDL 5.2 and IDL 5.3beta co-existing happily under Windows), but in UNIX I can tell you it's painful (licenses need to be changed, license manager re-started etc.).

However I think a more difficult problem is taking account of the differences in the IDL language in the different versions, even in IDL 5.x (I've given up supporting IDL 4.x). Some examples:

- The SIZE keywords DIMENSIONS, TNAME, N_DIMENSIONS were introduced in 5.1,
- The DEVICE keyword GET_VISUAL_DEPTH was introduced (but not documented) in 5.1,
- The DEVICE keyword GET_DECOMPOSED was introduced in IDL 5.2,
- The QUERY_* functions (e.g. QUERY_GIF) were introduced in IDL 5.2,
- The READ_IMAGE function was introduced in IDL 5.3.

All these enhancements are useful (I use them often), but if I want my code to be backwards compatible within IDL 5.x, I must choose one of the following approaches:

(1) Use any and all functionality in a particular version (e.g. IDL 5.2) and don't support users with earlier IDL versions (they must upgrade),

(2) Don't use any new functionality beyond a particular version (e.g. IDL 5.0), which requires all kinds of workarounds. For example, see my IMDISP procedure which is designed to work with all IDL 5.x versions: http://cimss.ssec.wisc.edu/~gumley/imdisp.html

(3) Write wrappers which detect and account for version differences. I've done this sort of thing in my COLORINFO function: http://cimss.ssec.wisc.edu/~gumley/colortools.html

I think the best thing to do is pick a strategy for supporting IDL versions, and make sure your customers know well in advance if and when they will need to upgrade.

Cheers,
Liam.

--
Liam E. Gumley
Space Science and Engineering Center, UW-Madison
http://cimss.ssec.wisc.edu/~gumley