Trilinos Release Improvement Issues

Roscoe A. Bartlett

Department of Optimization & Uncertainty Estimation
Trilinos Software Engineering Technologies and Integration Lead

Sandia National Laboratories

Trilinos User Group Meeting, November 5, 2009
Trilinos Release Improvement Issues

• Branch early vs. branch late

• Release-related testing

• Improved release processes

• Improved release-related activities

• Managing late release branching

See:
Trilinos/doc/DevGuide/TrilinosSoftwareEngineeringImprovements/*.tex
Branch Early vs. Branch Late

• Last minute changes before a release *always* happen:
  • End of FY deliverables
  • Porting work
  • “Cleaning up” (code, documentation, etc.) ..

• Reasons to branch as late as possible:
  • Difficulty merging changes between branches
  • Results in reduced testing of all types (APP Trilinos Integration, nightly testing, etc.)

• Reasons to give some time between branch and release dates:
  • Buffer for portability problems (3 days should be plenty)
  • Allow for new non-release development activity to continue?
    => No: These are rare and can be done on a temp local git branches

• Mitigation strategies for problems with late branching
  => Stay tuned …
Release-Related Testing

• Create release-like tarballs every night:
  • Mark non-release packages
    • Add a release/non-release column in TriinosPackages.cmake
  • Disabling and excluding non-release code
    • PACKAGE_EXCLUDE_FROM_RELEASE(...).
  • Do installation testing using release-like tarball
    • Untar release tarball (not from working dir)
    • Do installation testing procedure:
      • Build and install headers and libraries
      • Configure tests/examples against installed headers and libs

• Perform this testing every night
  • Different enable configurations
  • Just one platform should be enough

• Come release time you are ready to go!
Improved Release Process

• Avoid branching until just days before initial release (3-4 days)

• Minimize the changes on the release branch (just change one file)
  • Change Trilinos_version.h and that is it!

• Auto package versioning
  return "MyPackage in Trilinos " TRILINOS_VERSION;

• Minimize changes for minor releases
  • Major bugs only
Release-Related Activities

• Tasks to complete before the release branch is created
  • Implement all functionality for the upcoming release
  • Keep all documentation and examples for “up to date
  • Do all code “clean ups”
  • Define files/dirs to exclude from next release tarball,
  • Promote “Experimental” Code to “Stable” Code weeks before branch date
  • Add new test platforms weeks before branch date
  • Keep “Stable” code in a releasable state
  • Perform ports and acceptance tests with Trilinos Dev (APP tests).
  • Create the release branch only a few days (3 days max) before putting out release

• Tasks to complete after the release branch is created
  • Change the version in Trilinos version.h. (All other logic is automatic)
  • Fix serious defects only
  • (Optional) Final round of ports and acceptance tests against APPs
  • Create the final tag.
  • Release the code as the auto-generated tarball.
  • Fix other bugs and do minor releases
Managing Late Release Branching

• Incremental refactoring
  • Develop a new feature “under the hood”
  • Okay to release but users still will not use
  • See “Daily Deployment” in “Extreme Programming: 2nd edition”

• Disable new features
  • Just exclude the key files from the release tarball

• Temporary local git repositories
  • Big changes incompatible with the imminent release
Trilinos Release Improvement Issues

- Branch early vs. branch late
- Release-related testing
- Improved release processes
- Improved release-related activities
- Managing late release branching

See:
Trilinos/doc/DevGuide/TrilinosSoftwareEngineeringImprovements/*.tex