

# **Trilinos Framework: Update and Future Plans**

**November 8, 2007**

**8:30-9:30 a.m.**

**Jim Willenbring**

**Tim Shead**





# Overview

---

- **Using CMake to Build Epetra**
- **Merging Changes From a Release Branch**
- **Trilinos Tutorial Updates**
- **Source Control for Experimental Code**
- **Future Trilinos Releases\***
- **Local Management of Source Code\***
- **Trilinos Lifecycle Model\***
- **General Framework Discussion\***



**\*These are discussions that can be continued later in the day**



# Using CMake to Build Epetra

---

**Tim Shead will supply these slides**



# Merging Changes From a Release Branch

---

- In CVS, multiple merges from one branch to another cause collisions
- The `-j` option can be used to specify a beginning and an end point for a merge



# Merging Changes From a Release Branch

---

- For example:

**cv**s update -j merge-point -j trilinos-release-8-0-branch

**Will merge changes after the ‘merge-point’ tag, up until the most recent version of the ‘trilinos-release-8-0-branch’ branch**

- Don’t forget to create ‘merge-point-2’ on the ‘trilinos-release-8-0-branch’ branch for next time
- This technique may not work when adding new files

Thanks to Ross for pointing this out to me.



# Trilinos Tutorial Updates

---

- **The Trilinos Tutorial is incomplete**
  - **Some packages are not mentioned**
  - **An index of terms would be useful**



# Source Control for Experimental Code

---

- If you want to commit, but the code is unstable
  - **cv**s tag -b Anasazi\_Working\_20071030
  - **cv**s update -r Anasazi\_working\_20071030
  - **cv**s commit
- These commands
  - Create a new branch at the point of the last **cv**s update
  - Make the sandbox a copy of the new branch
  - Commit the changes to that branch



# Source Control for Experimental Code

---

- The changes will have to be merged back to the previous branch
- This technique allows a developer to commit changes, even at unstable points
- It is increasingly important not to purposely destabilize the HEAD branch as Trilinos grows

Thanks to Chris Baker for pointing this out to me.



# Future Trilinos Releases

---

- **We are still planning to release all of Trilinos**
- **Some packages need to release separately**
- **Should we consider logical meta-package releases?**
- **Which packages are planning to release separately?**



# Local Management of Source Code

---

- **Decentralized version control**
  - No central repository
  - Can be used locally and amongst small teams while continuing to use CVS
  - Examples include Git and Bazaar
  - Allows local commits
  - Can be adopted by any subset of the team



# Trilinos Lifecycle Model

---

- **Three phase meta lifecycle model**
  - Research
  - Production Growth
  - Production Maintenance
  - Two promotional events
- **Production Maintenance Phase is still just an idea**
- **What aspects of the model should we flesh out more?**



# General Framework Discussion

---