Trilinos Development Team

Chris Baker
Developer of Anasazi, RBGen, Tpetra

Ross Bartlett
Lead Developer of Thyra and Stratimikos
Developer of Rythmos

Pavel Bochev
Project Lead and Developer of Intrepid

Paul Boggs
Developer of Thyra

Eric Boman
Lead Developer of Isorropia
Developer of Zoltan

Todd Coffey
Lead Developer of Rythmos

David Day
Developer of Komplex and Intrepid

Karen Devine
Lead Developer of Zoltan

Clark Dohrmann
Developer of CLAPS

Michael Gee
Developer of ML, NOX

Bob Heaphy
Lead Developer of Trilinos SQA

Mike Heroux
Trilinos Project Leader
Lead Developer of Epetra, AztecOO,
Kokkos, Komplex, IFPACK, Thyra, Tpetra
Developer of Amesos, Belos, EpetraExt, Jpetra

Ulrich Hetmaniuk
Developer of Anasazi

Robert Hoekstra
Lead Developer of EpetraExt
Developer of Epetra, Thyra, Tpetra

Russell Hooper
Developer of NOX

Vicki Howe
Lead Developer of Meros
Developer of Belos and Thyra

Jonathan Hu
Developer of ML

Sarah Knepper
Developer of Komplex

Tammy Kolda
Lead Developer of NOX

Joe Kotulsik
Lead Developer of Pliris

Rich Lehoucq
Developer of Anasazi and Belos

Kevin Long
Lead Developer of Thyra,
Developer of Teuchos

Roger Pawlowski
Lead Developer of NOX

Eric Phipps
Developer of LOCA, NOX, and Sacado

Denis Ridzal
Lead Developer of Aristos and Intrepid

Marzio Sala
Lead Developer of Didasko and IFPACK
Developer of ML, Amesos

Andrew Salinger
Lead Developer of LOCA

Paul Sexton
Developer of Epetra and Tpetra

Bill Spotz
Lead Developer of PyTrilinos
Developer of Epetra, New_Package

Ken Stanley
Lead Developer of Amesos and New_Package

Heidi Thornquist
Lead Developer of Anasazi, Belos, RBGen, and Teuchos

Ray Tuminaro
Lead Developer of ML and Meros

Jim Willenbring
Developer of Epetra and New_Package.
Trilinos library manager

Alan Williams
Lead Developer of Isorropia
Developer of Epetra, EpetraExt, AztecOO, Tpetra
Outline of Talk

- Background / Motivation / Evolution.
- New Linear Algebra Foundation:
  - Teuchos / Tpetra / Kokkos
  - Zoltan
- Next Generation Algorithm Packages
  - Anasazi / Belos
- Trilinos and Computational Topology
  - Challenges
Sandia Physics Simulation Codes

- **Element-based**
  - Finite element, finite volume, finite difference, network, etc...

- **Large-scale**
  - Billions of unknowns

- **Parallel**
  - MPI-based SPMD
  - Distributed memory

- **C++**
  - Object oriented
  - Some coupling to legacy Fortran libraries

- **Fluids**
- **Combustion**
- **MEMS**
- **Circuits**
- **Plasmas**
- **Structures**
Motivation For Trilinos

- Sandia does LOTS of solver work.
- 10 years ago …
  - Aztec was a mature package. Used in many codes.
  - FETI, PETSc, DSCPack, Spooles, ARPACK, DASPK, and many other codes were (and are) in use.
  - New projects were underway or planned in multi-level preconditioners, eigensolvers, non-linear solvers, etc…
- The challenges:
  - Little or no coordination was in place to:
    - Efficiently reuse existing solver technology.
    - Leverage new development across various projects.
    - Support solver software processes.
    - Provide consistent solver APIs for applications.
  - ASCI was forming software quality assurance/engineering (SQA/SQE) requirements:
    - Daunting requirements for any single solver effort to address alone.
Evolving Trilinos Solution

- Trilinos\(^1\) is an evolving framework to address these challenges:
  - Fundamental atomic unit is a *package*.
  - Includes core set of vector, graph and matrix classes (Epetra/Tpetra packages).
  - Provides a common abstract solver API (Thyra package).
  - Provides a ready-made package infrastructure (new_package package):
    - Source code management (cvs, bonsai).
    - Build tools (autotools).
    - Automated regression testing (queue directories within repository).
    - Communication tools (mailman mail lists).
  - Specifies requirements and suggested practices for package SQA.
- In general allows us to categorize efforts:
  - Efforts best done at the Trilinos level (useful to most or all packages).
  - Efforts best done at a package level (peculiar or important to a package).
  - Allows package developers to focus only on things that are unique to their package.

1. Trilinos loose translation: “A string of pearls”
Evolving Trilinos Solution

- Beyond a “solvers” framework
- Natural expansion of capabilities to satisfy application and research needs

- Discretization methods, AD, Mortar methods, …
# Trilinos Package Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Package(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discretizations</strong></td>
<td></td>
</tr>
<tr>
<td>Spatial Discretizations (FEM,FV,FD)</td>
<td>Intrepid</td>
</tr>
<tr>
<td>Time Integration</td>
<td>Rythmos</td>
</tr>
<tr>
<td><strong>Methods</strong></td>
<td></td>
</tr>
<tr>
<td>Automatic Differentiation</td>
<td>Sacado</td>
</tr>
<tr>
<td>Mortar Methods</td>
<td>Moertel</td>
</tr>
<tr>
<td><strong>Core</strong></td>
<td></td>
</tr>
<tr>
<td>Linear algebra objects</td>
<td>Epetra, Jpetra, Tpetra</td>
</tr>
<tr>
<td>Abstract interfaces</td>
<td>Thyra, Stratimikos, RTOp</td>
</tr>
<tr>
<td>Load Balancing</td>
<td>Zoltan, Isorropia</td>
</tr>
<tr>
<td>“Skins”</td>
<td>PyTrilinos, WebTrilinos, Star-P, <em>ForTrilinos</em></td>
</tr>
<tr>
<td>C++ utilities, (some) I/O</td>
<td>Teuchos, EpetraExt, Kokkos, Triutils</td>
</tr>
<tr>
<td><strong>Solvers</strong></td>
<td></td>
</tr>
<tr>
<td>Iterative (Krylov) linear solvers</td>
<td>AztecOO, Belos, Komplex</td>
</tr>
<tr>
<td>Direct sparse linear solvers</td>
<td>Amesos</td>
</tr>
<tr>
<td>Direct dense linear solvers</td>
<td>Epetra, Teuchos, Pliris</td>
</tr>
<tr>
<td>Iterative eigenvalue solvers</td>
<td>Anasazi</td>
</tr>
<tr>
<td>ILU-type preconditioners</td>
<td>AztecOO, IFPACK</td>
</tr>
<tr>
<td>Multilevel preconditioners</td>
<td>ML, CLAPS</td>
</tr>
<tr>
<td>Block preconditioners</td>
<td>Meros</td>
</tr>
<tr>
<td>Nonlinear system solvers</td>
<td>NOX, LOCA</td>
</tr>
<tr>
<td>Optimization (SAND)</td>
<td>MOOCHO, Aristos</td>
</tr>
</tbody>
</table>
Satisfying Our Goals: Templates

- How do we support multiple data types?
  - C++ templating of the scalar type.
  - Neither new nor difficult.
  - Compiler support is good enough now.

- This provides generic programming capability, independent of data types.

- Templating implements compile time polymorphism.

- Pro: No runtime penalty.

- Con: Potentially large compile-time penalty.
  - Compiling is a good use of multiple cores.
  - Techniques exist for alleviating this for common and user data types (specifically, explicit instantiation).
Teuchos

- Portable utility package of commonly useful tools:
  - ParameterList class: key/value pair database, recursive capabilities.
  - LAPACK, BLAS wrappers (templated on ordinal and scalar type).
  - Dense matrix and vector classes (compatible with BLAS/LAPACK).
  - FLOP counters, timers.
  - Ordinal, Scalar Traits support: Definition of ‘zero’, ‘one’, etc.
  - Reference counted pointers / arrays, and more…

- Takes advantage of advanced features of C++:
  - Templates
  - Standard Template Library (STL)

Developers: Roscoe Barlett, Kevin Long, Heidi Thornquist, Mike Heroux, Paul Sexton, Kris Kampshoff, Chris Baker
Teuchos::ScalarTraits
(Arbitrary Datatypes)

- Any datatype that defines zero, one, addition, subtraction, and multiplication can use nearly all BLAS functions
  - Some require square root or division
  - Not necessary to define full Teuchos::ScalarTraits functionality

- ARPREC
  - Uses arrays of 64-bit floating-point numbers
  - Maximum working precision (in decimal digits) must be specified before any calculations are done
    ```cpp
    mp::mp_init(200);
    ```
  - dd (32 digits), qd (64 digits), mp_int

- GMP
  - Integer, rational, and floating point numbers
  - Uses fullwords (32 or 64 bits)
  - Dynamic space allocation
Teuchos::Comm

- Teuchos::Comm is a pure virtual class:
  - Has no executable code, interfaces only.
  - Encapsulates behavior and attributes of the parallel machine.
  - Defines interfaces for basic comm services between “nodes”, e.g.:
    - collective communications
    - gather/scatter capabilities
  - Allows multiple parallel machine implementations.

- Implementation details of parallel machine confined to Comm subclasses.

- Any package that uses Teuchos::Comm has no dependence on any particular API (e.g., MPI).
Petra is Greek for “foundation”.

Petra provides a “common language” for distributed linear algebra objects (operator, matrix, vector).

- Petra provides distributed matrix and vector services.
- Epetra (Essential Petra):
  - Current production version.
  - Restricted to real, double precision arithmetic.

- Tpetra (Templated Petra):
  - Next generation C++ version.
  - Uses namespaces, and STL: Improved usability/efficiency.

- Jpetra (Java Petra):
  - Pure Java. Portable to any JVM.
  - Interfaces to Java versions of MPI, LAPACK and BLAS via interfaces.

Developers: Chris Baker, Mike Heroux, Rob Hoekstra, Alan Williams
Tpetra Abstract Interfaces

- Tpetra is a successor to Trilinos’ Epetra package.
- These classes provide data services for many other packages in the Trilinos project (e.g., linear solvers, eigensolvers, non-linear solvers, preconditioners).
- Tpetra centered around the following interfaces:
  - `Comm` objects provide communication between nodes.
  - `DistObject` efficiently communicates data for distributed objects.
  - `Map` describes layout of data in distributed objects.
  - Linear algebra object interfaces (Operator, Vector) for writing abstract numerical algorithms.
Kokkos Node Package

- Trilinos/Kokkos: Trilinos compute node package.
- Generic Node object defines:
  - Memory structures for parallel buffers
  - Parallel computation routines (e.g., parallel_for, parallel_reduce)
- Kokkos also employs this API to provide local linear algebra objects for use in Tpetra distributed objects.

Example:
Kokkos::LocalCrsMatrix<int,double,NODE> lclA;
lclA.submitEntries(...); // fill the matrix
Kokkos::SparseMatVec<int,double,NODE> multOp(lclA);
Kokkos::LocalMultiVector<int,double,NODE> lclX(...), lclY(...);
multOp.apply(lclX,lclY); // apply the matrix operator
Zoltan

Data Services for Dynamic Applications

- Dynamic load balancing
- Graph coloring
- Data migration
- Matrix ordering

Partitioners:

Geometric (coordinate-based) methods:
- Recursive Coordinate Bisection (Berger, Bokhari)
- Recursive Inertial Bisection (Taylor, Nour-Omid)
- Space Filling Curves (Peano, Hilbert)
- Refinement-tree Partitioning (Mitchell)

Hypergraph and graph (connectivity-based) methods:
- Hypergraph Repartitioning PaToH (Catalyurek)
- Zoltan Hypergraph Partitioning
- ParMETIS (U. Minnesota)
- Jostle (U. Greenwich)

Developers: Karen Devine, Eric Boman, Robert Heaphy
Belos

Next-generation linear solver library, written in templated C++.

- Provide a generic framework for developing iterative algorithms for solving large-scale, linear problems.
- Algorithm implementation is accomplished through the use of traits classes and abstract base classes:
  - Operator-vector products: Belos::MultiVecTraits, Belos::OperatorTraits
  - Orthogonalization: Belos::OrthoManager, Belos::MatOrthoManager
  - Status tests: Belos::StatusTest, Belos::StatusTestResNorm
  - Iteration kernels: Belos::Iteration
  - Linear solver managers: Belos::SolverManager

- AztecOO provides solvers for $AX = b$, what about solvers for:
  - Simultaneously solved systems w/ multiple-RHS: $AX = B$
  - Sequentially solved systems w/ multiple-RHS: $AX_i = B_i$, $i=1,...,t$
  - Sequences of multiple-RHS systems: $A_iX_i = B_i$, $i=1,...,t$

- Many advanced methods for these types of linear systems
  - Block methods: block GMRES [Vital], block CG/BICG [O'Leary]
  - “Seed” solvers: hybrid GMRES [Nachtigal, et al.]
  - Restarting techniques, orthogonalization techniques, ...

Developers: Heidi Thornquist, Mike Heroux, Mike Parks, Rich Lehoucq, Teri Barth
Anasazi

- Next-generation eigensolver library, written in templated C++.

- Provide a generic framework for developing iterative algorithms for solving large-scale eigenproblems.

- Algorithm implementation is accomplished through the use of traits classes and abstract base classes:
  - Operator-vector products: Anasazi::MultiVecTraits, Anasazi::OperatorTraits
  - Orthogonalization: Anasazi::OrthoManager, Anasazi::MatOrthoManager
  - Status tests: Anasazi::StatusTest, Anasazi::StatusTestResNorm
  - Iteration kernels: Anasazi::EigenSolver
  - Eigensolver managers: Anasazi::SolverManager
  - Eigenproblem: Anasazi::Eigenproblem
  - Sort managers: Anasazi::SortManager

- Currently has solver managers for three eigensolvers:
  - Block Krylov-Schur
  - Block Davidson
  - LOBPCG

- Can solve:
  - standard and generalized eigenproblems
  - Hermitian and non-Hermitian eigenproblems
  - real or complex-valued eigenproblems

Developers: Heidi Thornquist, Mike Heroux, Chris Baker, Rich Lehoucq, Ulrich Hetmaniuk
Linear Algebra Interface

- **MultiVecTraits<ST,MV>**
  - Interface to define the linear algebra required by most iterative solvers:
    - creational methods
    - dot products, norms
    - update methods
    - initialize / randomize
  - Implementations:
    - `MultiVecTraits<double,Epetra_MultiVector>`
    - `MultiVecTraits<ST,Thyra::MultiVectorBase<ST>>`

- **OperatorTraits<ST,MV,OP>**
  - Interface to enable the application of an operator to a multivector.
  - Implementations:
    - `OperatorTraits<double,Epetra_MultiVector,Epetra_Operator>`
    - `OperatorTraits<ST,Thyra::MultiVectorBase<ST>,Thyra::LinearOpBase<ST>>`
Trilinos & Computational Topology

- Much effort has been spent on the real field …

- Challenges:
  - Tpetra operate over rationals, integers, $\mathbb{Z}_2$, {-1, 0, +1}
    - Develop new Kokkos node for local computation
    - Using GMP may be problematic for parallelism

  - Develop new and/or standard algorithms in Trilinos?

  - Leverage algorithms from LinBox
    - Better understanding of current linear algebra design
    - Similarity in “traits” mechanisms
Trilinos and related packages are available via LGPL.

Current release (9.0) is “click release”. Unlimited availability.
- 3100+ Downloads (not including internal Sandia users).
- 3880 registered users:
  - 61% university, 11% industry, 15% gov’t.
  - 38% European, 33% US, 15% Asian.


Trilinos Awards:
- 2004 R&D 100 Award.
- SC2004 HPC Software Challenge Award.
- Sandia Team Employee Recognition Award.
- Lockheed-Martin Nova Award Nominee.

More information:

7th Annual Trilinos User Group Meeting in November 2009 @ SNL
- talks available for download