Proposal for obf extlib Organization

Requirements

1. For Linux, prefer to derive automatically from Flight Software release
2. Produce identically-structured installed form on all supported operating systems
3. Has to support dynamic loading of certain shareables by flight software code. These shareables may have duplicate names.
4. Minimize link time and complexity for applications linked against obf
5. Minimize distributed size

Release Build vs. Installation

The current organization of the obf external library is identical to that used by Flight Software. This is a necessary stage in producing an installed version, but there are drawbacks to using it as ***the*** installed version (see 4. and 5. above).

The fsw external library (or, rather, the procedure - hereinafter called "the script" - which produces it) uses the flight software build as a starting point but copies only files which are needed to build applications against the library, and changes the organization to simplify linking by putting all shareables in a single lib directory. This won't quite work for obf because of the (small number of) shareables with duplicate names, but the idea is to produce something very similar for obf from a full (flight software-style organization) build of needed packages. It also omits some include files needed to build GlastRelease because, in the flight software organization, these files are buried under source directories. So, roughly the procedure on Linux would be

- run enhanced version of the script used now to produce the fsw external library. Its inputs would be, approximately
    - list of packages of interest producing "regular" libraries
    - list of packages (and versions?) of interest producing libraries which may be dynamically loaded
    - list of packages (and versions?) with include files which have to be copied
    - pointer to flight software release build

On any other OS

- create build with organization identical to flight software release builds
- run script as above (ideally the identical script) but last input would be pointer to the created build

To do

- Determine what file organization is required by Flight Software to dynamically load libraries. Is it identical to release build organization?
- Determine packages which have to be treated specially because of libraries or includes.
- Upgrade script now used to generate the fsw external. ~~Bryson is willing to look into this.~~ [No longer an option, unfortunately.]