LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-PROC-232530

# Interoperable mesh and geometry tools for advanced petascale simulations

L. Diachin, A. Bauer, B. Fix, J. Kraftcheck, K. Jansen, X. Luo, M. Miller, C. Ollivier-Gooch, M. Shephard, T. Tautges, H. Trease

July 6, 2007

SciDAC 2007
Boston, MA, United States
June 24, 2007 through June 28, 2007

Updated October 14, 2003

# Interoperable mesh and geometry tools for advanced petascale simulations

**L. Diachin[1], A. Bauer[2], B. Fix[3], J. Kraftcheck[4], K. Jansen[2], X. Luo[2], M. Miller[2], C. Ollivier-Gooch[5], M. S. Shephard[2], T. Tautges[6] and H. Trease[7]**

[1] Lawrence Livermore National Lab, Livermore, CA, [2] Rensselaer Polytechnic Institute, Troy, NY, [3] SUNY Stony Brook, Stony Brook, NY, [4] University of Wisconsin, Madison, WI, [5] University of British Columbia, Vancouver, BC, [6] Argonne National Laboratory, Argonne, IL, [7] Pacific Northwest National Laboratory, Richland, WA

E-mail: diachin2@llnl.gov, acbauer@scorec.rpi.edu, brian@ams.sunysb.edu, kraftche@cae.wisc.edu, xluo@scorec.rpi.edu,miller86@llnl.gov, cfog@mech.ubc.ca, shephard@scorec.rpi.edu, tautges@mcs.anl.gov, het@pnl.gov

**Abstract.** SciDAC applications have a demonstrated need for advanced software tools to manage the complexities associated with sophisticated geometry, mesh, and field manipulation tasks, particularly as computer architectures move toward the petascale. The Center for Interoperable Technologies for Advanced Petascale Simulations (ITAPS) will deliver interoperable and interchangeable mesh, geometry, and field manipulation services that are of direct use to SciDAC applications. The premise of our technology development goal is to provide such services as libraries that can be used with minimal intrusion into application codes. To develop these technologies, we focus on defining a common data model and data-structure neutral interfaces that unify a number of different services such as mesh generation and improvement, front tracking, adaptive mesh refinement, shape optimization, and solution transfer operations. We highlight the use of several ITAPS services in SciDAC applications.

## 1. Introduction

The advent of petascale computing will enable increasingly complex, realistic simulations of PDE-based applications. Numerous software tools are available to help manage the complexity of these simulations, including computer-aided design systems used to represent the geometry of the computational domain, mesh generation tools to discretize those domains, solution adaptive methods (AMR) to improve the accuracy and efficiency of simulation techniques, and parallel tools such as dynamic partitioning to ease implementation on today's computer architectures. These tools would be particularly effective if they could be easily integrated and used in concert in existing simulations or if they could enable integration of existing simulations into multi-physics, multi-scale codes. At present, however, this type of integration is exceedingly difficult, due in large part to software compatibility issues.

The goal of the Interoperable Technologies for Advanced Petascale Simulations (ITAPS) enabling technology center is to address this challenge through the development of interoperable and interchangeable mesh, geometry, and field manipulation tools that are of direct use to

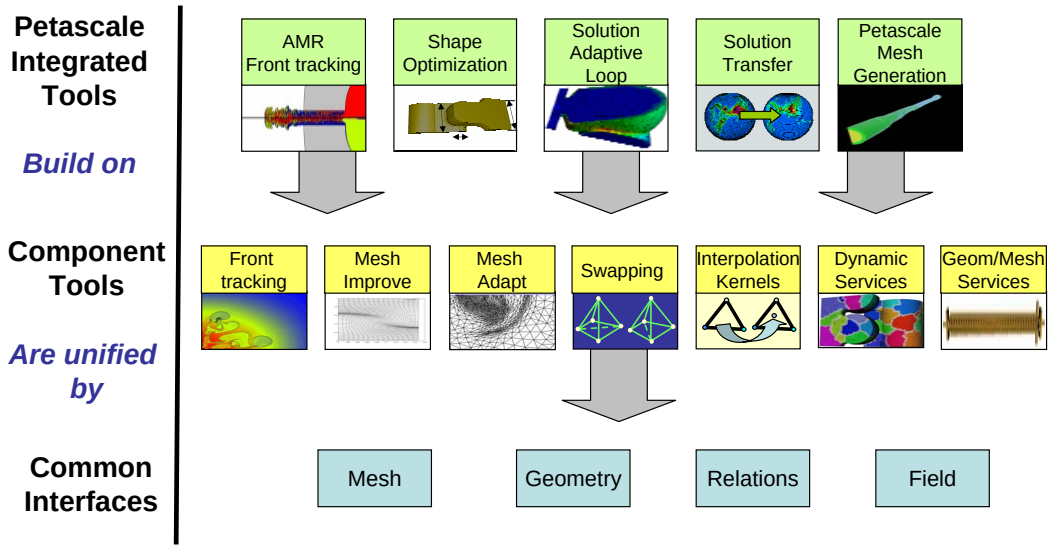


**Figure 1.** The ITAPS center is developing integrated services that build on multiple component services and common interfaces for geometry, mesh and field information.

SciDAC applications. The hierarchical approach we take to our technology development goals is summarized in Figure 1. We start with component services such as mesh quality improvement, adaptive loops, front tracking at the middle level of the figure. These services are of direct use to applications as stand-alone tools, and many of them pre-date the ITAPS project. However, to use these tools in concert to form higher-level integrated services, such as AMR-front tracking or shape optimization (top row of the figure), application scientists must often provide multiple interfaces to the same data which is a costly and error prone process. To address this problem, the ITAPS team is developing common interfaces that provide data-structure and implementation neutral access to mesh, geometry, and field information. These interfaces provide access to all ITAPS services in a uniform way and are fundamental to creating interoperability for the integrated services. Moreover, a uniform interface allows easier experimentation with different, but functionally similar, technologies to determine which is best suited for a given application.

Figure 2 shows the primary ways that applications can use ITAPS services. In the left figure (a), the application already has geometry, mesh, and/or field data and implements the ITAPS interfaces as wrapper functions around their own data structures. Once the necessary interface functions are implemented and tested, the ITAPS services can easily access the application data they need through the ITAPS interface. In the right figure (b), the application scientist can take advantage of data implementations already available from ITAPS for rapid prototyping of new applications. This mode of access also allows easy access to ITAPS services using a very small number of function calls to copy the necessary data to a fully compliant implementation of the interface. In some cases the memory overhead associated with the data copy is relatively small

and this access mode is sufficient for long term use. In other cases, it provides the mechanism for easy access to and experimentation with ITAPS services; as the benefits of the services become clear, application scientists can implement the interfaces more efficiently on top of their own data structures.

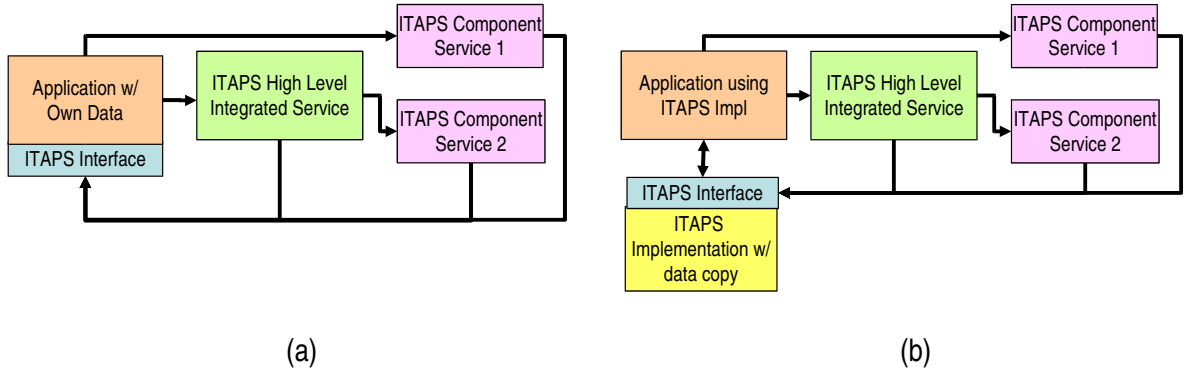


**Figure 2.** A schematic showing the use of ITAPS interfaces and services in application codes. The figure on the left (a) shows the use of the interfaces implemented directly on the application data structures. The figure on the right (b) shows the use of a reference implementation of the interfaces to provide immediate access to ITAPS services at the cost of a data copy

In this paper, we describe the framework and philosophy that we have used to create interoperable mesh, geometry and field tools. In Sections 2 and 3, we discuss the abstract data model and common interfaces that we have worked as a group to define, as well as the implementations that provide those interfaces. Use of these interfaces to provide services to SciDAC applications is described in Section 4.

## 2. The ITAPS Data Model

We use the information flow through a mesh-based simulation as the framework for developing interoperable geometry, mesh and solution field components. A simulation's information flow begins with a problem definition which consists of a description of the geometric and temporal domain annotated by *attributes* designating mathematical model details and parameters. The geometric domain is then often decomposed into a set of piecewise components, *the mesh*, and the continuous PDEs are then approximated on that mesh using, for example, finite difference or finite element techniques. Simulation automation and reliability often imply feedback of the PDE discretization information back to the domain discretization (i.e. in adaptive methods) or even modification of the physical domain or attributes (e.g., for design optimization).

Based on this model of information flow, ITAPS researchers have defined an abstract data model that supports a wide array of supporting technologies and encompasses a broad spectrum of usage scenarios. The data model divides the data required by a simulation into three *core data types*: the geometric data, the mesh data, and the field data. These core data types are associated with each other through *data relation managers*. The data relation managers control the relationships among two or more of the core data types, resolve cross references between entities in different groups, and can provide additional functionality that depends on multiple core data types. The building blocks within these data models are the concepts of *entities*, *entity sets*, and *tags*.

- *Entities* are used to represent atomic pieces of information such as a vertices in a mesh or edges in a geometric model. Entity adjacency relationships define how the entities connect to each other and both first-order and second-order adjacencies are supported.
- *Entity sets* are arbitrary collections of entities that may be an ordered list or unordered. The two primary supported relationships among entity sets are *contained in* and *parent/child* to allow for subsetting and hierarchical applications. In addition, entity sets also have "set operation" capabilities such as set subtraction, intersection, or union.
- *Tags* are used as containers to attach user-defined data to ITAPS entities and entity sets. Tags can be multi-valued which implies that a given tag handle can be associated with many different entities. We support specialized tag types for improved performance as well as the more general opaque case that allows any type of data to be attached.

As a particular example, consider the discrete representation of the computational domain, or the mesh. ITAPS *mesh entities* correspond to the individual pieces of the mesh, namely, vertices, edges, faces, and regions. Specific examples include a hexahedron, edge, triangle or vertex. Mesh entities are classified by their entity type (topological dimension) and entity topology (shape). Higher-dimensional entities are defined by lower-dimensional entities using canonical ordering relationships. To determine which adjacencies are supported by an underlying implementation, an adjacency table is defined which can be returned by a query through the interface. The implementation can report that adjacency information is always, sometimes, or never available; and to be available at a cost that is constant, logarithmic (i.e., tree search), or linear (i.e., search over all entities) in the size of the mesh. ITAPS *mesh entity sets* are extensively used to collect mesh entities together in meaningful ways, for example, to represent the set of all faces classified on a geometric face or the set of regions in a domain decomposition for parallel computing.

To support many of the services that applications desire, such as adaptive mesh refinement, it is important that the data model include the concept of modification to allow changes to geometry, topology, or set structure. In the case of the mesh, capabilities include changing vertex coordinates and adding or deleting entities. Modification often requires interactions between the mesh, geometry and field data models and is one of the primary uses for the data relations manager. For example, when refining a mesh, it is often critical to associate or classify the mesh entity with one or more specific entities in the underlying geometric model to ensure accuracy, particularly on curved or complex geometries.

## 3. The ITAPS Interfaces.

The next step to creating interoperable technologies is to define common interfaces that support the abstract data model. A key aspect of our approach is that we do not enforce any particular data structure or implementation with our interfaces, requiring only that certain questions about the geometry, mesh, or field data can be answered through calls to the interface. One of the most challenging aspects of this effort remains balancing performance of the interface with the flexibility needed to support a wide variety of data types. Performance is critical for kernel computations involving mesh and geometry access, and to address this need, we provide a number of different access patterns including individual iterator-based and agglomerated array-based requests. Further challenges arise when considering the support of many different scientific programming languages which we address using a two-pronged approach. First, we provide a C-language binding for our interfaces that is compatible with most needs in scientific computing. Additional flexibility, albeit at a somewhat higher cost, is supported through the use of the SIDL/Babel technology [1] provided by the Common Component Architecture Forum (CCA).

The ITAPS mesh interface has been under development for several years, and we provide a simple example of using the C-binding version of the interface in Figure 3. Lines 7-9 show the creation of a new mesh instance which creates the opaque handle `mesh` that is used in later calls

```
1  #include "iMesh.h"
3
4  int main( int argc, char *argv[] )
5  {
6      // create and populate the Mesh instance
7    iMesh_Instance mesh;
8    int geom_dim, ierr;
9    iMesh_newMesh("", &mesh, &ierr, 0);
10   iMesh_load(mesh, 0, "125hex.vtk", "", &ierr, 10, 0);
11
12    // get the geometric dimension of the mesh
13   iMesh_getGeometricDimension(mesh, &geom_dim, &ierr);
14
15    // get all 3d elements
16   iMesh_EntityHandle *ents;
17   int ents_alloc = 0, ents_size;
18   iMesh_getEntities(mesh, 0, iBase_REGION, iMesh_ALL_TOPOLOGIES,
19                     &ents, &ents_alloc, ents_size, &ierr);
20 }
```

**Figure 3.** Example use of the C-binding of the iMesh interface.

to refer to this instance of the interface. Line 10 shows a the use of the `iMesh_load` function to populate the mesh interface using a string name identifier. How the data is created is not specified; for example,it may be loaded from a file or generated on-the-fly. Line 13 shows a simple query for the geometric dimension of the mesh. Lines 16-19 show another query to retrieve all of the three-dimensional entities in the mesh, regardless of their particular topology.

Several implementations of the ITAPS mesh interface are well underway and are supported by mesh management toolkits such as FMDB (RPI) [2], MOAB (ANL) [3], NWGrid (PNL) [4], and GRUMMP (University of British Columbia) [5]. In addition to the development of underlying implementations, the ITAPS mesh interface has also been used in a variety of ITAPS services including the *Frontier-Lite* front tracking library [6], the *Mesquite* mesh quality improvement toolkit [7], the *Zoltan* partitioning toolkit [8], along with unstructured mesh refinement [9] and swapping tools [10].

## 4. Use of ITAPS Software in SciDAC Applications

The ITAPS team works closely with many different application teams to develop and deploy capabilities critical to their success. Early in SciDAC-1, application impact was made by applying existing tools to meet the needs of the application scientists. As SciDAC-1 progressed and new interoperable tools were developed, the impact was achieved with more sophisticated combinations of tools made possible by the SciDAC collaborative approach. For example, in the area of accelerator modeling, the ITAPS team is working closely with scientists from the Stanford Linear Accelerator Center (SLAC), to provide an adaptive simulation capability with error indicators, field function libraries, and mesh modification procedures based on ITAPS services and interfaces. To date, this work has enabled an order of magnitude improvement in the accuracy of predicted field quantities that influence wall losses in the Rare Isotope Accelerator device [11]. We also working on a collaboration among the TOPS (Toward Optimal Petascale Simulations) center, ITAPS, and SLAC to develop optimization procedures that allow automatic tuning of accelerator geometries to significantly increase the speed and decrease the cost by which new accelerators can be designed [12]. ITAPS researchers provide services for varying design

geometry, smoothing the mesh onto new models, and automatically computing sensitivities of the mesh with respect to design parameters [12]. In the area of fusion simulations, ITAPS researchers are contributing to a new effort at Princeton Plasma Physics Lab (PPPL) to develop an adaptive, high-order finite element discretization for their new M3D-C1 code. Ongoing work focuses on insertion of adaptive mesh refinement services and error estimators directly into the new code.

**More Information about ITAPS**

More information on the ITAPS project including detailed descriptions of the ITAPS services, interfaces and interactions with SciDAC application teams can be found at `http://www.itaps-scidac.org`. In addition, for those interested in ITAPS software, it can be downloaded from `http://www.itaps-scidac.org/software`.

**Acknowledgments**

This work was performed under the auspices of the U.S. Department of Energy by the University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48 (UCRL-CONF-232530); the Canadian Natural Sciences and Engineering Research Council under Special Research Opportunities Grant SRO-299160; and by Rensselaer Polytechnic Institute under DOE grant number DE-FC02-01ER25460.

**References**

[1] Dahlgren T, Epperly T, Kumfert G and Leek J 2005 *Babel User's Guide* CASC, Lawrence Livermore National Laboratory Livermore, California version 0.10.10

[2] Remacle J F and Shephard M 2003 *International Journal for Numerical Methods in Engineering* **58** 349–374

[3] Tautges T J, Meyers R E, Merkley K, Stimpson C and Ernst C 2004 *Sandia report SAND 2004-1592* (Sandia National Laboratories)

[4] Trease H 2006 The NWGrid mesh generation system Pacific Northwest National Laboratory - http://www.emsl.pnl.gov/nwgrid

[5] Ollivier-Gooch C F 1998–2005 GRUMMP — Generation and Refinement of Unstructured, Mixed-element Meshes in Parallel http://tetra.mech.ubc.ca/GRUMMP

[6] Fix B, Glimm J, Li X, Li Y, Liu X, Samulyak R and Xu Z 2005 *Journal of Physics: Conf. Series* **16** 471 – 475

[7] Brewer M, Diachin L, Knupp P, Leurent T and Melander D 2003 *Proceedings of the 12th International Meshing Roundtable* pp 239–250

[8] Devine K, Boman E, Heaphy R, Hendrickson B and Vaughan C 2002 *Computing in Science and Engineering* **4** 90–97

[9] Shephard M, Flaherty J, Jansen K, Li X, Luo X J, Chevaugeon N, Remacle J F, Beall M and OBara R 2005 *J. for Applied Numerical Mathematics* **53** 251–271

[10] Ollivier-Gooch C 2006 A mesh-database-independent edge- and face-swapping tool AIAA Paper 2006-0533. Presented at the 44th AIAA Aerospace Sciences Meeting

[11] Ge L, Lee L, Zenghai L, Ng C, Ko K, Luo Y and Shephard M 2004 *IEEE Conference on Electromagnetic Field Computations*

[12] Tautges T J 2005 *8th U. S. National Conference on Computational Mechanics*