FMS: the GFDL Flexible Modeling System

ESMF Components Workshop

SCI/GFDL Princeton University

V. Balaji

14 May 2003 GFDL, Princeton, NJ
GFDL Strategic Objectives:

- Advance expert assessment of global and regional climate change through research.
- Develop Earth Systems Models (ESMs) for climate variability and change.
- Provide timely and reliable knowledge for the nation on natural climate variability and anthropogenic change.

GFDL is a NOAA Climate modeling centre. The primary focus is the use and development of coupled climate models for simulations of climate variability and climate change on short (seasonal-interannual) and long (decadal-centennial) time scales.
Reliance on Cray vector architecture in previous decades.

Transition to scalable computing began in 1997 with the acquisition of Cray T3E.

Current computing capability: $2 \times 320 + 6 \times 128 + 2 \times 64$ Origin 3000.

GFLOPS Computing
Technological trends

In climate research...

Increased emphasis on detailed representation of individual physical processes governing the climate; requires many teams of specialists to be able to contribute components to an overall coupled system.

In computing technology...

Increased in hardware and software complexity as we shift toward the use of scalable high-performance computing architectures, as we shifts toward use of scalable high-performance computing, as we shift toward the use of scalable high-performance computing architectures.
In software design for broad communities...

"Rough consensus and working code." [IETF]

...through consultation and prototyping across the user community. The open standards evolve diversity requirements through "open standards." The open source communities...
The GFDL response:

- Modernization of modeling software
- RAMeterizations, development of community-wide standards for components
- Modular design for interchangeable dynamical cores and physical parameters
- Abstraction of underlying hardware to provide uniform programming
- Abstracted development model: many contributing authors. Use high-level abstract language features to facilitate development process.
- Distributed development model: many contributing authors. Use high-level abstract language features to facilitate development process.
- Model across vector, uniprocessor, and scalable architectures.

Modernization of modeling software
Develop high-performance kernels for the numerical algorithms underlying non-linear flow and physical processes in complex fluids.

Establish standards, and provide a shared software infrastructure implementing those standards for the construction of climate models and model components portable across a variety of scalable architectures.

Maintain high-level code structure needed to harness component models and re-purpose model subsystems developed by independent groups of researchers.

Benchmarked on a wide variety of high-end computing systems.

Run in production on very different architectures: parallel vector (PVP), distributed massively-parallel (MPP) and distributed shared-memory (NUMA).

FMS: the GFDL Flexible Modeling System
Architecture of FMS
FMS shared infrastructure:

Machine and grid layers

MPP modules

Communication kernels, domain decomposition and update, parallel I/O.

Time and calendar manager

Tracking of model time, scheduling of events based on model time.

Diagnostics manager

Runtime output of model fields.

Scientific libraries

Uniform interface to proprietary and open scientific library routines.
Communication kernels provide uniform interface to:

- Pointer-sharing and direct copy on shared-memory and distributed-memory (NUMA).
- MPI or SHMEM on tightly-coupled distributed memory (T3E).

MPI message-passing across clusters.

Provide uniform interface to:

**Communication Kernels**
User Interface to Communication Kernels

```fortran
!perform computations on f
    call mpp-define-domains (f, domain)
    ...
    ( ni, nj, domain, xhalo=2, yhalo=2 )
    call mpp-update-domains (f, domain)
    type(domain2D) :: domain
```

```fortran
call mpp-define-domains((/1,ni,1,nj/),domain,xhalo=2,yhalo=2)...
```

```
...,
```

```fortran
10
```

```fortran
call mpp-update-domains(f,domain)
!perform computations on f
```

```
10
```
Parallel I/O interface

mpp_io_mod is a set of simple calls to simplify I/O from a parallel processing environment.

mpp_io_mod supports three types of parallel I/O:

- Single-threaded I/O: a single PE acquires all the data and writes it out.
- Multi-threaded, single-fileset I/O: many PEs write to a single file.
- Multi-threaded, multi-fileset I/O: many PEs write to independent files (requires post-processing).

mpp_io_mod is designed to deliver high-performance I/O from distributed data in the form of self-describing files (verbose metadata).

mpp_io_mod uses the domain decomposition and communication interfaces of mpp_mod and mpp_domains_mod.

mpp_io_mod is a set of simple call interfaces to parallel I/O.
mpp_io_init() •
mpp_write_meta() •

mpp_write() •

mpp_read_meta() •

mpp_read() •

mpp_close() •

mpp_open() •

mpp_io-mod API
User Interface to Parallel I/O

```fortran
call mpp-define-domain(unit, \text{domain}, \text{field}, \text{field}, t, 
\text{field}, x, y, z, t, \text{field}, x, y, z, \text{field})

\ldots

\ldots

\ldots

\ldots

call mpp-open(unit, file, action=MPP_WRONLY, format=MPP_IEEE32, 
access=MPP_STENCIL, threading=MPP_SINGLE)

call mpp-write-meta(unit, x, 'X', 'km', ...)

call mpp-write-meta(unit, field, (/x, y, z, t/), 'Temperature', 'kelvin', ...)

\ldots

\ldots

call mpp-write(unit, field, domain, f, tstamp)
```
Physically identical grids (e.g. ocean and sea ice) exchange data without interpolation.

- Change software, and non-blocking.
- All calls exchange local data; data-sharing among processors is internal to the exchange software.
- Conservative interpolation up to second order.
- Each cell on exchange grid “belongs” to one cell on each parent grid.

Exchange grid
Features of the FMScoupler:

- Encapsulated boundarystate and boundary fluxes.
- Single location for initialization and linking of boundary fields.
- Use of field manager to organize operations on individual fields and field bundles.
- Implicit coupling between land-ocean surface and atmosphere on atmospheric timestep; explicit coupling between land-ocean surface and ocean on coupling timestep.
- Support for serial and concurrent coupling within single executable.

on coupling timestep.
ENDDO

// fast loop

END main

END DO
coupler_main

fastloop
dona=1,num_atmos_calls

Time=Time+Time_step_atmos
callsfc_boundary_layer(Atm,Land,Ice,&
Land_ice_atmos_flux)
callupdate_atmos_model_down(Land_ice_atmos_flux,Atm)
callflux_down_from_atmos(Time,Atm,Land,Ice,&
Atmos_land_flux,Atmos_ice_flux)
callupdate_land_model_fast(Atmos_land_flux,Land)
Atmos_ice_flux
Atmos-land-flux ATM
Land-ice-atmos-flux ATM
ice ATM
A call update-ice-model-fast ATM
Atmos-ice-flux ATM
Land-ice-atmos-flux ATM
A call update-atmos-model-up ATM
Land-ice-atmos-flux ATM
Ice ATM
Land ATM
Ice ATM
A call update-land-model-fast ATM
Land-ice-atmos-flux ATM
A call update-atmos-model-up ATM
Land-ice-atmos-flux ATM
Ice ATM
Land ATM
Ice ATM
A call stc-boundary-layer ATM
Ice ATM
Time = Time + Time-step-atmos
A do na = 1,num-atmos-calls

couple-main fast loop
Example: ocean boundary
Flux exchange

Three types of flux exchange are permitted: REGRID, REDIST and DIRECT.

REGRID physically distinct grids, requires exchange grid.
REDIST identical global grid, different domain decomposition.
DIRECT identical grid and decomposition.

Current use: REGRID between atmosphere, land, ice, ocean.

 tween ocean ice. Flux exchange exchange
FMS component models

Atmosphere:
- BGRID: hydrostatic finite difference model on a staggered Arakawa B grid and 1 soil/vegetation layer's, 1 temperature layer's (Wyman);
- SPECTRAL: hydrostatic spectral transform model also with the hybrid $f/\sigma$ vertical coordinate (Wyman);

Ocean:
- MOM primitive equation ocean climate model with generalized horizontal coordinates and vertical coordinates; full suite of physics options, compatible with state-of-art adjoint compiler (Pacanowski, Griffies, Rosati, Harrison);
- Spectral shallow water, 2D energy balance, data model, etc.
- $\sigma$ vertical coordinate (Held, Phillips);
- SPECTRAL: hydrostatic spectral transform model also with the hybrid $f/\sigma$ vertical coordinate (Wyman);
- BGRID: hydrostatic finite difference model on a staggered Arakawa B grid and...

Land:
- Land Dynamics model (LaD) 5 temperature layers, 1 soil/vegetation types;
- Ice: Sea Ice Simulator (SIS) full ice; ice dynamics with elastic-plastic-viscous rheology (Harrison);
- Ice: Sea Ice Simulator (SIS) full sea ice dynamics with elastic-plastic-viscous rheology (Winton);
- Ocean: MOM primitive equation ocean climate model with generalized horizontal coordinates and vertical coordinates; full suite of physics options, compatible with state-of-art adjoint compiler (Pacanowski, Griffies, Rosati, Harrison);
- Spectral shallow water, 2D energy balance, data model, etc.
- $\sigma$ vertical coordinate (Held, Phillips);
- SPECTRAL: hydrostatic spectral transform model also with the hybrid $f/\sigma$ vertical coordinate (Wyman);
- BGRID: hydrostatic finite difference model on a staggered Arakawa B grid and...
To incorporate your own ocean model (say) into FMS, you have to provide a few key routines (ocean-model_update, ocean-model_init, and ocean-boundary-type). It helps to use the FMS infrastructure but not essential.

Fitting into FMS
Atmospheric physics options

- Radiation
  - Diurnal cycle, radiative effects of trace gases; Simplified Exchange Approximation
  - Liquid/ice cloud radiative properties (SW), Liquid/ice cloud radiative properties (LW)
- Clouds
  - 3 prognostic tracers, prognostic stratiform clouds;
  - Relaxed Arakawa-Schubert, convective-stratiform detrainment;
- Convection
  - Relaxed Arakawa-Schubert, convective-stratiform detrainment;
  - PBL Mellor-Yamada 2.5, Monin-Obukhov similarity theory;
- Gravity wave drag
  - Pierrehumbert-Stern

Options:

- Prognostic stratiform clouds;
- Simplified Exchange Approximation
The FMS user interface

Comprehensive website for all information and documentation:

http://www.gfdl.noaa.gov/~fms

- Standard and custom diagnostic suites;
- Relational database for archived model results;
- Source code maintenance under CVS; browse over the net using webCVS;
- Model configuration, Launching and Regression testing encapsulated in XML;
- Comprehensive website for all information and documentation:
Current GFDL activities using FMS

- Incorporation of global biogeochemical models into coupled model for carbon cycle modeling.
- Development toward upcoming IPCC cycle.
- Development of seasonal-interannual forecasting capabilities.
- Mesoscale eddy-permitting simulations of the southern ocean.
Future developments: algorithms and models

- Development of non-hydrostatic atmospheric model options.
- Capabilities, currently testing:
  - Donner deep convection including convective updrafts and MCCs, convective and mesoscale downdrafts;
  - Bretherton-Grenier PBL;
  - Held-Klein very stable PBL;
  - Enhanced convective (Alexander-Dunkerton) and mountain gravity wave drag;
  - New atmospheric physics options currently under testing:
    - Lady model architecture to permit modular sub-systems for vegetation and soil hydrology.
  - New atmospheric physics options currently under testing:
A staged public release of FMS is currently underway.

Future developments: public release

- Complete model configurations.
- Atmospheric dynamical cores and ocean model (MOM4) code (June 2003)
- Infrastructure (March 2002)
Future developments: FMS and community standards

FMS authors are now active participants in the design of the Earth Systems Modeling Framework (ESMF) community-wide modeling standard and framework, for which FMS is a design prototype. ESMF is currently scheduled for public release between 2003 and 2005.

But this is not enough...
The "standard benchmarks" (LINPACK, SPEC, etc.) do not yet reflect this trend.

 Freedoms of innovation in software and hardware architectures for scalable systems. By developing an open standard for the distributed grid layer, we permit much greater freedom of innovation in software and hardware architectures for scalable systems.

 Standards currently sit in the machine layer (e.g., MPI, netCDF).

 **Upward evolution of standards**
The FMS infrastructure demonstrates that it is possible to write a data-sharing layer spanning that shared memory, distributed memory, and thread nesting. The current standardization efforts (ESMF, PRISM) depart from BLAS, MPI, etc in that they are explicitly formulated in high-level language constructs (classes, modules, types) that they are expressed in high-level language abstractions. The “standard benchmarks” do not yet stress the high-level language abstractions used by this community. The HPC industry and the standards bodies must be actively involved in this effort.

• The current standardization efforts (ESMF, PRISM) depart from BLAS, MPI, etc in that they are explicitly formulated in high-level language constructs (classes, modules, types) that they are expressed in high-level language abstractions. The “standard benchmarks” do not yet stress the high-level language abstractions used by this community. The HPC industry and the standards bodies must be actively involved in this effort.

• It is possible to write another layer that expresses these operations in a manner natural to our algorithms (“halo update”, “data transpose” instead of “buffered send”, “thread nesting”). The API is not as extensive as, say, MPI, but has been designed to serve the climate/weather modeling community.

• It is possible to write another layer that expresses these operations in a manner natural to our algorithms (“halo update”, “data transpose” instead of “buffered send”, “thread nesting”). The API is not as extensive as, say, MPI, but has been designed to serve the climate/weather modeling community.

The FMS infrastructure demonstrates that...
The jaws of code complexity

multithreaded speculative prefetch

load a to vreg

common a (int *j, n) { do

error = atmos()