IBM T. J. Watson Research Center

NEAMS Survey and Report on HPC Frameworks

John Magerlein

October 19, 2010

© 2010 IBM Corporation

Impediments to Broader Use of HPC Simulation

- Writing parallel codes is beyond the ability of many domain experts
- Difficult or impossible to couple multiple analysis codes
- Difficult to create and manage meshes
- Manual, multi-step modeling process
- Requires intimate familiarity with many pieces of software
- Few tools available to manage massive simulation data, manage workflows, or perform analytics
- No easy user interfaces
- Building and distributing code for a variety of platforms is complex

Need a comprehensive HPC simulation framework

© 2010 IBM Corporation

What Is an HPC Simulation Framework?

- Widely varying definitions in the HPC community
- Definition based on July 2010 NEAMS framework meeting:

  HPC simulation framework: The total collection of software needed to develop large-scale simulation codes and productively carry out complete simulation studies other than the domain-specific science and engineering code.
- A framework consists of many parts with different functions
  - Modular design, not highly integrated
- We have categorized framework parts into these groups:
  - Software backplane
  - Services
  - Data management, workflow, analytics, and user interface
  - Development framework
- HPC frameworks are at an early stage of design and development

© 2010 IBM Corporation

HPC Framework Software Categories - 1

- **Software backplane:** The part of the framework which provides coupling and allows plug-and-play usage for analysis codes and services. For mesh-oriented modeling, it generally includes a mesh and field database accessible to multiple codes and mesh management tools including partitioning and mesh-mesh coupling. If applications use a PDE toolkit such as PETSc, its parallel vector and matrix classes, solvers, and other tools can be considered part of the backplane.
- **Services:** Typically separate programs which provide widely useful capabilities such as mesh generation or visualization and interact with other parts of the framework using established interfaces.

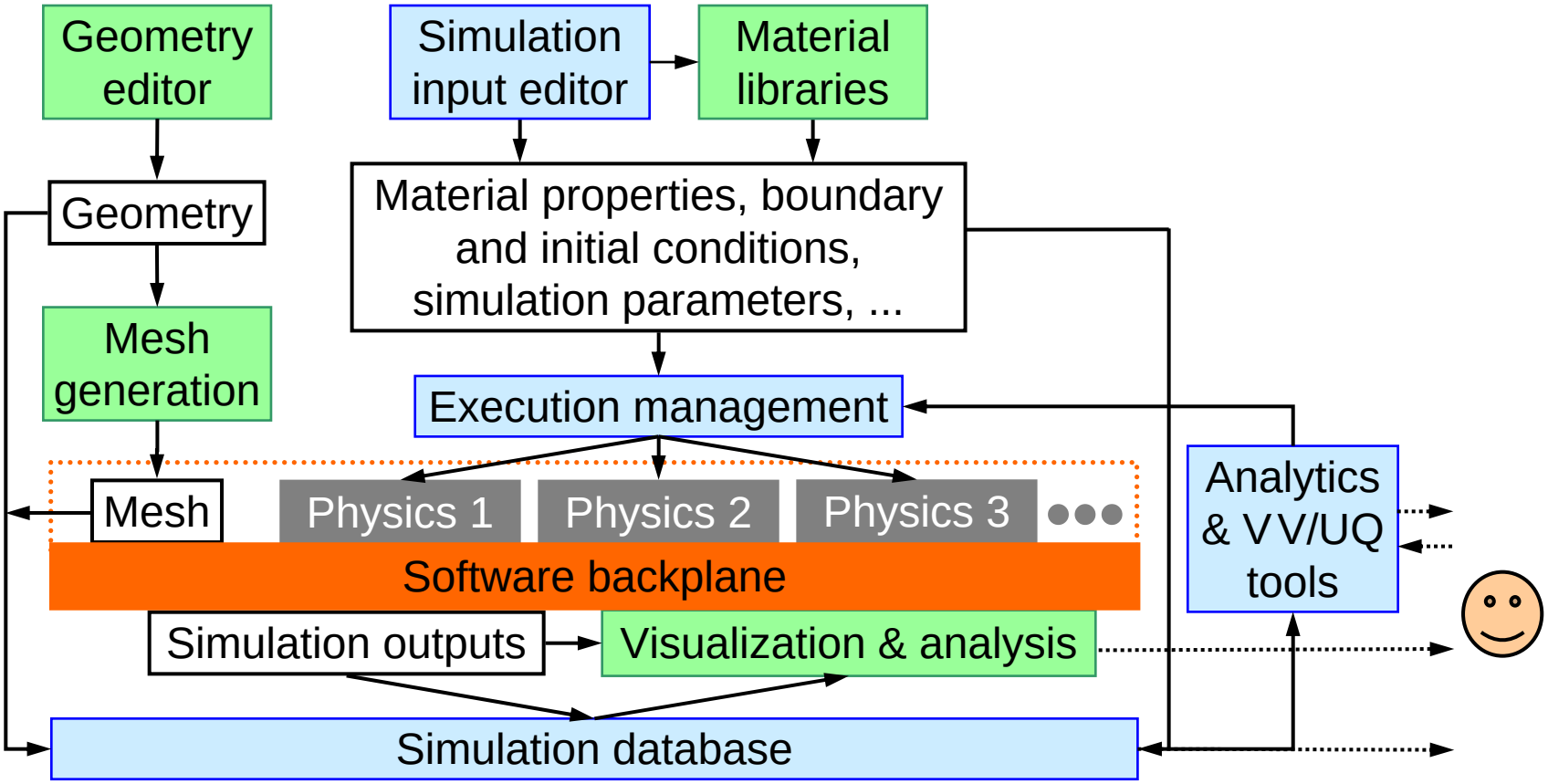
© 2010 IBM Corporation

HPC Framework Software Categories - 2

- **Data management, workflow, analytics, and user interface:** Software to store and catalog simulation inputs and outputs, manage complex simulation workflows, and assist users in interpreting simulation results. The latter includes tools to aid in verification, validation, and uncertainty quantification (VV&UQ). The user interface is also included in this category.
- **Development framework:** Tools used to develop HPC analysis and framework codes including configuration and build tools, software release and distribution tools, and software quality assurance tools.

© 2010 IBM Corporation

Example Simulation Framework



Framework categories:

Software backplane	Services	Data management, workflow, analytics, and user interface

Development framework (not shown) used to develop analysis and framework codes

© 2010 IBM Corporation

General Considerations for an HPC Framework

- Modular framework based on selected standards-based parts
- Must work with codes written in various styles
  - Different languages (Fortran, C, C++)
  - Older codes without making major internal changes
  - Codes which use latest PDE or other toolkits for fast development without knowledge of parallel programming details
- Ability to handle largest problems of interest
  - 10^{10} mesh cells in the near future; much larger by exascale timeframe
  - Will require significant innovation
- All critical parts, especially the backplane, must be open source
  - Must support use of proprietary codes when appropriate

© 2010 IBM Corporation

Software Backplane - PDE Solution Toolkits

Package	Source	Lic.	Lang support	Features
PETSc	Argonne	LGPL	C, C++, Fortran	Parallel vectors and matrices, linear and nonlinear solvers, time steppers, and other tools; interface to part of Trilinos and Sundials; well integrated; relatively good documentation
Trilinos	Sandia	LGPL BSD?	C++	Wide range of capabilities through separate packages, some better integrated and documented than others; includes STKmesh database
Sundials	LLNL	BSD	C, C++, Fortran	Robust time integrators and nonlinear solvers, optional user defined data structures
AMP framework	ORNL	?	C++	Wrapper around major parts of above packages; wrapper for libMesh mesh and field database, could use STKmesh or others; wrappers allow changing the underlying packages
Chombo	LBNL	BSD	C++	Finite difference methods to solve PDEs on **block-structured, adaptively-refined rectangular grids**; includes mesh generation and mesh database

© 2010 IBM Corporation

Software Backplane - Mesh and Field Databases Most Attractive for NEAMS

Package	Source	Lic.	Lang support	Features
MOAB/ iMesh	Argonne	LGPL	Fortran, C, C++, Python	iMesh mesh and field management, geometry management, parallelized, interface for code coupling
STKmesh	Sandia	LGPL BSD?	C++ only	Mesh and field management part of STK; parallelized, rich feature set; available as part of Trilinos
Salome (MED)	EDF	LGPL	Fortran, C, C++, Python	Mature comprehensive framework including geometry, MED mesh and field database, YACS coupling, GUI

These three databases are compared in more detail below

© 2010 IBM Corporation

Other Mesh and Field Databases

Package	Source	Lic.	Lang support	Features
libMesh	U Texas Austin CFDLab	LGPL	C++ only	Mesh management for unstructured meshes emphasizing adaptive mesh refinement, parallelized, supports CUBIT, uses METIS and ParMETIS, various linear algebra solvers
Rocstar, Roc...	Univ of Illinois	Similar to LGPL	C++, C, Fortran	Code coupling and related functions for solid rocket modeling
FMDB	RPI	?	?	Another implementation of iMesh, parallelized, little recent activity

Others not considered due to restrictive licenses: UG (Univ. of Heidelberg), deal.II

© 2010 IBM Corporation

Other Backplane-Related Software

Package	Source	Lic.	Lang support	Features
MOOSE	INL	?		Strong multi-physics coupling using JFNK
LIME	Sandia	?		Strong coupling of matrices from different codes (JFNK)
Overlink	LLNL	Will be open source	Stand-alone; file I/O	Code coupling library, second-order interpolation, now file based, working on in-line interface

© 2010 IBM Corporation

Services: Geometry and Mesh Generation

Package	Source	Lic.	Lang support	Features
Open Cascade	Open Cascade SA	Similar to LGPL		2-D and 3-D surface and solid modeling; used by Salome
CUBIT	Sandia	Not open source	Stand-alone; file I/O	Powerful interactive 2-D and 3-D mesh generation, editing, refinement, and visualization; 2-D and 3-D surface and solid modeling
GMSH	Univ of Liege	GPL	Stand-alone; file I/O	Simple geometry entry; mesh generation, editing, and visualization; used by Salome

- Many commercial solid modeling and mesh generation codes are also available
- Open-source mesh generation codes are very limited
- Work is required on parallel mesh generation and domain decomposition for large meshes

© 2010 IBM Corporation

Services: Visualization

Package	Source	Lic.	Lang support	Features
VisIt	LLNL	BSD	C++, Python, Java	Parallel visualization and analysis, GUI, powerful plug-ins; based on VTK
Paraview	KItware, Sandia, LANL	Similar to LGPL	C++, Python	Parallel visualization and analysis, GUI, powerful plug-ins; based on VTK

- Two excellent, very similar codes

© 2010 IBM Corporation

Data Management, Workflow, Analytics, and User Interface

Package	Source	Lic.	Lang support	Features
Salome	EDF	LGPL	Python	Some workflow management and an associated user interface; mature, but uses older protocols; active development
GS^3	PNNL	LGPL	N/A	Engineering and workflow environment based on wikis and data management extensions; easy to use, but very limited capabilities
NiCE	ORNL	?	N/A	Job launching system with GUI for beginning users; plans to develop other workflow tools

- These codes have limited capability and cannot handle large amounts of data
- Several heavy duty commercial packages are available, but not well suited to HPC modeling

© 2010 IBM Corporation

Development Framework

- Configure and build tools
  - GNU autotools – universally available on Unix systems
  - cmake – more flexible and supports builds on Windows
- Release and distribution tools
  - Git – gaining in popularity
  - Subversion
  - CVS
  - Mercurial
  - RSICC (Oak Ridge) for distribution of export-controlled packages
- Software quality assurance (SQA) tools
  - Plan from Livermore which establishes 5 risk levels

© 2010 IBM Corporation

Mesh Database Comparison: General (1 of 2)

Significant deficiencies

Capability	MOAB (Argonne)	STKmesh (Sandia)	Salome MED (EDF)
Languages supported	C, Fortran, C++, Python	C++ only	C, Fortran; MEDMEM C++
Number of API calls or classes	Implements the 120 iMesh API calls; MOAB 58 classes, 639 class members	264 classes, 679 class members	97 API calls
Input file types supported	MOAB (hdf5), Exodus, vtk, Cubit, SLAC, GMV, Ansys, Gmsh, STL	A separate coordinated library will provide this capability in the future	.med (hdf5); can import .unv and others
Types of meshes which can be stored	Unstructured and (with MOAB) structured meshes	Unstructured meshes	Unstructured and structured meshes
Mesh elements supported	Linear and quadratic triangle, quadrilateral, tetrahedron, pentahedron, pyramid, hexahedron, general hedra	Can plug in any desired element types	Linear and quadratic triangle, quadrilateral, tetrahedron, pentahedron, pyramid, hexahedron, general hedra

© 2010 IBM Corporation

Mesh Database Comparison: General (2 of 2)

Capability	MOAB (Argonne)	STKmesh (Sandia)	Salome MED (EDF)
Support for fields	General, but tag storage is inefficient; working on improved design	Rich and heterogeneous fields; high-performance,	Heterogeneous fields; high performance,
Other items supported	Entity sets with parent/child relationships; tags	Entity sets with subset-superset, defined-intersection, and stencil relationships	Groups, families
Coordinate types and units	Arbitrary; can store units as a tag	Arbitrary	Cartesian, spherical, cylindrical; stores axis units
Language for messages, documentation, and code	English	English	Documentation and messages in choice of English or French; variable & function names in French

© 2010 IBM Corporation

Mesh Database Comparison: Code Coupling

Capability	MOAB (Argonne)	STKmesh (Sandia)	Salome MED (EDF)
File-based data exchange	Yes	Yes	Yes
In-memory data exchange (languages)	Yes (Fortran, C, C++)	Yes (C++)	Yes (MEDMEM) (C++, Python)
Locations where fields can be defined	Elements including vertices	Elements including vertices	Elements including vertices, Gauss points
Can store fields on subsets of elements	Yes	Yes	Yes
Types of coupling supported	Weak; might be extended to strong	Weak; might be extended to strong with a separate coordinated library	Weak; might be extended to strong
Interpolation between different meshes	Linear interpolation is available; want to add quadratic	Plan to implement in the future with a separate coordinated library; capabilities to be determined	New interpolation capability is available

© 2010 IBM Corporation

Mesh Database Comparison: Parallelism

Capability	MOAB (Argonne)	STKmesh (Sandia)	Salome MED (EDF)
Provide interface for mesh decomposition	Yes	Does dynamic partitioning; application defines partition	Yes
Read mesh files in parallel efficiently	Parallel HDF5	A separate coordinated library will provide this capability in the future	Parallel HDF5
Tracks connections between cells on different processors	Yes	Yes	Yes (“mortar”)
Handling of ghost cells	Multiple levels automatic	1 level automatic; support for application defined	By application based on mortar data
Handling of mesh adaptation	Limited support for adapting mesh based on application requirements	Supports adapting mesh based on application requirements	Not at present
Write output files in parallel efficiently	Parallel HDF5	A separate coordinated library will provide this capability in the future	Parallel HDF5
Analysis codes of interest using parallelism	Codes are being adapted to use this capability	Codes are being adapted to use this capability	Codes are being adapted to use this capability

© 2010 IBM Corporation

Mesh Database Comparison: Quality and Maturity

Capability	MOAB (Argonne)	STKmesh (Sandia)	Salome MED (EDF)
Ease of build (build system)	Reasonably easy (autotools)	Trilinos build is complex but workable (cmake)	Complex from sources; prebuilt binary not difficult
Perceived quality	Occasional bugs found as might be expected for new code	Could not evaluate due to poor documentation; claims extensive unit and regression testing	MED mesh management seems good; other Salome parts have some stability issues
Documentation	Good	Poor; a tutorial manual is absolutely essential	Very good (~450 pages)
Maturity	Version 4 in beta test; new capabilities under active development	Available under Trilinos April 2010; new capabilities under active development	In production for several years (v. 2.3.6); new capabilities under active development
Maximum practical mesh size	Not known; limitations not clearly understood	Not known; limitations not clearly understood	Not known; limitations not clearly understood
Analysis codes using the backplane	Prototype usage by NEK5000 and UNIC	None known	Saturne, Aster, and others for modest problem sizes

© 2010 IBM Corporation

MOAB and MED Relative Performance

- Evaluated with a test program to read and analyze an example mesh
  - 6712434 vertices, 468000 faces, and 6480000 hexahedral regions

Characteristic	MOAB (Argonne)	Salome MED (EDF)
Initial memory usage	Reads entire mesh into memory before processing it	Only 4.4 MB
Memory to read and process example mesh	1755 MB (64-bit) ~1100 MB (32-bit)	241 MB (32-bit)
Comments about memory usage	▪ Memory requirement doubled by need to get and store entity handles ▪ Current tag storage method results in much larger memory usage – Could be improved	
Time to read and process example mesh	15 sec 14 sec user, 1 sec system	45 sec 14 sec user, 31 sec system

- IBM has exercised the parallel part of MOAB as part of ongoing performance optimization work

© 2010 IBM Corporation

Summary of Mesh and Field Database Comparison

- MOAB, STKmesh, and Salome MED are very similar in approach and general capabilities
  - Mesh-based with mesh and field database
  - Weak coupling; possible extension to strong coupling in similar ways
  - Support for parallelism, especially MOAB and STKmesh
- Any of the three could be quite satisfactory, but all require significant work to add required features and to scale for larger problems
  - MOAB's documentation and its implementation of the community-developed ITAPS iMesh interface are advantages
  - STKmesh supports only C++; needs good tutorial documentation
  - Salome is most mature; needs better parallel support (in progress)
  - MOAB and STKmesh development teams are too small to implement critical new capabilities in a timely way
- PDE toolkits should include a mesh and field database

© 2010 IBM Corporation

Framework Usage for Nuclear Energy Applications

Application	Framework Parts Expected To Be Used
NEAMS reactor codes	MOAB software backplane including mesh and field management, VisIt visualization; not defined how other capabilities will be provided
NEAMS fuels	Use AMP, which includes wrappers around much of PETSc, Trilinos, and SUNDIALS; mesh and field database AMPmesh is a set of wrappers currently calling libMesh, considering switch to calling STKmesh; could call MOAB if desired; NiCE user environment; not defined how other capabilities will be provided
NEAMS waste forms	September 2009 report lists available framework codes, but makes no choices; considering a complex mixture of commercial codes for workflow management; leaning toward mostly STKmesh/Trilinos tools
NEAMS SafeSep	Mostly interested in plant-level (tier 1) modeling, which requires a very different type of framework; have not thought much about needs for tier 2 and 3 modeling
Oak Ridge nuclear energy hub	Not yet well defined; expect to use AMP; LIME for strong code coupling
All EDF nuclear applications	Salome provides a full range of framework capabilities including MED mesh and field management; considering how best to provide backplane function for large simulations

© 2010 IBM Corporation

Developing an HPC Framework: Common Software

Common software

- Much easier interoperability
- Reduced development time and/or cost
- Greatly reduced support and training costs
- Developers see a clear direction and adopt framework sooner

Multiple similar codes

- Valuable in early development to promote innovation, give backup
- Competition may lead to better software
- Don't need to coordinate multiple development efforts
- Developers can use tools with which they are most comfortable

ECT approach

- Common user interface, configure and build, data management and analytics
- Plan to survey the required science and engineering codes used by the IPSC's to clarify the importance of a common software backplane
  - Need of backplane to interact with many codes is likely to drive toward commonality

© 2010 IBM Corporation

Proposed Plan to Develop an HPC Framework

- Complete an open-source mesh and field database
  - Need a larger development effort than present MOAB or STKmesh teams
  - Make prioritized list of necessary added capabilities
  - Releases after 1 and 2 years with all high-priority capabilities
- Encourage use of a PDE toolkit (e.g. PETSc) where appropriate
- Provide funded documentation and user education and support
  - Trilinos on-line support is a good example
- Complete development of remaining services
  - Open-source unstructured mesh generation would be of great value
- Develop data management, workflow, analytics, and user interface
  - Design and then develop and deploy in a staged fashion over 5 years
- Centralize and focus development framework activity
- Goal to design and deploy a reasonably comprehensive HPC framework within 5 years to enable broad range of HPC simulations

© 2010 IBM Corporation

Summary

- HPC frameworks are at an early stage of design and development
- A framework consists of many parts with different functions
  - Software backplane
  - Services
  - Data management, workflow, analytics, and user interface
  - Development framework
- Software backplane is a high priority
  - Mesh and field management, code coupling
  - Three leading mesh and field databases: any could be satisfactory, but all require significant additional development
- Could complete development of an HPC framework within 5 years
- Successful framework development will be a key enabler for broader usage of high performance computing in NEAMS and elsewhere

© 2010 IBM Corporation