2008-7715P

CMake For Trilinos Developers

Roscoe A. Bartlett
http://www.cs.sandia.gov/~rabartl/
Department of Optimization & Uncertainty Estimation

Esteban J. Guillen
Department of Information Engineering

Sandia National Laboratories

Trilinos User Group Meeting, October 23, 2008

Overview of CMake

- CMake = “Cross-platform Make”
- CMake:
  - Build system primarily for C/C++ code
  - Front-ends to configure a software package
    - Command-line, Scripts, CURSES, GUIs
  - Back-ends that build code
    - Unix Makefiles, MS Visual C++ Projects, Eclipse Projects, ...
  - Packaging and installing
    - Tar/gzip, Windows self-extracting installers, PackageMaker, RPM, ...
- Platforms and usage:
  - Platforms:
    - Unix/Linux, MAC OSX, MS Windows, AIX, IRIX, ...
  - Internal Sandia use:
    - VTK/Titan, ParaView, ThreatView, ...
  - External use:
    - KDE, MySql, MiKTeX, (and many many more) ..

CMake is a full featured mature build system!

Sandia National Laboratories

Current Status of Trilinos/CMake

- Our detailed evaluation of CMake for Trilinos is finished:
  - Roscoe A. Bartlett, Daniel Dunlavy, Guillen Esteban, and Tim Shead. *Trilinos CMake Evaluation.* SAND2008-xxxx, October 2008
    - http://www.cs.sandia.gov/~rabartl/publications.html
- We have a nearly complete CMake build system design in Trilinos Dev
- Current CMake enabled packages:
  - Teuchos, RTOp, Epetra, Triutils, EpetraExt, Thyra, RBGen
- Trilinos community close to making a decision to move to CMake?

Gains & (Initial) Looses Switching to CMake for Builds

- What we gain:
  - Full dependency tracking of every kind possible on all platforms (i.e. header to object, object to library, library to executable, and build system files to all built files)
  - Support for shared libraries on a variety of platforms
  - Support for MS Windows (i.e. Visual Studio projects, Windows installers, etc.)
  - Simplified build system and easier maintenance (extremely easy to add new packages and maintain existing packages)
  - Improved mechanism for extending capabilities (as compared to M4 in autotools)
  - Ability to affect the development of the build tools with good existing collaborations (i.e. with both Kitware and with organization 1420)
  - Significant ``in house'' knowledge-base (i.e. visualization group in 1420).
  - One hundred percent automated intra-package dependency tracking and handling (built into the prototype Trilinos/CMake build system)

- What we lose (at least initially):
  - CMake requires that all uses have 'cmake' installed on their machine when building from source and users will need to have at a very recent version of cmake. (However, cmake is very easy to build from source)
  - Support for circular test/example and package libraries is not provided in the current prototype Trilinos/CMake build system

Gains & (Initial) Looses Switching to CTest/CDash for Testing

- What we gain:
  - Test time-outs (this is a major maintenance issue for the current Perl-based test harness)
  - Memory testing with Valgrind and purify that is backed up by Kitware and a larger development community
  - Line coverage testing that is backed up by Kitware and a large development community
  - Support for selecting and excluding subsets of tests based on regular expressions (but better support for keywords would be welcomed)
  - Better integration with the build system (e.g. easier to support more advanced features like PBS batch systems and flexible testing control)
  - Better tracking of specific tests (i.e. each and every test can have a unique name that is easy to find)

- What we lose (at least initially):
  - Separate reporting of test results for different Trilinos packages on the web page and in emails sent out (however, such support could be layered on top of CTest and CDash)
  - Support for selectively disabling package tests/examples and entire packages when a build fails (however, such support could be layered on top of CTest for driving the test harness)

Sandia National Laboratories

Design Principles for Trilinos/CMake Build System: #1

- Make it exceedingly easy to define CMake files for new packages and to define libraries, tests, and examples in those packages.
- Create a design for building individual package CMake files that automatically results in uniformity of how things are done. This is needed to support a number of important features and support maintenance. Use standard macros to define every package's main features to facilitate this.
- Allow changes to logic and functionality that apply to all Trilinos packages without having to touch each individual Trilinos package's CMake files.
- Provide 100% automatic intra-package dependencies handling. This helps to avoid mistakes, avoid duplication, and robustifies a number of important features.
- Provide built-in automated support for as many critical software engineering practices a possible. This includes proper and complete pre-checkin testing when continuous integration is being performed.

Sandia National Laboratories

Design Principles for Trilinos/CMake Build System: #2

- Avoid duplication of all kinds as much as possible. This is just a fundamental software maintenance issue.
- The build system should be able to reproduce 100% update-to-date output by simply typing make. We will endeavor to provide 100% correct dependency management in all situations (e.g. coping test input files to binary directory).
- Aggregate as much common functionality as possible to the top-level CMake files but allow individual CMake packages to refine the logic if they really need to.
- Where there is a tradeoff between extra complexity at the global framework level verses at the package level, we will always prefer greater complexity at the framework level where we can apply solid software engineering design principles to manage the complexity and spare package developers.
- Allow Trilinos packages that want/need to be built separately from Trilinos to do so but don't force this on all Trilinos packages.

Quickstart: Getting Help

```
(*) Getting CMake help

    http://www.cmake.org


(*) Viewing available configure-time options with documentation

  $ cd $BUILD_DIR
  $ rm CMakeCache.txt
  $ cmake -LAH $TRILINOS_BASE_DIR


(*) Viewing available configure-time options without documentation

  $ cd $BUILD_DIR
  $ rm CMakeCache.txt
  $ cmake -LA $TRILINOS_BASE_DIR
```

See:

Trilinos/cmake/TrilinosCMakeQuickstart.txt

Quickstart: Configuring, Building, Testing, Installing

```
#!/bin/sh
EXTRA_ARGS=$@
cmake \
  -D CMAKE_CXX_FLAGS:STRING="-g -O0 -ansi -pedantic -Wall" \
  -D DART_TESTING_TIMEOUT:STRING=600 \
  -D Trilinos_ENABLE_NOX:BOOL=ON \
  -D Trilinos_ENABLE_ALL_OPTIONAL_PACKAGES:BOOL=ON \
  -D Trilinos_ENABLE_EXAMPLES:BOOL=ON \
  -D Trilinos_ENABLE_TESTS:BOOL=ON \
  ... \
  $EXTRA_ARGS \
  ../../../Trilinos
```

```
$ ./do-configure –D VEROBSE_CONFIGURE:BOOL=ON
$ make –j4
$ ctest
$ make install
```

See example scripts:

Trilinos/sampleScripts/*cmake

Quickstart: Enabling Packages and Examples

```
(*) Configuring Trilinos to build all packages with all tests and examples:

  $ ./do-configure \
     -D Trilinos_ENABLE_ALL_PACKAGES:BOOL=ON \
     -D Trilinos_ENABLE_TESTS:BOOL=ON \
     -D Trilinos_ENABLE_EXAMPLES:BOOL=ON

  NOTE: Specific packages can be disabled with
  Trilinos_ENABLE_PACKAGE:BOOL=OFF.


(*) Configuring a package(s) along with all of the packages it can use

  $ ./do-configure \
     -D Trilinos_ENABLE_ALL_PACKAGES:BOOL=OFF \
     -D Trilinos_ENABLE_NOX:BOOL=ON \
     -D Trilinos_ENABLE_ALL_OPTIONAL_PACKAGES:BOOL=ON \
     -D Trilinos_ENABLE_TESTS:BOOL=ON \
     -D Trilinos_ENABLE_EXAMPLES:BOOL=ON

  NOTE: This set of arguments allows a user to turn on NOX
  as well as all packages that NOX can use.  However, tests
  and examples will only be turned on for NOX.
```

Running Tests with CTest: Serial Tests

```
$ ctest -R '(^Teuchos_|^Epetra_)' -W 70

Start processing tests
Test project /home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/SERIAL_DEBUG
 12/118 Testing Teuchos_BLAS_test ..........................................   Passed
 13/118 Testing Teuchos_DefaultMpiComm_UnitTests ...........................   Passed
 14/118 Testing Teuchos_Comm_test ..........................................   Passed
 15/118 Testing Teuchos_Containers_test ....................................   Passed
 16/118 Testing Teuchos_UnitTest_UnitTests .................................   Passed
 17/118 Testing Teuchos_UnitTest_BadUnitTest_final_results .................   Passed
 18/118 Testing Teuchos_UnitTest_BadUnitTest_end_result_failed .............   Passed
 19/118 Testing Teuchos_UnitTest_BadUnitTest_end_result_totals .............   Passed
 20/118 Testing Teuchos_UnitTest_BadUnitTest_Int_BadAssignment_failed_0 ....   Passed
 21/118 Testing Teuchos_UnitTest_BadUnitTest_Int_BadAssignment_failed_1 ....   Passed
 22/118 Testing Teuchos_MemoryManagement_UnitTests .........................   Passed
 ...
 71/118 Testing Epetra_BlockMap_test .......................................   Passed
 72/118 Testing Epetra_BasicPerfTest_test ..................................   Passed
 73/118 Testing Epetra_Comm_test ...........................................   Passed
 74/118 Testing Epetra_CrsGraph_test_unit ..................................   Passed
 75/118 Testing Epetra_CrsMatrix_test ......................................   Passed
 76/118 Testing Epetra_RowMatrix_test ......................................   Passed
 77/118 Testing Epetra_CrsRectMatrix_test ..................................   Passed
 78/118 Testing Epetra_Directory_test ......................................   Passed
 79/118 Testing Epetra_FECrsGraph_test .....................................   Passed
 80/118 Testing Epetra_FEVbrMatrix_test ....................................   Passed
 81/118 Testing Epetra_ImportExport_test ...................................   Passed
 ...

100% tests passed, 0 tests failed out of 60
```

Detailed results in: Testing/Temporary/LastTest.log

Running Tests with CTest: MPI Tests

```
$ ctest -R '(^Teuchos_|^Epetra_)' -W 70

Start processing tests
Test project /home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/MPI
 12/109 Testing Teuchos_BLAS_test_MPI_1 ...........................................   Passed
 13/109 Testing Teuchos_DefaultMpiComm_UnitTests_MPI_1 ............................   Passed
 14/109 Testing Teuchos_Comm_test_MPI_4 ...........................................   Passed
 15/109 Testing Teuchos_Containers_test_MPI_1 .....................................   Passed
 16/109 Testing Teuchos_UnitTest_UnitTests_MPI_1 ..................................   Passed
 17/109 Testing Teuchos_UnitTest_BadUnitTest_final_results_MPI_1 ..................   Passed
 18/109 Testing Teuchos_UnitTest_BadUnitTest_end_result_failed_MPI_1 ..............   Passed
 19/109 Testing Teuchos_UnitTest_BadUnitTest_end_result_totals_MPI_1 ..............   Passed
 20/109 Testing Teuchos_UnitTest_BadUnitTest_Int_BadAssignment_failed_MPI_1_0 .....   Passed
 21/109 Testing Teuchos_UnitTest_BadUnitTest_Int_BadAssignment_failed_MPI_1_1 .....   Passed
 ...
 72/109 Testing Epetra_BlockMap_test_MPI_4 ........................................   Passed
 73/109 Testing Epetra_BasicPerfTest_test_MPI_1 ...................................   Passed
 74/109 Testing Epetra_Comm_test_MPI_1 ............................................   Passed
 75/109 Testing Epetra_CrsGraph_test_unit_MPI_4 ...................................   Passed
 76/109 Testing Epetra_CrsMatrix_test_MPI_4 .......................................   Passed
 77/109 Testing Epetra_RowMatrix_test_MPI_1 .......................................   Passed
 78/109 Testing Epetra_CrsRectMatrix_test_MPI_1 ...................................   Passed
 79/109 Testing Epetra_Directory_test_MPI_1 .......................................   Passed
 80/109 Testing Epetra_FECrsGraph_test_MPI_1 ......................................   Passed
 81/109 Testing Epetra_FEVbrMatrix_test_MPI_1 .....................................   Passed
 82/109 Testing Epetra_ImportExport_test_MPI_1 ....................................   Passed
 ...

100% tests passed, 0 tests failed out of 61
```

MPI options automatically specified!

Running Tests with CTest: Serial Tests Memory Checking

```
$ ctest -R '(^Teuchos_|^Epetra_)' -E Dense -W 70 -T memcheck

   Site: gabriel.sandia.gov
   Build name: Linux-c++
Create new tag: 20081023-0422 - Experimental
Start processing tests
Memory check project /home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/SERIAL_DEBUG
 12/118 Memory Check Teuchos_BLAS_test ....................................................   Passed
 13/118 Memory Check Teuchos_DefaultMpiComm_UnitTests .....................................   Passed
 14/118 Memory Check Teuchos_Comm_test ....................................................   Passed
 15/118 Memory Check Teuchos_Containers_test ..............................................   Passed
 16/118 Memory Check Teuchos_UnitTest_UnitTests ...........................................   Passed
 17/118 Memory Check Teuchos_UnitTest_BadUnitTest_final_results ...........................   Passed
 18/118 Memory Check Teuchos_UnitTest_BadUnitTest_end_result_failed .......................   Passed
 19/118 Memory Check Teuchos_UnitTest_BadUnitTest_end_result_totals .......................   Passed
 20/118 Memory Check Teuchos_UnitTest_BadUnitTest_Int_BadAssignment_failed_0 ..............   Passed
 21/118 Memory Check Teuchos_UnitTest_BadUnitTest_Int_BadAssignment_failed_1 ..............   Passed
 ...
 85/118 Memory Check Epetra_MultiVector_test ..............................................   Passed
 86/118 Memory Check Epetra_Object_test ...................................................   Passed
 87/118 Memory Check Epetra_RowMatrixTransposer_test ......................................   Passed
 90/118 Memory Check Epetra_Vector_test ...................................................   Passed

100% tests passed, 0 tests failed out of 54
-- Processing memory checking output: ##
Memory checking results:
Uninitialized Memory Conditional - 3
Uninitialized Memory Read - 3
```

Results: Testing/Temporary/LastDynamicAnalysis_20081023-0422.log

Need some work to get Valgrind to work with MPI mode

Running Tests with Perl ‘runtests’ Script: Serial Tests

```
$ make runtests-serial

...

/usr/bin/perl /home/rabartl/PROJECTS/Trilinos.base/Trilinos/commonTools/test/utilities/runtests --
trilinos-dir=/home/rabartl/PROJECTS/Trilinos.base/Trilinos --comm=serial --build-
dir=/home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/SERIAL_DEBUG --category=FRAMEWORK --output-
dir=/home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/SERIAL_DEBUG/runtests-results
thyra - sillyCgSolve_mpi.exe...                            passed      3 seconds
thyra - test_std_ops.exe...                                passed      2 seconds
thyra - sillyPowerMethod_serial.exe...                     passed     <1 second
thyra - sillyCgSolve_serial.exe...                         passed      3 seconds
thyra - sillyCgSolve_serial.exe...                         passed      6 seconds
thyra - test_product_space.exe...                          passed      1 second
thyra - test_block_op.exe...                               passed     <1 second
thyra - test_handles.exe...                                passed      5 seconds
thyra - test_linear_combination.exe...                     passed      1 second
thyra - test_scalar_product.exe...                         passed     <1 second
...
rtop - runSpmdTests.exe...                                 passed     <1 second
rtop - testLapackWrappers.exe...                           passed     <1 second
rtop - testLapackWrappers.exe...                           passed     <1 second
rtop - testLapackWrappers.exe...                           passed     <1 second
rtop - testLapackWrappers.exe...                           passed     <1 second

  Tests Passed: 108
  Tests Failed:   0
 -------------------
  Tests Total:  108
```

Running Tests with Perl ‘runtests’ Script: MPI Tests

```
$ make runtests-mpi

...
/usr/bin/perl /home/rabartl/PROJECTS/Trilinos.base/Trilinos/commonTools/test/utilities/runtests --
trilinos-dir=/home/rabartl/PROJECTS/Trilinos.base/Trilinos --comm=mpi --mpi-
go="/usr/local/mpi/bin/mpiexec -n \ " --max-proc=4 --build-
dir=/home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/MPI --category=FRAMEWORK --output-
dir=/home/rabartl/PROJECTS/Trilinos.base/BUILDS/CMAKE/MPI/runtests-results
thyra - sillyCgSolve_mpi.exe...                         passed      5 seconds
thyra - test_std_ops.exe...                             passed      3 seconds
thyra - test_product_space.exe...                       passed      3 seconds
thyra - test_block_op.exe...                            passed      2 seconds
thyra - test_handles.exe...                             passed      2 seconds
thyra - test_linear_combination.exe...                  passed      1 second
thyra - test_composite_linear_ops.exe...                passed      2 seconds
thyra - test_linear_op_with_solve.exe...                passed      1 second
thyra - test_linear_op_with_solve.exe...                passed     <1 second
...
rtop - supportUnitTests.exe...                          passed      1 second
rtop - opsUnitTests.exe...                              passed     <1 second
rtop - runSpmdTests.exe...                              passed      1 second
rtop - testLapackWrappers.exe...                        passed     <1 second
rtop - testLapackWrappers.exe...                        passed     <1 second
rtop - testLapackWrappers.exe...                        passed      1 second
rtop - testLapackWrappers.exe...                        passed     <1 second

  Tests Passed:  92
  Tests Failed:   0
 ------------------
  Tests Total:   92
```

MPI options automatically specified!

Outline of CMake Build System Files

```
Trilinos/
    CMakeLists.txt    # Top-level build file
    cmake/            # Helper macros, etc.
        TrilinosGlobalHelpers.cmake
        TrilinosPackageHelpers.cmake
        TrilinosPackageLibraryHelpers.cmake
        Trilinos_Add_Executable.cmake
        ...
    packages/
        ...
        epetraext/
            CMakeLists.txt    # Top-level package build file
            cmake/
                Dependencies.cmake    # Defines intra-package dependency lists
                Teuchos_Config.h.in    # Copied from ../src/ and hand modified
            src/
                CMakeLists.txt    # Defines library sources and library(s)
            test/
                CMakeLists.txt
                MatrixMatrix/
                    CMakeLists.txt   # Define actual test executables and test runs
        ...
```

Adding Packages in Trilinos/CMakeLists.txt

```
SET(Trilinos_PACKAGES
 Teuchos
 RTOp
 Epetra
 Triutils
 EpetraExt
 Thyra
 #Anasazi
 RBGen
 )
```

What it needs to be

```
SET(Trilinos_PACKAGES_AND_DIRS
 Teuchos          teuchos
 RTOp             rtop
 Epetra           epetra
 Triutils         triutils
 EpetraExt        epetraext
 Thyra            thyra
 RBGen            rbgen
 ForTrilinos      ForTrilinos
 PyTrilinos       PyTrilinos
 )
```

- Adding a new Trilinos Package is a 1-line addition at the Framework Level!
- NOTE: The packages must be listed in a order of strictly increasing dependences!
- NOTE: If you get the ordering wrong, the automated dependency handling CMake scripts will automatically detect this and issue a very good error messages <u>before</u> the build is performed!

Defining a Package's CMake Build Files

Trilinos/packges/
epetraext/CMakeLists.txt

```
INCLUDE(TrilinosPackageHelpers)
INCLUDE(Trilinos_Add_Option)

#
# A) Define the package
#

TRILINOS_PACKAGE(EpetraExt)

#
# B) Set up package-specific options
#

TRILINOS_ADD_OPTION(${PACKAGE_NAME}_BUILD_TRANSFORM
  HAVE_TRANSFORM
  "Enable transform functionality."
  ON )

...

#
# C) Add the libraries, tests, and examples
#

ADD_SUBDIRECTORY(src)

TRILINOS_PACKAGE_ADD_TEST_DIRECTORIES(test)

#
# D) Do standard postprocessing
#

TRILINOS_PACKAGE_POSTPROCESS()
```

- Utility macros provide framework hooks into package functionality
  - Defines common behavior across all packages
  - Avoids duplication
  - Facilitates maintenance

epetraext/cmake/Depencencies.cmake

```
SET(LIB_REQUIRED_DEP_PACKAGES Epetra Teuchos)
SET(LIB_OPTIONAL_DEP_PACKAGES Triutils)
SET(TEST_REQUIRED_DEP_PACKAGES "")
SET(TEST_OPTIONAL_DEP_PACKAGES "")
```

- Intra-package dependencies defined once only!
  - Used in all intra-package dependency handing
  - All header-file paths and link libraries and directories handled automatically
  - These dependencies can not be wrong! (i.e. the libs and execs would not build and link)

Automatic Intra-Package Dependency Handling

```
$ ./do-configure -D DUMP_PACKAGE_DEPENDENCIES:BOOL=ON -D Trilinos_PACKAGES_OVERRIDE:BOOL=OFF

Configuring Trilinos build directory


Printing package dependenies ...

-- Teuchos_FORWARD_LIB_REQUIRED_DEP_PACKAGES='RTOp;EpetraExt;Isorropia;Thyra;Galeri;Amesos;Ifpack;Belos;RBGen'
-- Teuchos_FORWARD_LIB_OPTIONAL_DEP_PACKAGES='AztecOO;ML'

-- Epetra_FORWARD_LIB_REQUIRED_DEP_PACKAGES='Triutils;EpetraExt;Isorropia;AztecOO;Galeri;Amesos;Ifpack;Belos'
-- Epetra_FORWARD_LIB_OPTIONAL_DEP_PACKAGES='Thyra;ML;RBGen'

-- Zoltan_FORWARD_LIB_REQUIRED_DEP_PACKAGES='Isorropia'

...

-- EpetraExt_LIB_REQUIRED_DEP_PACKAGES='Epetra;Teuchos'
-- EpetraExt_LIB_OPTIONAL_DEP_PACKAGES='Triutils'
-- EpetraExt_FORWARD_LIB_OPTIONAL_DEP_PACKAGES='Isorropia;Thyra;Galeri;Amesos;ML'
-- EpetraExt_FORWARD_TEST_OPTIONAL_DEP_PACKAGES='Belos'

-- Isorropia_LIB_REQUIRED_DEP_PACKAGES='Teuchos;Epetra;Zoltan'
-- Isorropia_LIB_OPTIONAL_DEP_PACKAGES='EpetraExt'
-- Isorropia_FORWARD_LIB_OPTIONAL_DEP_PACKAGES='ML'

-- Thyra_LIB_REQUIRED_DEP_PACKAGES='RTOp;Teuchos'
-- Thyra_LIB_OPTIONAL_DEP_PACKAGES='EpetraExt;Epetra'
-- Thyra_FORWARD_LIB_REQUIRED_DEP_PACKAGES='Stratimikos'

...

-- Stratimikos_LIB_REQUIRED_DEP_PACKAGES='Thyra'
-- Stratimikos_LIB_OPTIONAL_DEP_PACKAGES='Amesos;AztecOO;Belos;Ifpack;ML'
-- Stratimikos_TEST_OPTIONAL_DEP_PACKAGES='Triutils'
```

Adding a Library in PACKAGE/src/CMakeLists.txt

```
INCLUDE(TrilinosPackageLibraryHelpers)

# A) Package-specific configuration options

TRILINOS_PACKAGE_CONFIGURE_FILE(${PROJECT_NAME}_config.h)

# B) Define the header and source files (and directories)

INCLUDE_DIRECTORIES(${CMAKE_CURRENT_SOURCE_DIR})

SET(HEADERS
  EpetraExt_ConfigDefs.h
  ...
  )

SET(SOURCES
  EpetraExt_ProductOperator.cpp
  ...
  )

...

# C) Define the targets for package's library(s)

TRILINOS_PACKAGE_ADD_LIBRARY(
  epetraext
  HEADERS ${HEADERS}
  SOURCES ${SOURCES}
  )

# D) Export the dependency variables of this package for ...

TRILINOS_PACKAGE_EXPORT_DEPENDENCY_VARIABLES()
```

- Dependent package header directories, libraries, and library link directories handled automatically!
- Macros provide uniform behavior across all libraries across all pacakges!

Adding a Test in PACKAGE/test/CMakeLists.txt

epetraext/test/CMakeLists.txt

```
# Compile against epetra_test_err.h in all tests?
INCLUDE_DIRECTORIES(${CMAKE_CURRENT_SOURCE_DIR})


...

ADD_SUBDIRECTORY(MatrixMatrix)
```

- Just include the subdirectories

epetraext/test/MatrixMatrix/CMakeLists.txt

```
INCLUDE(Trilinos_Add_Executable_And_Test)
INCLUDE(Copy_Files_To_Binary_Dir)


TRILINOS_ADD_EXECUTABLE_AND_TEST(
  MatrixMatrix_test
  SOURCES cxx_main.cpp
  COMM serial mpi
  )

COPY_FILES_TO_BINARY_DIR(EpetraExtMatrixMatrixCopyFiles
  DEST_FILES
    infileAB infileATBT infileAB2 infiles infileABT infileAB3 infileATB infileATB2
    C.mtx C4x4.mtx C4x12x12x4.mtx C4x6.mtx C6x4.mtx C6x6.mtx mat6x4.mtx mat6x6.mtx
    mat12x4.mtx mat4x12.mtx mat4x4.mtx mat4x6.mtx Y.mtx YTC.mtx Y_transp.mtx
    roman roman2 romancase romancase2 cdt cdt_case cdt_d.mtx cdt_m.mtx cdt_tce.mtx
  SOURCE_DIR ${${PACKAGE_NAME}_SOURCE_DIR}/test/MatrixMatrix
  SOURCE_PREFIX "src_"
  )
```

- All header paths, link libraries etc are handled automatically!
- Define executable and test in one shot!
- 100% correct dependency tracking!

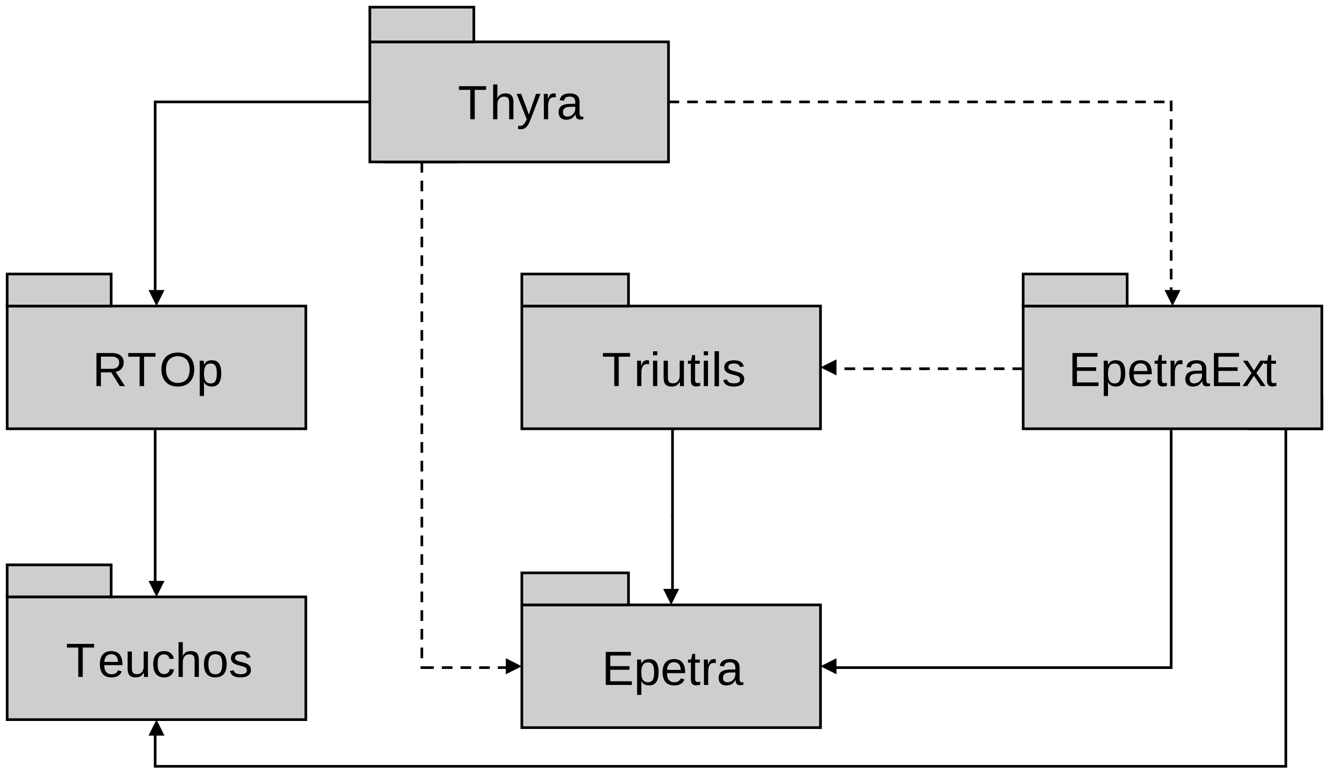
Defining More Complex Tests

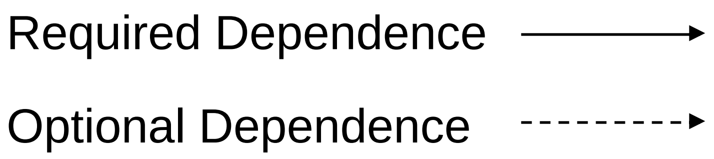
thyra/test/operator_solve/CMakeLists.txt

```
INCLUDE(Trilinos_Add_Executable_And_Test)


TRILINOS_ADD_EXECUTABLE(
  test_linear_op_with_solve
  SOURCES test_linear_op_with_solve.cpp
  COMM serial mpi
  )
TRILINOS_ADD_TEST(
  test_linear_op_with_solve
  NAME test_linear_op_with_solve_n1_n2
  ARGS "--n=1" "--n=2"
  NUM_MPI_PROCS 1
  COMM serial mpi
  )
TRILINOS_ADD_TEST(
  test_linear_op_with_solve
  NAME test_linear_op_with_solve_n4
  ARGS "--n=4"
  NUM_MPI_PROCS 1
  COMM serial mpi
  XHOST s858352 s903186
  )
```

- Define test cases separately from executable if needed!

Package Dependency Structure for Thyra





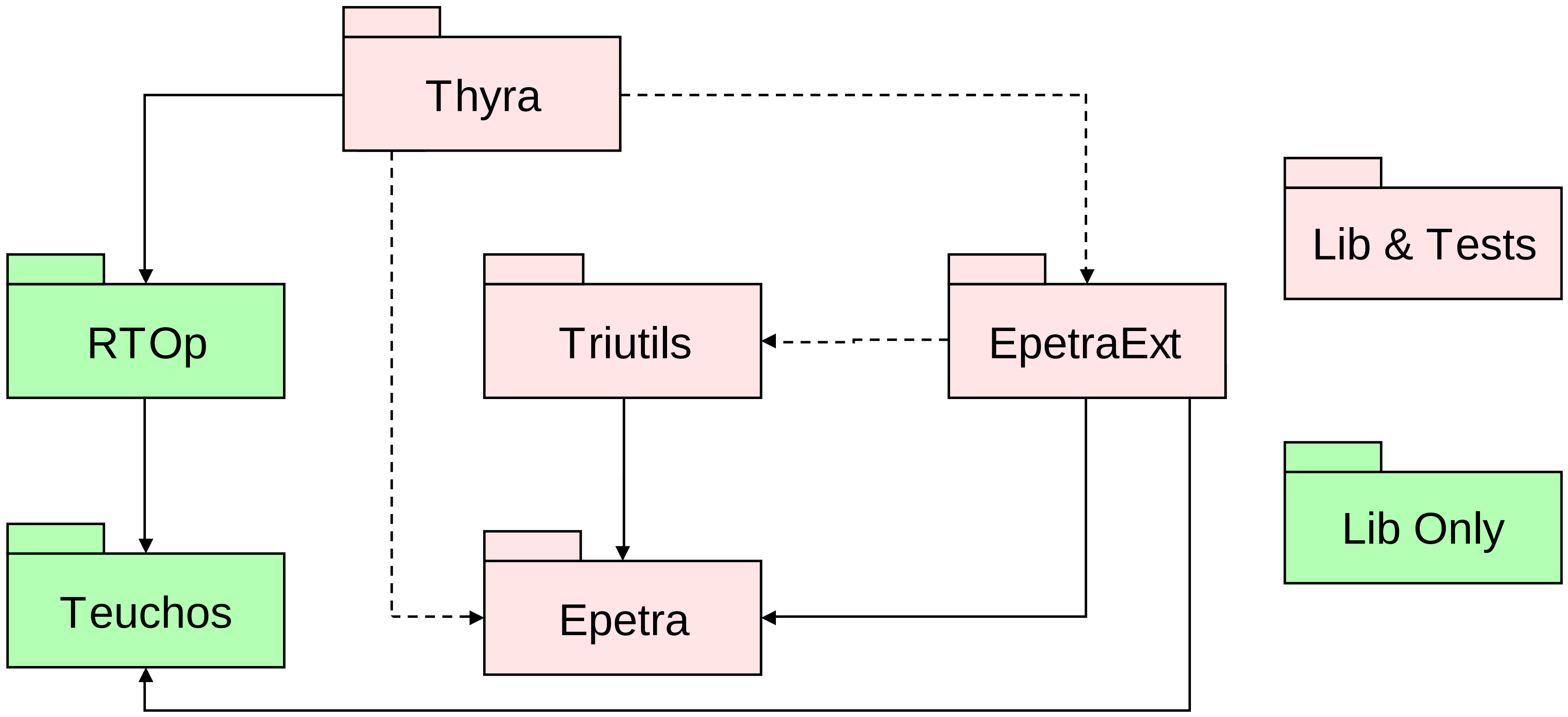
Example: Enabling a Package and All Optional Packages

```
$ ./do-configure -DTrilinos_ENABLE_ALL_PACKAGES:BOOL=OFF \
    -DTrilinos_ENABLE_Thyra:BOOL=ON \
    -DTrilinos_ENABLE_ALL_OPTIONAL_PACKAGES:BOOL=ON
```

```
Configuring Trilinos build directory

...

Enabling all optional packages for current set of enabled packages ...

-- Setting Trilinos_ENABLE_EpetraExt=ON because Trilinos_ENABLE_Thyra=ON
-- Setting Trilinos_ENABLE_Epetra=ON because Trilinos_ENABLE_Thyra=ON
-- Setting Trilinos_ENABLE_Triutils=ON because Trilinos_ENABLE_EpetraExt=ON

Enabling all remaining required packages for the current set of enabled packages ...

-- Setting Trilinos_ENABLE_RTOp=ON because Trilinos_ENABLE_Thyra=ON
-- Setting Trilinos_ENABLE_Teuchos=ON because Trilinos_ENABLE_Thyra=ON

Enabling all optional intra-package enables that can be if both sets of packages are enabled ...

-- Setting EpetraExt_ENABLE_Triutils=ON since Trilinos_ENABLE_EpetraExt=ON AND Trilinos_ENABLE_Triutils=ON
-- Setting Thyra_ENABLE_EpetraExt=ON since Trilinos_ENABLE_Thyra=ON AND Trilinos_ENABLE_EpetraExt=ON
-- Setting Thyra_ENABLE_Epetra=ON since Trilinos_ENABLE_Thyra=ON AND Trilinos_ENABLE_Epetra=ON

Final set of enabled packages:  Teuchos RTOp Epetra Triutils EpetraExt Thyra 6
```

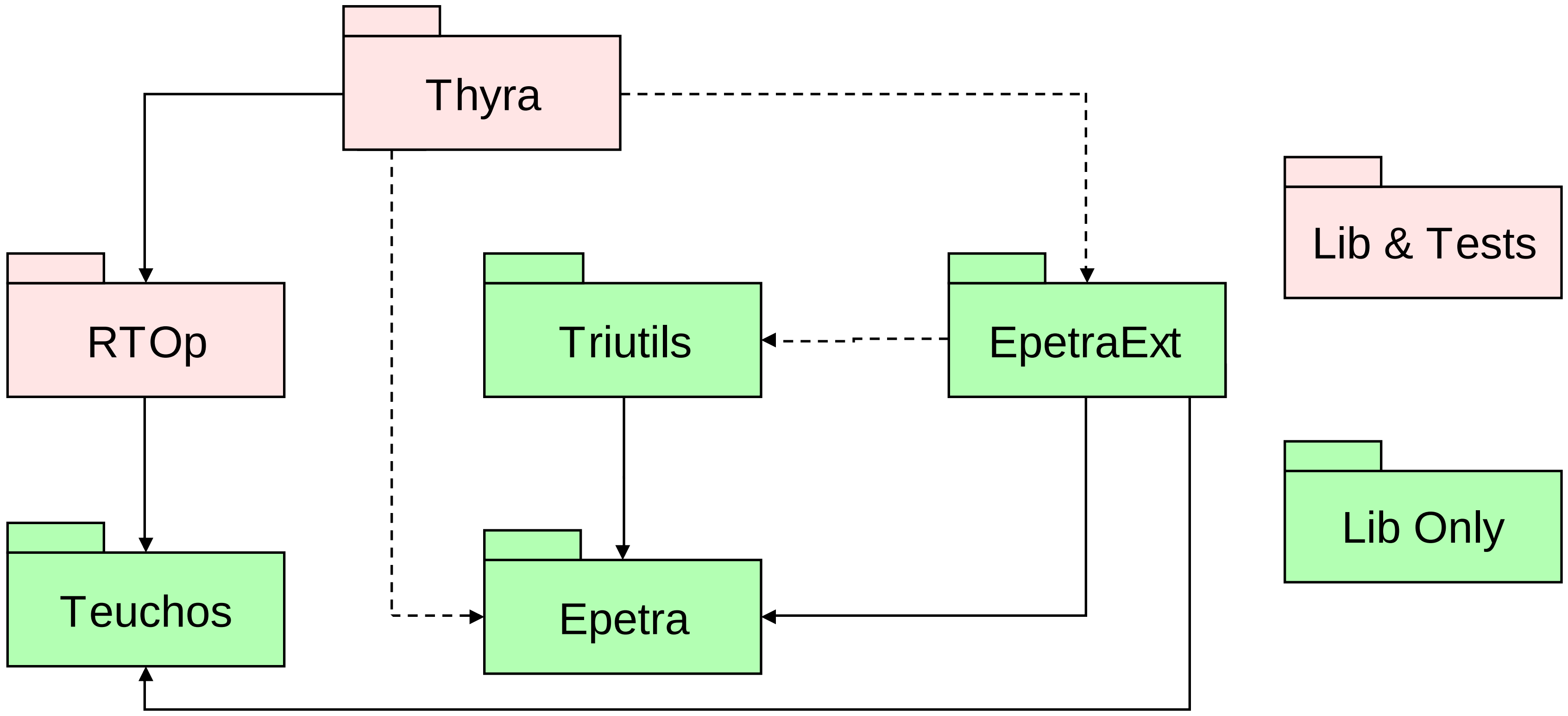
Dependency Handling for Pre-Checkin Testing

```
$ ./do-configure \
   -D Trilinos_ENABLE_ALL_PACKAGES:BOOL=OFF \
   -D Trilinos_ENABLE_Epetra:BOOL=ON \
   -D Trilinos_ENABLE_ALL_FORWARD_DEP_PACAKGES:BOOL=ON \
   -D Trilinos_ENABLE_TESTS:BOOL=ON \
   -D Trilinos_ENABLE_EXAMPLES:BOOL=ON
```



Dependency Handling for Pre-Checkin Testing

```
$ ./do-configure \
   -D Trilinos_ENABLE_ALL_PACKAGES:BOOL=OFF \
   -D Trilinos_ENABLE_RTOp:BOOL=ON \
   -D Trilinos_ENABLE_ALL_FORWARD_DEP_PACAKGES:BOOL=ON \
   -D Trilinos_ENABLE_TESTS:BOOL=ON \
   -D Trilinos_ENABLE_EXAMPLES:BOOL=ON
```



Testing and Results Display with CTest & CDash

http://trilinos.sandia.gov/cdash

http://datamining.sandia.gov/CDash (2008/10/03)

Final Recommendations

- CMake Build System:
  - Provide full support for CMake libs, tests/examples in all packages ASAP
  - Maintain support for Autotools for only building/installing libraries
    - Test Autotools built/installed headers & libraries with CMake tests/examples

- CTest/CDash Testing System:
  - Maintain current perl-based test system
    - Update perl runharness script(s) to drive CMake build of Trilinos
    - Maintain current test results web pages and email updates
  - Provide CTest versions of all tests and examples
    - Memory testing
    - Code coverage
    - Test timeouts
  - Work on improving CTest/CDash system:
    - Package-specific dashboard displays
    - Package-specific email test results notifications
    - Package build, disable, build,...., testing system

Sandia National Laboratories

The End

The End