SCons and ASP

There a couple ways to adapt ASP to SCons:

- Make it into a new kind of "override" container ✅
- Make it a full-fledged container, just like ST and GR.

We plan to go the override route. It pins an ASP release to a particular ST release, as the current system does; that's Jim's preference.See issues encountered to date in this effort, particularly run-time issues, and conversion status.

Override

ASP is similar in all respects to containers ScienceTools, GlastRelease and CHS with one notable exception: it depends on ScienceTools, a full-fledged container, more or less as if it were an external library. When building or running ASP, one must have access to ScienceTools installed files (includes, libraries, python scripts). This is just the way an override directory behaves. The base installation is (or should be) read-only and packages in the override directory may access any of the installed products from the base directory, as well as files installed under the override directory.

In the "normal" use of override, all or most of the override packages are developer versions of packages which already exist in the base directory, but that's not a requirement. In the case of the ASP override directory there would be no packages in common.

So far we know of one detail in the way override directories are treated which doesn't have quite the right behavior for the ASP situation, but there is a simple way to extend the existing behavior to suit the new situation as well as the old. There may be other small glitches; we don't expect anything major.

What is where

The files comprising the SCons machinery are of different types and reside in different places:

1. **package-specific** For each package there must be a file called **SConscript**. For packages building libraries there must also be a file ***pkgName*_Lib.py**. Both are in the package root directory
2. **global, container-specific** The file **package.scons** and the file **externals.scons**, kept in the top-level directory of the container
3. **global, generic** These files are identical for all containers. Only a single copy is kept in CVS (under SConsFiles). This category includes the files **SConstruct**, **processExternals.scons**, **allExternals.scons** and several tools in site_scons/site_tools

Up till now, in the case where there is an override directory, all the global files come from the base, but if the override directory is container-like it might need its own copy of the category 2 files. ASP needs its own **package.scons** (which should override the ScienceTools one) but, as currently constituted, does not need its own **externals.scons**.

Modified ASP

CVS organization depends upon the approach chosen.

CVS organization (override)

Make an ASP-scons container package containing

- an ASP-specific **package.scons** file **Done** as of Nov. 3
- src directory with **mainpage.h**
- symbolic links to all code packages belonging to ASP (AspHealPix, AspLauncher, etc.) but no link to AspPolicy.

The job AspPolicy does will be handled some other way, perhaps involving adding code to **package.scons** or inventing a new top-level file.

CVS organization (complete container)

Make an ASP-scons container package containing

- **package.scons**
- **externals.scons** (must at a minimum reference all externals used by ScienceTools packages ASP requires)
- symbolic links to files in category 3 above (SConstruct, allExternals.scons, site_scons/site_tools/) as is the case for other containers
- symbolic links to all code packages specific to ASP and any used from ScienceTools

To-do

The steps are the same in either case, but details of #1 and #4 depend on approach chosen:

1. Implement appropriate CVS organization **Done** as of Nov. 16
2. Add SConscript files to all ASP packages **Done** as of Nov. 3
3. Add xxxLib.py files to all ASP packages building libraries **Done** as of Nov. 3
4. Design and implement SCons code equivalent to AspPolicy. **Done** as of Oct. 19; needs further testing.

Issues encountered

ASP versions of ST packages

asp_healpix, asp_skymaps and asp_pointlike are copies or near copies of similarly-named packages in ScienceTools (which may have evolved somewhat since the copy was made). Only the package name has changed. Internally include directories have the original name, libraries are built with the original names, etc. SCons can handle this as long as the version line in SConscript uses the new package name. GoGui is a little confused - the gui part of it expects the target name to match the directory name - but it can still be used to build these packages.

If the asp_ versions were expected to be permanent I would prefer to change internal names to match, but Jim thinks ultimately ASP might switch back to use the corresponding ST pacakges.

With a little work GoGui could probably be enhanced to handle packages with a different outer name more gracefully but this is not a priority.

Proposed action: None. We can live with the current arrangement, but be aware of the possible confusion. Confirm that the intended include file, library, etc. is being used in all cases.

Resolution: I made a small change to registerTargets so that packages can register their includes to a directory different from package-name. This makes it possible for GoGui to handle these packages more naturally

healpix, asp_healpix hidden includes

There are includes under healpix/src which, though not directly referenced as public include files, are themselves included by public include files. This causes problems when building with SCons since, when building a package, SCons expects mapall required includes not belonging to that package to be in the install area. skymaps needs access to such a hidden include. The work-around has been to add lines like the following to skymaps SConscript:

libEnv.Append(CPPPATH = ['#/healpix/','#/healpix/src'])

but this won't work for asp_skymaps since the # refers to the root of the ScienceTools installation (because that's where SConstruct is) and asp_healpix, which is what asp_skymaps should be using, lives under ASP.

Proposed action: Reorganize asp_healpix so includes needed by other packages can be installed. Perhaps do the same for healpix.(Joanne)

Resolution: asp_healpix has been reorganized.

_setup, wrapper scripts for supersede directory

It's not doing the right thing. It did once, I'm pretty sure, but probably not since I rewrote some of the relevant stuff.

Proposed action: Navid was working on a rewrite for other reasons; he will test in supersede situation. (Navid)

Resolution: As of mid-September, It's all working!

BayesianBlocks

This package, which builds a shared library and a python wrapper library, had only a minimal SConscript and no xxxLib.py. I made something up including these lines:ons

```
BayesianBlocksLib = libEnv.SharedLibrary('BayesianBlocks' ,
                                         ['src/Exposure.cxx','src/BayesianBlocks.cxx'])
swigEnv.Tool('BayesianBlocksLib')

lib_BayesianBlocks = swigEnv.SwigLibrary('lib_BayesianBlocks', 'src/BayesianBlocks.i')

swigEnv.Tool('registerTargets', package="BayesianBlocks",
             libraryCxts=\[\[BayesianBlocksLib, libEnv\]\],
             swigLibraryCxts=\[\[lib_BayesianBlocks, swigEnv\]\],
            includes=listFiles(\['BayesianBlocks/*.h'\]) )
```

This built without error and installed libraries libBayesianBlocks.so and lib_BayesianBlocks.so, but when running python/test_BayesianBlocks.py

1. failure of line **import BayesianBlocks**. There is a file build/<variant>/source/BayesianBlocks.py which is presumably the right file, but it isn't installed anywhere. Temporary work-around was to copy it to python directory
2. in file BayesianBlocks.py failure of **import _BayesianBlocks** Work-around was to change line to **import lib_BayesianBlocks**.
3. unable to find liboptimizers.so This is because of problem with _setup described above. The file exists in ST lib directory.

Proposed actions: Install file BayesianBlocks.py (and similar files created by Swig) in a suitable place, e.g. top-level python directory. Adjust naming of swig wrapper library so all references agree. (Joanne?)

Resolution: Done, but there still is a problem finding data/caldb/software/tools/caldb.config. Program looks relative to supersede root only. File exists under base (ScienceTools installation) directory.

Run-time issues

Jim discovered these while testing the first-attempt build.

commonUtilities

The caldb problem above is actually more pervasive. commonUtilities routines need to be enhanced for support of installations with a supersede directory.

Resolution: facilities-02-18-13 has a smarter commonUtilities which handles almost everything properly in case there is a supersede directory. The one exception is getPfilesPath. The interface needs to be changed so it can return a list. But it looks like no one actually uses this routine so it hardly matters how it's handled.

Wrong python in path

At run time python was resolving to default installation rather than "our" python. Jim tracked this down to unfriendly behavior of one of the set-up files asp_setup.sh calls: it resets PATH.

Resolution: Split asp_setup.sh into two pieces, one run before invoking _setup.sh and the other after. Tagged in SConsFiles-00-02-01.

Over-resolution of nfs path

In case an installation at SLAC included a supersede directory, environment variable for base directory if in nfs space was being resolved to something including a server name like "sulkey35". At a later date the server might no longer be serving that particular file so the path should be represented by something starting /nfs...

Resolution: Use utility supplied by Jim to translate back to /nfs... path. Tagged in SConsFiles-00-02-01

PFILES behavior

ASP jobs may run without an afs token in which case they can't write to $HOME. Old PFILES-handling code would make a pfiles subdirectory in $HOME if it didn't already exist. It's bad form for programs run in batch to access $HOME anyway, so Jim proposed eliminating all PFILES handling in _setup.sh for batch processes.

Resolution: Compromised by getting rid of references to $HOME for batch jobs. The PFILES path in this case will include pfiles directory of base installation, pfiles directory of supersede area (if there is one), prepended with process's original value of PFILES (if any). Tagged in SConsFiles-00-02-01, except for one last fix for .csh only which is committed but not yet tagged.