
Subject: How to create a stable "package" in IDL?

Posted by [Paul Van Delst\[1\]](#) on Wed, 24 Jul 2013 17:33:34 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello,

I have a bunch of IDL software used to process satellite data. Works great.

The problem is I'm a tinkerer. I'm always trying to either "improve" algorithms or make them run a bit faster. And that leads to slight differences in the results - which is generally o.k. for me as I can identify where/how the differences originated, but for other users that checkout our IDL codebase it can be a bit frustrating.

What I want to do is create a release of a *subset* of IDL software within our IDL repository. The problem I have is how to do that within the IDL directory structure. I can't guarantee that other users will have their IDL_PATH variable set such that they will get to the stable release directory first, as opposed to the still-in-development directory.

Does anyone with experience in both IDL and version control have a solution for this? I have to admit I'm experiencing a case of brainfade about it.

I thought of the compiled-language approach (i.e. build a library from a checked out tag and install in a default location), but then how would a developer switch between the stable and development codes without mucking about with IDL_PATH? (Which I would forget at some point)

Hopefully my question makes sense. Thanks for any info.

cheers,

paulv

Subject: Re: How to create a stable "package" in IDL?

Posted by [Phillip Bitzer](#) on Wed, 24 Jul 2013 19:43:33 GMT

[View Forum Message](#) <> [Reply to Message](#)

I typically use Git for this. (Used to use Subversion, but since all the cool kids use Git these days....here I am.) Easy to setup multiple branches. I typically have (at least) these branches:

- 1) master - used for stable releases
- 2) develop - where I tinker
- 3) rc - release candidate - when I think my tinkering is enough
- 4) bugfix??? - a branch for each bug. Gets merged into master.

The (bare) repo is on a server all interested parties can access. Developers have a choice to checkout whatever branch - their files are updated with that particular branch. No worrying about IDL_PATH.

End users are only supposed to checkout the master branch.

When I'm tinkering, I branch off the master into the develop branch and get to work. Sometimes, I'll even branch off the develop, depending on how nutty I'm feeling. Sometimes these changes get merged back into the master, and result in a new release. Other times, the changes weren't worth it and the branch is thrown away.

When I release a version of the code, I tag the master with the version number and release into the wild.

The IDL Workbench implementation of Git is rather limited (the version of eGit packaged is outdated), so I typically use the Workbench to keep an eye on which branch I currently have checked out. I use SourceTree to handle all Git push/pull/merge/etc in a GUI, although you could certainly do it at the command line.

Here's a typical workflow, starting from some stable release in the master branch:

- 1) Branch from master to develop. Start working on some algorithm improvement. Other developers can do the same thing.
- 2) After I'm happy with my tinkering, merge the develop branch to the rc branch. Let others that want to do some beta testing work checkout this out and use it.
- 3) Fix bugs in the rc (and maybe even in the master - which is still the previous release). When it's stable, merge this into the master branch. Tag it, and let end users know a new version is available.

Hopefully, this rambling is understandable. I know others around here use Git as well - maybe they can provide a different/better way....
