Trilinos Overview

Roscoe A. Bartlett
Department of Optimization and Uncertainty Estimation
Sandia National Laboratories

Challenges and Opportunities at the Interfaces of Scientific Computing and Computational Geodynamics

October 17, 2006
Computational Sciences at Sandia: PDEs and More …

- Chemically reacting flows
- Climate modeling
- Combustion
- Compressible flows
- Computational biology
- Circuit modeling
- Inhomogeneous fluids
- Materials modeling
- MEMS modeling
- Seismic imaging
- Shock and multiphysics
- Structural dynamics
- Heat transfer
- Network modeling

Multi-Physics is a Major Theme!
Motivation For Trilinos

• Sandia does LOTS of solver work.

• When Mike Heroux started at Sandia in May 1998:
  – 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.
Overview of Trilinos

Trilinos\(^1\) is being developed to:

- Provide a **suite of numerical solvers** to support **massively parallel** predictive simulation capabilities for Sandia’s customers
- Provide a **decoupled and scalable development environment** to allow for **algorithmic research** that can transition to a **production capability**
  
  => “Package”
- Provide **support for growing SQA requirements**
- **Strategic Goals?**

At its most basic level Trilinos provides:

- A **common source code repository** and management system (CVS based)
- A **scalable configuration and build support** (autoconf/automake based)
- A common infrastructure for SQA
  - Bug reporting and tracking (i.e. Bugzilla)
  - Automated **regression testing and reporting** (test harness, results emails and webpage)
- Developer and user **communication** (i.e. Mailman email lists)
- Common integrated **documentation** system (Trilinos website and Doxygen)

**Trilinos website**

http://software.sandia.gov/trilinos

1. Trilinos loose translation: “A string of pearls”
Trilinos Development Team

Ross Bartlett
Lead Developer of Thyra and MOOCHO
Developer of Rythmos

Paul Boggs
Developer of Thyra

Todd Coffey
Lead Developer of Rythmos

Jason Cross
Developer of Jpetra

David Day
Developer of Komplex

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 Howle
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 Kotulska
Lead Developer of Pliris

Rich Lehoucq
Developer of Anasazi and Belos

Kevin Long
Lead Developer of Thyra,
Developer of Belos and Teuchos

Roger Pawlowski
Lead Developer of NOX

Michael Phenow
Trilinos Webmaster
Lead Developer of New_Package

Eric Phipps
Developer of LOCA and NOX

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 and Teuchos

Ray Tuminaro
Lead Developer of ML and Meros

Jim Willenbring
Developer of Epetra and New_Package.
Trilinos library manager

Alan Williams
Developer of Epetra, EpetraExt, AztecOO, Tpetra
Trilinos Strategic Goals

- **Scalable Solvers**: As problem size and processor counts increase, the cost of the solver will remain a nearly fixed percentage of the total solution time.

- **Hardened Solvers**: Never fail unless problem essentially unsolvable, in which case we diagnose and inform the user why the problem fails and provide a reliable measure of error.

- **Full Vertical Coverage**: Provide leading edge capabilities from basic linear algebra to transient and optimization solvers.

- **Grand Universal Interoperability**: All Trilinos packages will be interoperable, so that any combination of solver packages that makes sense algorithmically will be possible within Trilinos.

- **Universal Solver RAS**: Trilinos will be:
  - **Reliable**: Leading edge hardened, scalable solutions for each of these applications
  - **Available**: Integrated into every major application at Sandia
  - **Serviceable**: Easy to maintain and upgrade within the application environment.

Courtesy of Mike Heroux, Trilinos Project Leader
Outline

• Trilinos Overview [Just finished this]

• Abstract Numerical Algorithms (ANAs), and Interoperability

• Trilinos Global and Local (Package) Architecture

• Nonlinear Model Evaluator => Nonlinear Applications/Solvers

• Wrapping It Up
Outline

• Trilinos Overview Overview

• Abstract Numerical Algorithms (ANAs), and Interoperability

• Trilinos Global and Local (Package) Architecture

• Nonlinear Model Evaluator => Nonlinear Applications/Solvers

• Wrapping It Up
Categories of Abstract Problems and Abstract Algorithms

- **Linear Problems:** Given linear operator (matrix) $A \in \mathbb{R}^{n\times n}$
  - **Linear equations:** Solve $Ax = b$ for $x \in \mathbb{R}^n$
  - **Eigen problems:** Solve $Av = \lambda v$ for (all) $v \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$

- **Nonlinear Problems:** Given nonlinear operator $f(x, p) \in \mathbb{R}^{n+m} \rightarrow \mathbb{R}^n$
  - **Nonlinear equations:** Solve $f(x) = 0$ for $x \in \mathbb{R}^n$
  - **Stability analysis:** For $f(x, p) = 0$ find space $p \in P$ such that $\frac{\partial f}{\partial x}$ is singular

- **Transient Nonlinear Problems:**
  - **DAEs/ODEs:** Solve $f(\dot{x}(t), x(t), t) = 0, t \in [0, T], x(0) = x_0, \dot{x}(0) = x_0'$
    for $x(t) \subset \mathbb{R}^n, t \subset [0, T]$

- **Nonlinear Optimization Problems:**
  - **Unconstrained:** Find $p \in \mathbb{R}^m$ that minimizes $g(p)$
  - **Constrained:** Find $x \in \mathbb{R}^n$ and $p \in \mathbb{R}^m$ that: minimizes $g(x, p)$
    such that $f(x, p) = 0$

**Trilinos Packages**

- Belos
- Anasazi
- NOX
- LOCA
- Rythmos
- MOOCHO
What is an abstract numerical algorithm (ANA)?

An ANA is a numerical algorithm that can be expressed abstractly solely in terms of vectors, vector spaces, linear operators, and other abstractions built on top of these without general direct data access or any general assumptions about data locality.

**Example: Linear Conjugate Gradients**

Given:
- \( A \in \mathcal{X} \rightarrow \mathcal{X} \): s.p.d. linear operator
- \( b \in \mathcal{X} \): right hand side vector

Find vector \( x \in \mathcal{X} \) that solves \( Ax = b \)

**Linear Conjugate Gradient Algorithm**

Compute \( r^{(0)} = b - Ax^{(0)} \) for the initial guess \( x^{(0)} \).

For \( i = 1, 2, \ldots \):
- \( \rho_{i-1} = \langle r^{(i-1)}, r^{(i-1)} \rangle \)
- \( \beta_{i-1} = \frac{\rho_{i-1}}{\rho_{i-2}} \) \((\beta_0 = 0)\)
- \( p^{(i)} = r^{(i-1)} + \beta_{i-1} p^{(i-1)} \) \((p^{(1)} = r^{(1)})\)
- \( q^{(i)} = Ap^{(i)} \)
- \( \gamma_i = \langle p^{(i)}, q^{(i)} \rangle \)
- \( \alpha_i = \frac{\rho_{i-1}}{\gamma_i} \)
- \( x^{(i)} = x^{(i-1)} + \alpha_i p^{(i)} \)
- \( r^{(i)} = r^{(i-1)} - \alpha_i q^{(i)} \)

Check convergence; continue if necessary.

**Key Points**
- ANAs can be very mathematically sophisticated!
- ANAs can be extremely reusable!
- Flexibility needed to achieve high performance!

**Types of operations**
- Linear operator applications
- Vector-vector operations
- Scalar operations
- Scalar product \( \langle x, y \rangle \) defined by vector space

**Types of objects**
- Linear Operators: \( A \)
- Vectors: \( r, x, p, q \)
- Scalars: \( \rho, \beta, \gamma, \alpha \)
- Vector spaces: \( \mathcal{X} \)
**Trilinos Strategic Goals**

- **Scalable Solvers**: As problem size and processor counts increase, the cost of the solver will remain a nearly fixed percentage of the total solution time.

- **Hardened Solvers**: Never fail unless problem essentially unsolvable, in which case we diagnose and inform the user why the problem fails and provide a reliable measure of error.

- **Full Vertical Coverage**: Provide leading edge capabilities from basic linear algebra to transient and optimization solvers.

- **Grand Universal Interoperability**: All Trilinos packages will be interoperable, so that any combination of solver packages that makes sense algorithmically will be possible within Trilinos.

- **Universal Solver RAS**: Trilinos will be:
  - **Reliable**: Leading edge hardened, scalable solutions for each of these applications
  - **Available**: Integrated into every major application at Sandia
  - **Serviceable**: Easy to maintain and upgrade within the application environment.

*Thyra is being developed to address this issue*

**Courtesy of Mike Heroux, Trilinos Project Leader**
Interoperability is Especially Important to Optimization

Numerous interactions exist between layered abstract numerical algorithms (ANAs) in a transient optimization problem.

What is needed to solve problem?
- Standard interfaces to break $O(N^2)$ 1-to-1 couplings
  - Operators/vectors
  - Linear Solvers
  - Nonlinear solvers
  - Transient solvers
  - etc.

Thyra is being developed to address interoperability of ANAs

Key Points
- Higher level algorithms, like optimization, require a lot of interoperability
- Interoperability and layering must be “easy” or these configurations will not be achieved in practice
Outline

• Trilinos Overview

• Abstract Numerical Algorithms (ANAs), and Interoperability

• Trilinos Global and Local (Package) Architecture

• Nonlinear Model Evaluator => Nonlinear Applications/Solvers

• Wrapping It Up
Trilinos Packages

- Trilinos is a collection of Packages
- Each package is:
  - Focused on important, state-of-the-art algorithms in its problem regime
  - Developed by a small team of domain experts
  - Self-contained: No explicit dependencies on any other software packages (with some special exceptions)
  - Configurable/buildable/document on its own
- Sample packages: AztecOO, ML, IFPACK, …
- Special package collections:
  - Petra (Epetra, Tpetra): Distributed-Memory Vectors, Matrices, …
  - Thyra: Abstract Interfaces and related tools
  - Teuchos: Common C++ Utilities
- Global vs. Package Responsibilities:
  - Useful to most or all packages -> best done at the Trilinos level
  - Peculiar or important to single package -> best done at a package level
  - Allows package developers to focus only on things that are unique to their package
<table>
<thead>
<tr>
<th>Capabilities</th>
<th>Package(s)</th>
<th>Trilinos Package Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>C++ utilities, (some) I/O</td>
<td>Teuchos, EpetraExt, Kokkos</td>
<td>Trilinos 7.0 September 2006</td>
</tr>
<tr>
<td>Parallel linear algebra objects</td>
<td>Epetra, Jpetra, Tpetra</td>
<td></td>
</tr>
<tr>
<td>Krylov linear solvers</td>
<td>AztecOO, Belos, Komplex</td>
<td></td>
</tr>
<tr>
<td>ILU-type preconditioners</td>
<td>AztecOO, IFPACK</td>
<td></td>
</tr>
<tr>
<td>Multilevel preconditioners</td>
<td>ML, CLAPS</td>
<td></td>
</tr>
<tr>
<td>Eigen problems</td>
<td>Anasazi</td>
<td></td>
</tr>
<tr>
<td>Block preconditioners</td>
<td>Meros</td>
<td></td>
</tr>
<tr>
<td>Direct sparse linear solvers</td>
<td>Amesos</td>
<td></td>
</tr>
<tr>
<td>Direct dense solvers</td>
<td>Epetra, Teuchos, Pliris</td>
<td></td>
</tr>
<tr>
<td>Abstract interfaces</td>
<td>Thyra</td>
<td></td>
</tr>
<tr>
<td>Nonlinear system solvers</td>
<td>NOX, LOCA, CAPO</td>
<td></td>
</tr>
<tr>
<td>Time Integrators/DAEs</td>
<td>Rythmos</td>
<td></td>
</tr>
<tr>
<td>Trilinos Tutorial</td>
<td>Didasko</td>
<td></td>
</tr>
<tr>
<td>“Skins”</td>
<td>PyTrilinos, WebTrilinos, Star-P, Stratimikos</td>
<td></td>
</tr>
<tr>
<td>Simulation-Constrained Optimization</td>
<td>MOOCCHO</td>
<td></td>
</tr>
<tr>
<td>Archetype package</td>
<td>NewPackage</td>
<td></td>
</tr>
<tr>
<td>Other new in 7.0 release</td>
<td>Galeri, Isorropia, Moertel, RTOp</td>
<td></td>
</tr>
</tbody>
</table>
Interoperability (“Can Use”) vs. Dependence (“Depends On”)

- Although most Trilinos packages have few explicit dependencies, each package can use capabilities of *some* other packages:
  - NOX needs operator, vector and solver objects.
  - Interoperability is enabled at configure time. For example, NOX:
    - `--enable-nox-lapack` compile NOX lapack interface libraries
    - `--enable-nox-epetra` compile NOX epetra interface libraries
    - `--enable-nox-petsc` compile NOX petsc interface libraries

- *Trilinos global-level configure* script is vehicle for:
  - Establishing interoperability of Trilinos components…
  - … without compromising individual package autonomy

```
configure --enable-moocho --enable-epetra ...
```
Outline

• Trilinos Overview Overview

• Abstract Numerical Algorithms (ANAs), and Interoperability

• Trilinos Global and Local (Package) Architecture

• Nonlinear Model Evaluator => Nonlinear Applications/Solvers

• Wrapping It Up
Overview of Nonlinear Model Evaluator Interface

Approach: Develop a single, scalable interface to address many different types of numerical problems (e.g. nonlinear equations, stability/bifurcation methods, uncertainty quantification, ODEs/DAEs, optimization …) and combinations of problem types.

• (Some) Input arguments:
  • State and differential state: \( x \in \mathcal{X} \) and \( \dot{x} = \frac{dx}{dt} \in \mathcal{X} \)
  • Parameter sub-vectors: \( p_l \in \mathcal{P}_l \) for \( l = 1 \ldots N_p \)
  • Time (differential): \( t \in \mathbb{R} \)

• (Some) Output functions:
  • State function: \( (\dot{x}, x, \{p_l\}, t) \Rightarrow f \in \mathcal{F} \)
  • Auxiliary response functions: \( (\dot{x}, x, \{p_l\}, t) \Rightarrow g_j \in \mathcal{G}_j \), for \( j = 1 \ldots N_g \)
  • State/state derivative operator: \( (\dot{x}, x, \{p_l\}, t) \Rightarrow W = \alpha \frac{\partial f}{\partial x} + \beta \frac{\partial f}{\partial x} \)

Key Points
• Flexible/extendable specification of model inputs and outputs
• Address a large number steady-state and transient numerical problems and applications
• Designed for augmentation!
## Some Examples of Supported Nonlinear Problem Types

<table>
<thead>
<tr>
<th>Nonlinear equations:</th>
<th>Solve $f(x) = 0$ for $x \in \mathbb{R}^n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stability analysis:</td>
<td>For $f(x, p) = 0$ find space $p \in \mathcal{P}$ such that $\frac{\partial f}{\partial x}$ is singular</td>
</tr>
<tr>
<td>Explicit ODEs:</td>
<td>Solve $\dot{x} = f(x, t) = 0$, $t \in [0, T]$, $x(0) = x_0$, for $x(t) \in \mathbb{R}^n$, $t \in [0, T]$</td>
</tr>
<tr>
<td>DAEs/Implicit ODEs:</td>
<td>Solve $f(\dot{x}(t), x(t), t) = 0$, $t \in [0, T]$, $x(0) = x_0$, $\dot{x}(0) = x_0'$, for $x(t) \in \mathbb{R}^n$, $t \in [0, T]$</td>
</tr>
<tr>
<td>Explicit ODE Forward Sensitivities:</td>
<td>Find $\frac{\partial x}{\partial p}(t)$ such that: $\dot{x} = f(x, p, t) = 0$, $t \in [0, T]$, $x(0) = x_0$, for $x(t) \in \mathbb{R}^n$, $t \in [0, T]$</td>
</tr>
<tr>
<td>DAE/Implicit ODE Forward Sensitivities:</td>
<td>Find $\frac{\partial x}{\partial p}(t)$ such that: $f(\dot{x}(t), x(t), p, t) = 0$, $t \in [0, T]$, $x(0) = x_0$, $\dot{x}(0) = x_0'$, for $x(t) \in \mathbb{R}^n$, $t \in [0, T]$</td>
</tr>
<tr>
<td>Unconstrained Optimization:</td>
<td>Find $p \in \mathbb{R}^m$ that minimizes $g(p)$</td>
</tr>
<tr>
<td>Constrained Optimization:</td>
<td>Find $x \in \mathbb{R}^n$ and $p \in \mathbb{R}^m$ that: minimizes $g(x, p)$ such that $f(x, p) = 0$</td>
</tr>
<tr>
<td>ODE Constrained Optimization:</td>
<td>Find $x(t) \in \mathbb{R}^n$ in $t \in [0, T]$ and $p \in \mathbb{R}^m$ that: minimizes $\int_0^T g(x(t), p)$ such that $\dot{x} = f(x(t), p, t) = 0$, on $t \in [0, T]$ where $x(0) = x_0$</td>
</tr>
</tbody>
</table>
Nonlinear Algorithms and Applications: Thyra & Model Evaluator!

Nonlinear ANA Solvers in Trilinos

Key Points

• Provide single interface from nonlinear ANAs to applications
• Provide single interface for applications to implement to access nonlinear ANAs
• Provides shared, uniform access to linear solver capabilities
• Once an application implements support for one ANA, support for other ANAs can quickly follow
Outline

• Trilinos Overview Overview

• Abstract Numerical Algorithms (ANAs), and Interoperability

• Trilinos Global and Local (Package) Architecture

• Nonlinear Model Evaluator => Nonlinear Applications/Solvers

• Wrapping It Up
SciDAC-2 TOPS Project: Trilinos & PETSc Interoperability

- Five year (FY06-FY11), multi-institution project
- Trilinos tie to TOPS
  - Development of Fortran 90 interfaces to Trilinos
  - Multi-institution solver interoperability (Thyra-based?)
    - PETSc ↔ Trilinos Interoperability

- Result!
  - Application developers will be able to use any software they would like to define their problem (e.g. PETSc, Trilinos etc.)
  - Then they use any solver package they want (e.g. PETSc, HYPRE, Trilinos etc.) to solve their problem!
Trilinos Availability/Information

• Trilinos and related packages are available via LGPL.
• Current release (7.0) is “click release”, Unlimited availability
  – Version 7.0.0 released 9/28/06
  – 750 registered users:
    • 57% university, 11% industry, 20% gov’t
    • 35% European, 35% US, 10% 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:
  – http://software.sandia.gov/trilinos
  – Additional documentation at my website:

• 4th Annual Trilinos User Group Meeting: November 7-8, 2006 at SNL
In Conclusion: Trilinos Might be of Interest to you if …

• Your application includes non-PDE equations
• Multiple RHS, Block Krylov linear and eigen solvers needed
• Need more flexibility than just MPI
• Interested in multi-physics
• Applications are in C++ (or you just like C++)
• Interested in collaborating on Fortran interface
• Unstructured Data redistribution is important
• Multi-precision and complex useful
• Parallel Python (using MPI only) interest