

PRISM-ESMF Collaboration

V. Balaji

Princeton University and GFDL

ESMF Inter-Agency Meeting

NASA/GSFC

30 November 2004

Collaboration between PRISM and ESMF: overview

- FMS-MOM4 as case study: making a FMS component into ESMF and PRISM components.
- Can the PRISM coupler be an ESMF coupler?
- Merging of configuration files
- Grid standard
- What's happening to CF

Features of the FMS coupler

- Predefined list of components (atmosphere, ocean, land, ice).
- Encapsulated boundary state and boundary fluxes with predefined fields. (There is provision for adding an arbitrary bundle of tracers at runtime).
- Support for serial and concurrent coupling within single executable.
- Implicit coupling between land-ocean surface and atmosphere on atmospheric timestep; explicit coupling between ocean surface and ocean on ocean timestep. The atmosphere run method is split into down-up phases for the tridiagonal solver.
- Coupler serves as the scheduler for components, which return control to the coupler at the end of an independent execution segment.
- Support for ensembles (multiple instances of the same component).

Fitting into FMS

To incorporate your own ocean model (say) into FMS, you have to provide `init/run/exit` routines (`ocean_model_init`, `update_ocean_model`, `ocean_model_end`) and also encapsulate your ocean boundary state into `ocean_boundary_type`.

It helps to use the FMS infrastructure but not essential. For users with, say, a solo atmospheric model code that they wish to couple to MOM, the advantage is that MOM is distributed with the FMS coupler, and all that is needed is to write a few “wrapper” routines for the atmospheric model.

The ESMF coupler

The ESMF coupler closely follows the architecture of FMS, but with a more general and powerful notion of a component. A component is defined by

- its structure, consisting of an `init`, `run` and `exit` method. Unlike FMS, the coupler does not directly invoke these routines. These are registered for use by an `ESMF_SetServices()` call.
- its import and export states (analogous to the boundary fluxes and boundary states in FMS).
- a coupler must be written for every pair of components. This is not as onerous as it sounds: coupler functions are fairly generic and once one exists, other similar components can be plugged in with no effort at all. Efforts are underway to generalize this “middleware”.

Making a MOM ESMF component

- Register the `init/run/exit` methods.

```
type(ESMF_GridComp) :: comp
  call ESMF_GridCompSetEntryPoint(comp, ESMF_SETRUN, update_ocean_model, ...)
```

(1)

- Create the import and export states from FMS datatypes. This can be done entirely with pointers: no data movement is involved.

```
ocean%u_surf=>expFMSArray(1)%ptr
expESMF_Field(i)=ESMF_FieldCreate(ocnGrid, expFMSArray(i)%ptr, ...)
call ESMF_StateAddField(expState, expESMF_Field, ...)
```

(2)

- Run the component.

```
type(ESMF_GridComp) :: compOcn
compOcn=ESMF_GridCompCreate(vm, "Ocean", rc=rc)
call ESMF_GridCompSetServices(compOcn, OceanRegister, rc)
call ESMF_GridCompRun(compOcn, impOcn, expOcn, topClock, rc=rc)
```

(3)

The PRISM coupler

The PRISM coupler is designed with additional constraints: where components may not be able to be part of a common executable, where their execution sequence may not be easily parsed into **init**, **run**, and **exit** methods, and where components may not be able to return control to an external entity during execution. This has led to a different design, where concurrency is the norm, and components and the PRISM coupler may all be independent executables.

External configuration files (PMIOD/SMIOC/SCC) are used to share runtime configuration information between components and the PRISM system.

After configuration, components exchange data using **PRISM_Put** (non-blocking) and **PRISM_Get** (blocking) calls. All execution is concurrent, with the **PRISM_Get** block implicitly enforcing synchronization.

Making a MOM PRISM component

Write the SMIOC configuration information for MOM.

```
<transient local_name="ocean_SST">
  <standard_name>sea surface temperature</standard_name>
  <computation mask="true" mask_time_dependency="false"
                method_type="mean">
    <associated_gridfamily local_name="ocn_grid" />
    <associated_compute_space local_name="center" />
  </computation>

  <intent>
    <output transi_out_name="ocean_SST_out">
      <exchange_date>
        <period>
          <nbr_secs type="xs:integer">43200</nbr_secs>
        </period>
      </exchange_date>
      <corresp_transi_in type="xs:string">atm_SST_in</corresp_transi_in>
      <component_name type="xs:string">atm</component_name>
    </output>
  </intent>
</transient>
```

Making a MOM PRISM component

- Ingest the configuration information in `ocean_model_init`.

```
call prism_def_var ( var_id, var_name, grid_id, method_id, &  
    mask_id, var_nodims, actual_shape, var_type, ierr )
```

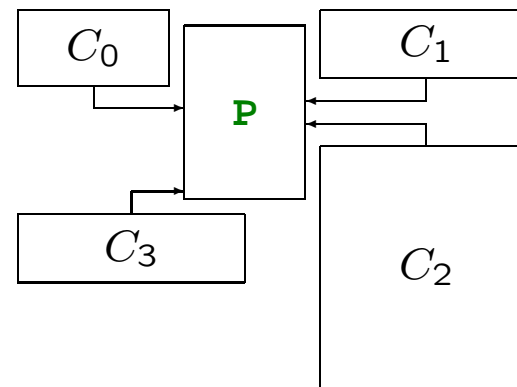
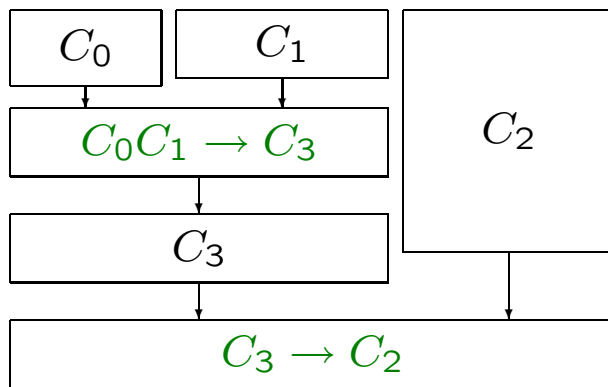
(4)

- Modify `update_ocean_model` to add the appropriate `PRISM_Put` and `PRISM_Get` calls.

```
call get_date (Time%model_time, date%year, date%month, &  
    date%day, date%hour, date%minute, second)  
call PRISM_calc_newdate ( ... )  
call PRISM_Put( var_id, date, date_bound, data, info, ierr )
```

(5)

Comparison of coupling models



- In FMS and ESMF, after each independent component run segment, control is returned to the coupler, which runs on the union of all PEs of its child components.
- PRISM uses a client-server model where all components execute concurrently, and the coupler **P** processes their **PRISM_Put** and **PRISM_Get** requests. Configuration of the coupler is through external files (SMIOC/SCC).

Configuration management

There are two places to put descriptive information about fields, components, and configurations: in ESMF's metadata-laden data structures and in external configuration files.

PRISM's XML configuration files are currently oriented toward just coupler configuration. But there are placeholders for "persistent" attributes which could be used for comprehensive model description.

Proposal:

- Static information (field metadata) will be in ESMF data structures.
- Runtime configuration will be in external files.
- External files will be in XML and based upon PRISM PMIOD/SMIOC/SCC files.
- Dataflow: static info: **ESMF_Field**→**PMIOD**.
- Configuration info: **SMIOC**→**ESMF_Attr**→**ESMF_Field**.
- Connectivity info: **SCC**→**ESMF_Attr**→Coupler.
- Include all model (including component) information into XML config files.
- Extend **ESMF_Attr** to ingest XML files directly.

ESMF's metadata-laden data structures

Earth system models can broadly be described as composed of components in which physical quantities are integrated on a physical grid. In a framework like ESMF, these are described in terms of 5 layers of abstractions consisting of *metadata-laden data structures*. These layers are:

grid describes the physical grid in a standard way, so that component-neutral regridding software can be used to transform quantities from one grid component to another, with no knowledge of those components themselves. We seek to inscribe the grid metadata within community standards and conventions, so that analysis tools cognizant of these conventions may take advantage of grid information.

field consists of the physical variable discretized on a **grid**, along with metadata describing the physical quantity itself. The field metadata in ESMF have been designed to resemble the CF convention, so that CF-compliant model output may be produced if desired.

state is the instantaneous state of some set of **fields** within a model component. Typically these are used as part of “import” and “export” states that are exchanged between components; but they are often used to contain the entire model state as well.

ESMF's metadata-laden data structures

attribute configuration attributes of a component: these are very generic, but are intended to contain all the physical input parameters used to configure a model.

component the top level entity of this design. Components are hierarchical: that is, they may be composed of other components. The top-level **component** is the application or model itself.

These software layers exist in the ESMF, and ESMF-compliant models in the near future will be using these abstractions, rich in metadata, to describe a wide range of models across the weather and climate community. Simply by using these abstractions and encoding them in model output, we are creating a layer of ***formal, structured, hierarchical metadata***. We call this the ***model metadata layer***, and it is the core of the proposed standard. The model metadata layer is what makes possible for either a fully-configured model configuration or a model dataset to be the result of a database query.

Needed conventions

- Shared time
- Shared grid information...
- CF standard names for variables
- Does CPR occur before or after a coupling event?
- What's exchanged, a pointer or a copy?

A related proposal calls for the development of a generic coupler layer with these conventions defined.

Grid metadata

Grid descriptors are in our data structures, covering a wide variety of actual grids. How to convert these to a grid standard?

- Should contain everything that is in PRISM grid definition routines such as: `prism_def_grid`, `prism_set_corners`, `prism_def_mask`, `prism_def_subgrid` (for the vector points). This includes coverage of logically rectangular as well as unstructured polygonal grids.
- Could contain information that is in the exchange grid.
- Could contain mask information that is to be shared between components (e.g land-sea mask)

Toward a layered approach

- In FMS and ESMF, the coupler is part of a *superstructure*, which exercises control and scheduling functions for components. Components must be structured in terms of *init*, *run* and *exit* methods. In PRISM it is part of the *infrastructure*, sharing configuration information and processing requests from components. This is the key architectural difference, not the MPMD/SPMD distinction. Synchronization in either architecture is provided by our applications, which remain in unison, never more than one coupling timestep out of synchronicity.
- The ESMF generic component interface, discussed in another talk, is a powerful abstraction for creating coupled models, in contrast with FMS, which is specifically targeted at coupled climate models with standard components. It is likely that a generic coupler with FMS-like structure and capabilities will emerge for coupled climate modeling, but based on ESMF components. This will be a significant community resource.
- The ESMF component and state data structures are rich in metadata, matching the information provided in PRISM configuration files. It is likely that a layered approach sharing configuration management standards will emerge shortly. Grid standards are an important emerging development in this area.
- Drivers for FMS/ESMF/PRISM can be automatically generated based upon this metadata.

Next steps

- A series of ESMF-PRISM workshops, of which the first was held at GFDL in September 2004.
- Independent proposals with explicit collaborative elements.
- Establishment of scientific partnerships: MOM4-ECHAM5, MOM4-NCEP.
- WGCM recommends formal funded activity for CF convention and standards.
- Joint ESMF-PRISM proposal to CF (grid standard).