CVS HOW-TO Series 1

How to maintain external software packages

Outline

- Initial Import and Documentation
- Managing updates from the 3rd Party package/module
- Collaboration and sending value-added changes upstream
- Roadmap forward and What next?

Initial Import and Documentation

- Create a file called “External_EPICS_MODULES”
  - Contents:
    - A list of all external software modules containing:
      - Package Name
      - Location (e.g. URL)
      - Author
      - Local maintainer/contact
      - Dependencies
- This file will be kept in CVS and available on the web.

Initial Import and Documentation (2)

- Initial Import:
  - cvs import <loc_in_repo> <vendorTag> <releaseTag>
    - For Vendor Tags I propose the following standard:
      - vendorTag = vendorName-packageName
      - Examples: **ANL**-asyn, **ANL**-motor, **ANL**-autosave **PSI**-streamdevice, **CARS**-ip330-asyn, **SLAC**-ChannelWatcher
    - For Release Tags I propose the following standard:
      - ReleaseTag = packageName-version
      - Examples: asyn-**R4-16**, motor-**R6-5**

Initial Import and Documentation (3)

- Note the vendorTag is very important as we will use it to make future imports when updates occur for external package. I call the vendorTag our merge anchor.
  - All successive imports will use the vendorTag
  - Example: Here is what I did for the asyn package:
    - Checkout the HEAD and perform a cvs log operation to determine the vendorTag.
      - cvs log Makefile |grep symbolic:
        - VendorTag is always the intial or very first symbolic tag:
        - ASYN: 1.1.1
        - VendorTag = ASYN

Initial Import and Documentation (4)

- Example for asyn continued:
  - Now we import the newly updated asyn package into our local CVS Repo:
    - cvs import -m "initial import of asyn R4-16 from ANL" **epics/site/src/asyn ASYN asyn-ANL-R4-16**
    - **104 conflicts created by this import (Oh Oh, what next?)**
    - **Well now we will have to some merge operations:**

Merging Updates (1)

- Example for asyn continued:
  - Strategy to reduce the number of conflicts merge against the most recent release tag in your local Repo:
  - cvs checkout -j <prev_rel_tag> -j asyn-ANL-R4-16 epics/site/src/asyn
  - Change directory to another sandbox
  - cvs co -d Merger -j asyn-R4-9-lcls6 -j asyn-ANL-R4-16 epics/site/src/asyn
  - cd Merger
  - cvs -n update
  - Now go ahead and commit all files that have the "M" status
  - Be sure to commit the Added files as well "A" status
  - Next make a decision on any removed files "R" status
  - Finally, resolve any files with conflicts "C" status

Merger Complete

- Asyn Example Continued:
  - I recommend doing a diff against the merged module and the original downloaded external package. The diffs should be very small and show only your local changes. This should give confidence that the merger was successful.
  - Also, do some testing and make sure that everything compiles.
  - Now, let's put our Production releaseTag in place and release the package for consumption first to our DEV then to PROD.
  - cvs tag **asyn-R4-16-lcls1**

Collaboration

- Site Specific local changes related to build and deployment environment; stay de-coupled from the third party package.
- Bug fixes and enhancements should be submitted upstream to author for future releases.

What Next?

- Document the Process as a standard.
- This process should be version control software agnostic.
- The documentation for how to maintain third party software should be placed on the web and kept up to date.
- Any other ideas here?
- What about Branching?