Architecture of the Earth System Modeling Framework

Climate

Weather

Data Assimilation

ESMF Project Description

GOALS: To increase software reuse, interoperability, ease of use and performance portability in climate, weather, and data assimilation applications

PRODUCTS:
• Core framework: Software for coupling geophysical components and utilities for building components
• Applications: Deployment of the ESMF in 15 of the nation’s leading climate and weather models, assembly of 8 new science-motivated applications

METRICS:

<table>
<thead>
<tr>
<th>Reuse</th>
<th>Interoperability</th>
<th>Ease of Adoption</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 applications use ESMF component coupling services and 3+ utilities</td>
<td>8 new applications comprised of never-before coupled components</td>
<td>2 codes adopt ESMF with less than 2% SLOC changed, or within 120 FTE-hours</td>
<td>No more than 10% overhead in time to solution, no degradation in scaling</td>
</tr>
</tbody>
</table>

RESOURCES and TIMELINE: $10.1M over 3 years
Computational Characteristics of Weather/Climate

- Applications solve time-dependent PDEs
- Mix of global transforms and local communications
- Load balancing for diurnal cycle, event (e.g. storm) tracking
- Applications typically require 10s of GFLOPS, 100s of PEs – but can go to 10s of TFLOPS, 1000s of PEs
- Required Unix/Linux platforms span laptop to Earth Simulator
- Multi-component applications, combined data/task parallelism
- Component hierarchies, ensembles, exchanges and multiple contexts
- Data and grid transformations between components
- Applications may be SPMD/MPMD, concurrent/sequential, combinations
- Most applications are implemented in Fortran, some in C/C++
- Parallelization via MPI, OpenMP, shmem, combinations
- Large applications (typically 100,000+ SLOC)
ESMF Class Structure

- **GridComp**: Land, ocean, atm, … model
- **State**: Data imported or exported
- **Bundle**: Collection of fields
- **Field**: Physical field, e.g. pressure
- **Array**: Hybrid F90/C++ arrays
- **Grid**: LogRect, Unstruct, etc.
- **PhysGrid**: Math description
- **DistGrid**: Grid decomposition
- **DELayout**: Virtual machine and comms
- **Route**: Stores comm paths
- **CplComp**: Xfers between GridComps
- **Regrid**: Computes interp weights
- **Utilities**: Machine, TimeMgr, LogErr, I/O, Config, Base etc.
- **Data**: Communications
- **Superstructure**
- **Infrastructure**
- **F90**
- **C++**