Trilinos Software Engineering Technologies and Integration Capability Area Overview

Roscoe A. Bartlett

http://www.cs.sandia.gov/~rabartl/

Department of Optimization & Uncertainty Estimation

Trilinos Software Engineering Technologies and Integration Lead

Sandia National Laboratories

Trilinos User Group Meeting, November 2, 2010

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories

Trilinos Software Engineering Technologies and Integration

- Numerical Algorithm Interoperability and Vertical Integration
  - Abstract Numerical Algorithms (ANAs)
  - Thyra (Interoperability and vertical integration of ANAs)
- General Software Interoperability and Integration
  - Memory management (Teuchos::RCP, ...)
  - User input and configuration control (Teuchos::ParameterList, ...)
  - User introspection (Teuchos::FancyOStream, ...)
- Skin packages (wrappers for other languages)
  - PyTrilinos, ForTrilinos, CTrilinos
- General Software Quality and Design
  - Separation of “Stable” vs. “Experimental” code
  - Day-to-day stability of “Stable” code
- Lean/Agile Software Engineering Principles and Practices
  - Internal Trilinos issues
  - External customer issues

Recent Trilinos Improvements of General Interest

- External repositories and add-on Trilinos packages
  - Allows users to add their own packages independently and use the Trilinos CMake/CTest/CDash system
- [Future] Generalize and externalize the Trilinos CMake/CTest/CDash system
  - Allow other projects to fully exploit the Trilinos SE infrastructure
  - Will be used by projects like NEAMS, CASL and perhaps others
- Regulated backward compatibility and Trilinos versioning
  - Deprecated warnings allow users to slowly refactor code
  - [Future] automated testing of backward compatibility
- Teuchos memory management classes:
  - Eliminate undefined behavior in C++ codes (single objects and arrays of objects).
  - www.cs.sandia.gov/~rabartl/TeuchosMemoryManagementSAND.pdf
- SIERRA Trilinos Almost Continuous Integration process:
  - Nightly testing (< 48 hour delay) of a lot of Trilinos (Teuchos through MOOCHO) on many platforms (GCC, Intel, AIX, Pathscale, PGI, etc.)
  - SIERRA takes snapshots of Trilinos for releases
- Greater Trilinos development stability:
  - Allow for daily integration testing and daily updating of customer APPs

External Trilinos Repositories and Add-On Packages

Example:

```
$ cd $TRILINOS_HOME_DIR
$ eg clone software.sandia.gov:/space/git/preCopyrightTrilinos
$ cd $BUILD_DIR
$ ./do-configure -DTrilinos_EXTRA_REPOSITORIES=preCopyrightTrilinos \
    -DTrilinos_ENABLE_Amesos2 …
```

After that, all of the extra packages defined in <EXTRAREPO> will appear in the list of official Trilinos packages and you are free to enable any that you would like just like any other Trilinos package.

For more details see:

$TRILINOS_HOME_DIR/cmake/TrilinosCMakeQuickstart.txt
$TRILINOS_HOME_DIR/cmake/HOWTO.ADD_EXTRA_REPO

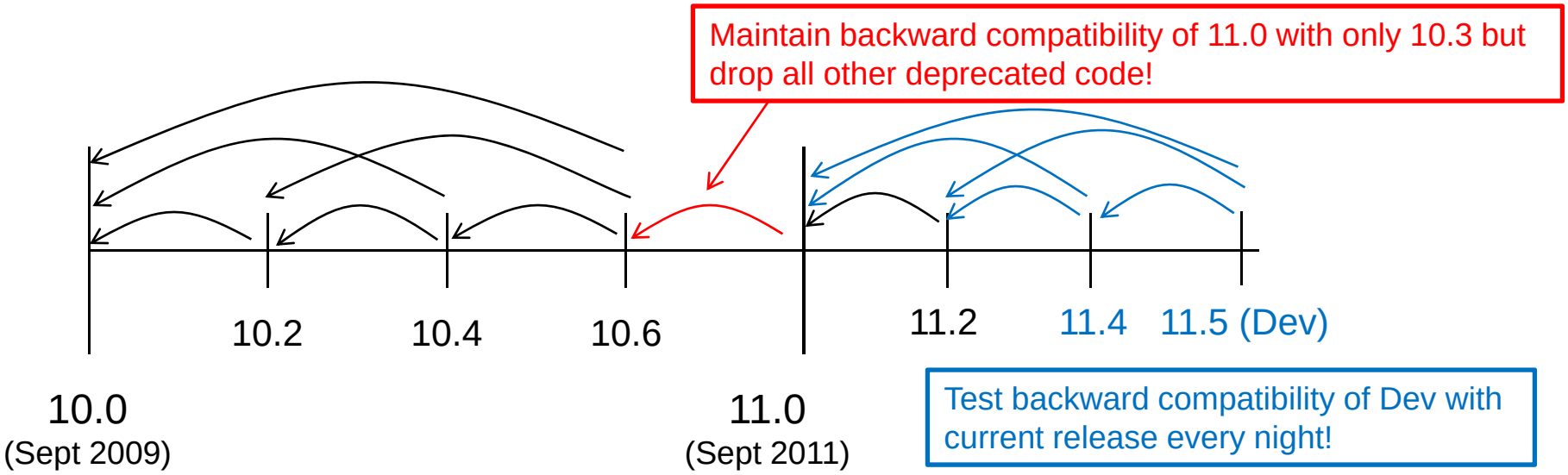
Sandia National Laboratories

Backward Compatibility Considerations

- Backward compatibility is critical for:
  - Safe upgrades of new releases
  - Composability and compatibility of different software collections
- Maintaining backward compatibility for all time has downsides:
  - Testing/proving backward compatibility is expensive and costly
  - Encourages not changing (refactoring) existing interfaces etc.
    - => Leads to software “entropy” which kills a software product
- A compromise: Regulated backward compatibility (Trilinos approach)
  - Maintain a window of “sufficient” backward compatibility over major version numbers (e.g. 1-2 years)
  - Provide “Deprecated” compiler warnings
    - Example: GCC’s __deprecated__ attribute enabled with –DTrilinos_SHOW_DEPRCATED_WARNINGS:BOOL=ON
  - Drop backward compatibility between major version numbers
  - [Future] Provide strong automated testing of Trilinos backward compatibility

Regulated Backward Compatibility in Trilinos

- Trilinos Version Numbering X.Y.Z:
  - X: Defines backward compatibility set of releases
  - Y: Major release (off the master branch) number in backward compatible set
  - Z: Minor releases off the release branch X.Y
  - Y and Z: Even numbers = release, odd numbers = dev
    - Makes logic with Trilinos_version.h easier
- Backward comparability between releases
  - Example: Trilinos10.6 is backward compatible with 10.0 through 10.4
  - Example: Trilinos 11.X is not compatible with Trilinos 10.Y



Example: Major Trilinos versions change every 2 years with 2 releases per year

Trilinos Software Engineering Capabilities Area Webpage

http://trilinos.sandia.gov/capability_areas.html