Trilinos Tutorial

Damian Rouson, Nicole Lemaster
Sandia National Laboratories

Karla Morris, Xiaofeng Xu
City University of New York

Salvatore Filippone
University of Rome

Sponsors: DOE/OASCR, ONR

Collaborators/Customers:
IBM, ORNL, LANL, NETL

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.

Trilinos Contributors

Chris Baker
Ross Bartlett
Pavel Bochev
Paul Boggs
Erik Boman
Cedric Chevalier
Todd Coffey
Eric Cyr
David Day
Karen Devine
Clark Dohrmann
Salvatore Filippone
Kelly Fermoyle
David Gay
Mike Heroux
Ulrich Hetmaniuk
Robert Hoekstra
Russell Hooper
Vicki Howle

Jonathan Hu
Joe Kotulski
Rich Lehoucq
Nicole Lemaster
Kevin Long
Karla Morris
Kurtis Nusbaum
Roger Pawlowski
Brent Perschbacher
Eric Phipps
Lee Ann Riesen
Damian Rouson
Marzio Sala
Andrew Salinger
Chris Siefert
Bill Spotz
Heidi Thornquist
Ray Tuminaro
Jim Willenbring

Alan Williams
Jim Xia
Xiaofeng Xu

Past Contributors

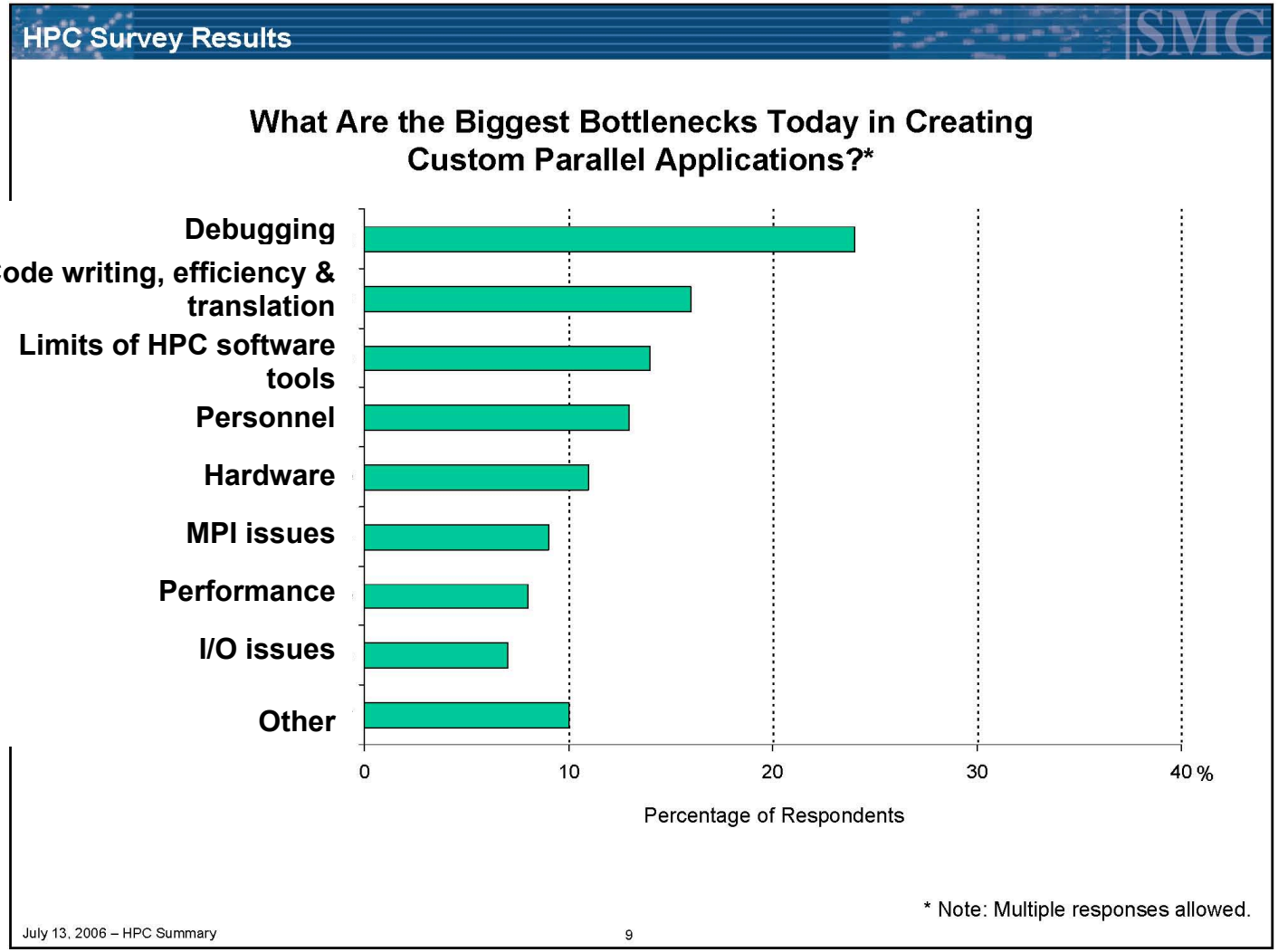
Jason Cross
Michael Gee
Esteban Guillen
Bob Heaphy
Kris Kampshoff
Ian Karlin
Sarah Knepper
Tammy Kolda
Joe Outzen
Mike Phenow
Paul Sexton
Bob Shuttleworth
Ken Stanley

Sandia National Laboratories

Outline

- Overview of Trilinos
- OOD patterns in Fortran 2003
- ForTrilinos
- Sample application

Motivation

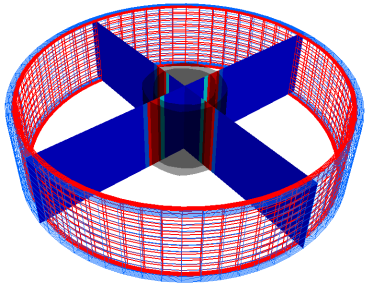


Sandia National Laboratories

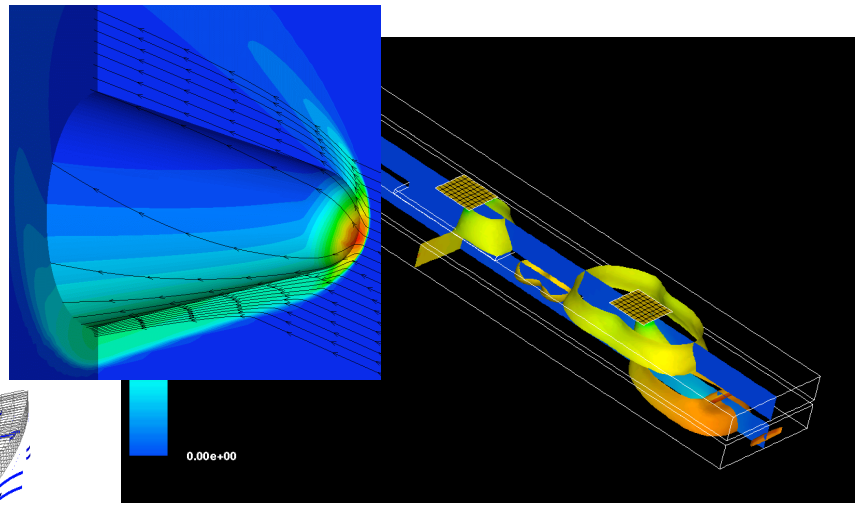
Objectives

- Trilinos:
  - Publish scalable, object-oriented numerical software using professional software engineering practices/tools:
    - Bugzilla, mailman, CMake, git, automated testing (CTest/CDash).
    - Agile development: test-driven, design patterns, frequent releases, etc.
- ForTrilinos "skin":
  - Maintain OOP philosophy of Trilinos while using idioms that feel natural to Fortran programmers.
  - Increase the adoption of Trilinos throughout DOE research communities that principally write Fortran, e.g. climate & combustion researchers.

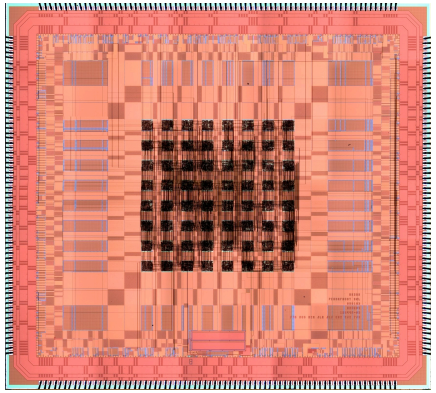
Target Problems: PDES and more…

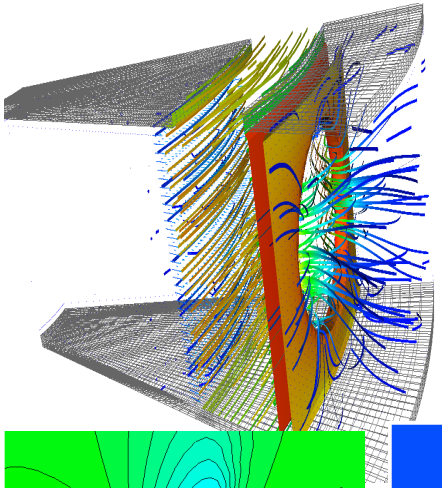


PDES

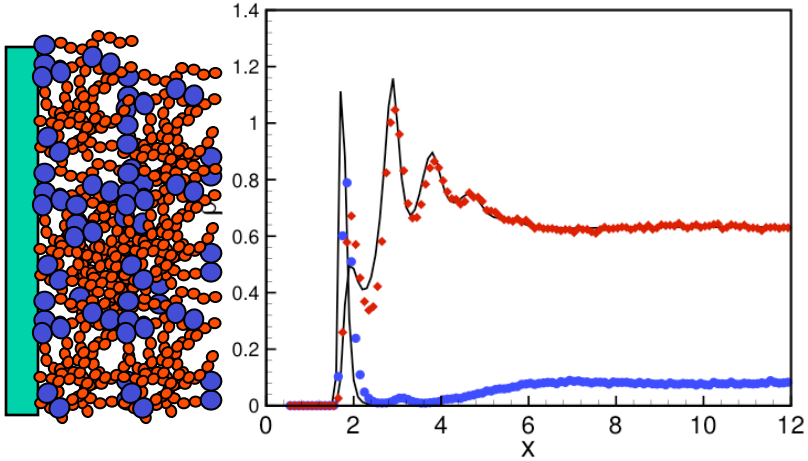


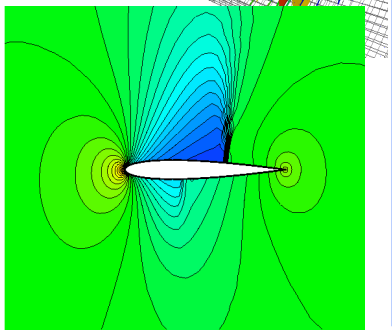
Circuits

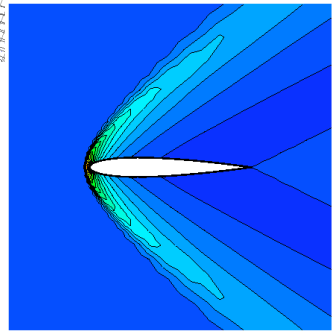




Inhomogeneous Fluids







And More…

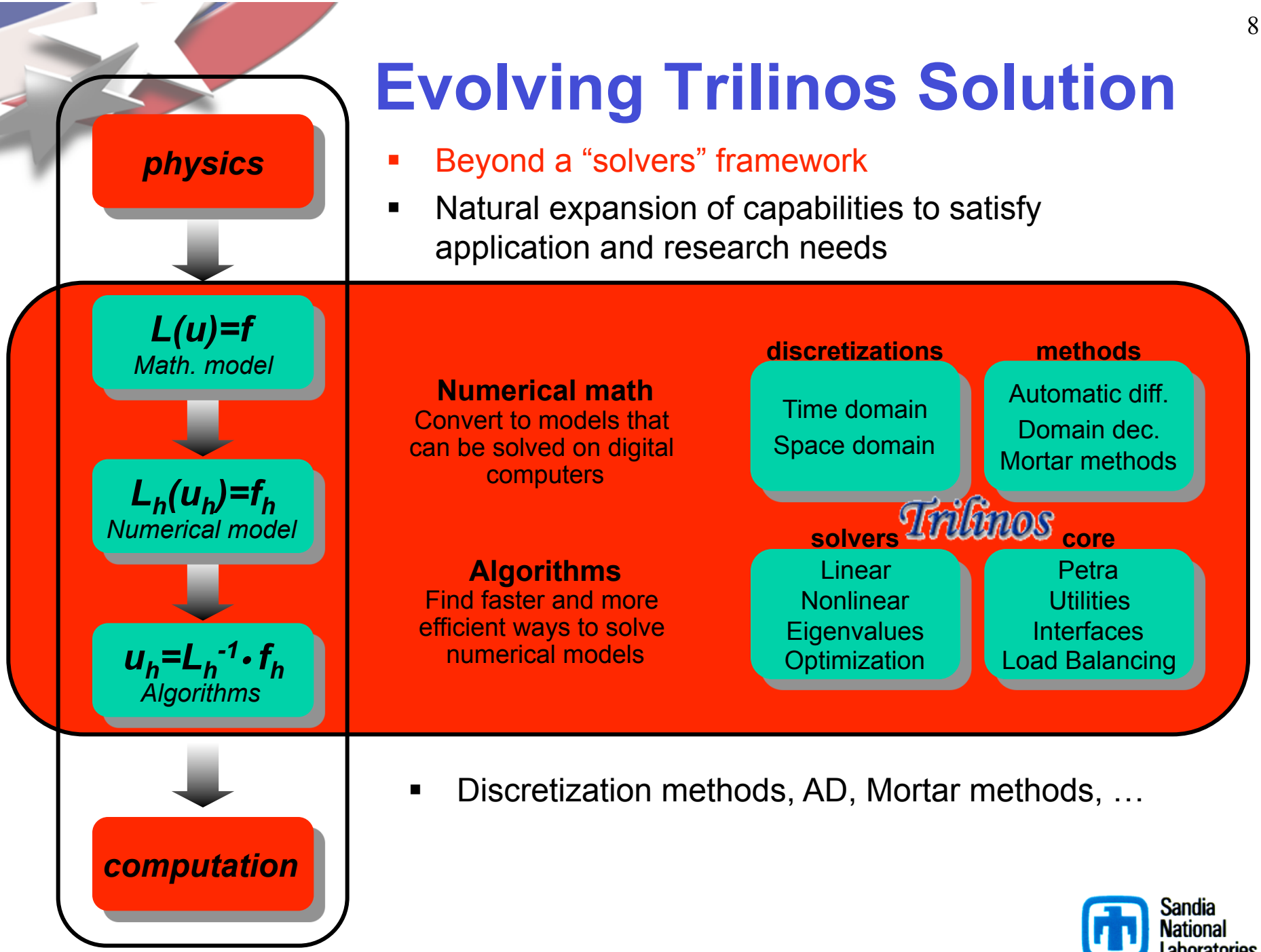
Target Platforms: Any and All
(Now and in the Future)

- Desktop: Development and more…
- Capability machines:
  - Redstorm (XT3), JaguarPF (XT5), Clusters
  - Roadrunner (Cell-based).
  - Multicore nodes.
- Parallel software environments:
  - MPI of course.
  - Threads, vectors, CUDA OpenCL, …
  - Combinations of the above.
- User "skins":
  - C++/C, Python
  - Fortran.
  - Web, CCA.

Sandia National Laboratories

Evolving Trilinos Solution

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



- Discretization methods, AD, Mortar methods, …

Sandia National Laboratories

Trilinos Package Summary

http://trilinos.sandia.gov

	Objective	Package(s)
Discretizations	Meshing & Spatial Discretizations	phdMesh, Intrepid, Pamgen, Sundance, ITAPS
	Time Integration	Rythmos
Methods	Automatic Differentiation	Sacado
	Mortar Methods	Moertel
Services	Linear algebra objects	Epetra, Jpetra, Tpetra, Kokkos
	Interfaces	Thyra, Stratimikos, RTOp, FEI, Shards
	Load Balancing	Zoltan, Isorropia
	“Skins”	PyTrilinos, WebTrilinos, ForTrilinos, Ctrilinos, Optika
	C++ utilities, I/O, thread API	Teuchos, EpetraExt, Kokkos, Triutils, ThreadPool, Phalanx
Solvers	Iterative (Krylov) linear solvers	AztecOO, Belos, Komplex
	Direct sparse linear solvers	Amesos
	Direct dense linear solvers	Epetra, Teuchos, Pliris
	Iterative eigenvalue solvers	Anasazi, Rbgen
	ILU-type preconditioners	AztecOO, IFPACK, Tifpack
	Multilevel preconditioners	ML, CLAPS
	Block preconditioners	Meros
	Nonlinear system solvers	NOX, LOCA
	Optimization (SAND)	MOOCHO, Aristos, TriKota, Globipack, Optipack
	Stochastic PDEs	Stokhos

Sandia National Laboratories

Categories of Abstract Problems and Abstract Algorithms

Trilinos Packages

- Linear Problems: Given linear operator (matrix) $A \in \mathbf{R}^{n \times n}$
  - Linear equations: Solve $Ax = b$ for $x \in \mathbf{R}^n$ **Belos**
  - Eigen problems: Solve $Av = \lambda v$ for (all) $v \in \mathbf{R}^n$ and $\lambda \in \mathbf{R}$ **Anasazi**
- Nonlinear Problems: Given nonlinear operator $c(x, u) \in \mathbf{R}^{n+m} \rightarrow \mathbf{R}^n$
  - Nonlinear equations: Solve $c(x) = 0$ for $x \in \mathbf{R}^n$ **NOX**
  - Stability analysis: For $c(x, u) = 0$ find space $u \in \mathcal{U}$ such that $\frac{\partial c}{\partial x}$ is singular **LOCA**
- 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 $x(t) \in \mathbf{R}^n, t \in [0, T]$ **Rythmos**
- Optimization Problems:
  - Unconstrained: Find $u \in \mathbf{R}^n$ that minimizes $f(u)$ **MOOCHO**
  - Constrained: Find $y \in \mathbf{R}^m$ and $u \in \mathbf{R}^n$ that:
    minimizes $f(y, u)$
    such that $c(y, u) = 0$ **Aristos**

Sandia National Laboratories

Trilinos Interoperability Mechanisms

(Acquired as Package Matures)

Package builds under Trilinos configure scripts.	➔	*Package* can be built as part of a suite of packages; cross-package interfaces enable/disable automatically
Package accepts user data as Epetra or Thyra objects	➔	Applications using Epetra/Thyra can use *package*
Package accepts parameters from Teuchos ParameterLists	➔	Applications using Teuchos ParameterLists can drive *package*
Package can be used via Thyra abstract solver classes	➔	Applications or other packages using Thyra can use *package*
Package can use Epetra for private data.	➔	*Package* can then use other packages that understand Epetra
Package accesses solver services via Thyra interfaces	➔	*Package* can then use other packages that implement Thyra interfaces
Package available via PyTrilinos	➔	*Package* can be used with other Trilinos packages via Python.

Sandia National Laboratories

Whirlwind Tour of Packages

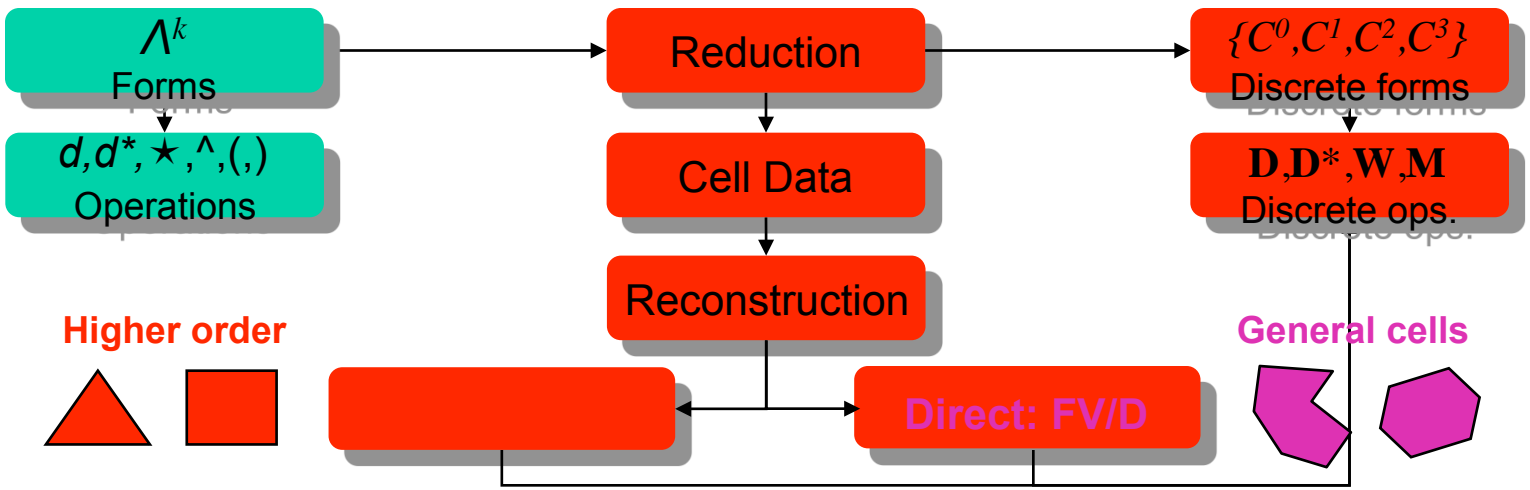
Discretizations **Methods** **Core** **Solvers**

Intrepid

Interoperable Tools for Rapid Development of Compatible Discretizations

Intrepid offers an **innovative software design** for compatible discretizations:

- allows access to FEM, FV and FD methods using a common API
- supports hybrid discretizations (FEM, FV and FD) on unstructured grids
- supports a variety of cell shapes:
  - standard shapes (e.g. tets, hexes): high-order finite element methods
  - arbitrary (polyhedral) shapes: low-order mimetic finite difference methods
- enables optimization, error estimation, V&V, and UQ using fast invasive techniques (direct support for cell-based derivative computations or via automatic differentiation)



Developers: Pavel Bochev and Denis Ridzal

Trilinos

Whirlwind Tour of Packages

Discretizations Methods Core Solvers

Sandia National Laboratories

Trilinos Common Language: Petra

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

- Petra[1] provides distributed matrix and vector services.
- Exists in basic form as an object model:
  - Describes basic user and support classes in UML, independent of language/implementation.
  - Describes objects and relationships to build and use matrices, vectors and graphs.
  - Has 3 implementations under development.

[1]Petra is Greek for “foundation”.

Petra Implementations

- Epetra (Essential Petra):
  - Current production version.
  - Restricted to real, double precision arithmetic.
  - Uses stable core subset of C++ (circa 2000).
  - Interfaces accessible to C and Fortran users.

- Tpetra (Templated Petra):
  - Next generation C++ version.
  - Templated scalar and ordinal fields.
  - 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

Sandia National Laboratories

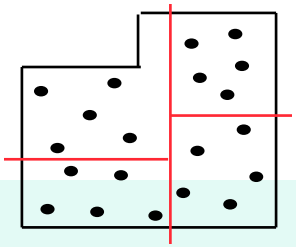
Teuchos

- 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)
- Teuchos::ParameterList:
  - Allows easy control of solver parameters.
  - XML format input/output.

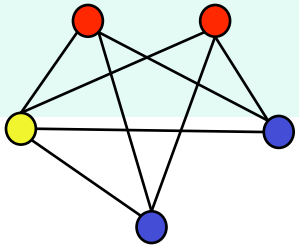
Developers: Roscoe Barlett, Kevin Long, Heidi Thornquist, Mike Heroux, Paul Sexton, Kris Kampshoff, Chris Baker

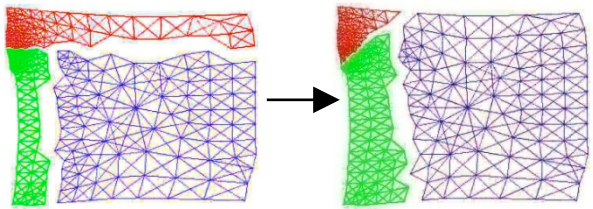
Sandia National Laboratories

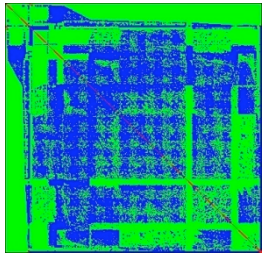
Zoltan



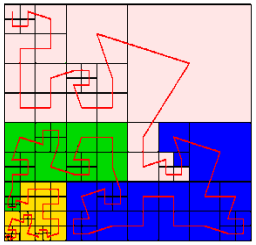
- Data Services for Dynamic Applications
  - Dynamic load balancing
  - Graph coloring
  - Data migration
  - Matrix ordering



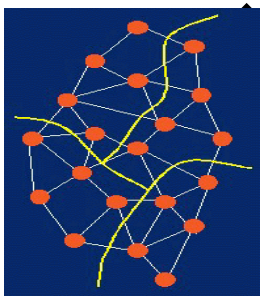




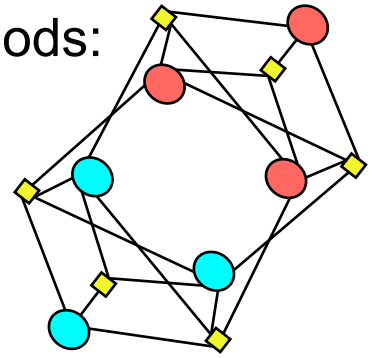
- Partitioners:



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

Sandia National Laboratories

Thyra

- High-performance, abstract interfaces for linear algebra
- Offers flexibility through abstractions to algorithm developers
- Linear solvers (Direct, Iterative, Preconditioners)
  - Abstraction of basic vector/matrix operations (dot, axpy, mv).
  - Can use any concrete linear algebra library (Epetra, PETSc, BLAS).
- Nonlinear solvers (Newton, etc.)
  - Abstraction of linear solve (solve $Ax=b$).
  - Can use any concrete linear solver library:
    - AztecOO, Belos, ML, PETSc, LAPACK
- Transient/DAE solvers (implicit)
  - Abstraction of nonlinear solve.
  - … and so on.

Developers: Roscoe Bartlett, Kevin Long

Sandia National Laboratories

"Skins"

- PyTrilinos provides Python access to Trilinos packages
- Uses SWIG to generate bindings.
- Epetra, AztecOO, IFPACK, ML, NOX, LOCA, Amesos and NewPackage are supported.

Developer: Bill Spotz

- CTrilinos: C wrapper (mostly to support ForTrilinos).
- ForTrilinos: OO Fortran interfaces.

Developers: Nicole Lemaster, Karla Morris, Damian Rouson, Salvatore Filippone, Jim Xia, Xiaofeng Xu

- WebTrilinos: Web interface to Trilinos
- Generate test problems or read from file.
- Generate C++ or Python code fragments and click-run.
- Hand modify code fragments and re-run.
- Will use during hands-on.

Developers: Ray Tuminaro, Jonathan Hu, and Marzio Sala

Whirlwind Tour of Packages

Discretizations Methods Core Solvers

Sandia National Laboratories

Amesos

- Interface to direct solvers for distributed sparse linear systems (KLU, UMFPACK, SuperLU, MUMPS, ScaLAPACK)

- Challenges:
  - No single solver dominates
  - Different interfaces and data formats, serial and parallel
  - Interface often changes between revisions
- Amesos offers:
  - A single, clear, consistent interface, to various packages
  - Common look-and-feel for all classes
  - Separation from specific solver details
  - Use serial and distributed solvers; Amesos takes care of data redistribution
  - Native solvers: KLU and Paraklete

Developers: Ken Stanley, Marzio Sala, Tim Davis

Sandia National Laboratories

MOOCHO & Aristos

- MOOCHO: Multifunctional Object-Oriented arCHitecture for Optimization
  - Large-scale invasive simultaneous analysis and design (SAND) using reduced space SQP methods.

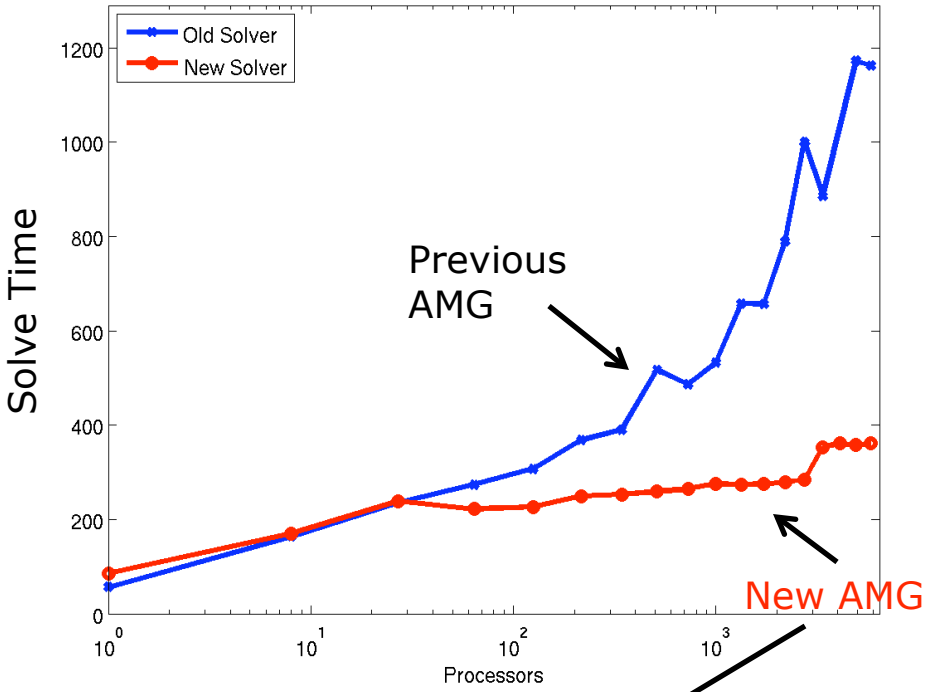
Developer: Roscoe Bartlett

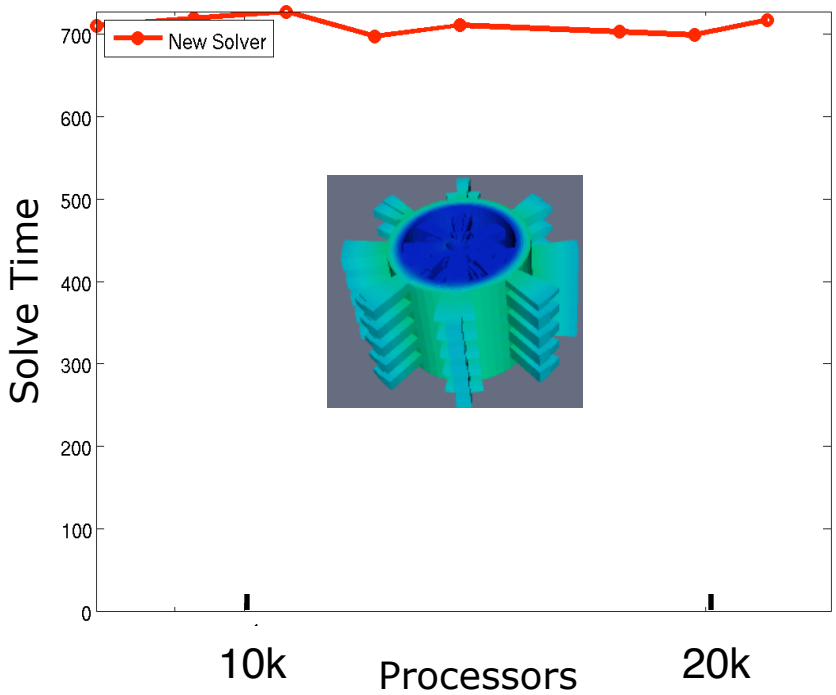
- Aristos: Optimization of large-scale design spaces
  - Invasive optimization approach based on full-space SQP methods.
  - Efficiently manages inexactness in the inner linear system solves.

Developer: Denis Ridzal

Sandia National Laboratories

: Scaling on Red Storm





- 10 times steps
- Chebyshev polynomial smoother
- Parallel load-balancing via Zoltan

Sandia National Laboratories

Multicore
(Inter-core comm. with MPI)

Nodes	Cores per node	Compute Jac+Prec		Linear Solve		Total	
		Time (sec)	Eff.	Time (sec)	Eff.	Time (sec)	Eff.
4096	1	16.9	-------	4.3	-------	21.2	------
2048	2	18.2	93%	4.5	95%	22.6	94%
1024	4	17.7	95%	4.9	88%	22.6	94%

Multi-core Efficiency Study
New 2.2 GHz Quad Cores
Cray XT3/4 (09/29/08)
Total of 4096 cores

12800x1280 mesh: ~65M unknowns; Agg = 33;
Coarse Operator: ~60K unknowns
ML: V(1,1) with ILU(1,2)/ ILU(1,2)/KLU and Petrov-Galerkin Projection

Our Largest Steady-state Simulation to Date:
1+ Billion unknowns
250 Million Quad elements
24,000 cores Cray XT3/4
Newton-GMRES / ML: PG-AMG 4 level
18 Newton steps
86 Avg. No. Linear Its. / Newton step
33 min. for solution

Sandia National Laboratories

Trilinos Availability / Information

- Trilinos and related packages are available via BSD License.
- Current release: 10.4.0. July 2010.
- 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://trilinos.sandia.gov
- Annual Forums:
  - DOE ACTS Tutorial (3rd week in August).
  - Annual Trilinos User Group Meeting in November @ SNL
    - talks available for download

Sandia National Laboratories

Useful Links

Trilinos website: http://trilinos.sandia.gov

Trilinos tutorial: http://trilinos.sandia.gov/Trilinos10.0Tutorial.pdf

Trilinos mailing lists: http://trilinos.sandia.gov/mail_lists.html

Trilinos User Group (TUG) meetings:

http://trilinos.sandia.gov/events/trilinos_user_group_2008

http://trilinos.sandia.gov/events/trilinos_user_group_2007

Outline

- Overview of Trilinos
- OOD patterns in Fortran 2003
- ForTrilinos
- Sample application

What is a Pattern?

Definition: A common solution to a recurring OOD problem.

Four essential elements:

- The name
- The problem
- The solution
- The consequences

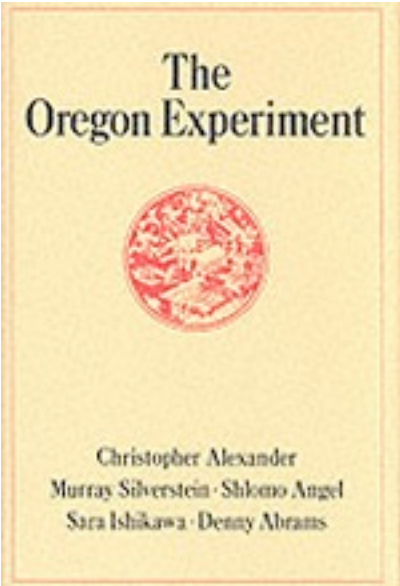
Additional elements:

- Also known as...
- Known uses.
- Related patterns.
- Sample implementation.

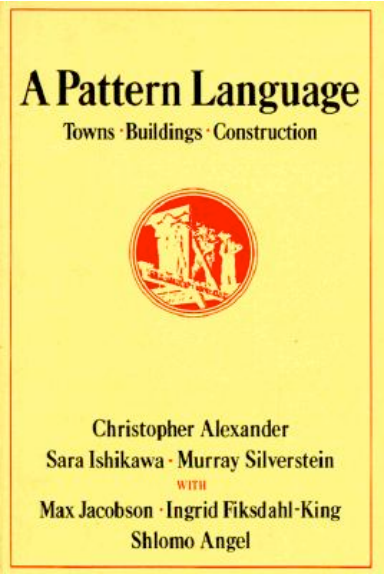
Sandia National Laboratories

Origin of Design Patterns

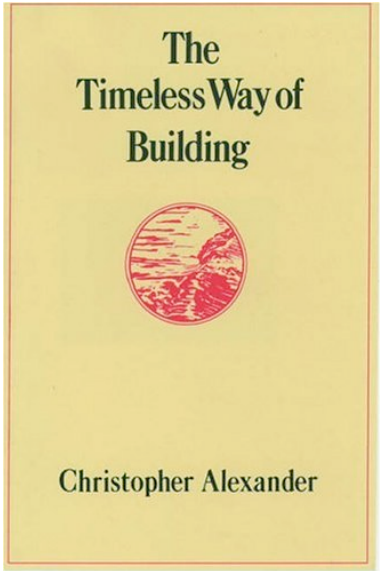
Building architecture: Alexander et al.



Vol. III
(1975)



Vol. II
(1977)



Vol. I
(1979)

Positive Outdoor Space
Julian Street Inn
Shelter for the Homeless
San Jose, CA

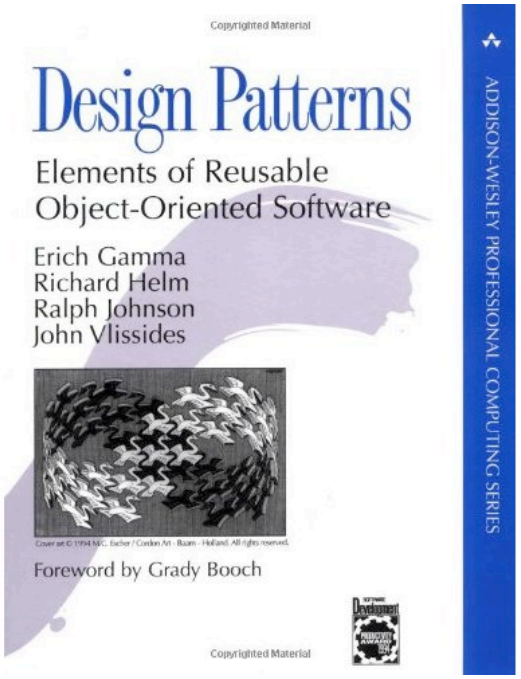
Sandia National Laboratories

OOD Patterns

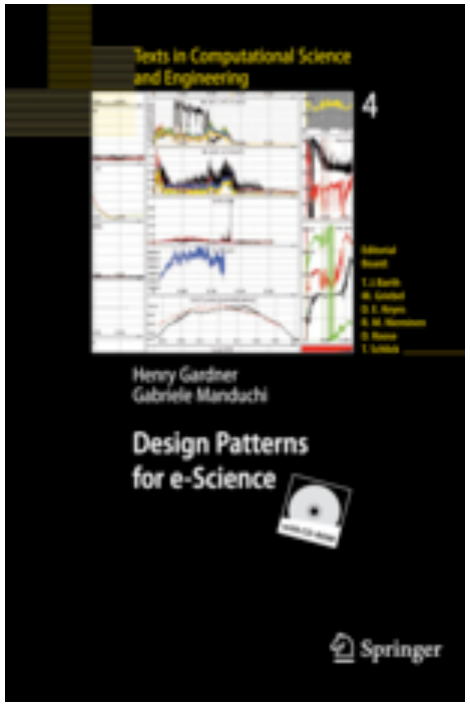
Software architecture: **Gamma et al. (1995)**

Scientific software architecture:

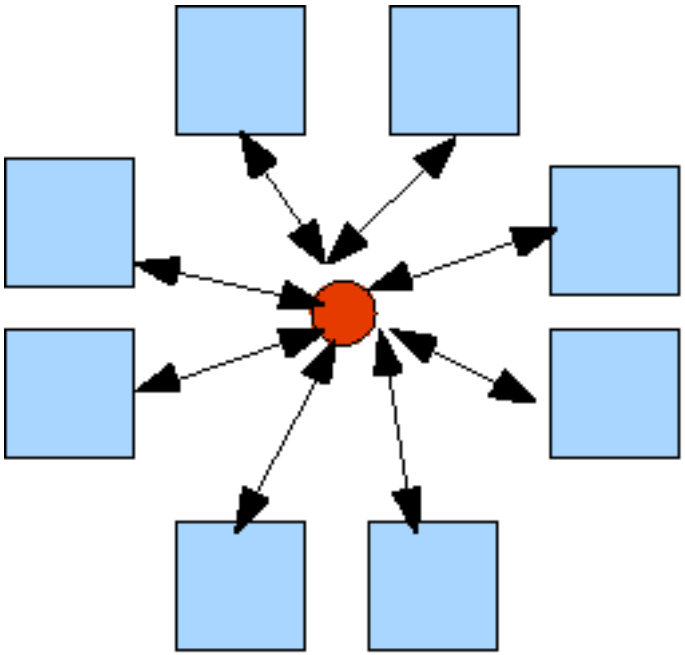
Gardner & Manduchi (2007)



1995



2007



Mediator

Sandia National Laboratories

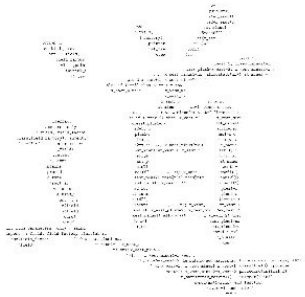
OOD Patterns in Fortran/C++

Multiphysics software architecture:

Rouson, Xia & Xu (2010)

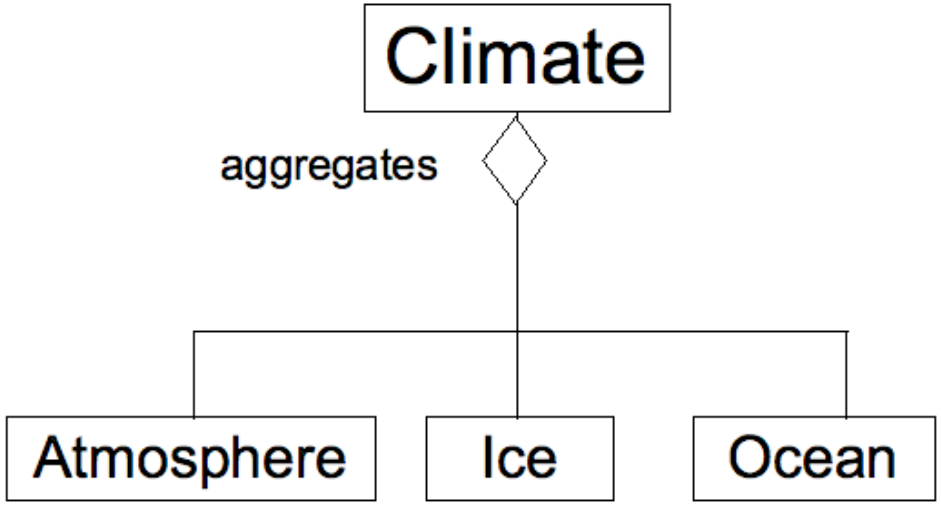
Scientific Software Design: The Object-Oriented Way

Damian W. I. Rouson
Jim Xia
Xiaofeng Xu



2011

Puppeteer:



Sandia National Laboratories

Why Fortran?

- Scientific programmers rule!
- Mathematical expressiveness:
  - User-defined operators: .div., .grad., .curl., etc.
- Multidimensional arrays
- Array operations
- Platform-agnostic parallelism (Fortran 2008)
- Automatic memory management:
  - Automatic re-sizing of arrays
  - Automatic destruction of dynamically allocated arrays/objects.

Sandia National Laboratories

Abstract Calculus Pattern

Blackboard abstraction	Software abstraction (Fortran 2003):
$u = u(x, y, z, t)$	`class(field),pointer::u,du_dt`
$u(x = 0, y, z, t) = u_0$	`call u%boundary(u0,x=0)`
$u_t \equiv \dfrac{\partial u}{\partial t}$	`u%t()`
$u^{n+1} = u^n + u_t^n \Delta t$	`u = u + u%t()*dt`
$u_t = \nu u_{xx} - u u_x$	`du_dt = nu*u%xx() - u*u%x()`

Sandia National Laboratories

Scalable Abstract Calculus

```
du_dt = nu*u%xx() - u*u%x()
```

Synchronization

Asyncrhonous

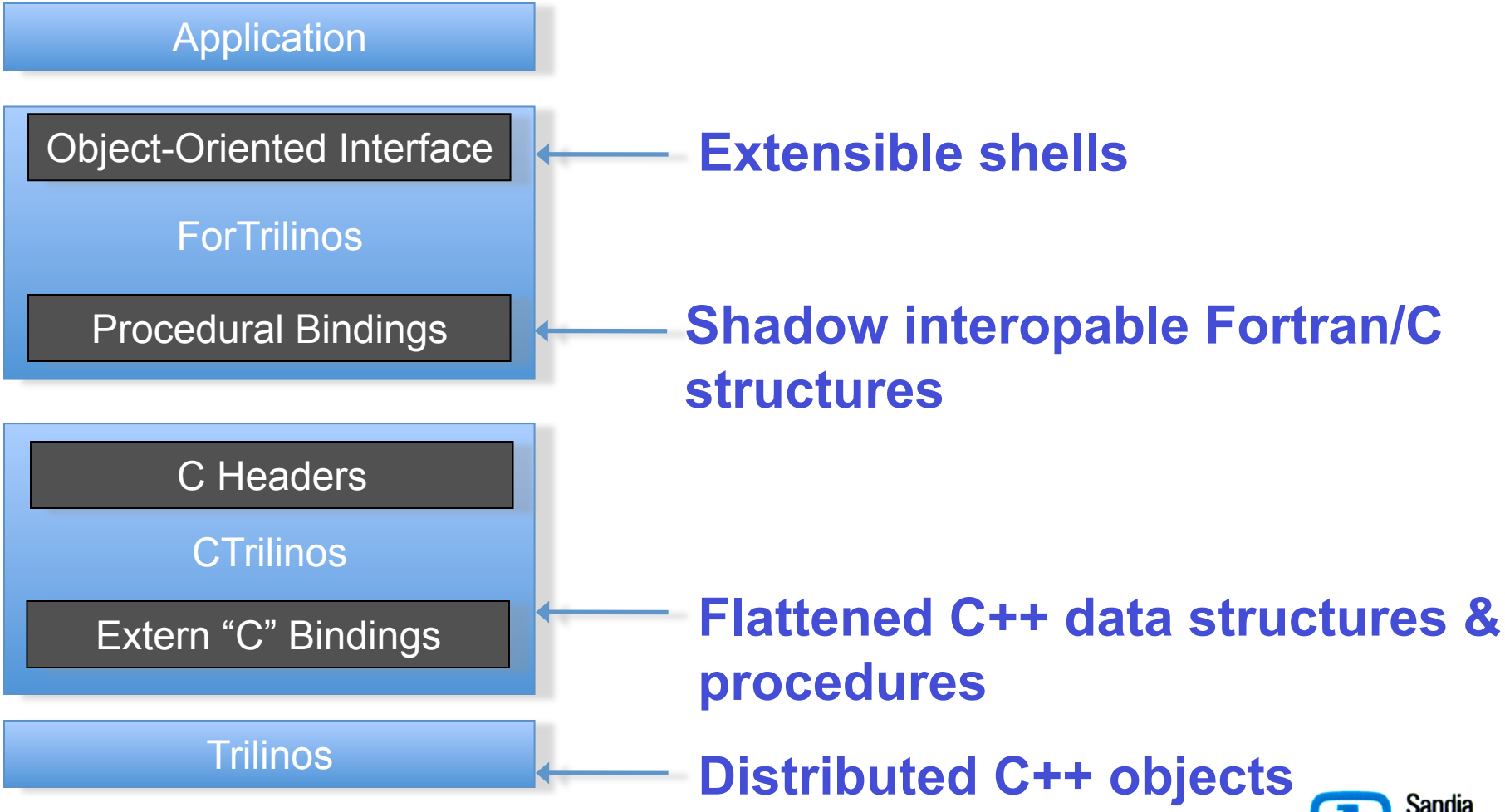
if purely functional,i.e. side-effect-free

Sandia National Laboratories

Outline

- Overview of Trilinos
- OOD patterns in Fortran 2003
- ForTrilinos
- Sample application

ForTrilinos Application Architecture



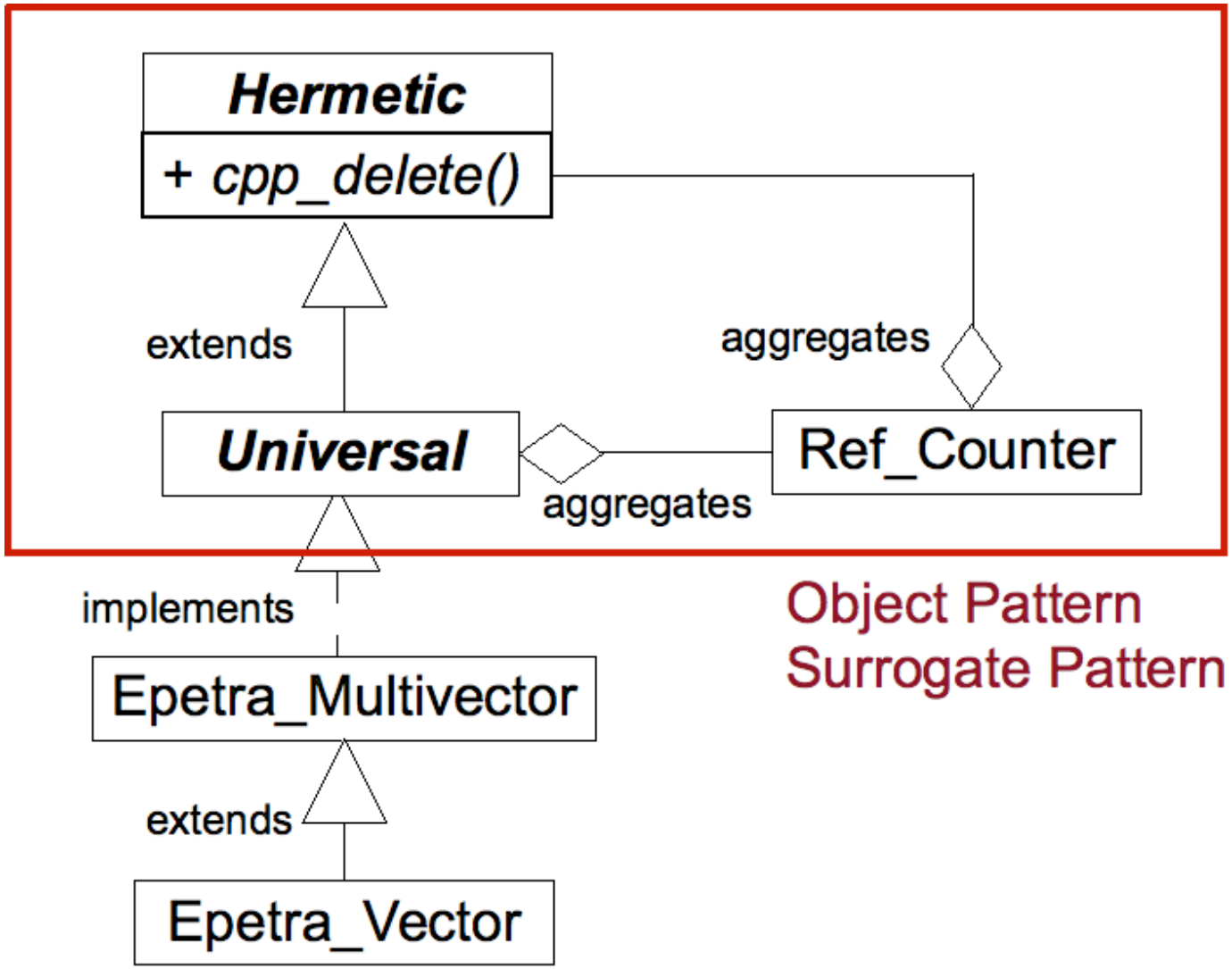
Sandia National Laboratories

Sample Fortran Main

```
program main

  use iso_c_binding,   only : c_double
  use Fepetra_MpiComm, only : epetra_mpicomm
  use FEpetra_Map,     only : epetra_map
  use FEpetra_Vector,  only : epetra_vector
  implicit none

  type(epetra_mpicomm) :: comm
  type(epetra_map)     :: map
  type(epetra_vector)  :: b

  comm = epetra_mpicomm()
  map  =
  epetra_map(numGlobalElements=64,IndexBase=1,comm=comm)
  b    = epetra_vector(map)
  call b%PutScalar(2.0_c_double)
  print *, "L2 norm of b = ",b%Norm2()

end program
```

Sandia National Laboratories

Class Model



Object Pattern
Surrogate Pattern

Sandia National Laboratories

Procedural Bindings

```
module ForTrilinos_enums
  use iso_c_binding ,only : c_int

  enum ,bind(C) enumerator ::
    FT_Epetra_SerialComm_ID, &
    FT_Epetra_Comm_ID,       &
    FT_Epetra_MpiComm_ID
  end enum

  integer(kind(c_int)) ,parameter :: ForTrilinos_Type_ID_t= c_int
  integer(kind(c_int)) ,parameter :: FT_boolean_t = c_int
  integer(FT_boolean_t) ,parameter :: FT_FALSE = 0
  integer(FT_boolean_t) ,parameter :: FT_TRUE  = 1

  type ,bind(C) :: ForTrilinos_Object_ID_t
    integer(ForTrilinos_Type_ID_t) :: type     ! Object data type
    integer(c_int)                 :: index    ! Array index
    integer(FT_boolean_t)          :: is_const ! Const status
  end type
  ! Additional structurally equivalent types…
end module
```

Sandia National Laboratories

Procedural Bindings

```
module forepetra
  use ForTrilinos_enums

  interface

    type(FT_Epetra_SerialComm_ID_t) function &
      Epetra_SerialComm_Create() &
      bind(C,name='Epetra_SerialComm_Create')
      import :: FT_Epetra_SerialComm_ID_t
    end function

    !...

  end interface

end module
```

Object-Oriented Fortran Interface

```
module FEpetra_Vector
  !...
  use forepetra ! Procedural bindings
  private       ! Hide everything by default
  public :: Epetra_Vector ! Expose type/constructors/methods
  implicit none

  type ,extends(Epetra_MultiVector) :: Epetra_Vector
    private
    type(FT_Epetra_Vector_ID_t)  :: vector_id
  contains
     procedure :: ReplaceGlobalValues
     procedure :: ExtractCopy_EpetraVector
     generic :: ExtractCopy => ExtractCopy_EpetraVector
     procedure :: get_element_EpetraVector
     generic :: get_Element => get_element_EpetraVector
  end type
  interface Epetra_MultiVector
    ! Specific constructor names
  end interface
contains ! Method impelementations
```

Sandia National Laboratories

Some Required Constructs

Construct	Cray	Gnu	G95	IBM	Intel	NAG	PG
`final`	x			x	beta		
`extends`	x	x		x	x	x	
`deferred`	x	x		x	x	x	
`select type`	x	x		x	x	x	
`abstract type`	x	x		x	x	x	
`bind(C)`	x	x	x	x	x	x	x

- Cray: Full F2003 support & partial F2008 support.
- IBM: Full F2003 support.
- Gnu: Partial F2003/F2008 support.
- Intel: Partial F2003/F2008 support with OOP in beta release.
- G95: Partial F2008 support, limited F2003 support.
- PGI: Limited F2003 w/one full-time F2003 developer.

Sandia National Laboratories

Outline

- Overview of Trilinos
- OOD patterns in Fortran 2003
- ForTrilinos
- Sample application