A Component-Based Strategy for Scientific Application Development

Andy Salinger, SNL

SWP4XS = (Scalability)$^3$
1. Scalable Algorithms \(O(\text{Cores}^1)\) E.g. Multi-Level Prec
2. Scalable Development \(O(\text{Developers}^1)\) E.g. Build/Test/Release
3. Scalable Capability Delivery \(O(\text{Libraries}^1)\) E.g. Abstract interfaces

Technical Strategy: Create, use, and improve a common base of modular, independent-yet-interoperable, software components.

“Components” = ✓ Libraries ✓ Software Quality Tools ✓ Interfaces ✓ Demonstration Applications
List of Reusable Math Libraries and Software Tool Components

Analysis Tools
- Optimization
- UQ (sampling)
- Parameter Studies
- V&V, Calibration
- OUU, Reliability

Analysis Tools (embedded)
- 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
- Multi-Core
- Accelerators

Composite Physics
- MultiPhysics Coupling
- System Models
- System UQ

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

Mesh Database
- Mesh Database
- Geometry Database
- Solution Database

Discretizations
- Discretization Library
- Field Manager

Derivative Tools
- Sensitivities
- Derivatives
- Adjoints
- UQ / PCE Propagation

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

Utilities
- Input File Parser
- Parameter List
- Memory Management
- I/O Management
- Communicators

PostProcessing
- Visualization
- Verification
- Model Reduction

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

Discretizations
- Discretization Library
- Field Manager

Derivative Tools
- Sensitivities
- Derivatives
- Adjoints
- UQ / PCE Propagation

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

Utilities
- Input File Parser
- Parameter List
- Memory Management
- I/O Management
- Communicators

Mesh Database
- Mesh Database
- Geometry Database
- Solution Database

PostProcessing
- Visualization
- Verification
- Model Reduction

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

Utilities
- Input File Parser
- Parameter List
- Memory Management
- I/O Management
- Communicators

Mesh Database
- Mesh Database
- Geometry Database
- Solution Database

PostProcessing
- Visualization
- Verification
- Model Reduction

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

Utilities
- Input File Parser
- Parameter List
- Memory Management
- I/O Management
- Communicators

Mesh Database
- Mesh Database
- Geometry Database
- Solution Database

PostProcessing
- Visualization
- Verification
- Model Reduction

Software Quality
- Version Control
- Regression Testing
- Build System
- Backups
- Verification Tests
- Mailing Lists
- Unit Testing
- Bug Tracking
- Performance Testing
- Code Coverage
- Porting
- Web Pages
- Release Process
Anatomy of a Component-Based Application:

- Analysis Tools
  - Optimization
  - UQ

- Application
  - Nonlinear Model
  - Problem Discretization

- Interoperability Use Case
  - Linear Algebra Data Structures
  - AD Seed/Extract

- Mesh Tools
  - Mesh Database
  - Mesh I/O
  - Load Balancing

- Demo Apps
  - Version Control
  - Build System
  - Regression Testing

- Libraries
  - Interoperability

- Interface
  - Nonlinear
  - Transient

- ManyCore Node
  - Node Kernels
  - Multi-Core Accelerators

- PDE Assembly
  - Field Manager
  - Discretization

- PDE Terms
  - Linear Algebra
  - Data Structures

- Software Quality Tools
  - Regression Testing

- Demo Apps
  - Version Control
  - Build System
  - Regression Testing
## Benefits of Building Applications from a Base of Components Include ...

<table>
<thead>
<tr>
<th>Benefits of building and using Libraries:</th>
<th>Benefits of adopting a standard set of Software Quality Tools:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Code reuse</td>
<td>• Tool development can be done by experts</td>
</tr>
<tr>
<td>• Amortization of verification/maturation expenses</td>
<td>✓ Sharing common tools improves agility of staff between projects</td>
</tr>
<tr>
<td>✓ Developed by expert</td>
<td>• Unified deployment of libraries decreases barriers to interdisciplinary research</td>
</tr>
<tr>
<td>– Access to continued development</td>
<td></td>
</tr>
<tr>
<td>• Efficient development by small teams</td>
<td></td>
</tr>
<tr>
<td>• Modularity improves agility</td>
<td></td>
</tr>
<tr>
<td>• Decreased application code base</td>
<td></td>
</tr>
<tr>
<td>– Better encapsulation of protected code</td>
<td></td>
</tr>
</tbody>
</table>

### Benefits of abstract Interfaces:

- Modularity improves agility, flexibility, extensibility, maintainability
- ✓ Multiple capabilities can be delivered together (e.g. all Trilinos linear solvers)

### Benefits of building Demonstration Apps:

- ✓ Provides a template for a new codes
- ✓ Demonstrates capability integration
- • Drives interface development
- • Defines scope of new libraries&interfaces
- • Identifies gaps in our library coverage
Challenges of Building Applications from a Base of Components Include ...

**Challenges:**
- Requires more sophisticated software engineering
- Dependence on library developers that are not bound to your project
  - Credit for a project success may not go to the component developers
- Short-term decreases in productivity
  - Learning curve for using a library versus hand-coding
- Software support mortgage
- General-purpose implementations may lag in performance behind hand-tuned
The Components Technical Strategy Implies a Business Strategy as well

Base of Software Components:
- Libraries
- Software Tools
- Interfaces
- Demo Applications

Leverage the Base

Project Milestones
- ASC
- ASCR
- WFO
- LDRD
- NE
- BER

More Strategic Benefits
- Agility (e.g. exascale)
- Diversification
- Managing IP
- Maintenance
- Prioritization tool

Grow the Base