THE WEATHER RESEARCH AND FORECAST MODEL: SOFTWARE ARCHITECTURE AND PERFORMANCE

J. MICHALAKES, J. DUDHIA, D. GILL, T. HENDERSON,
J. KLEMP, W. SKAMAROCK, W. WANG

Mesoscale and Microscale Meteorology Division, National Center for Atmospheric Research, Boulder, Colorado 80307 U.S.A.

The first non-beta release of the Weather Research and Forecast (WRF) modeling system in May, 2004 represented a key milestone in the effort to design and implement a fully-functioning, next-generation modeling system for the atmospheric research and operational NWP user communities. With efficiency, portability, maintainability, and extensibility as bedrock requirements, the WRF software framework has allowed incremental and reasonably rapid development while maintaining overall consistency and adherence to the architecture and its interfaces. The WRF 2.0 release supports the full-range of functionality envisioned for the model including efficient scalable performance on a range of high-performance computing platforms, multiple dynamic cores and physics options, low-overhead two-way interactive nesting, moving nests, model coupling, and interoperability with other common model infrastructure efforts such as ESMF.

1. Introduction

The WRF project has developed a next-generation mesoscale forecast model and assimilation system to advance both the understanding and the prediction of mesoscale precipitation systems and to promote closer ties between the research and operational forecasting communities. With the release of WRF version 2.0 to the community in May of 2004, the wide dissemination of the WRF modeling system to a large number of users and its application in a variety of areas including storm-scale research and prediction, air-quality modeling, wildfire simulation, hurricane and tropical storm prediction, regional climate, and operational numerical weather prediction are well underway. The number of registered downloads exceeded 2,500 at the end of 2004. 173 participants from 93 institutions in 20 countries attended the annual WRF Users Workshop in June 2004 at NCAR and heard 28 scientific presentations involving work being

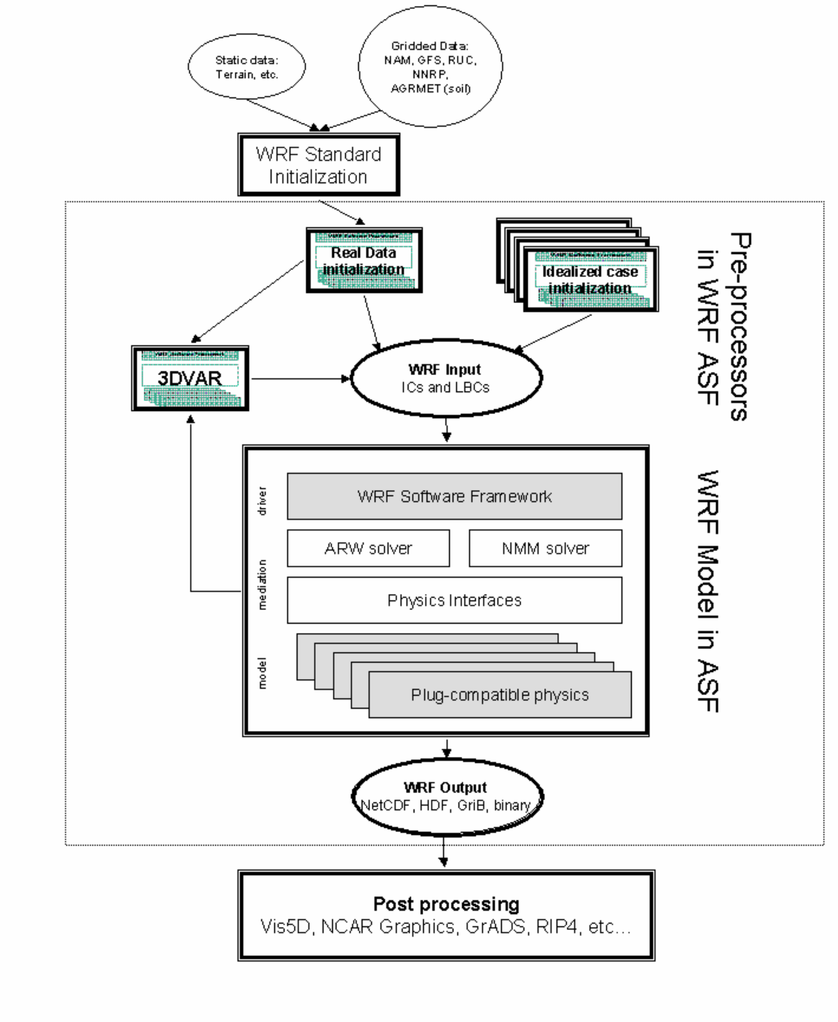

Figure 1 Schematic of WRF System

conducted with the WRF model. Operational implementation of WRF is underway at the NOAA National Centers for Environmental Prediction and at the U.S. Air Force Weather Agency. A joint NOAA/NCAR/DoD Developmental Testbed Center has been formed to facilitate the ongoing testing, evaluation, and transition of new developments from the research community into operations at NCEP, AFWA, and at the U.S. Navy through Operational Testbed Centers being established at the respective centers.

The WRF system, illustrated in Figure 1, consists of the WRF model itself, preprocessors for producing initial and lateral boundary conditions for idealized, real-data, and one-way nested forecasts, postprocessors for analysis and visualization, and a three-dimensional variational data assimilation (3DVAR) program. With the exception of the standard initialization (SI) program, each of the preprocessors and 3DVAR are parallel programs implemented using the WRF Advanced Software Framework (ASF). Data streams between the programs are input and output through the ASF's I/O and Model Coupling API. The WRF Model (large box in figure) contains two dynamical cores, providing additional flexibility across institutions and applications. The NCAR-developed Advanced Research WRF (ARW; originally the Eulerian Mass, or "EM" core) uses a time-split high-order Runga-Kutta method to integrate a conservative formulation of the compressible non-hydrostatic equations [16]. ARW is supported to the research community as WRF Version 2 and is undergoing operational implementation at the U.S. Air Force Weather Agency. NOAA/NCEP's operational implementation of WRF is using dynamics adapted to the WRF ASF from the Non-hydrostatic Mesoscale Model (NMM) [3][8][9][15].

The WRF ASF implements the WRF software architecture [11] and is the basis on which the WRF model and 3DVAR systems have been developed. It features a modular, hierarchical organization of the software that insulates scientific code from parallelism and other architecture-, implementation-, and installation-specific concerns. This design has also been crucial for managing the complexity of a single-source-code model for a range of users, applications, and platforms.

This paper describes the implementation and performance of WRF software, including new features provided in WRF 2.0: two-way interacting and moving nests, support for model coupling, and interoperability with emerging community modeling infrastructure such as the Earth System Modeling Framework.[1]

2. WRF Advanced Software Framework

The WRF ASF comprises a number of separable layers and supporting components: the Driver Layer, Mediation Layer, Model Layer, a meta-programming utility called the Registry, and application program interfaces (APIs) to external packages for interprocessor communication, data formats, and I/O. The benefits of the WRF ASF are facilitation of rapid development, ease of extension, leverage of development effort by the WRF community at large,

---

[1] www.esmf.ucar.edu

software reuse, and ready adaptation to community model infrastructure such as ESMF.

The Driver layer handles run-time allocation and parallel decomposition of model domain data structures; organization, management, interaction, and control over nested domains, including the main time loop in the model; high-level interfaces to I/O operations on model domains; and the interface to other components when WRF is part of a larger coupled system of applications. Within the driver, each domain is represented abstractly as a single object: a Fortran90 derived data type containing the dynamically allocated state data with pointers to other domains in the nest hierarchy. Nesting is represented as a tree of domains rooted at the top-level (most coarse resolution) domain. Each model time step involves a recursive depth-first traversal over this tree, advancing each node and its children forward to the next model time. Forcing, feedback, and nest movement is also handled in the Driver.

The Mediation Layer encompasses one time-step of a particular dynamical core on a single model domain. The *solve* routine for the dynamical core contains the complete set of calls to Model Layer routines as well as invocation of interprocessor communication (halo updates, parallel transposes, etc.) and multi-threading. The current WRF implementation uses the RSL communication library [12] that, in turn, uses the Message Passing Interface (MPI) communication package. Shared-memory parallelism over tiles – a second level of domain decomposition within distributed memory patches – is also specified in the solve routines using OpenMP.

The Model Layer comprises the actual computational routines that make up the model: advection, diffusion, physical parameterizations, and so forth. Model layer subroutines are called through a standard Model Layer Interface: all state data is passed as arguments, along with the starting and ending indices in each of the three grid dimensions for the tile that is being computed. Model layer subroutines may not include I/O, stop statements, multi-threading, or interprocessor communication, ensuring that they may be executed coherently for any tile-decomposition or order of execution over tiles. The Model Layer Interface is a contract between the ASF and the programmer/scientist working at the Model Layer. Adherence to the interface ensures that a Model Layer package incorporated into WRF will work on any parallel computer the framework itself is ported to. Model layer routines that have data dependencies rely on the mediation layer to perform the necessary interprocessor communication prior to their being called. The programmer describes the communication type and pattern by adding an entry to the Registry and then inserts a notation to perform the communication at the appropriate location in the solve routine.

The Registry is a concise database of information about WRF data structures and a mechanism for automatically generating large sections of WRF code from the notations in the database. The Registry data base is a collection of tables that

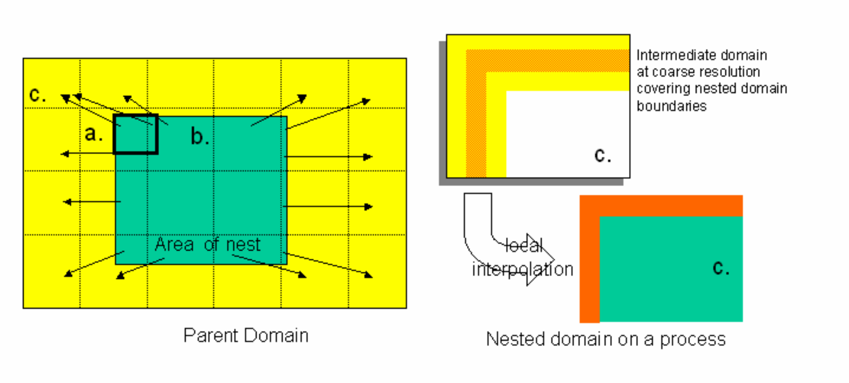

Figure 2 Nesting decomposition and communication.

lists and describes the WRF state variables and arrays with their attributes such as dimensionality, number of time levels, association with a particular dynamical core, association with a particular physics package, membership in an input, output, or restart dataset, communication operations on the data, and some descriptive meta-data such as what the variable or array represents and its units. From this database, the Registry generates code for interfaces between layers of the infrastructure, packing and unpacking code for communication and nesting, and field-by-field calls to routines for model I/O -- code that would otherwise be extremely time-consuming and error-prone to write and manage manually. Adding or modifying a state variable or array in WRF is a matter of modifying a line or two in the Registry. Currently, the Registry automatically generates 60-thousand of the total 250-thousand lines of WRF code.

APIs to external packages are also part of the WRF software framework. These allow WRF to use different packages for self-describing data formats, model coupling toolkits and libraries, and libraries for interprocessor-communication by simply adapting the external package to the interface. Clean APIs also support reuse in the other direction; for example, Earth System Modeling Framework developers are adapting the WRF I/O API for use within the ESMF software.

Documentation for the WRF infrastructure, including reference documentation for the Registry, the WRF I/O Application Program Interface specification, and a web-based WRF code and documentation browser [6] are maintained on-line.[1] Additional software documentation for WRF is in-progress.

---

[1] http://www.mmm.ucar.edu/wrf/WG2/software_2.0

3. Nesting and Moving Nests

Nesting is a form of mesh refinement that allows costly higher resolution computation to be focused over a region of interest. WRF 2.0 includes support for one-way and two-way interacting nested domains. Nests in WRF are non-rotated and aligned so that parent mesh points are coincident with a point on the underlying nest, which eliminates the need for more complicated generalized regridding calculations. Nest configurations are specified at run-time through the namelist. The WRF ASF supports creating and removing nests at any time during the simulation but the WRF model is currently constrained to starting nests at the beginning if runs require input of nest-resolution terrain or other lower boundary data; this limitation will be addressed in the near future. Nests may be telescoped (nests within nests) to an arbitrary level of horizontal refinement. Vertical refinement is not yet implemented. Refinement ratios are whole integers, typically 1:3. A prototype implementation of moving nests was released in version 2.0.3. This version was used for a 4km moving nest simulation of Hurricane Ivan (Sept. 2004). An animation is viewable on-line.[1]

Efficient and scalable implementation of nesting is a key concern. All domains in a nested simulation are decomposed over the same set of processes and nested domains run synchronously with the parent. Exchanging forcing and feedback information requires communication to scatter and gather data across processes every parent time step. In addition, interpolation of parent domain data to nest points is load imbalanced because it only occurs over regions of the domain shared by both parent and nest. This is partially alleviated by first rearranging the parent domain data to the processes storing the corresponding nest domain points, allowing interpolation to be performed locally and over a larger number of processes. Figure 2 shows parent domain data on the processes overlaying the nested boundary (including “a” and “b” for the northwest nest corner) being communicated to the processes that compute the nest boundary (including process “c”. Recall both domains are decomposed over the same set of processes). After rearrangement of the parent-domain data, the parent-to-nest grid interpolation is performed locally on the nest-boundary processes.

Nesting overhead has been measured by running equally dimensioned parent and nest domains as two-way interacting domains and then separately as stand-alone single domain runs. Overhead for nesting is between 5 and 8 percent, depending on the number of processes, and well within the target of 15 percent overhead observed in the parallel MM5 model. Most of the overhead appears related to the cost of the interpolation, which uses a relatively expensive non-linear algorithm.

The approach for moving nests is the same as two-way nesting, with some additional logic added to the framework and the model code:

---

[1] http://www.mmm.ucar.edu/wrf/WG2/wrf_moving_nest.gif

1. Determine whether it is time for a move and, if so, the direction and distance of the move.
2. Adjust the relationship between points on the nest and corresponding points on the parent domain.
3. Shift data in the 2- and 3-dimensional state arrays of the nest in the opposite direction of the nest movement.
4. Initialize the leading edge of the nest as it moves into a new position relative to the parent domain.

Additional work is needed in step 1, above, to incorporate an automatic feature-following nest movement mechanism and in step 4 to allow run-time ingest of nest-resolution lower boundary data such as topography and land use on the leading edge of the moved nest. Lastly, the issue of moving coupling to an external model such as an ocean model will be addressed.

4. I/O and Model Coupling

The I/O and Model Coupling API within the WRF ASF provides a uniform, package-independent interface between the WRF model and external packages for I/O and data formatting. Implementations of the API for NetCDF, Parallel HDF5, native-binary, and GRIB1 I/O are run-time assignable to the framework's I/O streams.[1] The WRF I/O and Model Coupling API also supports model coupling, an idea well developed under [5][6] and in the PRISM coupling framework [7]. "Coupling as I/O" is attractive in that it allows encapsulation of details of component data exchange within a model's control structures and interfaces that already exist for I/O. It requires little if any modification to the models themselves, it is readily and efficiently adaptable to different forms of coupling (sequential or concurrent), it can switch transparently (from the applications point of view) between on-line and off-line modes of coupling, and it is naturally suited to distributed computing environments such as Grid computing.

Two model coupling implementations of the WRF I/O and Model Coupling API have been developed: the Model Coupling Toolkit (MCT) [10] is the basis for the Community Climate System Model (CCSM) coupler; the Model Coupling Environment Library (MCEL) [2] is a CORBA-based client-server based coupling framework.

The MCT implementation of the WRF I/O and Model Coupling API supports regular, scheduled exchanges of boundary conditions for tightly to moderately coupled interactions between WRF and the Regional Ocean Modeling System

---

[1] Muqun Yang at NCSA contributed the HDF5 implementation of the WRF I/O API. Todd Hutchinson of WSI Inc contributed the GRIB1 implementation.

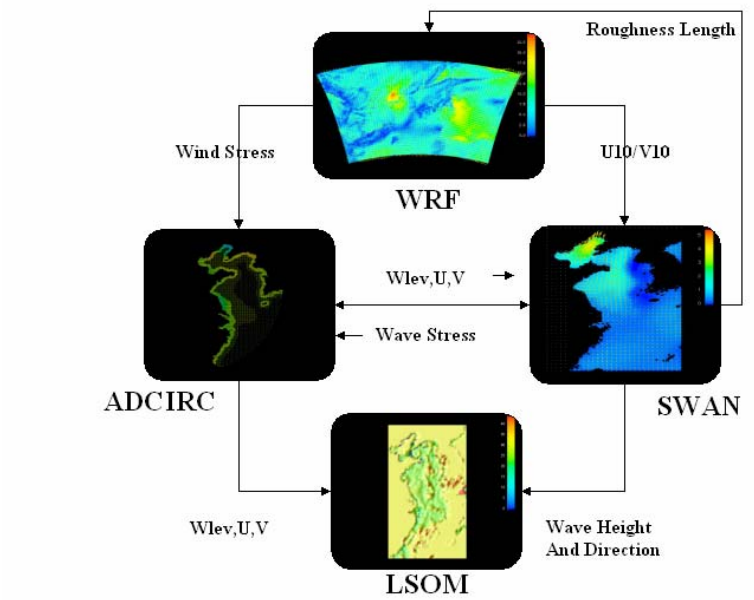

Figure 3 Coupling graph for Yellow Sea simulation showing WRF (atmosphere), ADCIRC (ocean circulation), SWAN (wave model), and LSOM (sediment optics). U and V velocities from WRF are air; U and from ADCIRC are water.

(ROMS). WRF wind stress and heat fluxes are sent to the ocean model and sea-surface temperature is received from ROMS. Three performance benchmarks of the WRF/ROMS coupling were conducted and coupling overhead was nominal, well under 1 percent of total run time for the coupled system. The coupled WRF/ROMS system has been used in follow-on scientific studies involving an idealized hurricane vortex described in [14]. The WRF/ROMS MCT implementation has been demonstrated using the MPICH-G2 library within the Globus toolkit over a rudimentary computational grid.[1] ROMS ran on one Intel Linux node at NOAA Pacific Marine Environmental Laboratory (PMEL) in Seattle; WRF on four Linux nodes at NOAA Forecast Systems Laboratory (FSL) in Boulder. Even over geographically distributed systems, overhead for the WRF/ROMS coupling using MCT over Globus was less than 2 percent.

The MCEL implementation of the WRF I/O and Model Coupling API supports coupling of sets of models with a wider range of spatial and temporal scales, or with irregular, data-driven interactions. Figure 3 shows output from a four-model simulation of the Yellow Sea littoral environment for a high-wind event in November, 1999. WRF is coupled to a system composed of the ADCIRC ocean and SWAN wave models. ADCIRC and SWAN, in turn, provide forcing to a sedimentation and optics model that simulates diver visibility [1]. ADCIRC,

---

[1] Work by Daniel Schaffer (NOAA/FSL) and Chris Moore (NOAA/PMEL).

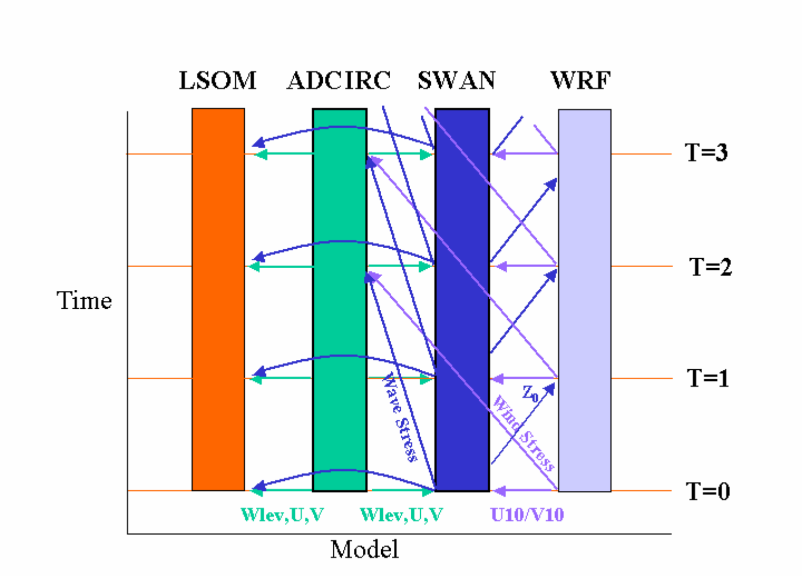

Figure 4 Schedule of data interaction between four concurrently executing models. The coupling interval, T, is one hour.

which uses an unstructured mesh, was also interfaced to MCEL through the WRF I/O API in order to demonstrate its applicability to other models and grid systems. MCEL supports concurrent coupling, meaning that the components run at the same time on different sets of processors. Figure 4 shows the timing of interactions between the components. As with MCT, the measured coupling overheads were small. WRF, the atmosphere, is the dominant cost of the simulation. Therefore, we measured the coupling overhead from the point of view of the WRF atmosphere. The cost of coupling measured from WRF is only 15-20 milliseconds per exchange in each direction. With WRF running on 4 processors, the coupling overhead is negligible – less than half a percent of total run time. On 32 processors, the overhead from coupling is under 5 percent of the cost of the run.

The Earth System Modeling Framework, an emerging community-based standard and software infrastructure employs a different approach to model coupling. As opposed to handling coupling as a form of I/O, ESMF components are restructured to conform to a top-level “ESMF-compliant” component interface. This allows an ESMF-based driver to control model initialization, integration, and finalization. Coupling data is exchanged between components by passing import and export state objects through the component’s top-level interface. Multi-executable execution and compatibility with distributed and Grid-computing environments is not currently supported.

WRF is being adapted to support top-level interface requirements to interoperate as an ESMF coupled component but will also continue to interoperate through I/O-like coupling mechanisms through the WRF I/O and Model Coupling API. Implementing a form of ESMF coupling that presents itself through the WRF I/O and Model Coupling API is also being explored.

5. Performance

Key goals for the WRF software are portability and efficiency over shared, distributed-memory, and hybrid parallel architectures and over both vector and scalar processor types. The WRF ASF supports a two-level decomposition strategy that first decomposes each model domain over distributed-memory *patches* and then, within each patch, over shared-memory *tiles.* The framework and the Model Layer Interface allow the Driver layer to decompose domains over arbitrarily shaped and sized rectangular patches and tiles, giving maximum flexibility for structuring the computation as efficiently as possible. Towards the goal of WRF performance-portability, routine benchmarking on a variety of target computer platforms has been ongoing.

Figure 5 represents a snapshot of WRF performance at the end of 2004; the latest results are maintained and routinely updated on the web.[1] The test case in the figure is a 48-hour, 12km resolution case over the Continental U.S. (CONUS) domain. The computational cost for this domain is about 22 billion floating point operations per average time step (72 seconds). Performance is defined as model speed, ignoring I/O and initialization cost, directly measured as the average cost per time step over a representative period of model integration, and is presented both as a normalized floating-point rate and as simulation speed. These are equivalent measures of speed, but floating-point rate expresses speed as a measure of efficiency relative to the theoretical peak capability while simulation speed, the ratio of model time simulated to actual time, is more relevant as a measure of actual time-to-solution.

The purpose of this WRF benchmark is to demonstrate computational performance and scaling of the WRF model on target architectures. The benchmarks are intended to provide a means for comparing the performance of different architectures and for comparing WRF computational performance and scaling with other similar models. In light of the continuing evolution and increasing diversity of high-performance computing hardware it is important to define what is being counted as a process. For this benchmark, a parallel process is the finest-grained sequence of instructions and associated state that produces a

---

[1] http://www.mmm.ucar.edu/wrf/bench

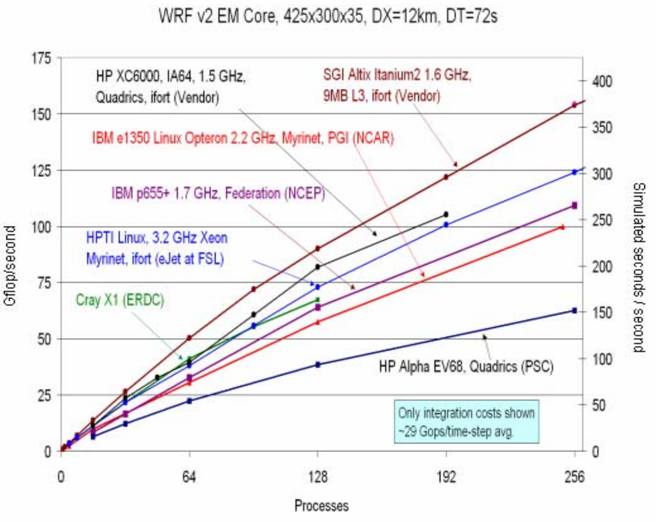

Figure 5 WRF performance results at end of 2004.

separable and disjoint part of the solution. Typically, the number of processes is the number of WRF tiles are executing in parallel during a given run.

6. Conclusion

Since its release in 2004, WRF Version 2 has seen increasing adoption within the research and operational communities. In addition to providing a common tool for mesoscale simulation and data assimilation, the WRF project is also beginning to serve its broader goal of fostering communication, cooperation, and collaboration, especially within WRF working groups specializing in regional climate, air quality simulation, and numerical weather prediction research. WRF software is supported to the user community through an ongoing effort involving workshops, tutorials, helpdesk services, and on-line documentation. Operational implementation at NCEP and the Air Force Weather Agency is underway, with the joint NOAA/NCAR/DoD Developmental Testbed Center facilitating the WRF project goal of transferring research into operations. Central to the multi-agency WRF development effort has been the WRF ASF: a flexible, maintainable, extensible framework allowing rapid development, support, and maintenance of the WRF software to a diverse set of users, institutions, and applications. Portability and efficiency over a full range of high-performance computing systems has been a key objective. Finally, the modular design of the WRF ASF and its interfaces facilitates model coupling and integration of WRF into community model infrastructure efforts such as PRISM and ESMF.

ACKNOWLEDGEMENTS

M. Yang (NCSA), T. Black, N. Surgi. and S. G. Gopalakrishnan (NOAA/-NCEP), D. Schaffer, J. Middlecoff, G. Grell (NOAA/FSL), M. Bettencourt (AFRL), Shuyi Chen and David Nolan (U. Miami), Alan Wallcraft (NRL), Chris Moore (NOAA/PMEL). JIN Zhiyan (CMA), D. Barker, A. Bourgeois, C. Deluca, R. Loft (NCAR), J. Wegiel (AFWA), T. Hutchinson (WSI), R. Jacob and J. Larson (ANL).

REFERENCES

1. Allard, R., C. Barron, C.A. Blain, P. Hogan, T. Keen, L. Smedstad, A. Wallcraft, C. Berger, S. Howington, J. Smith, R. Signell, M. Bettencourt, and M. Cobb, “High Fidelity Simulation of Littoral Environments,” in proceedings of UGC 2002, June 2002.
2. Bettencourt, M. T. Distributed Model Coupling Framework, in proceedings of HPDC-11, July 2002. (http://www.extreme.indiana.edu/~gannon/-hpdc/hpdc11.html)
3. Black, T., E. Rogers, Z. Janjic, H. Chuang, and G. DiMego. Forecast guidance from NCEP's high resolution nonhydrostatic mesoscale model. Preprints 15th Conference on Numerical Weather Prediction, San Antonio, TX, Amer. Meteor. Soc., J23-J24. 2002.
4. Coats, C.J., J. N. McHenry, A. Lario-Gibbs, and C. D. Peters-Lidard, 1998: MCPL(): A drop-in MM5-V2 module suitable for coupling MM5 to parallel environmental models; with lessons learned for the design of the weather research and forecasting (WRF) model. The Eighth PSU/NCAR Mesoscale Model Users' Workshop. MMM Division, NCAR, Boulder, Co., 117-120.
5. Coats, C. J., A. Trayanov, J. N. McHenry, A. Xiu, A. Gibbs-Lario, and C. D. Peters-Lidard, 1999: An Extension of The EDSS/Models-3 I/O API for Coupling Concurrent Environmental Models, with Applications to Air Quality and Hydrology, Preprints, 15th lIPS Conference, Amer. Meteor. Soc. Dallas, Tx., January 10-15, 1999.
6. Fiedler, B., J. Rafal, and C. Hudgin. F90tohtml tool and documentation. http://mensch.org/f90tohtml.
7. Guilyardi, E., R. Budich, G. Brasseur, and G. Komen, 2002: PRISM System Specification Handbook V.1.. PRISM Report Series No 1 239 pp.
8. Janjic, Z. I., J. P. Gerrity, Jr. and S. Nickovic. An Alternative Approach to Nonhydrostatic Modeling. Monthly Weather Review, 129, 1164-1178. 2001.

9. Janjic, Z. I. A Nonhydrostatic Model Based on a New Approach. Meteorology and Atmospheric Physics (in print). 2002.
10. Larson, J.W., R. L. Jacob, I. Foster, and J. Guo, “The Model Coupling Toolkit, ” 2001, Proc. 2001 Int'l Conf. on Computational Science.
11. Michalakes, J., J. Dudhia, D. Gill, J. Klemp, and W. Skamarock: "Design of a next-generation regional weather research and forecast model," *Towards Teracomputing*, World Scientific, River Edge, New Jersey (1999), pp. 117-124.
12. Michalakes, J.: RSL: A parallel runtime system library for regional atmospheric models with nesting, in *Structured Adaptive Mesh Refinement (SAMR) Grid Methods*, IMA Volumes in Mathematics and Its Applications (117), Springer, New York, 2000, pp. 59-74.
13. Michalakes, J., M. Bettencourt, D. Schaffer, J. Klemp, R. Jacob, and J. Wegiel, Infrastructure Development for Regional Coupled Modeling Environments Parts I and II, Final Project Reports to Program Environment and Training Program in the U.S. Dept. of High Performance Computing Modernization Office, Contract No. N62306-01-D-7110/CLIN4, 2003 and 2004. See also: http://www.mmm.ucar.edu/wrf/WG2/software_2.0/io.pdf
14. Nolan, D.S. and M.T. Montgomery, 2002: “Nonhydrostatic, three-dimensional perturbations to balanced, hurricane-like vortices. Part I: linearized formulation, stability, and evolution”, J. Atmos. Sci., 59, 2989-3020.
15. Pyle, M.E., Z. Janjic, T. Black, and B. Ferrier: An overview of real time WRF testing at NCEP, in proceedings of the MM5/WRF Users Workshop, NCAR, June 22-25, 2004.
16. Wicker, L. J., and W. C. Skamarock, 2002: Time splitting methods for elastic models using forward time schemes. Mon. Wea. Rev., 130, 2088-2097.