Panzer: A Finite Element Assembly Engine for Multiphysics Simulation

Roger Pawlowski, Eric Cyr, and John Shadid
Sandia National Laboratories

European Trilinos User Group Meeting
MArch 3rd, 2015

SAND2011-8261C

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

What is Panzer?

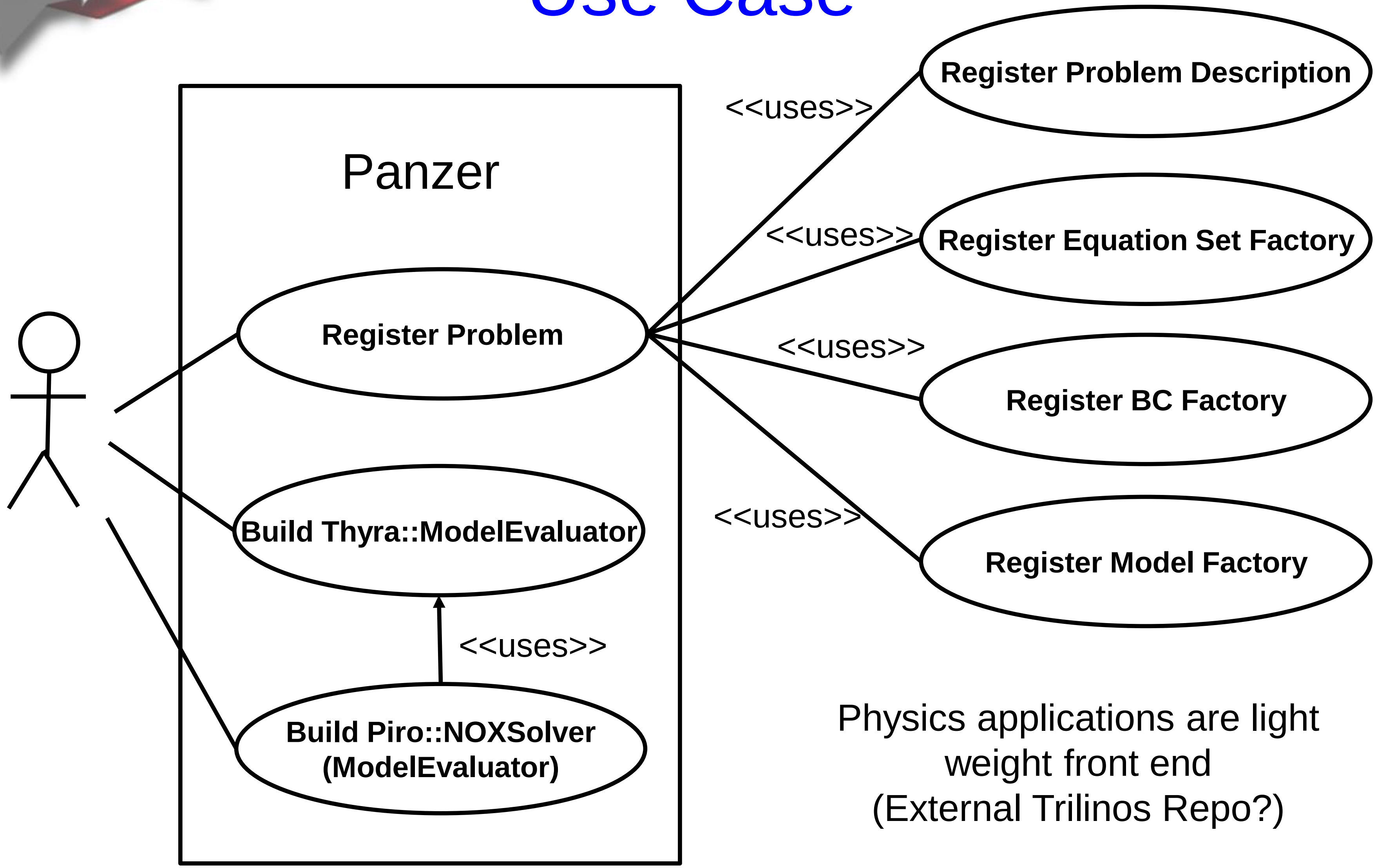
- A general finite element assembly engine for multiphysics simulation:
  - **User Physics Kernels + Problem Description = Thyra::ModelEvaluator**
    - Quantities need for advanced **solution** and **analysis** algorithms: residuals, Jacobians, parameter sensitivities, stochastic residual/Jacobians, etc.
  - A **unification** of Trilinos discretization tools: Shards, Intrepid, Phalanx, Sacado, Stokhos, (Optionally: STK, SEACAS)
  - Supports 1D, 2D, and 3D unstructured mesh calculations

- A library and a Trilinos package – NOT a terminal application

- Contains NO physics specific code
  - Generic assembly tools

- Leverages Template-based Generic Programming to assemble quantities of interest

Use Case



Physics applications are light weight front end (External Trilinos Repo?)

New Research Requirements

A Research Tool for DOE/OS: ASCR/AMR, ASCR/UQ

- **Formulations: fully coupled fully implicit, semi-implicit, FCT**

- **Compatible discretizations:**
  - Mixed basis for DOFs within element block
  - Arbitrary element types (not restricted to nodal basis)
  - “Node” specific code is eliminated (or treated as specializations)

- **Multiphysics:**
  - Fully coupled systems composed of different equation sets in different element blocks
  - Preconditioning: Approximate block factorization/physics based
  - Recent work on IMEX

- **Supports advanced analysis techniques:**
  - **Modern software techniques for advanced architectures**
  - Supports Template-based Generic Programming
  - Adjoint-based error analysis
  - Stability, bifurcation, embedded (SAND) optimization, embedded uncertainty quantification (Stokhos/PCE)

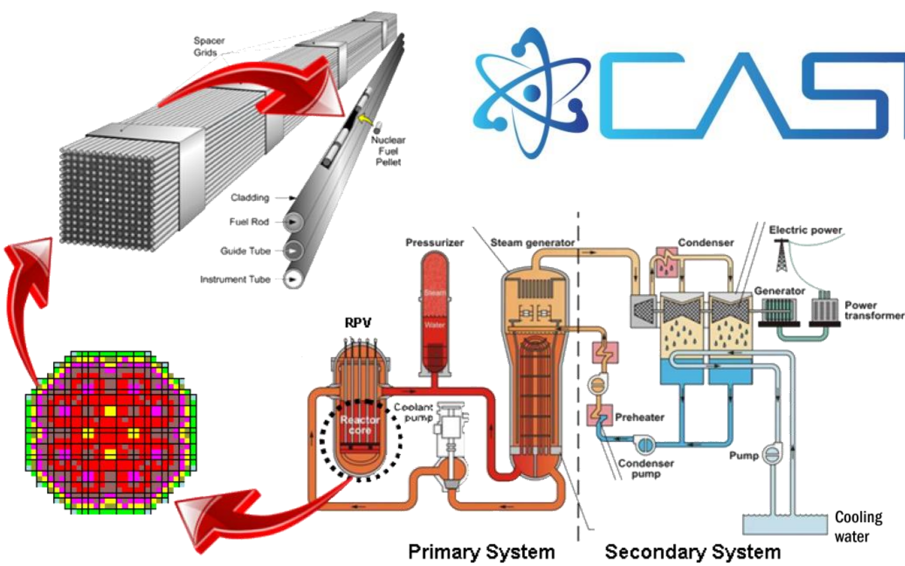
Sandia National Laboratories

Production Requirements

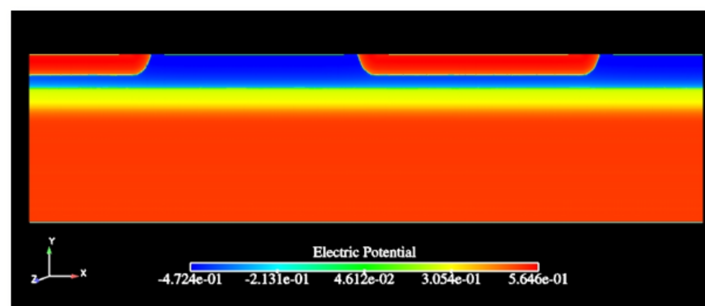
Production Quality Software (ASC, CASL)

- Strict and extensive unit testing (TDD)
- Integration with legacy code components
- NOT restricted to any mesh database or I/O format
- Control over granularity of assembly process (efficiency vs flexibility)
- Applications:
  - ASC: Semiconductor Device (Next-generation Charon) for QASPR
  - CASL: CFD component for VERA simulator

CASL



ASC™
DOE / NNSA



Sandia National Laboratories

Panzer Components

- Problem Description
  - Maps equations sets and boundary conditions into nodes of Phalanx assembly DAG.
- Assembly Engine
  - A collection of Phalanx Field Managers to control assembly
  - Produces a Model Evaluator for User
- Data Mapping Utilities
  - DOF Manager for mapping field values into linear algebra
  - Connection Manager: Abstraction of Mesh
- STK Adaptors (Optional)
  - Concrete implementation Panzer objects for using STK::Mesh and SEACAS for I/O
  - Specialized evaluators

Agile Components (A. Salinger): Trilinos has a coordinated integration effort (ASC) to support all aspects of a simulation!

Analysis Tools (*non-invasive*)
- Optimization
- Parameter Studies
- UQ (non-invasive)
- V&V, Calibration
- OUU, Reliability
- Computational Steering

Analysis Tools (*invasive*)
- Nonlinear Solver
- Time Integration
- Continuation
- Sensitivity Analysis
- Stability Analysis
- Constrained Solves
- Optimization
- UQ Solver

Linear Algebra
- Data Structures
- Iterative Solvers
- Direct Solvers
- Eigen Solver
- Preconditioners
- Matrix Partitioning
- Architecture-Dependent Kernels

Mesh Tools
- Mesh I/O
- Inline Meshing
- Partitioning
- Load Balancing
- Adaptivity
- Remeshing
- Grid Transfers
- Mesh Quality

Mesh Database
- Mesh Database
- Geometry Database
- Solution Database

Local Fill

Discretizations
- Discretization Library
- Variable Manager

Derivative Tools
- UQ / PCE Propagation
- Derivatives
- Sensitivities

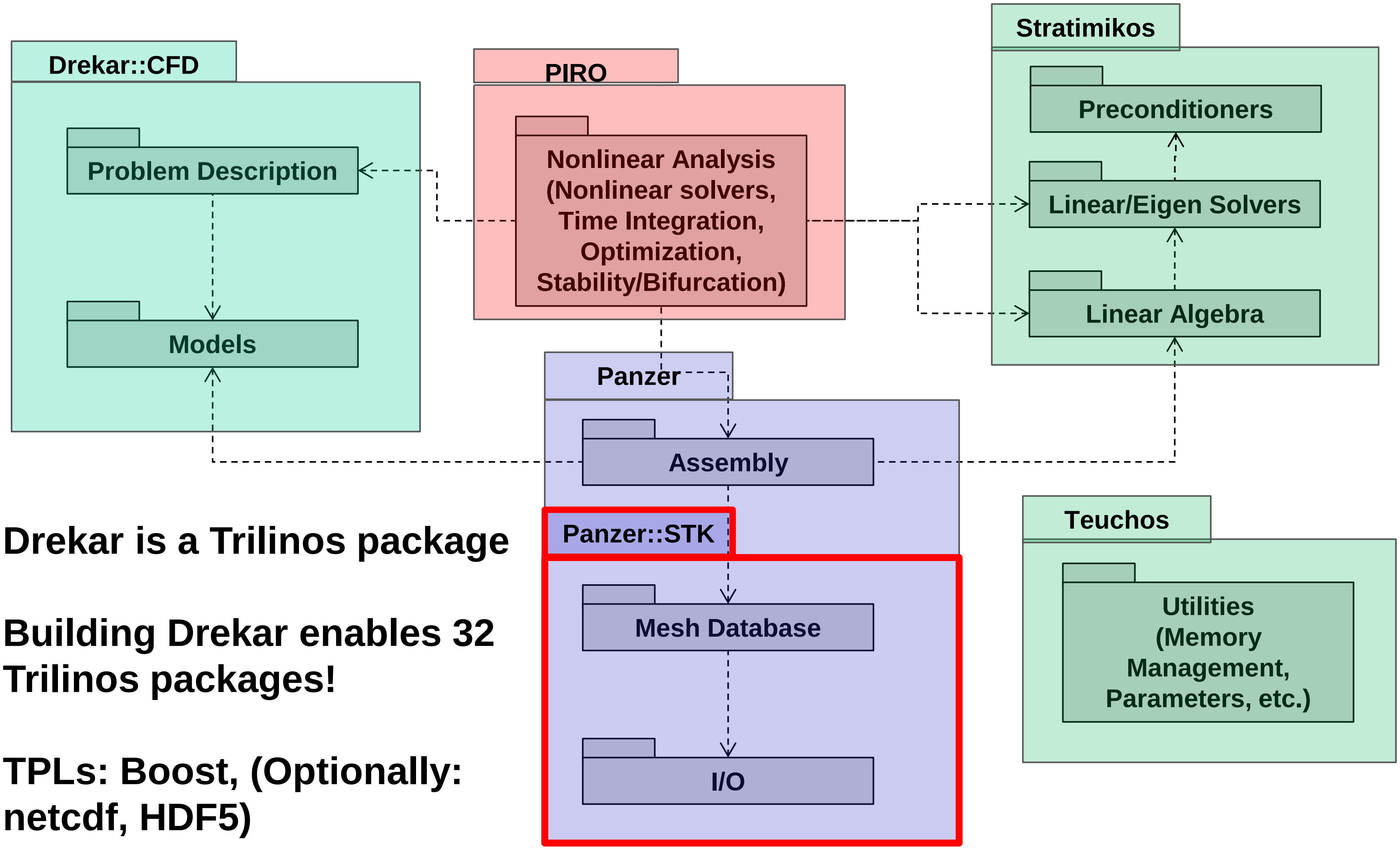
Physics Fill
- Element Level Fill
- Material Models
- Objective Function
- Constraints
- Error Estimates
- MMS Source Terms

Composite Physics
- MultiPhysics Coupling
- Solution Control
- System Models
- System UQ

Utilities
- Input File Parser
- Parameter List
- I/O Management
- Memory Management
- Communicators
- Runtime Compiler
- MultiCore Parallelization Tools

PostProcessing
- Visualization
- Verification Tools
- Feature Extraction
- Data Reduction
- Model Reduction

Software Design
(Composition of Trilinos Packages)



- **Drekar is a Trilinos package**
- **Building Drekar enables 32 Trilinos packages!**
- **TPLs: Boost, (Optionally: netcdf, HDF5)**

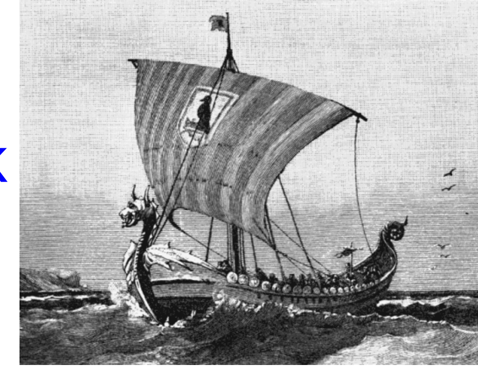
Introducing Drekar
(Named for the Viking Longship)

- A **light-weight front end** “Trilinos package” that provides Stabilized Galerkin CFD and MHD physics
- Provides mathematical kernels to evaluate the discretized PDEs using TBGP concepts
- Panzer/Drekar package dependencies:
  - 10 required
  - 9 optional
- Indirect dependencies: 32 enabled packages (including Drekar itself)

Panzer and Drekar

Trilinos Discretization Tool Stack

(Pawlowski,Cyr, Shadid, Smith)



PIRO (Solvers)
NOX
Rythmos
MOOCHO
LOCA
LIME

Thyra Model Evaluator

Drekar::CFD

- Input ParameterList
- Equation Set Factory
- BC Factory
- Evaluator Factory

Phalanx (TBGP)

Intrepid (FE Basis/IR)

Shards (Cell Topology)

Stokhos (UQ)

Sacado (AD)

Kokkos

Panzer

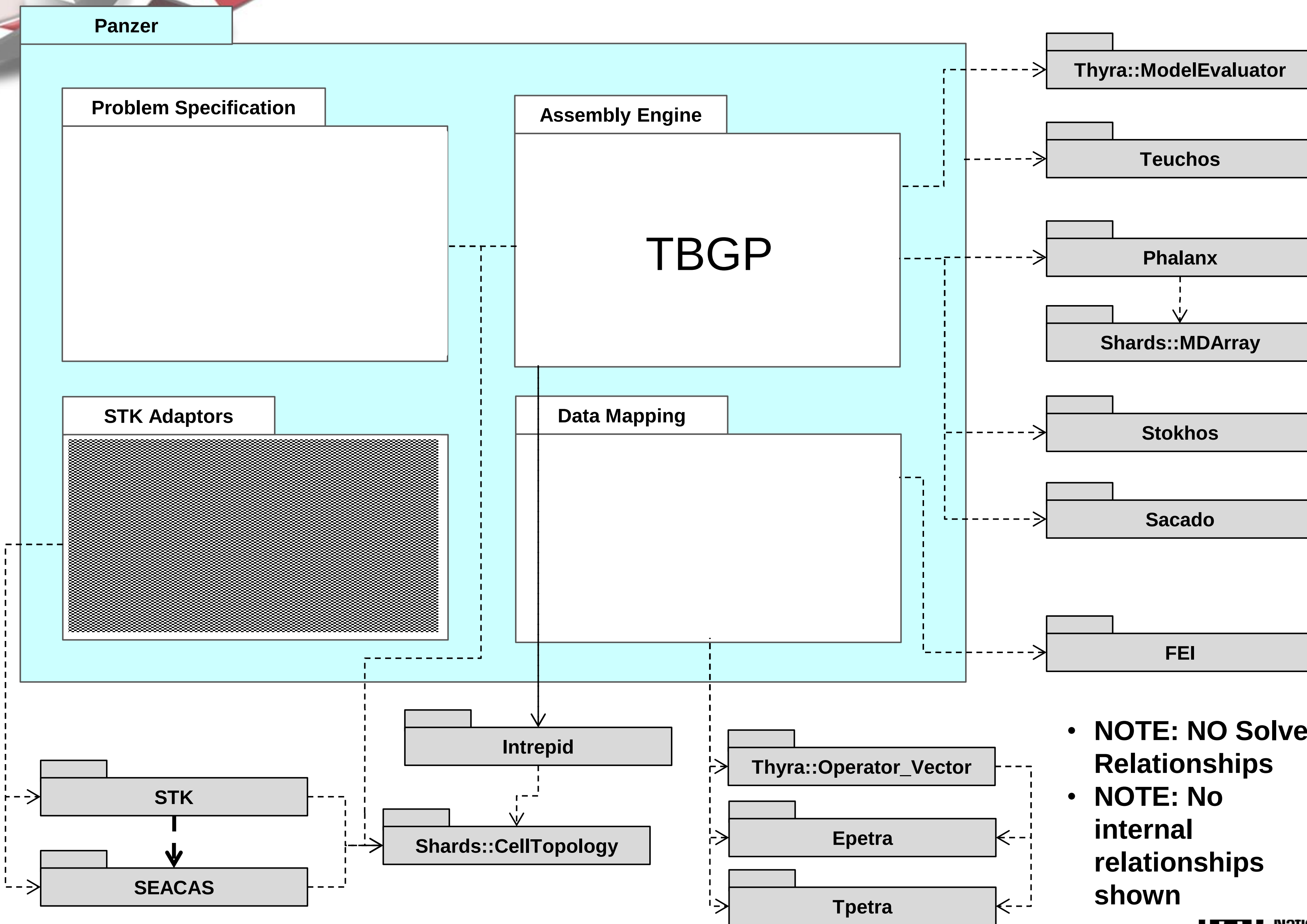
- **Multiphysics Assembly Engine:**
  - Fully coupled Multiphysics
  - Compatible discretizations
  - Multiple Equation sets
  - Arbitrary BCs
- **DOF Manager**
  - Mapping DOFs
  - ConnectionManager

FEI (DOF Mapping Strategy)

STK (Mesh Database)

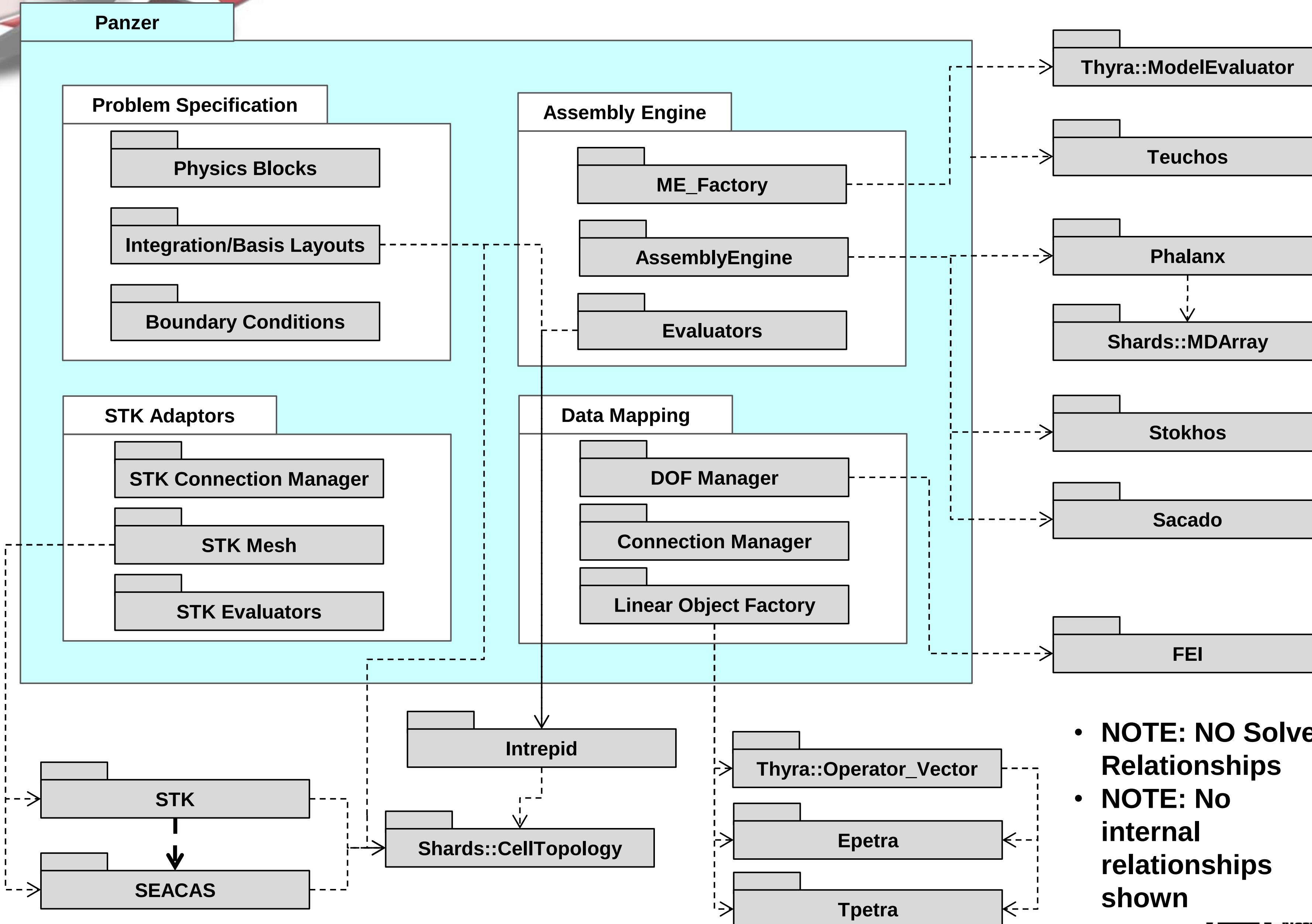
SEACAS (I/O, Partitioning)

Panzer Unifies Trilinos Discretization Tools



- **NOTE: NO Solver Relationships**
- **NOTE: No internal relationships shown**

Panzer Unifies Trilinos Discretization Tools



- NOTE: NO Solver Relationships
- NOTE: No internal relationships shown

Data Mapping

Computes global unknown indices

1. Serves as interface to mesh
2. Allows Panzer to be mesh agnostic
3. Handles unknowns for mixed discretizations
4. Handles unknowns for multiphysics (multiple element blocks)
5. Uses FEI for producing unknowns

Composed of 3 primary pieces

1. FieldPattern – Describes the basis layout and continuity of fields
2. DOFManager – Manages and computes unknown numbers on fields
3. ConnManager – (User implemented) Mesh topology from field pattern

Features not implemented but supported by design

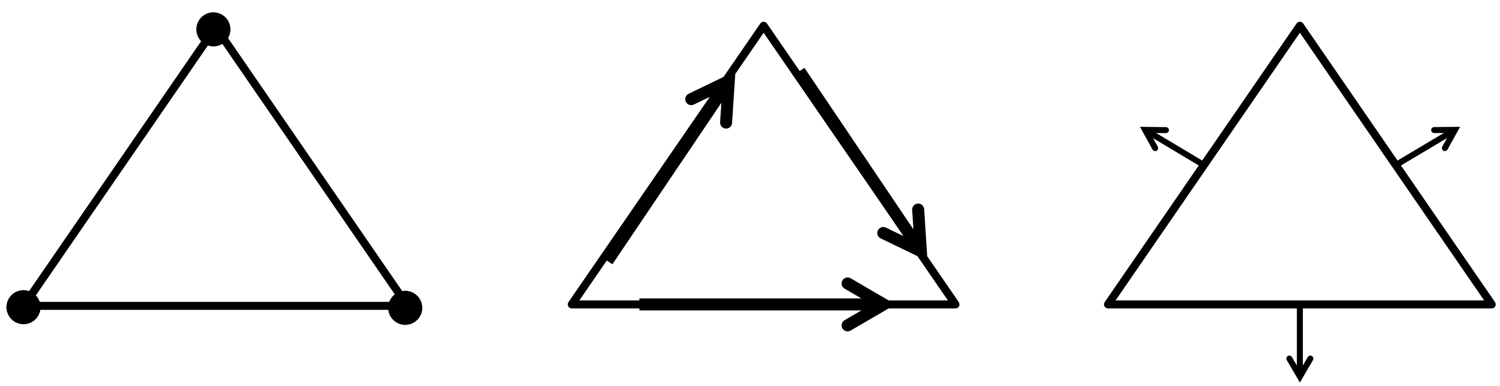
1. Higher order discretizations – geometric symmetries
2. Heterogeneous meshes – quadrilaterals and triangles

Data Mapping: New Directions

Finite Element discretizations have changed

- Charon used nodal-equal-order-finite elements
- New code embraces mixed discretizations
- Also using “Compatible Discretizations”
- Requires extra data management: orientations

H_{grad}(Nodal elements) H_{curl}(Edge elements) H_{div}(Face elements)



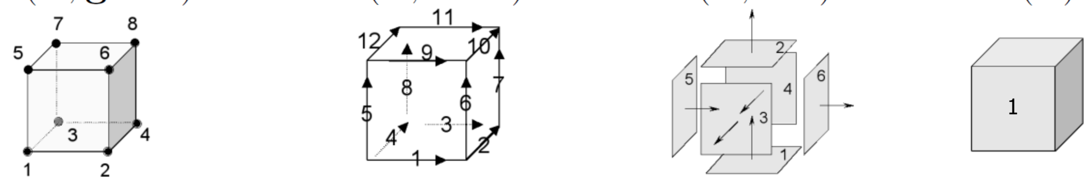
Data Mapping Handles These Elements

Advanced Discretizations

Intrepid: *Trilinos* **toolbox for discretizations** (**Bochev**, Ridzal, Peterson).

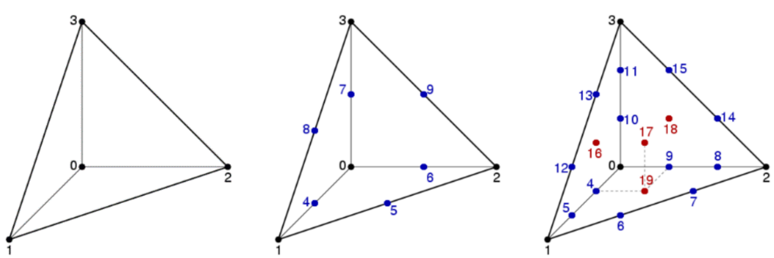
- allows access to finite element, finite volume, and finite difference methods via a common API
- compatible node-, edge-, face-, and cell-based discretizations

$$H(\Omega, \mathbf{grad}) \xrightarrow{\nabla} H(\Omega, \mathbf{curl}) \xrightarrow{\nabla\times} H(\Omega, \mathrm{div}) \xrightarrow{\nabla\cdot} L^2(\Omega)$$

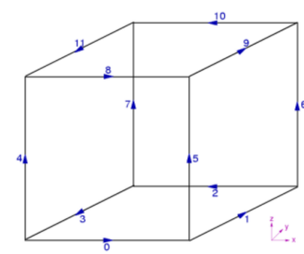


- enables hybrid discretizations (FE, FV, FD) on unstructured grids
- reference-map-based low- and high-order FE discretizations on standard cells
- "direct" low-order FV and FD discretizations on arbitrary polyhedral cells

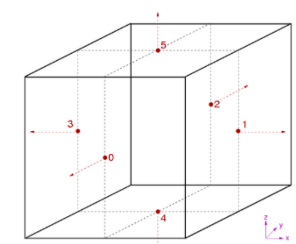
Completed development of **basic finite element** reconstruction operators (Bochev, Ridzal):



Lagrange elements of order 1,2,3



Nedelec element

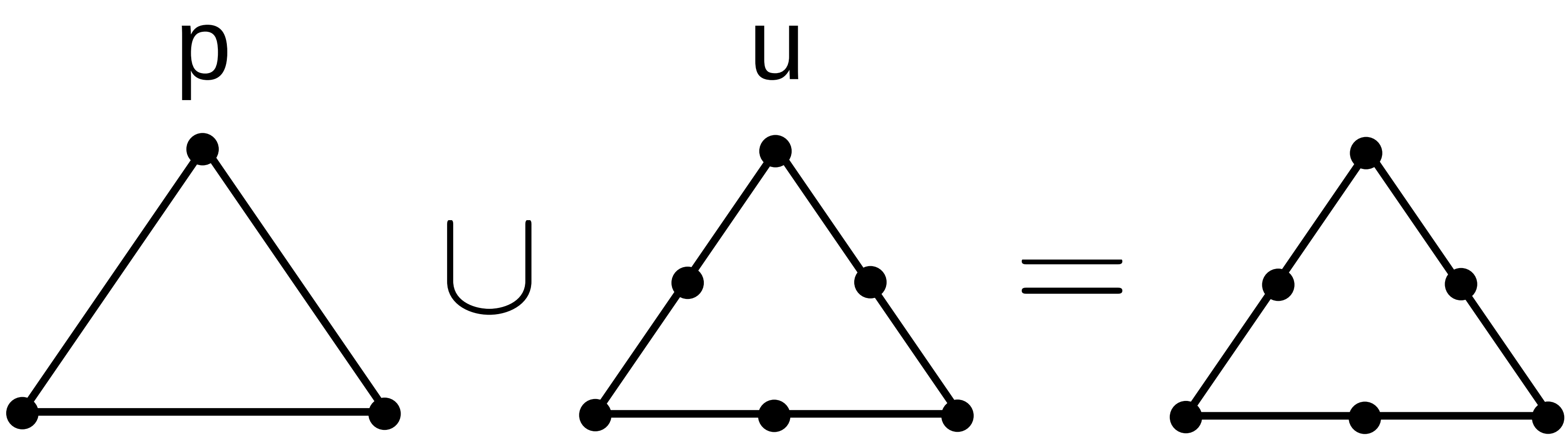


Raviart-Thomas element

Data Mapping: Field Pattern

For stable Navier-Stokes pair:

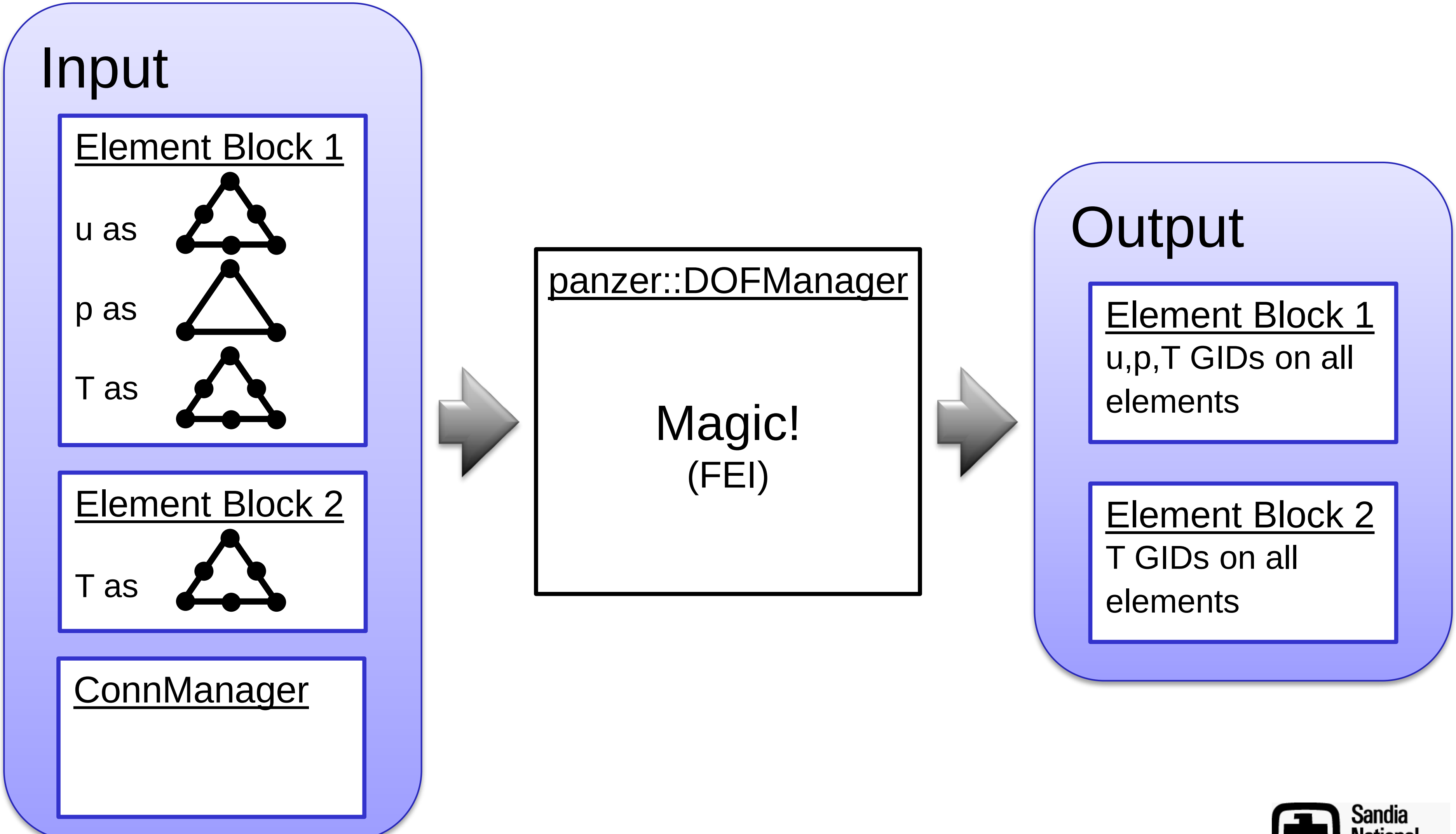
- Linear pressures
- Quadratic velocities



Field Pattern specifies basis layout

- Continuity across subcells (continuity of field)
- Unknowns on each element
- Communicates required topology

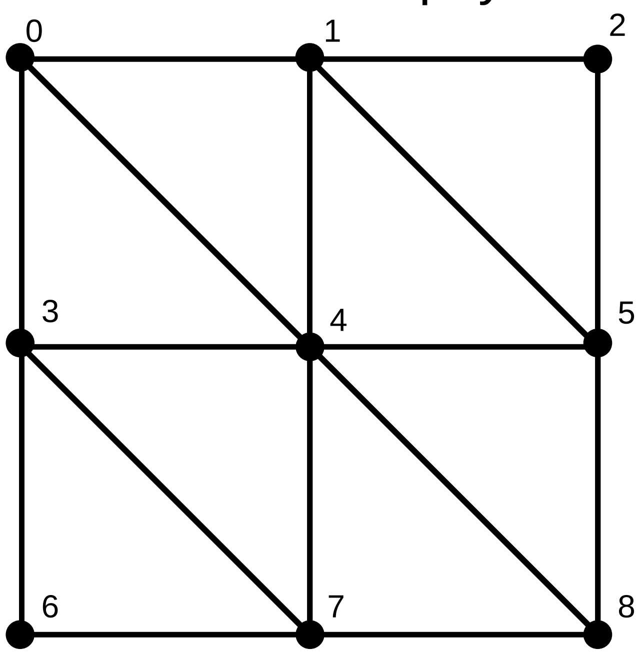
Data Mapping: DOFManager

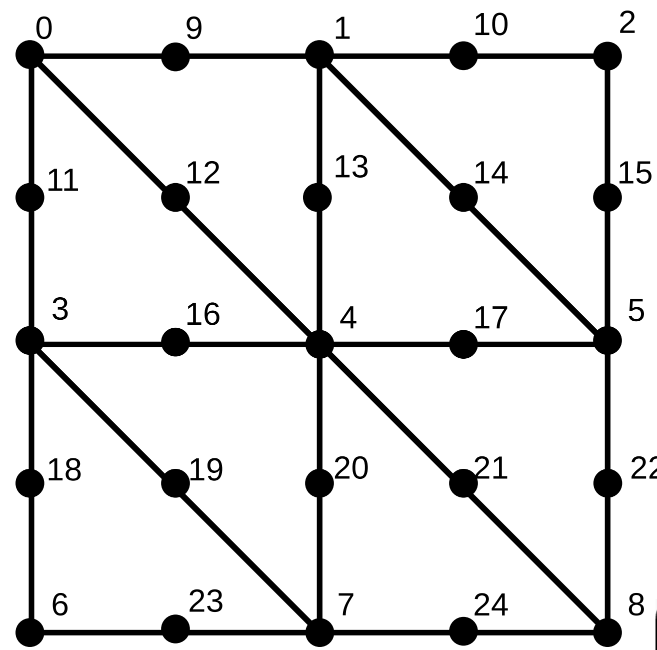


Data Mapping: ConnManager

Must generate mesh connectivity

- DOFManager passes in field pattern
- Provides unique global node, edge, volume ids for each element
- Optionally provides orientation for edge and face elements
- Uniform field pattern across all element blocks
  - Makes multiphysics easy

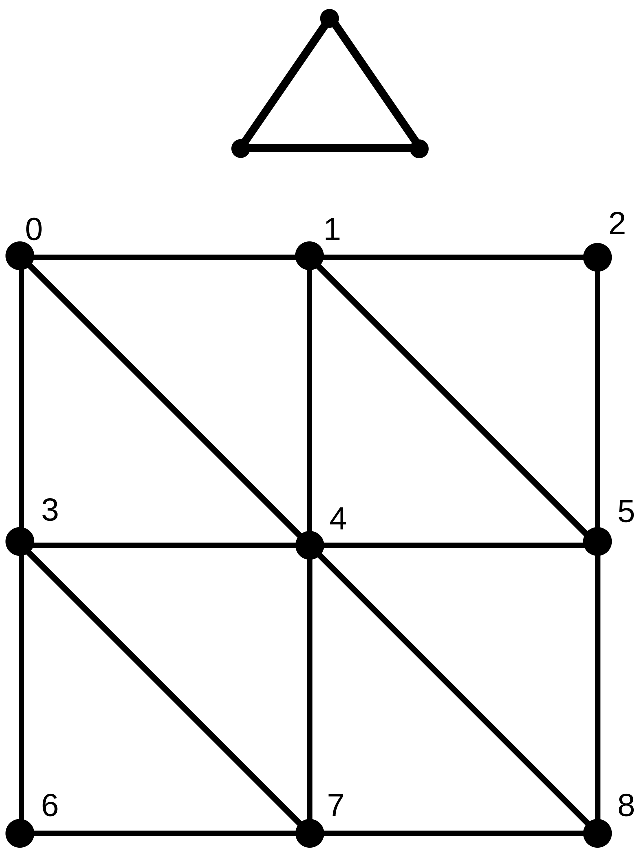




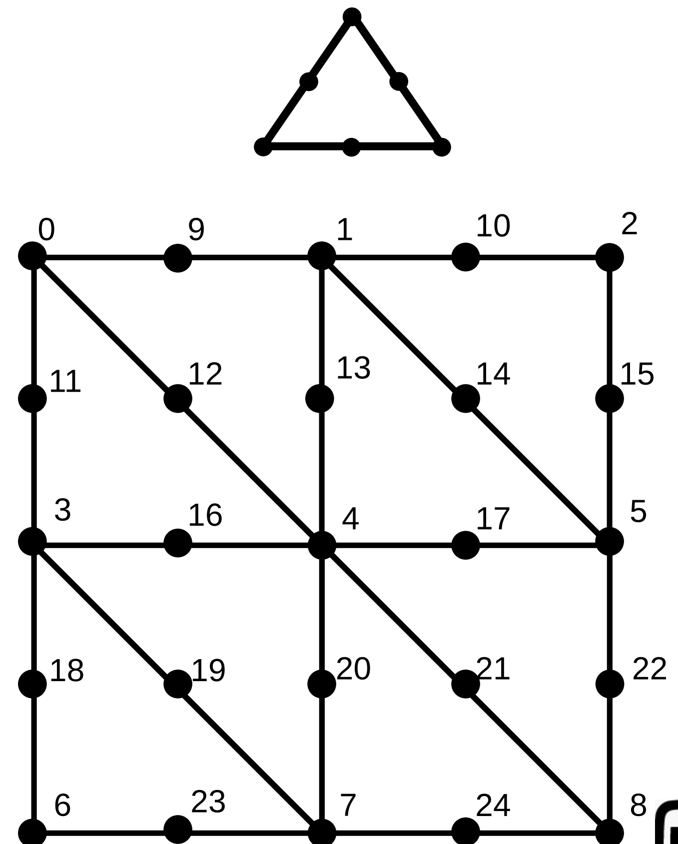
Sandia National Laboratories

Data Mapping: ConnManager

Piecewise linear p
Piecewise linear u



Piecewise linear p
Piecwise quadratic u



Data Mapping: Unknown Ordering

Old code used "interlaced" unknown ordering by node

$$[u_0, v_0, p_0, u_1, v_1, p_1, u_2, v_2, p_2, \ldots, u_N, v_N, p_N]^T$$

Panzer data mapping allows for greater control of ordering

- You can still interlace (the default)
- Blocked physics is also possible

Same ConnManager can be used multiple times

- Produce DOFManager for each type of physics
- Good for Block Preconditioning

$$\overset{u_0 v_0 p_0 \ldots u_8 v_8 p_8}{[A]} \longrightarrow \begin{array}{cc} u_0 v_0 \ldots u_8 v_8 & p_0 \ldots p_8 \end{array} \atop \begin{bmatrix} F & B^T \\ B & C \end{bmatrix}$$

Comments

- Adjoint capabilities supported
- Use of Kokkos MDArray for multi-/many-core/GPGPU support
- Expression templates for MDFields
- Phalanx: transition to Kokkos