Tags and Defining a Build

Current Practice

RM's determination of what to build is rooted in CVS tags:

LATEST builds are triggered when the RM detects a new package tag. For each watched container which includes the changed package, a new LATEST tag (e.g. GlastRelease-LATEST-1-4765) is added, package by package, to the latest package tags of all packages belonging to the container.

HEAD and release builds are triggered when a tag of the respective proper form (e.g. ScienceTools-HEAD-1-894 or ScienceTools-09-23-01) is detected and there is no existing RM build for it. Such tags are made by invoking the script tagCollector.py.

tagCollector.py logic

To create a release tag for a container like ScienceTools, tagCollector determines the most recent HEAD tag and adds a new tag to the same set of files, like this:

```
cvs rtag -r ScienceTools-HEAD-1-nnn ScienceTools-xx-yy-zz ScienceTools-scons
```

Support is envisioned but not yet implemented for tagging along branches as well.

The procedure for creating a new HEAD tag is more complicated because the caller specifies packages to be addeed, removed or upgraded, but it also involves adding a tag "all at once" to a collection of files which already have some other tag.

Why This Isn't Good Enough

On occasion CVS commands fail in such a way that they leave locks in the repository which can interfere with later operations. This might be related to the more pernicious problem of incomplete tags, which can cause obscure build failures.

Options for Improvement

1. Leave procedures as they are. Work on automating detection and possibly repair of stale CVS locks and incomplete tags.
2. Modify creation of HEAD and release tags to go package by package. The description of a HEAD or release tag is based on a list of package tags; the tag itself is then a derived quantity. The primary source for the list of package tags may be
    a. in a file, e.g. packageList.txt OR
    b. in the RM database
3. Get rid of container-wide tags altogether, at least for HEAD and releases. For container check-out, write script which can interpret list of package tags (in form of file or db entries) belonging to the build and check out files accordingly. Would also need a new trigger for RM to build the new HEAD or release tag.

My preference is for some form of **2. 1.** doesn't attempt to address the root of the problem. **3.** involves possibly significant changes to RM operation (but I defer to Tom as to whether this is a real concern).

LATEST tags appear to be much less subject to problems than HEAD. I suspect this is because the LATEST tags are made bit by bit. At the very least, the SConsFiles part of each container, in particular the SConstruct file, should receive its HEAD or release tag last to minimize contention. (RM is watching SConstruct to determine when there is a new tag to be built. When it finds one, it submits several jobs which will do check-outs. Check-outs require a read lock; tagging requires a write lock.)

2.b. is probably more robust than 2.a but also more work. Either tagCollector.py has to be modified to access the database or the currently-disabled tag collecting in RMViewer has to be revisited.