SCons and GR

GlastRelease Conversion to SCons: Status and Issues

Package Conversion

~~Five~~ ~~Four~~ ~~Six~~ ~~Four~~ Three packages remain to be converted to use newer tools (registerTargets rather than registerObjects). "Conversion" as used here also includes a re-examination of dependencies and other aspects of the package SConscripts generated by Emmanuel for a much older version of GlastRelease. ~~A few issues need to be addressed in order to complete this work:~~

- GlastClassify builds a static library and a guadi component which includes everything from the static. Requires some fancy footwork in one or both of the package SConscript and GlastClassifyLib.py. Done Nov. 20, tagged Dec. 7. ✓
- calibGenTkr depends on calibTkrUtil. The latter defines a macro for its source files which is then used in calibGenTkr. It's neither typical nor desirable to import source (.cxx) from one package into another. I'd like to find a way to avoid it. Done and tagged by Dec. 10 ✓
- ~~OnboardFilter will not build with current (SCons-specific) organization of the obf external.~~ As of Dec. 18, have a script to handle this. See details below.✓
- Overlay packages were added since Emmanuel's pass. Symbolic links within GlastRelease-scons hadn't even been made for them. All sym links, conversion, tagging are complete as of Dec. 7 for OverlayEvent, Overlay and overlayRootData.✓
- gui package probably needs some reorganization to build on Windows. Seems to be ok on Linux even though it's somewhat irregular.

obf External

The script offlineObf.py copies specified includes and libraries to an installation which may be used by SCons. It handles only Linux for now, but it shouldn't be too difficult to add Windows as long as the original build organization there is similar to that of flight builds on Linux. Here is the help output:

```
Usage: offlineObf.py [options] fsw_version
Options:
  -h, --help            show this help message and exit
  -p PKGLIST, --package_list=PKGLIST
                        file containing list of fsw packages (<mjr> <mnr>) and
                        what to do with them.  Default:
                        <fsw_version>_package_list.txt
  --input_binary=INPUTBINARY
                        input binary tag [default: linux-gcc]
  -c COMPILER, --compiler=COMPILER
                        compiler [default: substring of output binary tag
                        following final hyphen]
  --output_binary=OUTPUTBINARY
                        output binary tag [default: redhat4-i686-32bit-gcc34]
  --input_root=INPUTROOT
                        Input fsw build root [default: /afs/slac.stanford.edu/
                        g/glast/flight/ltx_root_<fsw_version>]
  --output_parent=OUTPUT
                        Output root parent [default: $GLAST_EXT/obf]
  --overwrite=OVERWRITE
                        Overwrite existing installation if any [default:
                        False]
```

The package list file used for B1-1-3 starts out like this:

```
# Input package list for offlineObf.py
# Format:
#   MAJ   MIN  VER   [inc] [src] [lib | (lib=LIBNAME)*]
#  [   ] signifies optional; | is "or";
# ( ) are just for grouping in the template and do not actually appear
#
#  inc  recursively copy include directory to new-include-root/MIN
#  lib  copy all libraries to new-lib-root
#  lib=LIBNAME  copy specific library  LIBNAME to new-lib-root
#  src  copy everything matching ..source/MIN/src/*.h to new-include-root/src
PHY  EFC  V4-3-0     inc src lib=efc
PHY  XFC  V0-1-2     inc
PHY  EDS  V2-9-1     inc lib
```

The package list file has to be hand-generated for each new Flight Software build.

To make this go, small changes were required to allExternals.scons (add an additional include directory for the obf external) and processExternals (handle second include directory when there is one).

Update: As of June 22 have corresponding script for Windows. Probably the two can be combined with judicious use of

```
    if sys.platform == "win32":
```

Gaudi Component Libraries

Windows builds

On Windows Gaudi component libraries are (that is, should be) built differently from non-Gaudi dlls. For the non-Gaudi dlls only the link command must be preceded by a step to generate exported symbols. It is neither necessary nor desirable for Gaudi component libraries. Current SCons code does not provide a way to make a distinction; the symbol generation step will happen in all cases. Work required includes

- modifications to processExternals to support both kinds of builds
- modifications to registerTargets, makeStudio and our private msvs to provide a way to ask for the appropriate kind of build and to get necessary information into project files
- trivial modifications to SConscript files for those packages building Gaudi component libraries

Dependencies

Since Gaudi component libraries may be loaded dynamically, test programs in other packages may require libraries at run time which are not needed at link time. Whether or not they are required at run time can, for example, be a function of job options. Should such dependencies be known to SCons? I've discussed this with Tracy and Leon and believe the following satisfies everyone's needs:

- do not make such a dependence on component libraries for most packages.
- Gleam *will* depend on all the component libraries (as well as all other package libraries). As a side-effect, the created Gleam solution file will reference all library project files.

Job options

The current arrangement causes most programs dependent on job options to fail because the files can't be found. They need to be *installed:* that is, copied to a top-level directory as is now done for xml files, include files, data files, python files, etc. Because there is a lot of duplication of names, there should be per-package subdirectories as is the case for include. Work required to implement this includes

- modifications to registerTargets and other tools to recognize the new category of installable files (easy) Done Jan. 7, 2010; committed Jan. 13.; since tagged ✓
- modifications to package SConscripts needing to install such files (easy but tedious because of the number of packages involved). Also add a line to define compile-time macro PACKAGE_NAME. Done and committed for all GR packages as of June 11, 2010. As of June 25 almost all are tagged.✓
- modifications to Pipeline code?
- mods to job options files so all job options files can be located at run time. Committed as of June 11, 2010; mostly not tagged
- (maybe) modifications to TestMain to take care of standard simple test programs. Done Jan. 7, 2010. Committed, tagged as of June 11, 2010. ✓
- modifications to facilities::commonUtilites to define per-package job option path variables, e.g. GLASTSVCJOBOPTIONSPATH. Done Jan. 7, 2010; committed Jan. 14; since tagged. ✓
- modifications to GlastRelease package.scons, changing build procedure for Gaudi programs slightly. Done Jan. 7, 2010; committed Jan. 13. Various updates to GR externsals.scons also. Committed and tagged as of June 11, 2010 ✓

Most likely the above will not cover everything. Package maintainers would be responsible for the rest.

June 11 update: there are indeed other loose ends. Some have been dealt with; others require more general discussion, perhaps. Various files needed at run time which cannot be installed nicely by SCons because they aren't currently in the "right" place, for example the file G4Generator/src/test /test_sources.xml, which several jo files refer to. I'd like to move it to G4Generator/xml, G4Generator/data or a subdirectory, then let SCons install it. Because of this and similar problems (another example: ideal Cal calibration files which are in CalXtalResponse/src/test instead of under xml or data), test_Gleam can't yet run. However all of GR builds on rhel4 with SCons and some test programs **do** run, so the new jo architecture appears to be ok.

External Libraries

Certain external libraries are not yet available for vc90.

Several small Windows-specific changes had to be made to allExternals.scons

Outstanding Issues on Windows

Gaudi is apparently built non-debug. GlastSvc - and very likely other packages - returns memory allocated by Gaudi or vice versa. This means that on vc71 we must exclude msvcrtd.