TriBITS

TriBITS Foundations and Updates

Roscoe A. Bartlett, Ph.D.

bartlettra@ornl.gov

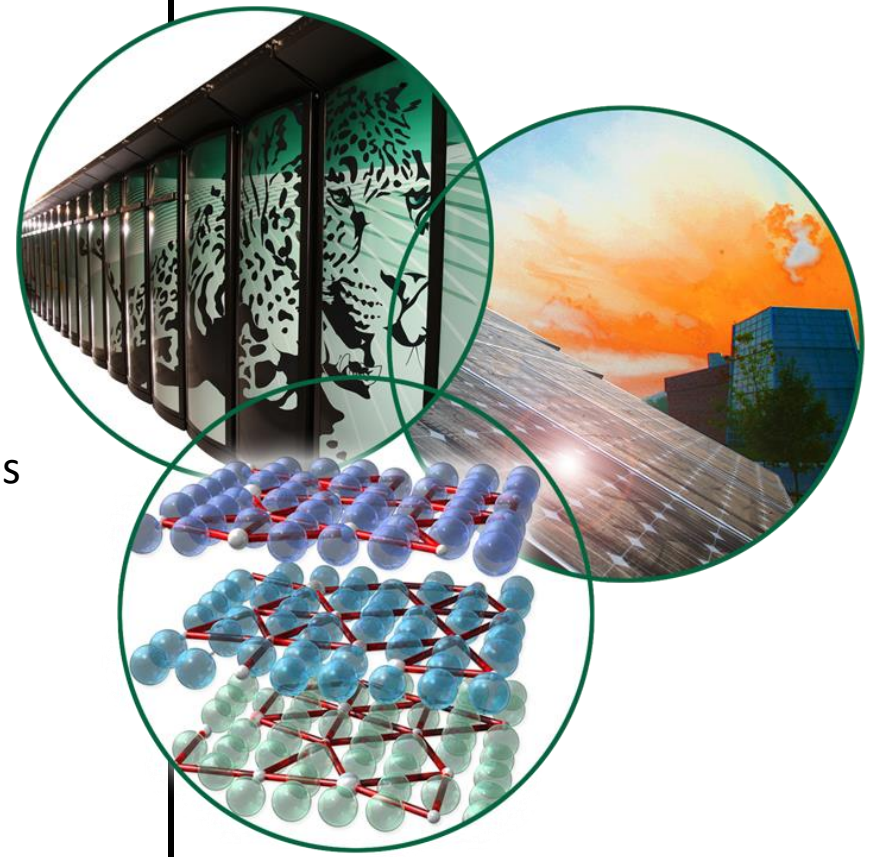
http://web.ornl.gov/~8vt/

Computational Engineering and Energy Sciences Group,

Oak Ridge National Laboratory

Trilinos Developers Meeting

October 29, 2015



U.S. DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY

MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

What is TriBITS?

- Framework for large, distributed multi-repository CMake projects
- Reduce boiler-plate CMake code and enforce consistency across large distributed projects
- Subproject dependencies and namespacing architecture (packages)
- Automatic package dependency handling (directed acyclic graph)
- Additional functionality missing in raw CMake
- Change default CMake behavior when necessary
- Additional tools for agile software development processes (e.g. Continuous Integration (CI))

History of TriBITS:

- 2007: Initially developed as a CMake package architecture for Trilinos
- 2011: Generalized and extended for CASL VERA
- 2014: Source code hosted on GitHub

Raw CMake vs. TriBITS

Raw CMakeLists.txt File

```
# Build and install library
SET(HEADERS hello_world_lib.hpp)
SET(SOURCES hello_world_lib.cpp)
ADD_LIBRARY(hello_world_lib ${SOURCES})
INSTALL(TARGETS hello_world_lib DESTINATION lib)
INSTALL(FILES ${HEADERS} DESTINATION include)

# Build and install user executable
ADD_EXECUTABLE(hello_world hello_world_main.cpp)
TARGET_LINK_LIBRARIES(hello_world hello_world_lib)
INSTALL(TARGETS hello_world DESTINATION bin)

# Test the executable
ADD_TEST(test ${CMAKE_CURRENT_BINARY_DIR}/hello_world)
SET_TESTS_PROPERTIES(test PROPERTIES PASS_REGULAR_EXPRESSION "Hello World")

# Build and run some unit tests
ADD_EXECUTABLE(unit_tests hello_world_unit_tests.cpp)
TARGET_LINK_LIBRARIES(unit_tests hello_world_lib)
ADD_TEST(unit_test ${CMAKE_CURRENT_BINARY_DIR}/unit_tests)
SET_TESTS_PROPERTIES(unit_test  PROPERTIES PASS_REGULAR_EXPRESSION "All unit tests passed")
```

TriBITS Package CMakeList.txt File

```
TRIBITS_PACKAGE(HelloWorld)
TRIBITS_ADD_LIBRARY(hello_world_lib
  HEADERS hello_world_lib.hpp SOURCES hello_world_lib.cpp)
TRIBITS_ADD_EXECUTABLE(hello_world NOEXEPREFIX SOURCES hello_world_main.cpp INSTALLABLE)
TRIBITS_ADD_TEST(hello_world NOEXEPREFIX PASS_REGULAR_EXPRESSION "Hello World")
TRIBITS_ADD_EXECUTABLE_AND_TEST(unit_tests SOURCES hello_world_unit_tests.cpp
   PASS_REGULAR_EXPRESSION "All unit tests passed")
TRIBITS_PACKAGE_POSTPROCESS()
```

- Avoid duplication and reduce boiler-plate commands
- Install libs & headers but not (test) execs by default (most common use case)
- Library linking automatically handled by default (between Packages, execs/libs)
- Automatic namespacing of test & exec names
- Allows for all library names to be prefixed (needed for Linux distributions)
- Consistent handling of test enables/disables based on various criteria

TriBITS Structural Elements

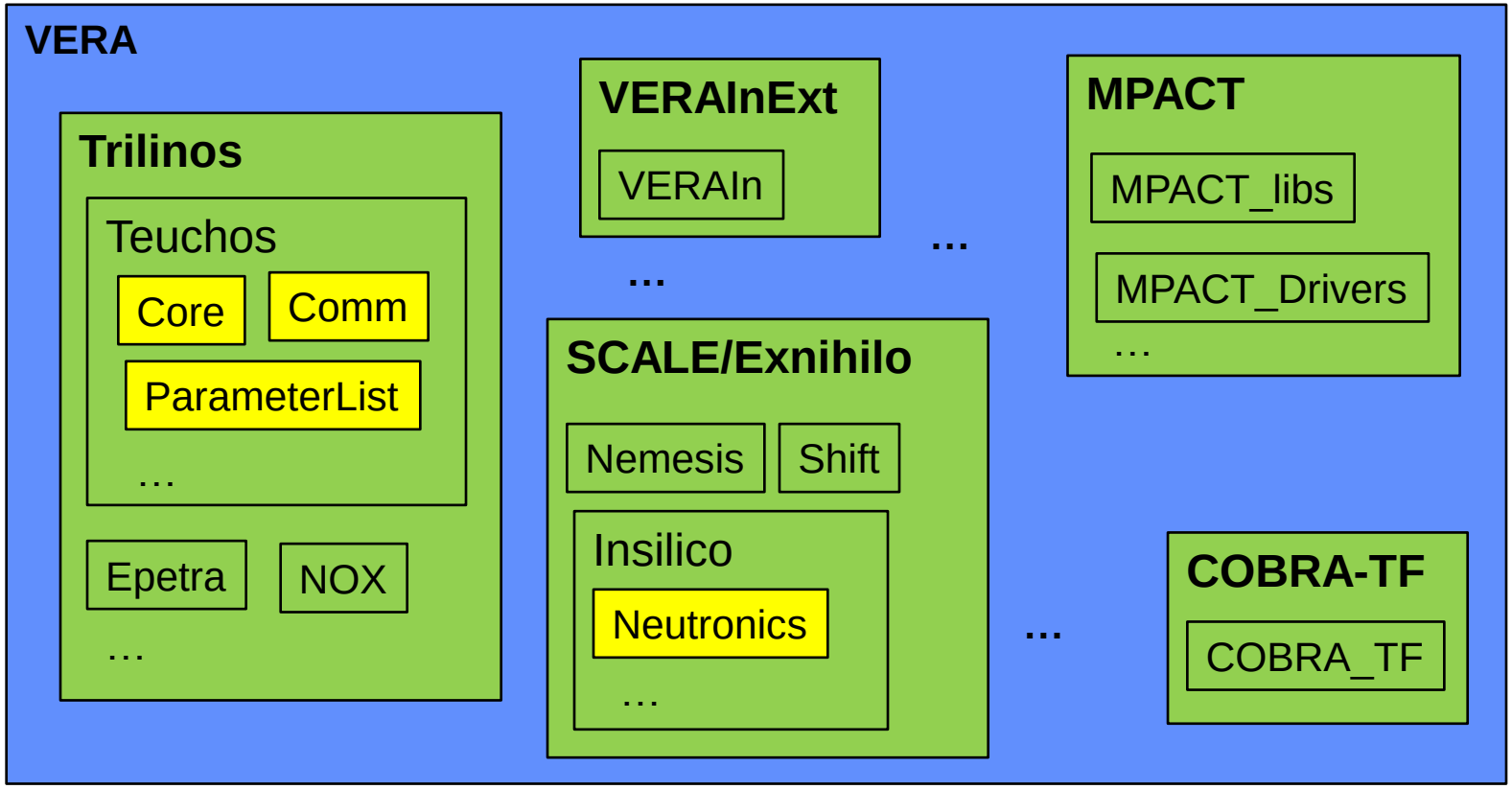
TriBITS Structural Units

- TriBITS Project:
  - Complete CMake “Project”
  - Overall projects settings
- TriBITS Repository:
  - Collection of Packages and TPLs
  - Unit of distribution and integration
  - Typically a version control (git) repository
- TriBITS Package:
  - Encapsulated collection of related software & tests
  - Unit of testing, namespacing, documentation, and reuse
  - Lists dependencies on SE Packages & TPLs
- TriBITS Subpackage:
  - Optional partitioning of package software & tests
  - Primarily for dependency management (SE principles)
- TriBITS TPLs (Third Party Libraries):
  - Specification of external dependencies (libs)
  - Required or optional dependency
  - Single definition across all packages
  - Can use native CMake Find<Package>.cmake modules

Packages + Subpackages = Software Engineering (SE) Packages

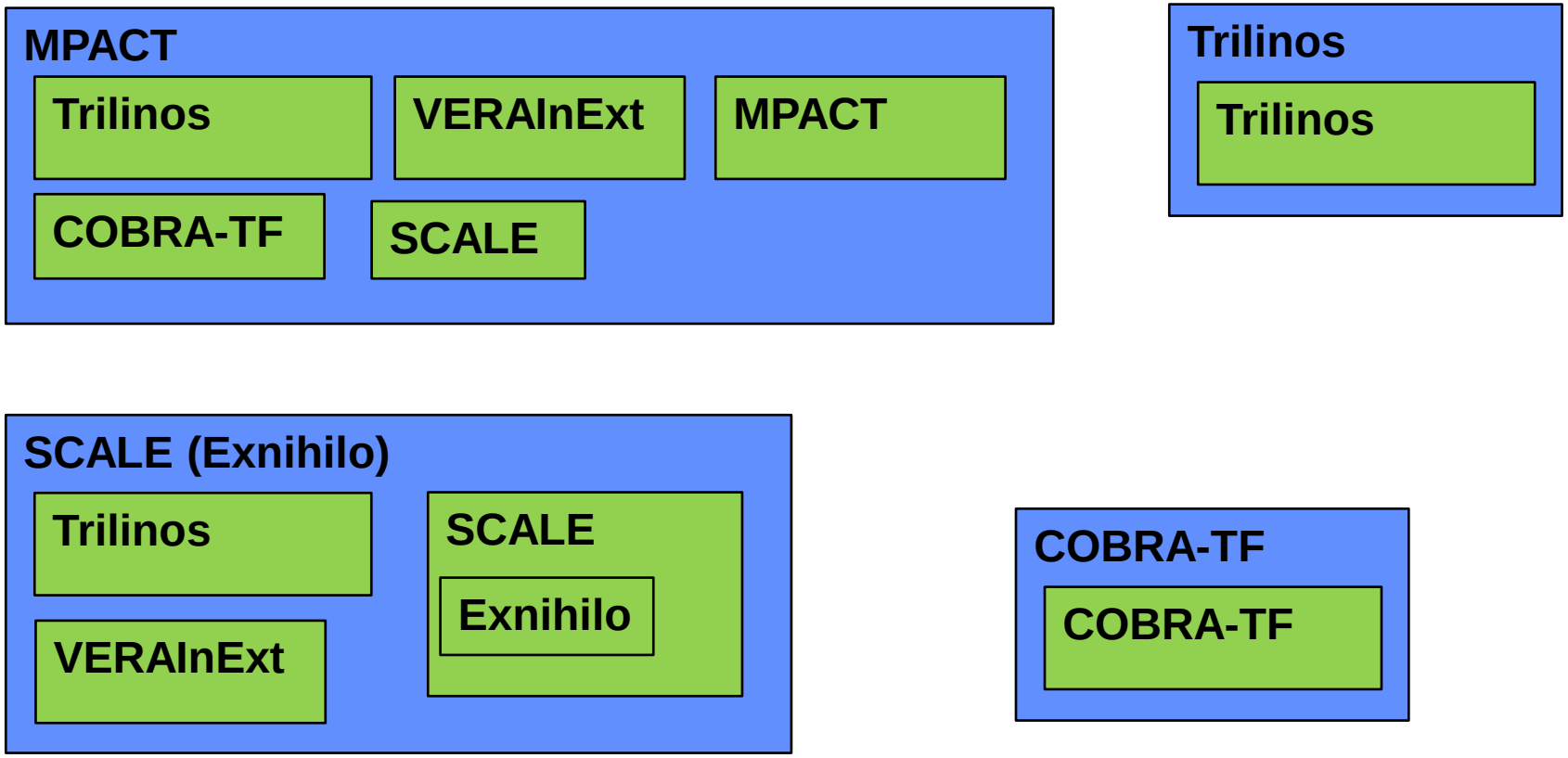
See: https://tribits.org/doc/TribitsDevelopersGuide.html

VERA Meta-Project, Repositories, Packages & Subpackages



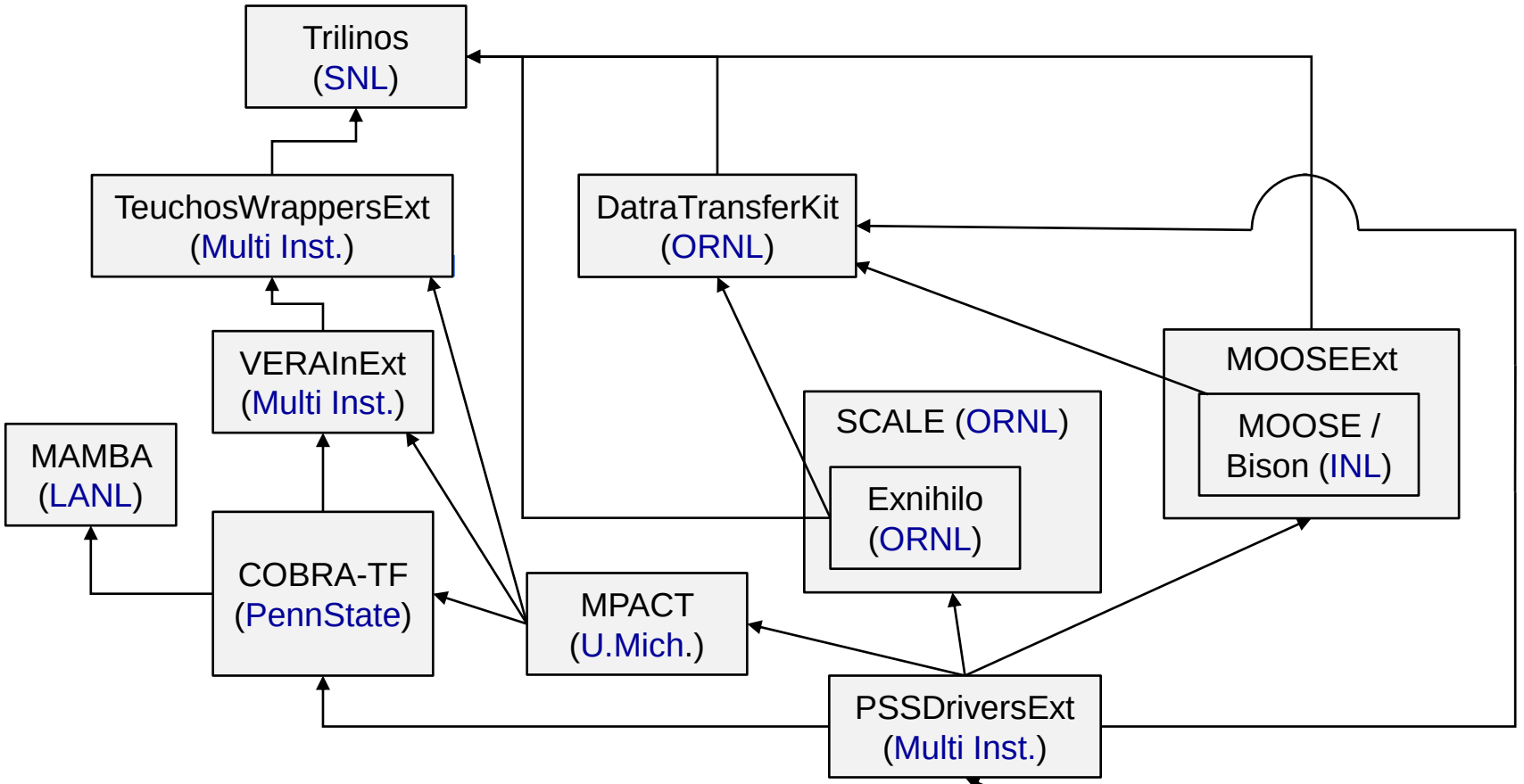
- **VERA meta-project:** Git repository and TriBITS meta-project (contains no packages)
- **TriBITS repos:**: Trilinos, VERAInExt, COBRA-TF, MPACT, SCALE …
- **TriBITS packages**: Teuchos, Epetra, VERAIn, Insilico, COBRA_TF, MPACT_Drivers, …
- **TriBITS subpackages**: TeuchosCore, InsilicoNeutronics …
- **TriBITS SE packages**: TeuchosCore , Teuchos, VERAIn, Insilico, InsilicNeutronics, …

Flexibility in TriBITS Projects and Repositories



The same TriBITS repositories can be arranged into multiple TriBITS CMake projects!

TriBITS and VC Repos for CASL VERA



- Outer boxes are all TriBITS and VC repos (all native git or snapshotted to git)
- Inner boxes (e.g. Exnihilo, MOOSE, Dakota) are VC repos but not TriBITS repos

VERA/cmake/ExtraRepositoriesList.cmake

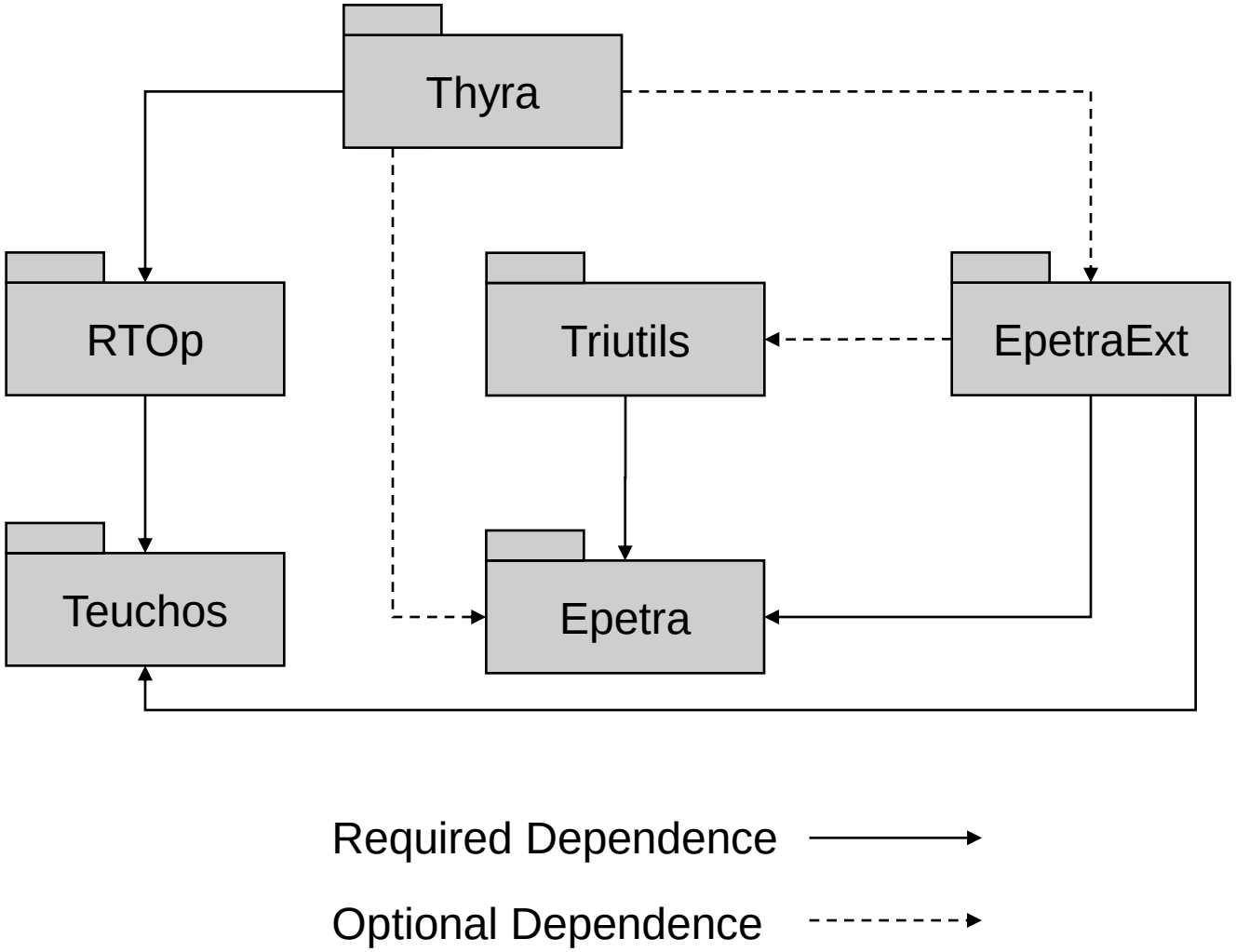
```
TRIBITS_PROJECT_DEFINE_EXTRA_REPOSITORIES(
  TriBITS          ""   GIT  git@casl-dev:TriBITS          ""  ${TriBITS_REPO_TYPE}
  Trilinos         ""   GIT  git@casl-dev:Trilinos         ""  Continuous
  TeuchosWrappersExt "" GIT git@casl-dev:TeuchosWrappersExt ""  Continuous
  MAMBA            ""   GIT  git@casl-dev:MAMBA            ""  Continuous
  COBRA-TF         ""   GIT  git@casl-dev:COBRA-TF         ""  Continuous
  VERAInExt        ""   GIT  git@casl-dev:VERAInExt        ""  Continuous
  VERAData         ""   GIT  git@casl-dev:VERAData         NOPACKAGES ${VERADATA_CAT}
  DataTransferKit  ""   GIT git@casl-dev:DataTransferKit  ""  Continuous
  MOOSEExt         ""   GIT  git@casl-dev:MOOSEExt        ""  Continuous
  MOOSE            MOOSEExt/MOOSE   GIT
      git@casl-dev:MOOSE   NOPACKAGES  Continuous
  SCALE            ""   GIT  git@casl-dev:SCALE          ""  Continuous
  Exnihilo         SCALE/Exnihilo   GIT
      git@casl-dev:Exnihilo                        NOPACKAGES  Continuous
  MPACT            ""   GIT  git@casl-dev:MPACT          ""  Continuous
  LIMEExt          ""   GIT  git@casl-dev:LIMEExt        ""  Nightly
  PSSDriversExt    ""   GIT  git@casl-dev:PSSDriversExt  ""  Continuous
  DakotaExt       ""   GIT  git@casl-dev:DakotaExt ""  Continuous
  Dakota  DakotaExt/Dakota  GIT  git@casl-dev:Dakota  NOPACKAGES  Continuous
  VUQDemos         ""   GIT  git@casl-dev:VUQDemos        ""  Nightly
)
```

Official version of VERA in on master branch used for CI & Nightly testing

- Partial set of repos can be cloned (protected by different groups)
- Non-git repos are converted into git repos: Dakota (svn), SCALE (hg), MOOSE (git submodules)

Automated Package Dependency Handling

Package Dependency Structure (e.g. Old Trilinos)



Package Dependencies.cmake Files

Teuchos

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_TPLS  BLAS  LAPACK
  LIB_OPTIONAL_TPLS Boost  )
```

Epetra

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_TPLS  BLAS  LAPACK  )
```

RTOp

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES  Teuchos  )
```

Triutils

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES  Epetra  )
```

EpetraExt

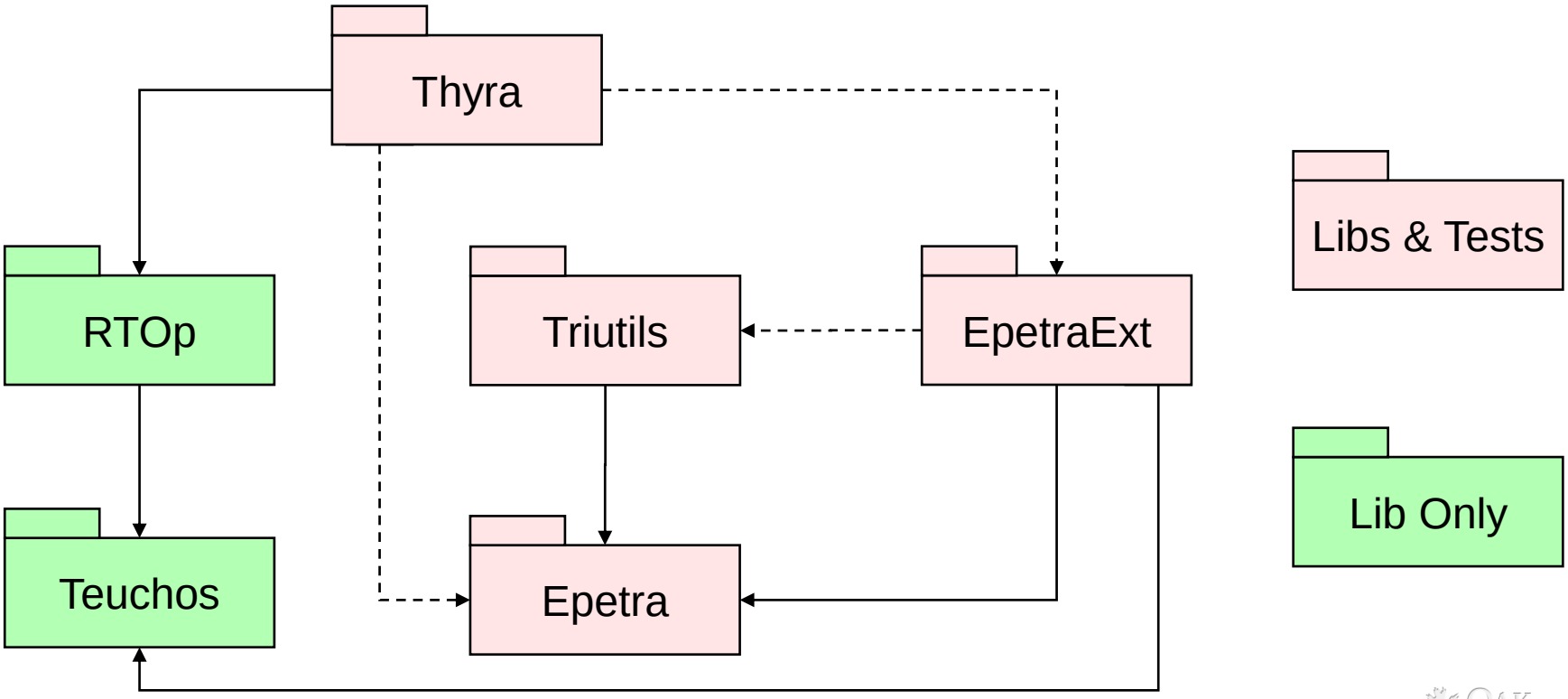
```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES  Epetra  Teuchos
  LIB_OPTIONAL_PACKAGES  Triutils  )
```

Thyra

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES  RTOp  Teuchos
  LIB_OPTIONAL_PACKAGES  EpetraExt  Epera  )
```

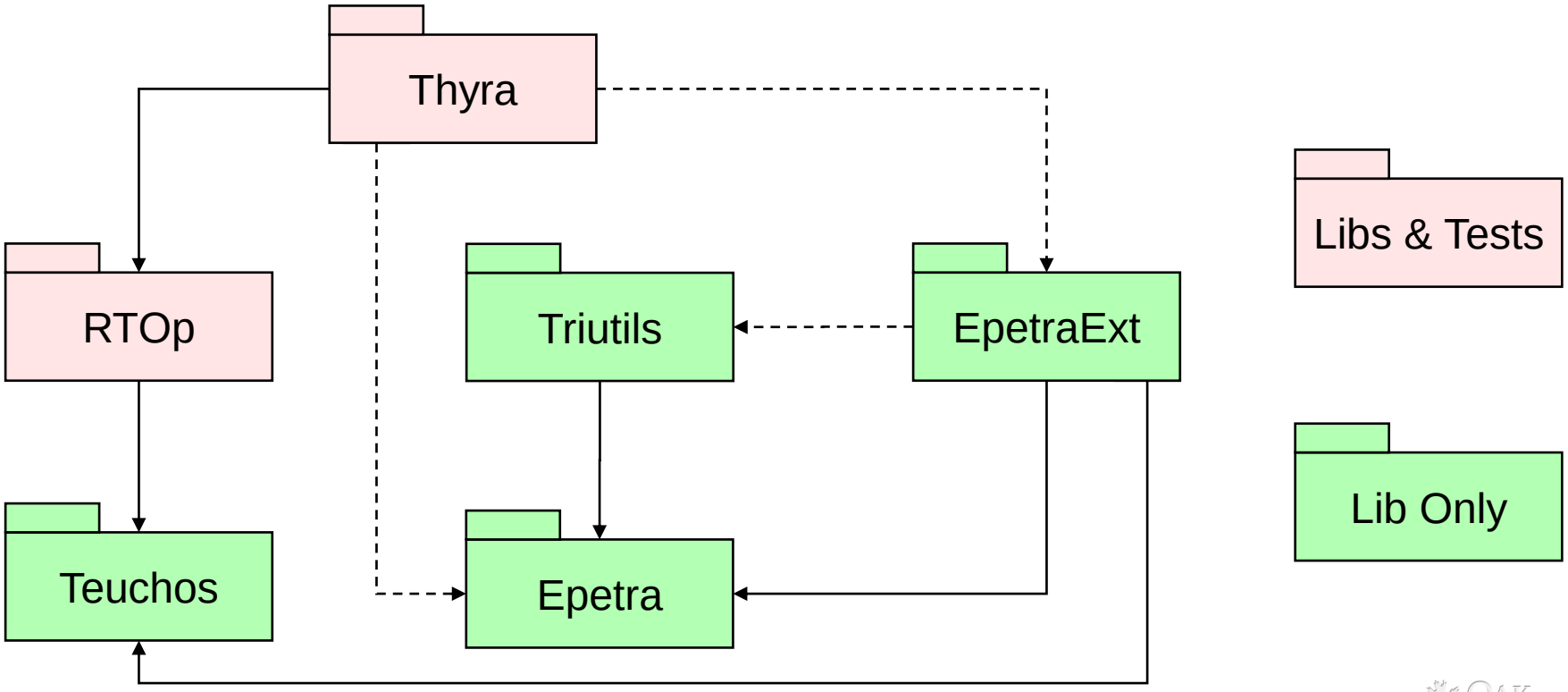
CI Testing: Change Epetra

```
$ ./do-configure \
   -D Trilinos_ENABLE_Epetra=ON \
   -D Trilinos_ENABLE_ALL_FORWARD_DEP_PACKAGES=ON \
   -D Trilinos_ENABLE_TESTS=ON
```



CI Testing: Change RTOp

```
$ ./do-configure \
   -D Trilinos_ENABLE_RTOp=ON \
   -D Trilinos_ENABLE_ALL_FORWARD_DEP_PACKAGES=ON \
   -D Trilinos_ENABLE_TESTS=ON
```



Usage of TriBITS Packages and Subpackages

Software Engineering Theory about Packaging

Package Cohesion OO Principles:

- REP (Release-Reuse Equivalency Principle): The granule of reuse is the granule of release.
- CCP (Common Closure Principle): The classes in a package should be closed together against the same kinds of changes. A change that affects a closed package affects all the classes in that package and no other packages.
- CRP (Common Reuse Principle): The classes in a package are used together. If you reuse one of the classes in a package, you reuse them all.

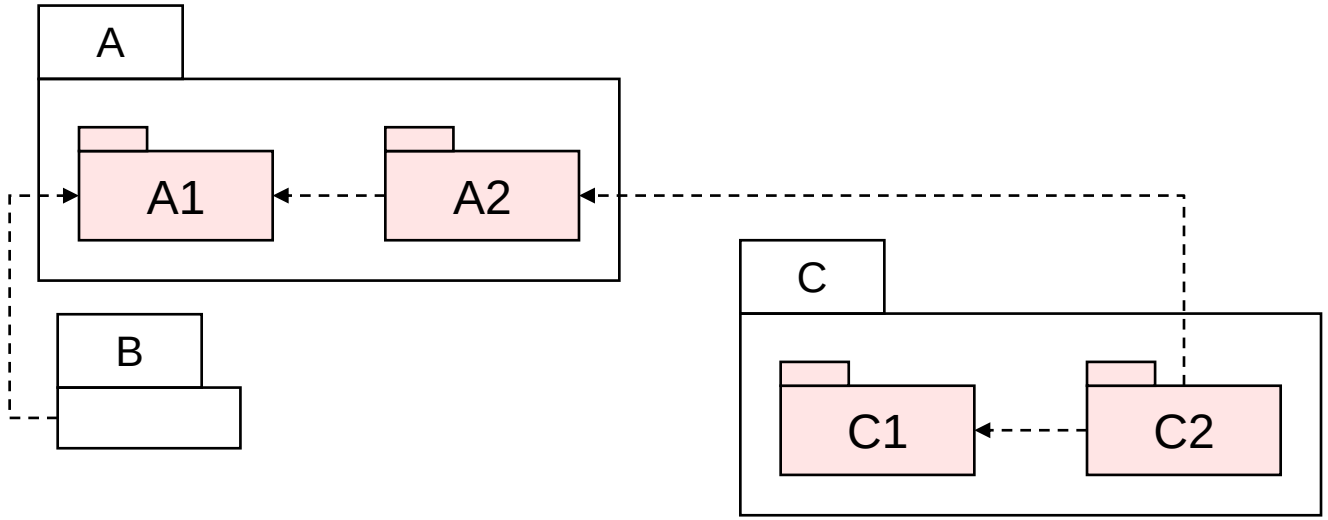
Package Coupling OO Principles:

- ADP (Acyclic Dependencies Principle): Allow no cycles in the package dependency graph.
- SDP (Stable Dependencies Principle): Depend in the direction of stability.
- SAP (Stable Abstractions Principle): A package should be as abstract as it is stable.

Problem: The Trilinos definition of a "Package" is not consistent with SE packaging principles most importantly the CRP

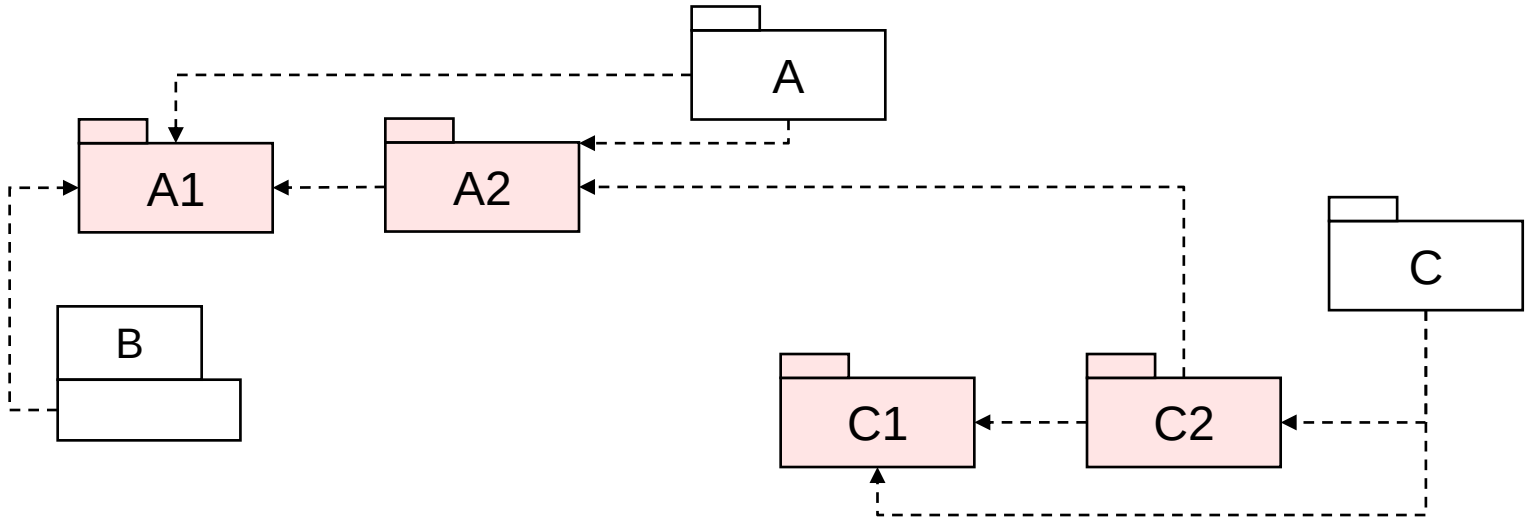
Source: Martin, Robert C. *Agile Software Development (Principles, Patterns, and Practices)*. Prentice Hall, 2003

TriBITS Packages and Subpackages: Overview



- TriBITS Parent Package:
  - Collection of related subpackages
  - Community of tightly integrated developers
  - Unit of documentation, package-by-package CTest driver (single email address)
  - Downstream (SE) packages should **not** list parent package as a dependency!
- TriBITS Subpackage:
  - Lightweight encapsulated collection of tightly related libs and tests/examples
  - Lightweight use the options of the parent package
  - Lists dependencies on upstream SE Packages & TPLs
  - Primary unit for dependency management!

TriBITS Packages and Subpackages: Dependencies



- This is how the TriBITS dependency management looks at packages and subpackages => **Packages and Subpackages are just are all just SE packages!**
- The parent package is just an SE Package that depends (optional or required) on all of its subpackages
  - New TriBITS packages should only have optional subpackages, **have no required subpackages**!
  - Support for **required subpackages is to maintain backward compatibility** when packages are broken into subpackages (when optional packages are disabled).

Depending only on subpackages: Example RTOp

rtop/cmake/Dependencies.cmake

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES TeuchosCore TeuchosComm TeuchosNumerics
  REGRESSION_EMAIL_LIST thyra-regression@software.sandia.gov
  )
```

Configure RTOp:

```
$ ./do-configure -DTrilinos_ENABLE_RTOp=OFF -DTrilinos_ENABLE_Kokkos=OFF
[…]
-- Setting Trilinos_ENABLE_TeuchosCore=ON because RTOp has a required dependence on TeuchosCore
-- Setting Trilinos_ENABLE_TeuchosComm=ON because RTOp has a required dependence on TeuchosComm
-- Setting Trilinos_ENABLE_TeuchosNumerics=ON because RTOp has a required dependence on
TeuchosNumerics
-- Setting Trilinos_ENABLE_TeuchosParameterList=ON because TeuchosComm has a required dependence
on TeuchosParameterList
[…]
Enabling all parent packages that have at least one subpackage enabled ...
-- Setting Trilinos_ENABLE_Teuchos=ON because Trilinos_ENABLE_TeuchosCore=ON
[…]
Final set of enabled packages:  Teuchos RTOp 2
Final set of enabled SE packages:  TeuchosCore TeuchosParameterList TeuchosComm TeuchosNumerics \
  Teuchos RTOp 6
[…]
Final set of non-enabled packages:  […] Kokkos […] 64
Final set of non-enabled SE packages:  Gtest ThreadPool KokkosCore KokkosContainers \
  KokkosAlgorithms KokkosExample Kokkos TeuchosRemainder TeuchosKokkosCompat \
  TeuchosKokkosComm […] 142
```

- NOTE: “Final set of enabled packages” means that at least one subpackage in that parent package is enabled and **not** that the parent package is explicitly enabled!

Depending only on subpackages: Example Thyra

thyra/cmake/Dependencies.cmake

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  SUBPACKAGES_DIRS_CLASSIFICATIONS_OPTREQS
    Core                 core                 PT  REQUIRED
    EpetraAdapters       adapters/epetra      PT  OPTIONAL
    EpetraExtAdapters    adapters/epetraext   PT  OPTIONAL
    TpetraAdapters       adapters/tpetra      PT  OPTIONAL
  )
```

thyra/core/cmake/Dependencies.cmake

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES TeuchosCore
   TeuchosParameterList TeuchosComm
   TeuchosNumerics RTOp
  )
```

thyra/adapters/epetraext/cmake/Dependencies.cmak

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES ThyraCore
    ThyraEpetraAdapters Epetra EpetraExt
  )
```

thyra/adapters/epetra/cmake/Dependencies.cmake

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES ThyraCore Epetra
  TEST_REQUIRED_PACKAGES Triutils
  )
```

thyra/adapters/tpetra/cmake/Dependencies.cmake

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  LIB_REQUIRED_PACKAGES ThyraCore Tpetra
  LIB_OPTIONAL_PACKAGES
    ThyraEpetraAdapters
  )
```

TriBITS Package and Subpackages : Details

- Packages that are broken into subpackages **should have no libs of their own**!
  - ⇒ Original design of subpackages was not to allow a parent package to have any libs or test/examples of its own! **(but was allowed by accident**)
  - ⇒ But packages can have their own tests/examples (if guarded correctly)
- Tight integration between the parent package and its subpackges:
  - <Project>_ENABLE_<ParentPackage>=ON => Equivalent to explicit **enabling** all subpackages
  - <Project>_ENABLE_<ParentPackage>=OFF => Equivalent to explict **disabling** all subpackages
  - Tests/examples will be enabled for all enabled subpackges when the package's tests/examples are enabled
  - Parent package's CMakeList.txt file is **always** processed when any subpackages are processed
    - ⇒ Parent package's options all defined (e.g. debug-mode checking, etc.)
      - ⇒ Subpackages are always processed in the context of the parent!
      - ⇒ **Does not allow interleaving subpackages between parent packages!**

Question: Should we really support parent packages to have their own libs?

⇒ If so, some work is needed to specify this but it will be constrained in packages enable/disable logic …

Consequences of move to GitHub on Dependencies

- No more need to get individual copyright on Trilinos packages

  $\Rightarrow$ Allows more freedom to break out new packages to get around the interleaving of subpackages!

Building Individual TriBITS packages as their own TriBITS CMake project

Allow an upstream TriBITS Package to Build Independently

- Put 'IF()' statement in package's top CMakeLists.txt for building as a TriBIITS project **or** a TriBITS package.

- Add <packageDir>/ProjectName.cmake

- Add <packageDir>/PackagesList.cmake
  - Package dir: "<PackageName> . PT"
  - Allow optional upstream packages to be missing TRIBITS_ALLOW_MISSING_EXTERNAL_PACKAGES()

- Add <packageDir>/TPLsList.cmake
  - Point to standard TPLs under "${${PROJECT_NAME}_TRIBITS_DIR}/common_tpls/"
  - Or, copy critical FindTPL<TPLNAME>.cmake to <packagesDir>/tpls/ and point to them there (no duplication)
  - Or, simply give it a dummy directory and make it EX so that it will not get enabled.

- Everything else is unchanged!

teuchos/cmake/Dependencies.cmake

```
TRIBITS_PACKAGE_DEFINE_DEPENDENCIES(
  SUBPACKAGES_DIRS_CLASSIFICATIONS_OPTREQS
    Core          core          PS  REQUIRED
    ParameterList parameterlist PS  REQUIRED
    Comm          comm          PS  REQUIRED
    Numerics      numerics      PS  REQUIRED
    Remainder     remainder     PS  REQUIRED
    KokkosCompat  kokkoscompat  PS  OPTIONAL
    KokkosComm    kokkoscomm    PS  OPTIONAL
  )
```

- Dependencies file remains completely unchanged!

teuchos/CMakeLists.txt

```
IF (TRIBITS_PROCESSING_PACKAGE)

# Processing TriBITS package!
[…]
TRIBITS_PACKAGE_DECL( Teuchos ENABLE_SHADOWING_WARNINGS CLEANED )
[…]
TRIBITS_PROCESS_SUBPACKAGES()
TRIBITS_PACKAGE_DEF()
[…]
TRIBITS_PACKAGE_POSTPROCESS()

ELSE(TRIBITS_PROCESSING_PACKAGE)

# Processing as a TriBITS project
CMAKE_MINIMUM_REQUIRED(VERSION 2.8.11 FATAL_ERROR)
INCLUDE("${CMAKE_CURRENT_SOURCE_DIR}/ProjectName.cmake")
PROJECT(${PROJECT_NAME} NONE)
SET(${PROJECT_NAME}_TRIBITS_DIR "${CMAKE_CURRENT_SOURCE_DIR}/../../cmake/tribits"
  CACHE PATH "By default assume Teuchos is in Trilinos")
INCLUDE("${${PROJECT_NAME}_TRIBITS_DIR}/TriBITS.cmake")
SET(TEUCHOS_STANDALONE_PACKAGE TRUE)
TRIBITS_PROJECT_ENABLE_ALL()

ENDIF(TRIBITS_PROCESSING_PACKAGE)
```

teuchos/PackagesList.cmake

```
TRIBITS_REPOSITORY_DEFINE_PACKAGES(
  Kokkos  kokkos  PT
  Teuchos  .  PT
  )

TRIBITS_ALLOW_MISSING_EXTERNAL_PACKAGES(Kokkos)
```

- Can build Teuchos by itself or with Kokkos
  => Need to fix gtest issue with Kokkos first (see Kokkos Issue #117)

teuchos/TPLsLists.cmake, Trilinos/TPLsList.cmake

teuchos/TPLsLists.cmake:

```
TRIBITS_REPOSITORY_DEFINE_TPLS(
  BinUtils    "${${PROJECT_NAME}_TRIBITS_DIR}/common_tpls/"    ST
  ARPREC      "${${PROJECT_NAME}_SOURCE_DIR}/cmake/tpls/"      ST
  QD          "${${PROJECT_NAME}_SOURCE_DIR}/cmake/tpls/"      ST
  MPI         "${${PROJECT_NAME}_TRIBITS_DIR}/core/std_tpls/"  PT
  BLAS        "${${PROJECT_NAME}_TRIBITS_DIR}/common_tpls/"    PT
  LAPACK      "${${PROJECT_NAME}_TRIBITS_DIR}/common_tpls/"    PT
  Boost       "${${PROJECT_NAME}_TRIBITS_DIR}/common_tpls/"    ST
  QT          "${${PROJECT_NAME}_SOURCE_DIR}/cmake/tpls/"      ST
  Eigen       "${${PROJECT_NAME}_SOURCE_DIR}/cmake/tpls/"      ST
  )
```

Trilinos/TPLsLists.cmake:

```
TRIBITS_REPOSITORY_DEFINE_TPLS(
  …
  ARPREC      "packages/teuchos/cmake/tpls/"      ST
  QD          "packages/teuchos/cmake/tpls/"      ST
  QT          "packages/teuchos/cmake/tpls/"      ST
  Eigen       "packages/teuchos/cmake/tpls/"      ST
  …
  )
```

- FindTPL<TPLCMAKE>.cmake modules that are specific to the package should go in the package source directory!

Configure of stand-alone Teuchos

```
$ cmake \
    -D TPL_ENABLE_MPI=ON -D CMAKE_BUILD_TYPE=DEBUG \
    -DTeuchos_ENABLE_TESTS=ON -D BUILD_SHARED_LIBS:BOOL=ON \
    $TEUCHOS_DIR
…
Processing Project, Repository, and Package dependency files and building internal
dependencies graph ...

NOTE: Setting Teuchos_ENABLE_TeuchosKokkosCompat=OFF because package
TeuchosKokkosCompat has a required dependency on missing package KokkosCore!
NOTE: Setting Teuchos_ENABLE_TeuchosKokkosComm=OFF because package TeuchosKokkosComm
has a required dependency on missing package KokkosCore!
…

$ time cmake .

real    0m4.567s
user    0m4.263s
sys     0m0.278s
```

- Get TriBITS from a) Trilinos (default), b) Install of TriBITS/tribits/ somewhere, c) Snapshot TriBITS/tribits/ into teuchos/cmake/

TriBITS Snapshotting into Trilinos

TriBITS Snapshots into Trilinos and local changes

```
$ cd Trilinos.base/Trilinos/cmake/tribits/
$ git checkout --track origin/tribits_github_snapshot
$ ../../TriBITS/tribits/snapshot_tribits.py
$ git checkout master
$ git merge --no-ff tribits_github_snapshot
```

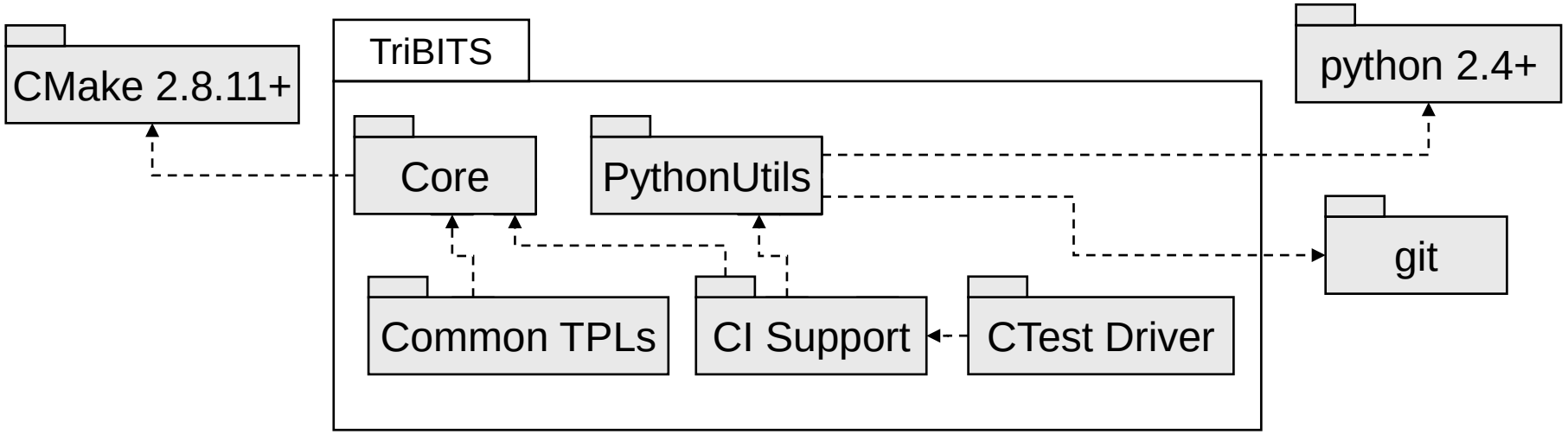
- Changes made to Trilinos/cmake/tribits/ and committed to 'master' branch are **not** overwritten!
- Quick urgent changes can be made directly to Trilinos copy of TriBITS by Trilinos developers!
- TriBITS Maintainer will address merge conflicts with new snapshots
- Commits can be pulled off with 'git format-patch' and applied to the TriBITS repo using 'git am'.
- Contributions to TriBITS must follow Contributing to TriBITS guidelines:
  - All new features, bug fixes, and changes in behavior must have automated tests.
  - All documentation is completed

See:

- Trilinos/TriBITS development and sync: http://trac.trilinos.org/wiki/TriBITSTrilinosDev
- https://github.com/TriBITSPub/TriBITS/wiki/Contributing-to-TriBITS

Summary

TriBITS Partitioning and Dependencies



- **TriBITS Core** (tribits/core/): Core TriBITS package-based architecture for CMake projects includes configure, build, test, install, deploy (tarballs) for multi-repo projects.
- **TriBITS Python Utils** (tribits/python_utils/): Some basic Python utilities that are not specific to TriBITS (e.g. gitdist, snapshot_dir.py).
- **TriBITS CI Support** (tribits/ci_support/): Support code for pre-push continuous integration testing (e.g. checkin-test.py).
- **TriBITS CTest Driver** (tribits/ctest_driver/): Support for package-by-package testing driven by CTest submitting to CDash (e.g. TribitsCTestDriverCore.cmake).
- **TriBITS Common TPLs** (tribits/common_tpls/): Used by many independent TriBITS projects (e.g. FindTPLBLAS.cmake, FindTPLLAPACK.cmake, FindTPLHDF5.cmake, …)
- …

TriBITS Summary & Future work

Summary:

- TriBITS enables interoperability and compatibility for large distributed projects
- TriBITS subpackages are designed for dependency management
  - ⇒ Don't depend on parent packages, depend on their subpackages!
  - ⇒ Don't add libs to a parent package with subpackages!
  - ⇒ Update your Dependencies.cmake files!
- Upstream TriBITS packages can be made to build as own TriBITS CMake projects
  - E.g. Teuchos + Kokkos
- TriBITS Core is only 1.4M and 10K lines of CMake code (no other dependencies)

Future Work:

- Combining concepts of packages and TPLs for large meta-projects (TriBITS #63)
- High-level and tutorial documentation
- Several other issues (see TriBITS Issues and TriBITS Backlog)
- But once these are done => TriBITS will be a good candidate for a universal meta-build and installation system for a **very** large amount of CSE software!