Interoperable Unstructured Mesh Technologies for Petascale Computations

Lori Diachin, LLNL
Representing the ITAPS Team
March 2011
Many thanks to the ITAPS team for contributions to this presentation

Lori Diachin
LLNL

Ed d’Azevedo
ORNL

Karen Devine
SNL

Jim Glimm
BNL/SUNY SB

Ken Jansen
U. Colorado

Pat Knupp
SNL

Xiaolin Li
SUNY SB

Ahmed Khamayseh
ORNL

Roman Samulyak
BNL

Carl Ollivier-Gooch
UBC

Mark Shephard
RPI

Tim Tautges
ANL

Harold Trease
PNNL
Introduction
Unstructured mesh methods are commonly used for numerical simulation

- A piece-wise space/time domain decomposition over which the simulation is to be run
- General topology-based mesh representation consists of
  - 0-3D topological entities (vertices, edges, faces and regions)
  - Connectivity between entities called *adjacencies*
- Mesh data structure provides
  - Infrastructure for solution data
  - Foundation for services to create and/or use the mesh data
There are advantages and disadvantages to this approach

• **Some Advantages**
  – Meshes of mixed topologies and order easy
  – Mesh adaptation can account for curved domains
  – General mesh anisotropy can be obtained
  – Easy to create strong mesh gradations without special numerical techniques
  – Alignment with multiple curved geometric features

• **Some Disadvantages**
  – Data structures larger and more complex
  – Solution algorithms can be more complex
Basic parallel solution on unstructured meshes has several key steps

**Construct** the initial mesh (serial or parallel)
Improve the mesh using **smoothing** and **swapping**
If necessary, **(re)partition** the mesh across processors
Solve the PDE on mesh and estimate the error
While error > tolerance
  **Refine, coarsen, improve and repartition** the mesh
  Solve the PDE on the mesh and estimate the error
End
Solution methods are complicated by the needs of advanced simulations

Examples:
- Design optimization requires geometry modification, remeshing, derivative computations
- Multi-physics applications require mesh to mesh transfer, interpolation methods, sophisticated adaptive methods
The ITAPS team has developed tools to address unstructured meshing needs

- CAD interaction: CGM
- Mesh generation: GRUMMP, NWGrid
- Mesh databases: FMDB, MOAB
- Mesh improvement: Mesquite, swapping tools
- Parallel Adaptive loops: FMDB, NWGrid, MeshAdapt
- Front tracking: Frontier
- Partitioning: Zoltan
- Visualization: VisIt plug-ins, eyeMesh, iMesh I/O

What is the best way to deliver these related technologies to application scientists?
ITAPS chose a component-based approach for delivery of unstructured mesh tools

- Construct applications from smaller software “pieces”
- Components provide ...
  - Services applications commonly need implemented by domain experts
  - Support for wide range of applications
  - Ability to select only the services needed
- Components avoid ...
  - Prescribed data structures
  - Heavy framework
  - Limited freedom for application developers

Hasbro, Inc.
A successful component model has many key attributes

- Shared interfaces and data models enable interoperability
  - Enable plug-and-play use of different implementations of similar functionality
  - Enable data sharing among components
- Well-abstracted data model
  - Generalizes the type of and flow of information
  - Describes the inputs and outputs of the interface
- Well-defined interfaces
  - Interfaces define how to access components’ functionality (inputs and outputs)
  - Interfaces do NOT specify how the underlying functionality is implemented; Data structures, language, algorithm are chosen by the developer.
- A healthy ecosystem of compliant implementations, tools and services
The ITAPS data model abstracts the mesh-based PDE-simulation data hierarchy

- **Core Data Types**
  - *Geometric Data*: provides a high level description of the boundaries of the computational domain; e.g., CAD, image, or mesh data \( (iGeom) \)
  - *Mesh Data*: provides the geometric and topological information associated with the discrete representation of the domain \( (iMesh) \)
  - *Field Data*: provides access to time dependent physics variables associated with application solution. These can be scalars, vectors, tensors, and associated with any mesh entity. \( (iField) \)

- **Data Relation Managers \( (iRel) \)**
  - Provides control of the relationships among the core data types
  - Resolves cross references between entities in different groups
  - Provides functionality that depends on multiple core data types
The ITAPS data model has four fundamental “types”

- **Entity**: fine-grained entities in interface (e.g., vertex, face, region)
- **Entity Set**: arbitrary collection of entities & other sets
  - Parent/child relations, for embedded graphs between sets
- **Tag**: named datum annotated to Entities and Entity Sets
- **Interface Instance**: object on which interface functions are called and through which other data are obtained
Unstructured Meshes on Parallel Computers

• Typically the mesh is distributed over independent memories
• A mesh partition groups mesh entities and places them into parts
• Applications using a partitioned mesh need
  – Communication links for between “shared” mesh entities on neighboring parts
  – Ability to move mesh entities between parts (while maintaining links)
  – Algorithms to maintain load balance of parts which minimizing communications
Distributed mesh data models have several functional requirements

- Entity ownership
  - Each mesh entity is owned by exactly one part
  - Ownership imbues right to modify
  - Ownership is not static during the course of a simulation
    - Repartitioning
    - Local micro-migration
  - Some entities have read-only copies on other parts (e.g. along part boundaries and ghosts)
- Communication links
  - Efficient mechanisms to update mesh partitioning and keep the links between parts are mandatory
ITAPS has a well-defined partition model data abstraction

- **Process**: a program executing; MPI process
  - # of processes == MPI_Comm_size
  - Process number == MPI_Comm_rank
- **iMesh instance**: mesh database provided by an implementation
  - One or more instances per process
- **Partition**: describes a parallel mesh
  - Maps entities to subsets called *parts*
  - Maps parts to processes
  - Has a communicator associated with it

- **Ownership**: right to modify an entity
- **Internal entity**: Owned entity not on an interpart boundary
- **Part-Boundary entity**: Entity on an interpart boundary
- **Ghost entity**: Non-owned, non-part-boundary entity in a part
- **Copies**: ghost entities + non-owned part-boundary entities
iMesh(P) interface provides access to the discrete representation of the domain

- iMesh supports local access to the mesh
- iMeshP complements iMesh with parallel support
- Required functionality:
  - Access to mesh geometry and topology
  - User-defined mesh manipulation and adaptivity
  - Grouping of related mesh entities together (e.g. for boundary conditions)
- Builds on a general data model that is largely suited for unstructured grids
- Implemented using a variety of mesh types, software, and for a number of different usage scenarios
The serial iMesh interface supports basic and advanced local operations

- Provides basic access to vertex coordinates and adjacency information
  - Mesh loading and saving
  - Global information such as the root set, geometric dimension, number of entities of a given type or topology
  - Access to all entities in a set as single entities, arrays of entities, or entire set
  - Set/remove/access tag data
- Set functionality
  - Boolean operations (union, subtract, intersect)
  - Hierarchical relationships
- Mesh modification
  - Adding / Deleting entities
  - Vertex relocation
  - No validity checks
The iMeshP interface supports the needs of parallel computation

- Build and maintain partition and entity ownership information
  - Partition creation and modification
  - Entity ownership status
  - Ghost entity creation
  - Tag data retrieval and exchange on owned and copy data
  - Information about part boundaries and neighboring parts

- Parallel operations
  - Large and small scale entity migration
  - Update of coordinate information
  - Coordination of new entities along part boundaries
  - Synchronous and asynchronous operations supported

One layer of ghost triangles for all boundary triangles sharing edges:
- Ghost-entity dimension = 2
- Bridge-entity dimension = 1
- Number of layers = 1

Ghost-entity updates of tag data from owners
The iGeom interface provides access to the computational domain

- Must support
  - automatic mesh generation
  - mesh adaptation
  - tracking domain changes
  - relating information between alternative discretizations
- Builds on boundary representations of geometry
- Used to support various underlying representations
  - Commercial modelers (e.g., Parasolid, ACIS)
  - Modelers that operate from standard files (e.g. IGES, STEP)
  - Models constructed from an input mesh
New iField interface enables access to solution data

- Represent field as weighted sum of distribution functions
- Key information:
  - SI Units, dimensions
  - Scalars, vectors, tensors, coordinates
  - Basis/distribution functions
  - Domains
- Use cases:
  - Solution evaluation
  - Representation of analytic data
  - Mesh-to-mesh solution transfer
  - Change of basis
  - Interactions with solvers, preconditioners, post-processing tools
iRel Relations interface enables other ITAPS interfaces to work together

- Relates entities between two interfaces without adding dependencies between them. E.g.,
  - Relate entities in iMesh to entities iGeom
  - Generation of spectral element points on curved geometry

- Relationships supported:

- Implementation available in Lasso.
  - Reference implementation that all services/ implementations can use to manage relationships
ITAPS interface design enables interoperability in many dimensions

- Interoperability across language, application, implementation
- C-based interface, but designed to be callable directly from Fortran and C++
  - Good portability, performance
  - Maintenance easier
  - iGeom, iRel too
- Quick startup for new users
Simple Example: HELLO iMeshP

```c
#include "iMesh.h"
#include "iMeshP.h"
#include <mpi.h>

int main(int argc, char *argv[]) {
    char *options = NULL;
    iMesh_Instance mesh;
    iMeshP_PartitionHandle partition;
    int num_vtx, num_parts, ierr, options_len=0;
    iBase_EntitySetHandle root;
    /* create the Mesh instance */
    iMesh_newMesh(options, &mesh, &ierr, options_len);
    iMesh_getRootSet(mesh, &root, &ierr);

    MPI_Init(&argc, &argv);
    /* Create the partition. */
    iMeshP_createPartitionAll(mesh, MPI_COMM_WORLD, &partition, &ierr);

    /* load the mesh */
    iMeshP_loadAll(mesh, partition, root, argv[1], options, &ierr,
                   strlen(argv[1]), options_len);

    /* Report number of Parts in Partition */
    iMeshP_getNumParts(mesh, partition, &num_parts, &ierr);
    iMeshP_getNumOfTypeAll(mesh, partition, root, iBase_VERTEX, &num_vtx, &ierr);
    ...
```

1) Instantiates Partition
2) Reads mesh into mesh instance and Partition
3) Reports # parts in Partition
4) Get global number of vertices
Performance in building a finite element stiffness matrix

- Set up a simple stiffness matrix for a 2D diffusion equation
- Examine costs of entity access via native data structures, arrays, entity iterators and workset iterators
- Arrays minimize time overhead but require a data copy
- Entity iterators are straightforward to program, minimize memory overhead, but maximize time cost
- Entity array iterators balance time/memory tradeoffs but are the most difficult to program

\[ \nabla^2 u = f \\
\]

\[ u(x=0)=1 \quad u(x=1) = 1 \]

\[ u_y(x=0, \, x=1) = 0 \]

<table>
<thead>
<tr>
<th></th>
<th>Time (ms)</th>
<th>iMesh Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native</td>
<td>10479</td>
<td></td>
</tr>
<tr>
<td>Array-based</td>
<td>10774</td>
<td>2.8%</td>
</tr>
<tr>
<td>Entity Iterator</td>
<td>11642</td>
<td>11.1%</td>
</tr>
<tr>
<td>Workset Iterator (1)</td>
<td>11351</td>
<td>8.3%</td>
</tr>
<tr>
<td>Workset Iterator (3)</td>
<td>11183</td>
<td>6.7%</td>
</tr>
<tr>
<td>Workset Iterator (5)</td>
<td>11119</td>
<td>6.1%</td>
</tr>
<tr>
<td>Workset Iterator (10)</td>
<td>11095</td>
<td>5.8%</td>
</tr>
<tr>
<td>Workset Iterator (20)</td>
<td>11094</td>
<td>5.8%</td>
</tr>
</tbody>
</table>
An Overview of ITAPS Services
ITAPS provides a healthy ecosystem of interoperable services

- ITAPS provides stand-alone services as libraries
- Improve applications’ ability to leverage advanced tools
  - Mesh quality improvement
  - Mesh adaptation loops
  - Mesh partitioning
  - Front tracking
  - Visualization
  - Mesh I/O

I’ll provide a brief overview of each of these tools; more information can be found on www.itaps-scidac.org
There are three primary techniques for improving the quality of existing meshes:

- **Node Movement**: Moving grid points without changing mesh topology.
- **Edge or Face Flipping**: Modify topology without changing grid point location.
- **Refinement/Coarsening**: Adding or deleting elements to improve local resolution.
Mesquite provides advanced mesh smoothing capabilities

- Mesquite is a comprehensive, stand-alone library for mesh quality improvement with the following capabilities
  - Shape Quality Improvement
  - Mesh Untangling
  - Alignment with Scalar or Vector Fields
  - R-type adaptivity to solution features or error estimates

- Uses node point repositioning schemes
- Parallel to O(13000) processors
- Tested with FMDB, MOAB, NWGrid, GRUMMP iMesh implementations
Swapping complements node point movement in improving mesh quality

- Changing topology can eliminate poorly-shaped mesh entities directly

- Service using standard interface handles error-prone aspects of implementation
  - Swapping decisions
  - Topological changes to the mesh

- Swapping service functionality
  - Triangular/tetrahedral edge and face swapping
  - Works on single entities, mesh subsets, or entire mesh
  - Built in and user-defined swapping criteria provide both ease of use and flexibility

- Tested with FMDB, MOAB, and GRUMMP
Performance of iMesh Swap for 3D Meshes

- Comparing GRUMMP native implementation to service with GRUMMP iMesh implementation
  - MANY calls to region-face, face-region, cell-vert, face-vert adjacencies; create/delete also called often
- Most remaining overhead is in transcribing data to return format expected by iMesh

<table>
<thead>
<tr>
<th>Case</th>
<th># of Tets</th>
<th>Native Swaps</th>
<th>Native Rate (1/s)</th>
<th>iMesh Swaps</th>
<th>iMesh Rate (1/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rand1</td>
<td>5104</td>
<td>10632</td>
<td>29500</td>
<td>10838</td>
<td>21300</td>
</tr>
<tr>
<td>Rand2</td>
<td>25704</td>
<td>65886</td>
<td>27700</td>
<td>67483</td>
<td>22100</td>
</tr>
<tr>
<td>Airplane</td>
<td>251140</td>
<td>25448</td>
<td>3380</td>
<td>28629</td>
<td>2800</td>
</tr>
<tr>
<td>Rocket</td>
<td>464080</td>
<td>53331</td>
<td>3540</td>
<td>59330</td>
<td>2790</td>
</tr>
</tbody>
</table>
Mesh adaptation is critical for many applications

- Determining optimal initial mesh sizes is not possible for complex geometry/physics
- CFD adaptivity example impact:
  - *Isotropic*: same accuracy as uniform with one order of magnitude fewer elements
  - *Anisotropic*: same accuracy as uniform with two orders of magnitude fewer elements
- ITAPS Mesh Adapt Service
  - Starts with an arbitrary initial mesh with a solution
  - Given a new mesh size field, alters the mesh via local mesh modifications
  - Supports
    - Curved Boundaries
    - Anisotropy
    - Parallel mesh adaptation
Scaling studies of uniform refinement on up to 32K processors

- Weak scaling uniform refinement
  - Mesh adaptation characterized by small, but variable, work per operation - “perfect scaling” too costly
    - Can run on the large numbers of parts used in the analysis
    - Will still be a small % of total solution time - the mesh adapt example given is 0.04% of the estimated solve time
- Improvements to message passing can improve scaling
Zoltan partitioners access mesh data through ITAPS interfaces

- Partitioning Methods
  - Geometric (RCB, Space filling curves)
  - Connectivity-based (ParMetis, Scotch, Hypergraph)
- iMesh and iMeshP versions available
- Tested and adopted by FMDB, GRUMMP, MOAB, MeshAdapt, accelerator and fusion scientists
Performance of Zoltan Partitioning for 3D Meshes

- Comparing MOAB native implementation linking to Zoltan partitioning service with the MOAB iMesh implementation
- Using a coordinate bisection geometric partitioner on tetrahedral meshes and array-based access to the data

<table>
<thead>
<tr>
<th>Number of Tets</th>
<th>Native (sec)</th>
<th>iMesh (sec)</th>
<th>iMesh Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>15591</td>
<td>0.866</td>
<td>0.869</td>
<td>0.35%</td>
</tr>
<tr>
<td>20347</td>
<td>0.971</td>
<td>0.976</td>
<td>0.51%</td>
</tr>
<tr>
<td>34750</td>
<td>1.28</td>
<td>1.31</td>
<td>2.34%</td>
</tr>
<tr>
<td>54383</td>
<td>1.72</td>
<td>1.75</td>
<td>1.74%</td>
</tr>
<tr>
<td>100630</td>
<td>2.76</td>
<td>2.82</td>
<td>2.17%</td>
</tr>
</tbody>
</table>
MeshAdapt and iZoltan used to prepare strong scaling study on 128K processors

- **PHASTA CFD solver**
  - Implicit time integration - iterative system solution at each time step
  - Employs the partitioned mesh for system formulation and solution

- **PHASTA’s work characterized as:**
  - Organized and regular communication between parts that “touch” each other
  - A specific number of ALL-REDUCE communications also required

- **ITAPS Services used**
  - FMDB for the mesh database
  - MeshAdapt for refinement up to 32K
  - iZoltan to partition the mesh to 128K

- **Strong scaling highlights need for advanced partitioning algorithms**

  1 billion element anisotropic mesh on Intrepid Blue Gene/P

<table>
<thead>
<tr>
<th>#of cores</th>
<th>Rgn imb</th>
<th>Vtx imb</th>
<th>Time (s)</th>
<th>Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>16k</td>
<td>2.03%</td>
<td>7.13%</td>
<td>222.03</td>
<td>1</td>
</tr>
<tr>
<td>32k</td>
<td>1.72%</td>
<td>8.44%</td>
<td>112.43</td>
<td>0.987</td>
</tr>
<tr>
<td>64k</td>
<td>1.6%</td>
<td>11.18%</td>
<td>57.09</td>
<td>0.972</td>
</tr>
<tr>
<td>128k</td>
<td>5.49%</td>
<td>17.85%</td>
<td>31.35</td>
<td>0.885</td>
</tr>
</tbody>
</table>

Recent results extend these scalability tests to 288K processors on JuGene IBM BG/P system in Germany
Interface tracking is available through the FronTier library

- **Intended uses include**
  - Computational domains sharply different quantitatively or qualitatively, boundary dynamic
  - Tracking the dynamic motion of distinct bodies, or interfaces between distinct physical regions

- **Coupled with hyperbolic, parabolic, and elliptic PDE solvers**
- **Parallel to 16K processors**
- **DOE Applications**: fluid-fluid, fluid-structure, crystal growth, phase transition, elastic-plastic, and other moving interfaces

- **Available and tested with the iMesh interfaces through FMDB, MOAB, and GRUMMP**

FronTier meshed data structure
Each service requires only a fraction of the defined interface functions

<table>
<thead>
<tr>
<th></th>
<th>Coordinates</th>
<th>Adjacency</th>
<th>Other queries</th>
<th>Iterators</th>
<th>Modification</th>
<th>Basic Sets</th>
<th>Tags</th>
<th>Parallel</th>
<th>iGeom/iRel</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mesquite</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>13</td>
<td></td>
<td>6</td>
<td>32</td>
</tr>
<tr>
<td>Swapping</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>Mesh Adapt</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>13</td>
<td></td>
<td></td>
<td>34</td>
</tr>
<tr>
<td>FronTier Lite</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td>14</td>
<td></td>
<td>18</td>
</tr>
<tr>
<td>Zoltan</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14</td>
<td></td>
<td>22</td>
</tr>
<tr>
<td>VisIt</td>
<td>1</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td></td>
<td>1</td>
<td>16</td>
<td></td>
<td></td>
<td>31</td>
</tr>
</tbody>
</table>
Applications using ITAPS Tools
Applications can access ITAPS services in two ways

1. Implement ITAPS interfaces on top of application data structures

   - Application w/ Own Data
   - Interface
   - Component Service 1
   - High Level Integrated Service
   - Component Service 2
   - Component Service 3
   - Application using ITAPS Implementation
   - Interface
   - ITAPS Implementation

2. Use a reference implementation of the interfaces to provide access to ITAPS services at the cost of a data copy

   - Application using ITAPS Implementation
   - Interface
   - ITAPS Implementation
   - High Level Integrated Service
   - Component Service 2
   - Component Service 3

Successful development of applications has been accelerated by close collaboration.
The ITAPS team has facilitated new science in many application domains

- **Accelerator Science**
  - Curved mesh correction tools improve solution time by 30%
  - Mesh adaptation reduced time to solution by 10X

- **Fusion Science**
  - Front-tracking for pellet ablation revealed new physical properties to improve ITER fueling plans
  - Anisotropic refinement and high order elements allows efficient computation on curved reactor domains

- **Earth Systems**
  - Front-tracking for crystal precipitation should provide lower-cost, comparable accuracy simulations with SPH
  - Meshes/geometry for ice sheet modeling

- **Nuclear Reactors**: Interoperable interfaces enable multi-physics simulations on petascale computers
Mesh curving for SLAC accelerator simulations

- Reliably produces valid meshes from invalid curved elements.
  - Valid meshes allow solution to run and produce results that can be used.
  - Simulations also ran 30% faster due to better conditioning.
- Example:
  - 2.97 millions curved regions in input mesh
  - 1,583 invalid curved elements
  - All corrected by mesh curving tool using curved mesh modification operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge collapse</td>
<td>253</td>
</tr>
<tr>
<td>Region collapse</td>
<td>17</td>
</tr>
<tr>
<td>Edge swap</td>
<td>76</td>
</tr>
<tr>
<td>Double edge split+collapse</td>
<td>13</td>
</tr>
<tr>
<td>Recurving</td>
<td>32</td>
</tr>
</tbody>
</table>
Mesh adaptation for SLAC wakefield calculations

- Adaptively refined meshes have 1~1.5 million curved regions
- Reduced the number of elements needed by nearly an order of magnitude

Electric fields on the three refined curved meshes

Experiments for future accelerators at CERN driven by ITAPS simulation tools

- New liquid mercury targets considered for future accelerators
- Understanding the target response is critical for the accelerator design
- Previous simulations using ITAPS front tracking tools made predictions for the targetry experiment at CERN called MERIT and influenced its design
- The study group concluded that liquid mercury jet targets can work reliably in future accelerators
- New series of MHD simulations reproduced important details of MERIT experiment and currently focus on future designs

Surface instabilities of the mercury target interacting with a proton pulse in 5 T magnetic field
Anisotropic mesh adaption in M3D-C1 MHD fusion code at PPPL

- Compute size field
  - Hessian interpolation or *a priori* known vorticity mode
- Apply high order local solution field transfer together with the mesh adaptation
- Vertical integration with solvers technologies enabled previously unachievable simulations

![Initial mesh and vorticity field](image1)
![Computed size field](image2)
![Adapted mesh and transferred field](image3)
Simulation helps evaluate the concept of Plasma-Jet driven Magneto-Inertial Fusion

- Plasma-Jet driven Magneto-Inertial Fusion potentially provides a low-cost and fast R&D path towards the demonstration of practical fusion energy

- Simulations help to evaluate the feasibility of PJMIF. Simulations include
  - Propagation of Mach 60 plasma jets
  - Jet merger and the liner formation
  - Plasma liner implosion and target compression

- Simulation demonstrated successful formation of the plasma liner by the merger of 625 jets
  - Estimated liner uniformity and Mach number reduction
  - Verified scaling laws
  - Provide support for PLX experiment at LANL

FronTier simulation of the implosion of plasma liner formed by the merger of 625 jets
ITAPS tools used extensively in reactor simulation application

- **SHARP framework**
  - Couple multiple physics modules (T/H, neutronics, etc.)
  - Data substrate uses ITAPS interfaces, components, services

- **Advanced meshing tools**

  7-pin non-conformal hex mesh

  ARR 1/6 core generated with copy/move/merge

  Fuel assembly, 1/6 core model, Advanced Recycle Reactor
ITAPS tools lead to better understanding of accidents in nuclear reactors

- Advanced simulation capabilities are necessary to satisfy technological and safety standards for future Gen IV Reactors.

- ITAPS tools enabled multiscale simulations of an accident scenario involving fuel rod failure.

- FronTier was used for the simulation of overheating and melting of the nuclear fuel rod, the clad failure, and the ejection of fission gases into the sodium coolant. FronTier provided output to PHASTA for the simulation of larger scale dynamics.

FronTier simulation of the ejection of fission gas into sodium coolant: isosurfaces of pressure at 1 ms after the crack formation.
A recent collaboration addresses the geometry and mesh needs for ice sheet modeling

- Ice sheet bed location derived from remote sensing data represented as point set, elevations
- Need:
  - All-quad, all-hex mesh for ice sheet bed/ice sheet
  - Smooth normal field, for slip boundary condition modeling
- Ice sheet geometry, mesh capability assembled from combination of ITAPS and non-ITAPS tools

**Jakobshavn glacier, Greenland**

- Original (Triangle/MOAB): 5M tri
- Decimated (Qslim/MOAB): 20k tri
- 1km quad mesh (CAMAL/MeshKit/MOAB)
To learn more; please attend our minisymposia today

• MS99 (Reno Ballroom) @ 9:30 am
  – Tools to support unstructured meshes on massively parallel computers (Mark Shephard)
  – ITAPS Tools in computational evaluation of alternative methods of nuclear fusion (Roman Samulyak)
  – Introduction to the ITAPS Field Interface (Carl Ollivier-Gooch)
  – MeshKit: An open-source toolkit for mesh generation (Tim Tautges)

• MS111 (Reno Ballroom) @ 1 pm
  – Towards Large-Scale Predictive Flow Simulations using ITAPS services (Onkar Sahni)
  – Parallel Mesh Generation by Distributing CAD Geometry (Hong-Jun Kim)
  – Parallel Hybrid Mesh Adaptation (Aleksandr Ovcharenko)
For more information…

- Visit our web site at http://www.itaps-scidac.org
  - General project information
  - Software downloads
  - Petascale computing research
  - Applications impact
Auspices and disclaimer

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.