Final Report
# of the
Technical Workshop on WRF-ESMF Convergence

February 9-10, 2006
Boulder, CO

1. Software Framework Convergence Strategies

We see three possibilities for WRF to utilize ESMF.

1. WRF uses ESMF for high-level coupling (i.e., coupling of the entire WRF code to other models).
2. WRF uses ESMF low-level utilities.
3. WRF uses ESMF components internally – for example, using ESMF-compatible interfaces for physics and dynamics packages.

We believe that the first two options are appropriate near-term strategies. WRF has already been wrapped as an ESMF component so that it can be coupled to other ESMF-compatible applications using an ESMF coupler and driver. WRF also uses one of the ESMF low-level utilities, the ESMF calendar manager. The use of other ESMF low-level utilities will be considered for WRF based on their availability, features, maturity, and user demand.

The third option, using ESMF component concepts internal to WRF, is much more complex and requires further study and careful assessment of costs, user benefits, and user demand. It is possible that this approach could facilitate interoperation of other ESMF-compatible components with WRF. While it might add additional complication to the WRF infrastructure, that might not be the case, since movement toward ESMF-compatible interfaces may be a natural outgrowth of further simplification of the WRF architecture, as described in Section 3. At this time, there is no consensus on the relative merits of this option. However, we do not believe that there is an immediate need to explore these possibilities. Opportunities for further alignment between the two frameworks should be considered as they arise.

None of the three options above requires that the ESMF library be linked with WRF in order to build and run the WRF code – WRF could continue to be run without ESMF. This preserves continuity and transparency for users having no need for ESMF. Since its inception, WRF developers have employed a strategy of "defensive programming." As part of this strategy, the WRF code encapsulates external libraries under WRF interfaces and makes optional the inclusion of even standard libraries such as NetCDF and MPI.

This minimizes risks arising due to the transience of many HPC packages, provides WRF users with the simplest possible build-and-run environment, and preserves generality over a range of applications, installations, institutions, and computer platforms. The WRF development team intends to follow the same strategy with ESMF.

Allowing the use of ESMF interfaces directly (option 3), or requiring ESMF in order to build and run WRF, could be reconsidered under the following conditions:

- There was a clear and strong desire on the part of WRF users to do so; and
- ESMF demonstrated portability, ease of use, overall maturity, and a level of community acceptance that would ensure its smooth integration with WRF. (For example, one prerequisite might be that the ESMF library was required by other large community codes.)

Additional points that were raised were the following:

It is estimated that it will take 3-5 years for ESMF to fully support WRF infrastructure needs and another 3-5 years to rewrite WRF to use the ESMF infrastructure alone to reproduce WRF's current functionality.

The ESMF team is supporting a wide range of users and programs, while WRF software developers are focusing on WRF users only. WRF software developers are thus in a better position to respond to specific WRF support issues than the ESMF development team.

The WRF software is complex because of its requirements; ESMF is also complex because of its requirements. Both frameworks make use of layering --software engineering best practices of information hiding, separation of concern, and abstraction -- to make such complexity manageable. This comes at a cost of increased opacity, which some users find to be a significant impediment. Recognizing that developers and users sometimes have a need to work with and understand detailed aspects of the code, including the internal workings of the frameworks themselves, we feel it is critical that software be clear, well documented, and understandable. However, we also point out that dependence on code written by outside developers is an inescapable result of the decision to leverage community software, and an increase in opacity is a consequence of using any software framework.

Finally, it was emphasized that WRF is foremost a science model with specific needs and requirements. The WRF software infrastructure and ESMF are technical frameworks. However, the WRF framework, developed and supported alongside and in concert with the developing WRF model, is relatively specific to the intra-component needs and requirements of WRF. It addresses only peripherally more general inter-component interactions. ESMF will provide some general corresponding intra-component functionality, but its principal focus inter-component (i.e. coupling). They are fundamentally different entities, and one cannot and should not replace the other.

2. Assessment of ESMF and WRF for Grid Nesting

Coupling of ESMF components is somewhat similar to nesting within WRF. As ESMF development continues this is another area where components of the ESMF infrastructure might become useful inside WRF. However, NWP nesting requires scientific algorithms, such as mass balancing, that are not within the scope of ESMF (or WRF) infrastructure. Moreover, the existing WRF infrastructure is optimized for nesting (i.e. for frequently passing large 3-D data between nests). By contrast ESMF is optimized for coupling distinct geophysical components, such as land, sea, sea-ice and air components which typically involve 2-D data passed relatively infrequently. ESMF could in the future be optimized for the NWP nesting case, and there are advantages to the greater generality of ESMF – for example it can couple either serially or concurrently. WRF implements only serial nesting, i.e. one nest is active at a time and it is distributed across all processors. Concurrent nesting, running nests on different sets of processors, may have advantages. Concurrent nesting is not currently supported by WRF, but if ESMF were extended to support nesting it would automatically provide both serial and concurrent nesting, just as it does today for coupling.

3. Streamlining the WRF Framework

Unrelated to WRF-ESMF convergence questions, a number of other issues for making the WRF software easier to understand and work with were identified and discussed.

It was agreed that the complexity and layered organization of the software implementing I/O within WRF is appropriate, considering the degree of flexibility and amount of functionality it supports. The extent of layering and encapsulation in the I/O software has caused considerable difficulty and delay for those users who have been required to understand and work with it. It was recommended that detailed documentation of the I/O Stack within the WRF Software Framework be developed and disseminated, along with examples of use.

The WRF Registry, a concise data-dictionary and automatic code generation mechanism within the WRF software, follows an accepted approach within the high-productivity computing community and has proven useful for WRF users and developers. However, over-reliance on the Registry to deal with inadequacies in Fortran90 compilers early in the project is no longer required, given improvements in compilers since that time. This particular role for the Registry – which was not envisioned in the original design – will be eliminated by the end of calendar year 2006, resulting in a significant reduction in the amount of automatically generated code. More importantly, this will eliminate unnecessary and confusing compiler work-arounds in the WRF model software.

Two items handled by the Framework, how physics packages are initialized and the WRF build mechanism, can be improved. The interface to physics routines themselves is standard, straightforward and easily interchangeable with other models; however, initialization of physics packages that require it has been a hindrance and should be standardized. The WRF build mechanism – that is, the mechanism by which WRF is installed on a computer – is complicated and non-standard (though it is not clear that such a standard yet exists), and has caused difficulties for users attempting to incorporate WRF software into their codes or vice versa. It was recommended that effort be devoted to arriving at standard build mechanisms and conventions.

Also with regard to standardization, it was recommended that effort be devoted to arrive at standards for parameter naming and metadata conventions to enhance interoperability for both physics packages and for data interoperability. There was agreement that adoption of the CF (Climate Forecast) convention is a desired end-state and that current participation in GO-ESSP be continued so as to influence CF as it evolves to address requirements for WRF and other models of this type. A specific area identified was metadata conventions for accurately representing georegistration data at very high resolutions. Another identified area, possibly outside of CF, is model metadata conventions for representing more generic model configuration information such as start time, stop time, time step, coupling frequency, etc.

Finally, regarding the issue of array storage-order and loop nesting order – frequently referred to as IJK versus IKJ – it was agreed that this is not a framework issue but rather a coding convention in the models themselves. The WRF framework supports both IJK and IKJ storage orders (and the four others). However, neither the WRF framework nor ESMF, nor for that matter any currently existing programming languages, have facility for addressing interoperability between existing codes that use different storage/loop-nesting orders without incurring a run-time penalty for data movement when going from one ordering to another. Several ideas were discussed that might make different storage/loop-nesting orders less of an obstacle to physics interoperability between models:

- In cases where real-time performance is not critical, such as during testing when packages are imported from one modeling system to another, there are straightforward Fortran-based "wrapping" techniques for handling the transposition between differing storage orders.

- In the special case where the direction of an index varies (e.g. k runs up the atmosphere in one model and down in another), techniques for writing generalized physics that can work in either direction should be explored. There may also be a role for source-translation tools that can automate this; a number of computer science research groups have developed approaches and software for this type of application-domain specific source translation. Effort could be devoted to surveying this work, assessing approaches, and developing solutions that would be useful not only for the WRF project but for the larger scientific simulation community.

- It was noted during the workshop that the choice of IKJ as the storage/loop-nesting order for the WRF model (as distinct from framework) was based on a study conducted relatively early in the development of WRF. The IKJ ordering was found to be the best compromise between performance on vector and microprocessor based systems available at that time. Given that several generations of hardware have passed since then, it was suggested the issue be revisited. This would involve identifying a number of representative modules within the current WRF model and testing these using several different storage orders on the current set of computing platforms used by WRF users. It is not clear whether the more than hundred-thousand lines of scientific code currently in WRF model should be rewritten if a new study gave a different result; however, at a minimum, the results of such a study would inform future development and provide guidance for the effort, mentioned above, to explore source-translation techniques for handling conversion between different storage/loop-nesting orders.

4. Interdisciplinary Applications using WRF and ESMF

A future is envisioned in which WRF and ESMF each play critical, and very different, roles in the broad modeling community. One example of how this can work in practice is in the realm of air quality modeling, which increasingly relies on multi-component, integrated systems. For many years, the EPA and NOAA partnership has sought to develop a common modeling infrastructure based on Multimedia Integrated Modeling System (MIMS) to link atmospheric models with air quality, water, soil, and ecosystems models. However, the present MIMS framework works at a higher conceptual level rather than actually providing necessary computational tools to facilitate the linkage. ESMF's general high-level coupling methods can provide the actual functionality needed.

Integration of the WRF science model with other types of cross-media environmental models will frequently require tools that allow exchange of information among models with heterogeneous system designs with different spatial and temporal interpolation needs. For example, ESMF can be utilized to couple state variables of different domains of media with distinctly different data-structure requirements, such as air, water, soil, and ecosystems through its specially designed interpolator. Thus ESMF can be utilized as the overarching model driver with couplers developed utilizing ESMF low-level utilities that can significantly extend the range of problems for which WRF can be applied.

Workshop Participants:

Daewon Byun	U. of Houston	Daewon.Byun@mail.uh.edu
Tom Black	NCEP/EMC	Tom.Black@noaa.gov
Sue Chen	NRL/MRY	Sue.Chen@nrlmry.navy.mil
Nancy Collins	NCAR	Nancy@ucar.edu
Cecelia Deluca	NCAR	Cdeluca@ucar.edu
Tom Henderson	NCAR/MMM	Hender@ucar.edu
Mark Iredell	NCEP/EMC	Mark.Iredell@noaa.gov
Steve Lowder	NRL/MRY	Steve.Lowder@nrlmry.navy.mil
John Michalakes	NCAR/MMM	Michalak@ucar.edu
Jacques Middlecoff	ESRL	Jacques.Middlecoff@noaa.gov
Dan Sedlacek	AFWA	Dan.Sedlacek@afwa.af.mil
Alan.Wallcraft	NRL/SSC	Alan.Wallcraft@nrlssc.navy.mil
Todd Hutchinson	WSI	thutchinson@wsi.com

Workshop Organizers:

Mike Clancy	N7C/FNMOC	Mike.Clancy@fnmoc.navy.mil
Robert Gall	NCAR/RAL/DTC	Gall@ucar.edu
Nelson Seaman	NOAA/NWS/OST	Nelson.Seaman@noaa.gov