SVAC Tagging Strategy

I'm currently maintaing code tags and task versions as a single entity.

The downside to having the same version on all tasks is having a bunch of tasks that differ only in their name. The upside is that I don't have to keep track of which tasks have actually changed. Note that under the old scheme, changes in the setup script could necessitate new task versions even though the scripts run by that task had not changed.

The result of using the same code tag to cover all pipeline scripts is mainly to simplify directory structure.

Under the old scheme, $svacPlRoot looked like "/nfs/slac/g/svac/common/pipeline/Integration/svacPipeline", and that contained subdirectories "configReport/v3r1p0", "digiReport/v3r1p0", "digitization/v3r1p1", "eLogUpdate/v3r1p0", "lib/v3r1p1", "online/v3r1p0", "recon/v3r1p1", "reconReport /v3r1p1", "setup/v3r1p2", "svacTuple/v3r1p1", and links "lib-current -> lib/v3r1p1" and "setup-current -> setup/v3r1p2".
The links for the lib and setup directories were required because the wrappers had to find files in those directories before they could run the setup script, so they contained paths like "$svacPlRoot/lib-current". I had to keep track of which code needed new tags, install the new code in the correct locations with each update, and change the links to point to the new lib and setup directories.

Under the new scheme, $svacPlRoot looks like "/nfs/slac/g/svac/common/pipeline/Integration/svacPipeline/v3r1p3", and contains subdirectories "configReport", "digiReport", "digitization", "eLogUpdate", "lib", "online", "recon", "reconReport", "setup", "svacTuple", and no links. The wrappers contain paths like "$svacPlRoot/lib". This requires changing the value of svacPlRoot (currently an environment variable set in glast and glastdpf's .cshrc) every time the task versions change.

Reprocessing will work the same way under the new scheme as it did under the old: Make symlinks in the location that the new "downstream" task expects to find its input DSs, with names appropriate to the new task, that point to the output DSs produced by the old "upstream" task, and do a createRun on the new "downstream" task. For examples of reprocessing links created for old and new schemes, look at /nfs/farm/g/glast/u12/Integration/rawData/398000735 and /nfs/farm/g/glast/u14/Integration/rawData/398000571, respectively.

Every wrapper sources the script formerly known as $svacPlRoot/setup-current/svacPlSetup.cshrc, now $svacPlRoot/setup/svacPlSetup.cshrc. This script always changed any time any task changed, so the CCB was making an exception for it to avoid version cascades. That exception is not necessary under the new scheme.

The new scheme reduces the number of tags on the svac pipeline scripts, as I was maintaining a master tag on everything for convenience. This is now the only tag, the 10 confusingly-similar tags on the subdirectories are gone. The master tag for the last CCB round was prod-v3r1p2.