Real-time Storm Surge Ensemble Modeling in a Grid Environment

Lavanya Ramakrishnan[1], Brian O. Blanton[4], Howard M. Lander[1], Richard A. Luettich, Jr.[3],
Daniel A. Reed[1], Steven R. Thorpe[2]
[1] Renaissance Computing Institute, [2]MCNC, [3]UNC Chapel Hill Institute of Marine Sciences, [4]Science Applications International Corporation
{lavanya, howard, dan_reed}@renci.org, thorpe@mcnc.org,
rick_luettich@unc.edu, brian.o.blanton@saic.com

Abstract

Natural disasters such as hurricanes heavily impact the US East and Gulf coasts. This creates the need for large scale modeling in the areas of meteorology and ocean sciences, coupled with an integrated environment for analysis and information dissemination. In turn, this means there is an increased need for large-scale distributed high performance resources and data environments. In this paper, we describe a framework that allows a storm surge model-ADCIRC to be run in a distributed Grid environment. This framework was developed as a component of the Southeastern Universities Research Association's (SURA) Southeastern Coastal Ocean Observing and Prediction (SCOOP) program. SCOOP is creating an open-access grid environment for the southeastern coastal zone to help integrate regional coastal observing and modeling systems. Specifically this paper describes a set of techniques used for resource selection and fault tolerance in a highly variable ad-hoc Grid environment. The framework integrates domain-specific tools and standard Grid and portal tools to provide an integrated environment for forecasting and information dissemination.

1. Introduction

Year after year, the US East and Gulf coasts are heavily impacted by hurricane activity causing large number of deaths and billions of dollars in economic losses. For example in 2005, there were 14 hurricanes, exceeding the record of 12 in 1969, out of which 7 were considered major hurricanes [9]. To help reduce the impact of hurricanes, there is a need for an integrated response system that enables virtual communities [1] to evaluate, plan and react to such natural phenomena. The integrated system needs to handle real-time data feeds, schedule and execute a set of model runs, manage the model input and output data, make results and status available to the larger audience. In addition, to enhance the scientific validity of the models there is a need to be able to recreate scenarios and re-run the models for retrospective analysis[19]. The large-scale modeling and analysis has driven the use of high performance resources and Grid environments for such problems.

In this paper, we describe the distributed software infrastructure used to run a storm surge model in a Grid environment. The sensitivity to timely model completion drives the need for specific techniques for resource management and increased fault tolerance when the models run in a distributed Grid environment. This framework was developed as a component of the Southeastern Universities Research Association's (SURA) Southeastern Coastal Ocean Observing and Prediction (SCOOP) program[20]. The SCOOP program is a distributed project that includes Gulf of Maine Ocean Observing System, Bedford Institute of Oceanography, Louisiana State University, Texas A&M, University of Miami, University of Alabama in Huntsville, University of North Carolina, University of Florida and Virginia Institute of Marine Science. SCOOP is creating an open-access grid environment for the southeastern coastal zone to help integrate regional coastal observing and modeling systems. Specifically, our effort in this program is focused on two main areas: 1) storm surge modeling for the south east coast; and 2) experimenting with novel techniques to use grid resources to meet real-time constraints of the application. The storm surge component uses the Advanced Circulation (ADCIRC)[12] model that computes tidal and storm surge water and currents, forced by tides and winds. While our framework was developed in the context of ADCIRC, the solution is more general and is applicable for running other models and applications in grid environments. In fact the framework is currently being applied to other models in the context of the North Carolina Forecasting System[22].

Our solution builds on existing standard grid and portal technologies including the Globus toolkit [2], Open Grid Computing Environment (OGCE)[4] and lessons learned from grid computing efforts in other science domains, such as bioinformatics[21], astronomy[5] and other projects. A portal provides the front-end interface for users to interact with the ocean observing and modeling system. The users can conduct retrospective analysis, access historical data from previous model runs and observe the status of daily forecast runs from the portal. The real-time data for the ensemble forecast arrives through Unidata's Local Data Manager (LDM)[15], an event-driven data distribution system that selects, captures, manages and distributes meteorological data products. Once all the data for a given ensemble member has been received, available and suitable grid resources are discovered using a simple resource selection algorithm. The model run is then executed and the output data is staged back to the originating site. The final ensemble result of the surge computations is inserted back into the SCOOP LDM stream for subsequent analysis and visualization by other SCOOP partners [18].

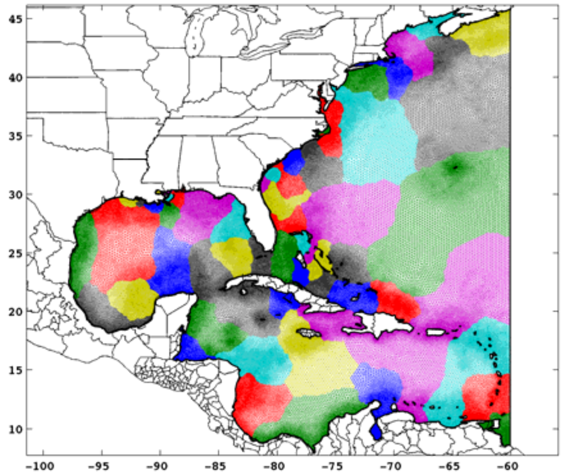


Figure 1. Domain decomposition of a high-resolution ADCIRC grid used in the SCOOP computational system.

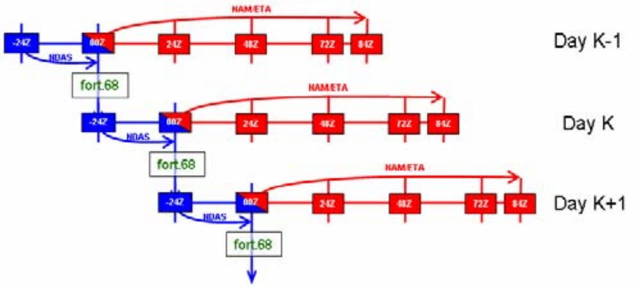


Figure 2. Timeline showing the computation of a hotstart file and a subsequent forecast. On Day K, the hotstart computed "yesterday" (Day K-1) is used to bring the hotstart sequence up to date, and an 84-hour forecast is subsequently computed. This same hotstart file is used "tomorrow" (Day K+1) to start the sequence over again.

In this paper, we describe the interaction of the Grid components and specific techniques used for resource selection and fault tolerance during model execution. The rest of the paper is organized as follows. In §2 the science drivers are described in greater detail. We describes our design philosophy in greater detail in §3. The architecture and technology components are presented in §4 and §5, experiences from our system and related work in §6 and §7, and we present our conclusions and future work in §8.

2. Science Drivers

Before we detail our design and techniques, we present a brief description of the science elements that are the motivation for our decisions. As mentioned earlier for the storm-surge forecasts, we use the tidal and storm-surge model ADCIRC[12]. ADCIRC is a finite element model that solves the shallow-water generalized wave-continuity equations for a thin fluid layer on a rotating platform. The ADCIRC model is parallelized using Message Passing Interface (MPI). In the current implementation, we use a relatively coarse representation of the western North Atlantic Ocean. Figure 1. shows a 32-processing element decomposition of this ADCIRC grid.

Storm surge modeling requires assembling input meteorological and other data sets, running models, processing the output and distributing the resulting information. In terms of modes of operation, most meteorological and ocean models can be run in 'hindcast' mode, as an after fact of a major storm or hurricane, for post-analysis or risk assessment, or in 'forecast' mode for prediction to guide evacuation or operational decisions[19]. The forecast mode is driven by real-time data streams while the hindcast mode is initiated by a user. Our framework is designed to support both these usage models for running ADCIRC and other models in a Grid environment.

Further, often it is necessary to run the ADCIRC model with different forcing conditions to analyze forecast accuracy. This results in a large number of parallel model runs, creating an ensemble of forecasts. The meteorological modeling community has long recognized that a consensus forecast, based on an ensemble of forecasts, generally has better statistical forecast skill than any one of the ensemble members[14, 11]. Thus, we have taken an ensemble approach to storm-surge forecasting that requires access to a large number of computational clusters, coordinated access to data and computational resources, and the ability to leverage additional resources that may become available over time.

Our operational cycle is tied to the typical 6-hour synoptic forecast cycle used by the National Weather Service and the National Centers for Environmental Prediction (NCEP). NCEP computes an atmospheric analysis and forecast four times per day, for which the forecast initialization times are 00Z, 06Z, 12Z, and 18Z. As ADCIRC solves discrete versions of partial differential equations, both initial and boundary conditions are required for each simulation. Boundary conditions include the wind stress on the ocean surface (an ensemble member, described below) and tidal elevations. The initial conditions for each simulation are taken from a previously computed “hindcast” that is designed to keep the dynamic model up-to-date with respect to the analyzed atmospheric model state. This is called hot-starting the model. For each synoptic cycle, a hot-start file is computed that brings the model state forward in time from the beginning of the previous cycle to the start of the current forecast cycle (**Figure 2**).

The wind field boundary conditions for each simulation are taken from a variety of sources, each of which constitutes one member of the ensemble. In addition to the atmospheric model forecasts provided by NCEP, the SCOOP project also uses tropical storm forecast tracks from the National Hurricane Center to synthesize “analytic” wind fields. Each forecast track is statistically perturbed and an analytic vortex model[13] is used to compute the wind and pressure fields for each track. In the SCOOP project, this service is provided by the University of Florida and the wind files arrive through LDM. We are currently investigating the skill of this ensemble approach, and results will appear in a separate communication.

3. Design Philosophy

The need for timely access to high performance resources for the large suite of ensemble runs makes it important to have a distributed, fault tolerant Grid environment for these model runs. Based on earlier experience in storm surge modeling and the lessons learned from other inter-disciplinary Grid efforts, we identified a set of higher level design principles that helped guide the architecture and implementation of the system.

Scalable real-time system: As discussed earlier, using ensemble modeling the forecast accuracy can be increased. Running multiple high resolution, large-scale simulations necessitates the need for a scalable and distributed real-time system. Thus, our system is based on Grid technologies and standards allowing us to leverage access to ad-hoc resources that may become available.

Extensible: While this effort has been largely focused in the context of the SCOOP ADCIRC model, our goal is to build a modular architecture to be able to support other applications and add additional resources as they become available.

Adaptable: The criticality and the timeliness aspects of the science and the variability in grid environments require the infrastructure to be adaptable at various levels. The infrastructure needs to have active monitoring and adaptation components that can react to these changes and ensure successful completion of the models using fault tolerance and failure recovery techniques.

Specifically, based on these underlying design principles, we are focused on building a framework that can be used for real-time storm surge ensemble modeling on the Grid that is triggered by arrival of wind data. The required timeliness of the model runs makes it important to address the following issues on the Grid: a) real-time discovery of available resources b) managing the model run on an ad-hoc set of resources c) continuous monitoring and adaptation to allow the system to be resilient to the variability in Grid environments.

4. Data and Control Flow of the NC SCOOP System

The ADCIRC storm surge model can be run in two modes. The “forecast” mode is triggered by real-time data arrival of wind data from different sites through the Local Data Manager[15]. In the “hindcast” mode, the modeler can either use a portal or a shell interface to launch the jobs to investigate prior data sets (post-hurricane). **Figure 3** shows the architectural components and the control flow for the NC SCOOP system:

1. In the forecast run the wind data arrives at the local data manager (Step 1.F. in Figure 3). In our current setup, the system receives wind files from University of Florida and Texas A&M. Alternatively, a scientist might log into the portal and choose the date and the corresponding data to re-run a model (Step 1.H. in Figure 3).
2. In the hindcast run, the application coordinator locates relevant files using the SCOOP catalog at UAH[23] and retrieves them from the SCOOP archives located at TAMU and LSU[17]. In the forecast runs, once the wind data arrives, the application coordinator checks to see if the hotstart files are available locally or are available at the remote archive. If they are not available and not being generated currently (through a model run), a run is launched to generate the corresponding hotstart files to initialize the model for the current forecast cycle.

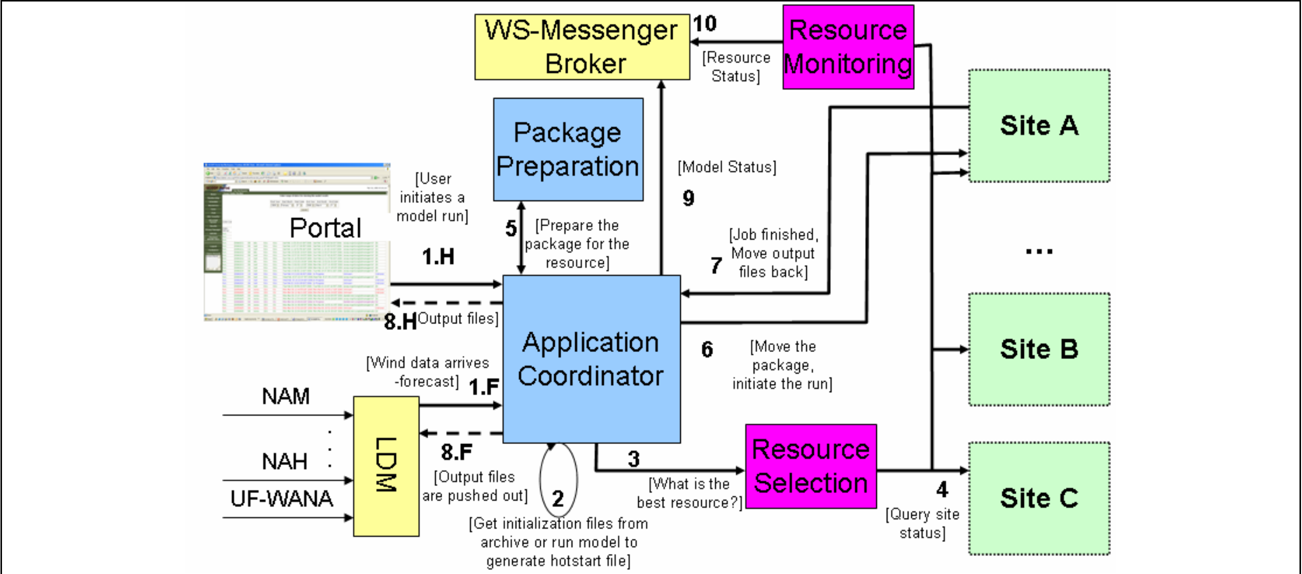


Figure 3. The control flow through the various components of the architecture

3. Once the model is ready to run (i.e. all the data is available), the application coordinator will use the resource selection component to select the best resource for this model run.
4. The resource selection component queries the status at each site and ranks the resources, accounting for queue delays and network connectivity between the resources.
5. The application coordinator then calls an application specific component that prepares an application package that can be shipped to remote resources. The application package is customized with specific properties for the application on a particular resource and includes the binary, the input files and other initialization files required for the model run.
6. The self-extracting application package is transferred to the remote resource and the job is launched using standard grid mechanisms.
7. Once the application coordinator receives the *"job finished"* status message, it retrieves the output files from the remote sites.
8. In case of the hindcast mode, the results are then available through the portal (Step 8.H in Figure 3). Additionally, in case of forecast mode, we push the data back through LDM (Step 8.F in Figure 3). Data is then archived and visualized by other SCOOP partners downstream.
9. The application coordinator publishes status messages at each of the above steps to a centralized messaging broker. Interested components such as the portal can subscribe to relevant messages to receive real-time status notification of the job run.
10. In addition the resource status information is also collected across all the sites and can be observed through the portal as well as used for more sophisticated resource selection algorithms.

5. Technology Components

We have described the flow through the control system and identified the key components of the architecture. In this section we will discuss in greater detail the design issues, technology choices and implementation of the architecture components.

As noted earlier, our architecture is based on existing open source grid middleware and web services tools such as Globus[2], Open Grid Computing Environment (OGCE)[4] and WS-Messenger[10]. We describe each of the components in detail below.

5.1. Data Management

The data transport system in SCOOP is based on Unidata's Local Data Manager (LDM). LDM allows us to select, capture, manage, and distribute arbitrary data products over a networked set of computers. LDM is designed for event-driven data distribution where a client may ingest data. In addition an LDM server can communicate with other LDM servers to either receive or send data. LDM is flexible and allows for site-specific configuration and processing actions on the data. The ADCIRC model receives its upstream wind and meteorological data through LDM and the model results are sent downstream to other SCOOP partners through LDM for archiving and visualization. LDM allows us to associate triggers with arriving data that can be used for launching automated model runs. In the long term we anticipate that there might be multiple ways that the data might arrive. In this case, the model runs may need to be triggered by a higher level component.

We also use GridFTP to manage data movement during model execution. In addition, we use the SCOOP catalog[23] to locate the data files that may have been generated previously. If available, the files are retrieved from the SCOOP archives[17]. The two types of files retrieved from the archive are the hotstart files to initialize the model run and the netCDF wind files. The wind files arrive through LDM for the forecast runs but may need to be retrieved from the archive for the hindcast runs. In addition, this gives us the ability to use the wind files from the archive to reduce data movement costs during forecast model execution.

5.2. Grid Middleware

In the last few years, there has been increased deployment of Grid technologies on commodity clusters. These clusters are used to run scientific applications and are shared across different organizations forming large interdisciplinary virtual communities. For our system we assume a minimal software stack composed of existing grid technologies and protocols to manage jobs and files, namely, Grid Resource Allocation and Management (GRAM)[2] and GridFTP[6] based in the Globus toolkit. Additionally, the Globus Monitoring Discovery System (MDS)[28] and Network Weather Service (NWS)[27] configured at a site is used to make a more informed resource selection. During the resource selection process, each of sites is queried for the queue status and the bandwidth to each site. The resource selection process is described in greater detail in §5.5. Once a resource is selected, a credential to be used at this site is obtained from a MyProxy[3] server. MyProxy server is a credential management service that stores Globus X.509 certificates. MyProxy allows users to store their certificates and private keys in the repository making it accessible from different distributed resources. MyProxy issues a short lifetime certificate to the system that can then be used to authenticate to the remote system.

5.3. Application Coordinator

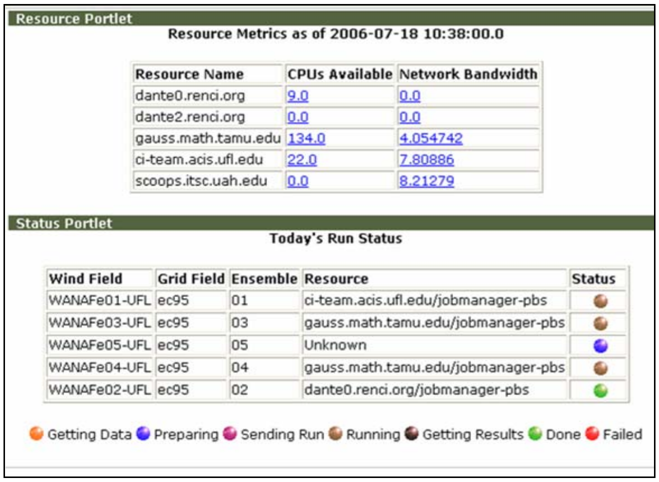
The Application Coordinator acts as a central component for each of the model runs whether initiated by the user through the portal or triggered by the arrival of data through LDM. It uses the resource selection component to select a grid site. After the user proxy is obtained, the Application Coordinator is able to perform Grid operations on behalf of the user (in case of the hindcast) or a pre-configured user (for the forecast). The application manager invokes a specified script to generate a self-extracting package of the application for the particular remote site. This self extracting package is transferred to the remote site using GridFTP. Once the file is transferred, the job is submitted to the Globus gatekeeper using the GRAM protocol. The GRAM protocol also allows users to poll for the status of the jobs or associate listeners that get invoked when the job status changes. Additionally, when the job completes, we use GridFTP to retrieve the compressed set of output files.

The Application Coordinator has been designed to take configuration parameters about the application, its requirements and environment. This module supports running ADCIRC with different grids for different geographical regions and configurations. More recently, the module is being customized to be used with different meteorological models.

5.4. Application Preparation

In this work, we assume that the need for urgent computing may necessitate situations that result in ad-hoc quick social arrangements to make resources available during a major storm or weather event.

This has implications on how and what we can expect a site to have installed and/or pre-configured. It is possible that the binaries may not be installed on the target resource. Once a resource for a particular ensemble member is selected, we need to create the application package that will be needed for the particular resource. We create a self-extracting archive file using an open source product called *makeself*. The self-extracting archive file contains everything that is needed for a model run and is the only file that is transferred to the selected grid resource. While in this particular work, the module contains the binary as well, it is possible to use this for applications which might be pre-installed at sites. The specific steps that are involved in creating this bundle include:



Resource Metrics as of 2006-07-18 10:38:00.0

Resource Name	CPUs Available	Network Bandwidth
dante0.renci.org	9.0	0.0
dante2.renci.org	0.0	0.0
gauss.math.tamu.edu	134.0	4.054742
ci-team.acis.ufl.edu	22.0	7.80886
scoops.itsc.uah.edu	0.0	8.21279

Today's Run Status

Wind Field	Grid Field	Ensemble	Resource	Status
WANAFe01-UFL	ec95	01	ci-team.acis.ufl.edu/jobmanager-pbs	●
WANAFe03-UFL	ec95	03	gauss.math.tamu.edu/jobmanager-pbs	●
WANAFe05-UFL	ec95	05	Unknown	●
WANAFe04-UFL	ec95	04	gauss.math.tamu.edu/jobmanager-pbs	●
WANAFe02-UFL	ec95	02	dante0.renci.org/jobmanager-pbs	●

Figure 4. Job status and resource status from the portal

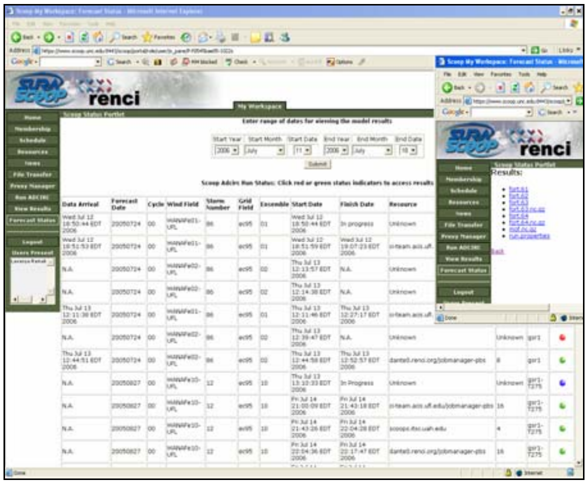


Figure 5. Job History and result files from the portal

Hindcast Model Runs

Enter data for running the model

Year: 2006 Month: January Date: 1 Wind Product: WANA-UFL Ensemble: 01 Storm Number: Cycle: 00

Model: ADCIRC

Figure 6. Hindcast mode from the portal

- Running a program that converts the netCDF version of the input wind file to a version compatible with the ADCIRC model.
- Select the correct set of ADCIRC executables for the given resource architecture and model run.
- Identifying specific arguments that are required at the remote end when the bundle is extracted, e.g. the actual MPI command for ADCIRC might vary slightly on different resources.
- Finally, create the compressed file containing the binary and all the input data.

As described previously in §2, these model runs are usually hotstarted with previous day's model results. The Application Preparation module checks a "*correspondence*" description file to identify the type of hotstart file required for a particular wind type and grid. It checks to see if this file has been generated previously and is available either locally or remotely in the archives. If the file does not already exist, it checks to see if another process is running right now that might generate it. If the file is being generated by another process it waits for the process to complete, otherwise a process is launched to generate the hotstart file.

5.5. Resource Selection

The Grid sites vary greatly in performance and availability. Even with pre-established arrangements for exclusive access, resources and/or services maybe down or unavailable. Hence, given the criticality of the model run completion, we choose to use a dynamic resource selection algorithm to select an appropriate site for the job submission.

During the resource selection process, each of the sites is queried for the queue status and the bandwidth. Globus MDS[28] is an information service that aggregates information about resources and services that are available at a site. Network Weather Service (NWS)[27] is a sensor based distributed system that periodically monitors and dynamically forecasts performance measurements such as CPU and bandwidth.

We have developed a simple plug-in based resource-ranking library. While currently we use only real-time information, the library is flexible in allowing us to collect historical information to make better and more accurate predictions. The question we try to answer in our resource selection is "Where should I run this job right now?" The library is built on top of Java CoG Kit[24] and uses the standard libraries for querying resources. The framework is completely extensible and can easily accommodate more sophisticated algorithms in the future.

The resource selection first searches a list of remote resources to confirm availability in terms of appropriate authentication and authorization access to the resource, ascertain running of the basic Globus services such as GridFTP and GRAM. All remote resources meeting the above requirements are then ranked according to the real-time information including queue status and bandwidth. This allows us to balance the implications of data movement costs with computational running time. Based on queue and the bandwidth a total time estimate on each resource is calculated to rank the resources. The algorithm takes approximate running times for the model and the data sizes as input to perform this calculation.

5.6. Portal

In addition to timely execution of the model, it is important to be able to share the data with the community at large while shielding the consumers of the information from the complexity of the underlying system. We use an Open Grid Computing Environment (OGCE) based portal interface to make available the status of the runs and output files from the daily forecast runs. Figure 4 shows the status of the model runs and Figure 5 shows the results available from the portal. A color marker shows the current state of the run (i.e. “data arrived”, “running”, *etc*.). In addition end users can use the portal to launch hindcast model execution (Figure 6) in a grid computing environment using the files from the SCOOP archives.

5.7. Fault Tolerance and Recovery

We apply a number of techniques to diagnose and repair errors that might occur during run-time, using a two-phase approach in the ADCIRC Application Manager. The first phase uses retries in the event of a failure or a timeout, and the step is retried a specified number of times. If the retries do not resolve the failure, a "persistent" error has occurred. The execution of the application coordinator has distinct phases (move files, run job, *etc*.). Persistent errors may occur in one of these labeled phases. A persistent error causes the decoder to retry beginning at an appropriate earlier phase. In addition, certain kinds of persistent errors, such as a failure to successfully transfer a file to a selected resource, cause that resource to be omitted from consideration during the resource selection phase of the retry. This error handling allows the complete execution of model runs under many different adverse circumstances, taking advantage of the inherent redundancy in a grid enabled environment. The application manager can easily detect errors and take appropriate rectification action. But sometimes errors might occur at the model level producing garbled data or a process might run longer than expected and not produce the output. In future implementations, we anticipate we will need additional error checking to detect these scenarios to decrease the probabilities of failures.

5.8. Monitoring and Notification

A central component of our design is proactive monitoring of the status of the application and data. This monitoring system is based on standard tools and techniques such as Network Weather Service[27] and instrumentation points at various points of the data flow. The key to managing a distributed adaptation framework is a standard messaging interface. Our messaging interface is based on the workflow tracking tools and eventing system (WS-Messenger)[10] being built as part of another NSF ITR project - LEAD (Linked Environments for Atmospheric Discovery)[26]. Every component in our system publishes status information such as “input data arrived”, “task started”, “task finished”, *etc.* This status information is available through the portal interface (Figure 5). In addition, the resource monitoring portlet reads a web service we created that serves

CPU availability and network bandwidth data. The data itself is currently collected using MDS, NWS and the LEAD eventing system and then stored in a MySQL database.

6. Deployment Experiences

Various components of the framework have been tested and deployed in the context of hurricane storm surge over the past two years. In this section we briefly describe our evaluation and experiences.

6.1. Resource Pool Management

The following SCOOP partner and SURAGrid sites have been tested and added to the resource pool for ADCIRC - local resources at Renaissance Computing Institute (RENCI), Texas A&M University (TAMU), University of Florida(UFL), University of Alabama at Huntsville(UAH), and University of Louisiana Lafayette (ULL). Each of the sites run basic Globus grid services such as the Gatekeeper for job submission, GridFTP for file transfer, and an information service and Network Weather Service. Our current infrastructure is based on the pre-web service protocol stack available in Globus versions 2.x through 4.x. It is important that the basic Globus services are configured correctly at all sites that might be used for the model runs. We have a test suite that is used to test all sites to verify the basic services are running and configured correctly. The test suite verifies the access rights, firewall, configuration of Globus services and the batch scheduler that might configured at the site. The sites are tested periodically to verify correct operation. The test suite helps detect, diagnose errors more proactively. To easily add resources to the pool, we use configuration properties. This allows us to add other resources to the pool, without any programmatic changes. The properties include the addresses for the Globus services, firewall port information and security credentials that can be used for a resource.

6.2. Application Coordinator

The application coordinator is configured using a property file allowing easy addition of model configuration parameters *etc*. An application can use the framework by supplying application specific properties and scripts for creating the packaging, *etc*. As mentioned the framework is being applied to the North Carolina Forecasting System to run ADCIRC with different grids and other meteorological models. Our early experiences showed the need for higher resilience and fault tolerance that was needed in the Application Coordinator to recover from various errors that might occur during execution. This was then built into the more recent version of the Coordinator. We are currently planning on wrapping these capabilities as web services allowing for more wide spread use in the Grid framework and workflow tools. Our resource selection algorithm is simplistic, but more generally the framework we have developed allows us to easily integrate other more sophisticated algorithms that are being researched in the Grid community.

7. Related Work

Grid computing has been increasingly used to run scientific applications from different domains including earthquake engineering, bioinformatics, astronomy, meteorology, *etc*. Our framework specifically addresses the problems of the need of increased reliability and fault tolerance and recovery that is needed in the context of time sensitive application such as storm surge prediction. Grid scheduling and adaptation techniques have been based on evaluating system and application performance are used to make scheduling and/or rescheduling decisions. Heuristic techniques are often used to qualitative select and map tasks to available resource pools[25]. Our resource selection algorithm is fairly simplistic and only considers queue status and bandwidth measurements to make a decision. While this is simplistic, it works effectively in our current resource environment. The API has been designed to be flexible to allow easy addition of other more sophisticated algorithms in the future.

8. Conclusions and Future Work

This framework provides a solid foundation on which to build a highly reliable Grid environment for applications that might be time sensitive and/or critical.

An enhancement to the computational system currently being developed is the selection of the ADCIRC model grid based on the predicted storm landfall location. We envision a suite of ADCIRC domains with the same basic open-ocean detail, but with different grids resolving different parts of the coastal region and supporting flooding and surge inundation.

Grid and portal standards have been a moving target for a few years now. Our software stack for this work was guided by state of the art at the time of the project inception. More recently technology implementations of the standard (i.e. JSR 168) and grid standards (WSRF) have stabilized and we will transition to support Globus 4.0 web services and OGCE-2 for our portlets.

Our experiences with building and deploying the framework emphasize the need for increased fault tolerance and recovery techniques to be implemented in real Grid environments. We are investigating standardized web services interfaces that will allow applications to be easily run in a Grid environment with capabilities such as resource selection and fault tolerance. In addition, user-friendly modules that allow scientists to specify the properties needed by the Application Coordinator are being investigated. Data collected from the operation of the framework during the hurricane season will drive further evolution of the framework.

9. Acknowledgements

This study was carried out as a component of the "SURA Coastal Ocean Observing and Prediction (SCOOP) Program", an initiative of the Southeastern Universities Research Association (SURA). Funding support for SCOOP has been provided by the Office of Naval Research, Award N00014-04-1-0721 and by the National Oceanic and Atmospheric Administration's NOAA Ocean Service, Award NA04NOS4730254. We would also like to thank the various SCOOP partners for discussion on the use cases - Philip Bogden (SURA and GoMOOS); Will Perrie, Bash Toulany (BIO); Charlton Purvis, Eric Bridger (GoMOOS); Greg Stone, Gabrielle Allen, Jon MacLaren, Bret Estrada, Chirag Dekate (LSU, Center for Computation and Technology); Gerald Creager, Larry Flournoy, Wei Zhao, Donna Cote and Matt Howard (TAMU); Sara Graves, Helen Conover, Ken Keiser, Matt Smith, and Marilyn Drewry (UAH); Peter Sheng, Justin Davis, Renato Figueiredo, and Vladimir Paramygin (UFL); Harry Wang, Jian Shen and David Forrest (VIMS); Hans Graber, Neil Williams and Geoff Samuels (UMiami); and Mary Fran Yafchak, Don Riley, Don Wright and Joanne Bintz (SURA). We would like to thank various SCOOP and SURAGrid partners for making resources available and special thanks to Steven Johnson (TAMU), Renato J. Figueiredo (UFL), Michael McEniry (UAH), Ian Chang-Yen (ULL), and Brad Viviano (RENCI), for providing valuable system administrator support.

10. References

1. I. Foster, C. Kesselman and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of Supercomputer Applications*, 15(3), 2001.
2. I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *International Journal of Supercomputer Applications*, 11(2):115-128, 1997.
3. J. Novotny, S. Tuecke and V. Welch, "An Online Credential Repository for the Grid: MyProxy," *Proceedings of the Tenth International Symposium on High Performance Distributed Computing* (HPDC-10), August 2001.
4. Open Grid Computing Environment (http://www.collab-ogce.org/nmi/index.jsp)
5. M. Russell, G. Allen, I. Foster, E. Seidel, J. Novotny, J. Shalf, G. von Laszewski and G. Daues, "The Astrophysics Simulation Collaboratory: A Science Portal Enabling Community Software Development," *Proceedings of the Tenth International Symposium on High Performance Distributed Computing* (HPDC-10), pp. 207-215, 2001.
6. W. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal and S. Tuecke, "Data Management and Transfer in High Performance Computational Grid Environments," *Parallel Computing*, 28 (5), pp. 749-771, May 2002
7. I. Foster, C. Kesselman, G. Tsudik and S. Tuecke, "A Security Architecture for Computational Grids," *Fifth ACM Conference on Computer and Communications Security*, pp. 83-92, 1998.
8. L. Pearlman, C. Kesselman, S. Gullapalli, B.F. Spencer Jr., J. Futrelle, K. Ricker, I. Foster, P. Hubbard and C. Severance, "Distributed Hybrid Earthquake Engineering Experiments: Experiences with a Ground Shaking Grid Application," *NEESGrid Technical Report*-2004-42, 2004.
9. Climate of 2005: Atlantic Hurricane Season. http://www.ncdc.noaa.gov/oa/climate/research/2005/hurricanes05.html, 2006.
10. Y. Huang, A. Slominski, C. Herath, and D. Gannon, "WS-Messenger: A Web Services based Messaging System for Service-Oriented Grid Computing ," *6th IEEE International Symposium on Cluster Computing and the Grid* (CCGrid06).

11. E. Kalnay, “Atmospheric Modeling, Data Assimilation and Predictability,” Cambridge University Press, 2003.

12. R.A. Luettich, J. J. Westerink, and N. W. Scheffner, ADCIRC: An advanced three-dimensional circulation model for shelves, coasts and estuaries; Report 1: theory and methodology of ADCIRC- 2DDI and ADCIRC-3DL, *Technical Report DRP-92-6, Coastal Engineering Research Center*, U.S. Army Engineer Waterways Experiment Station, Vicksburg, MS, 1992.

13. G. Holland. An Analytic Model of the Wind and Pressure Profiles in Hurricanes. *Monthly Weather Review*, Vol. 108, No. 8, pp. 1212–1218, 1980.

14. J. Sivillo, J. Ahlquist, and Z. Toth. An Ensemble Forecasting Primer, *Weather and Forecasting*, Vol. 12, pp. 809-818, 1997.

15. Unidata Local Data Manager. http://www.unidata.ucar.edu/software/ldm/, 2006

16. P. Bogden, et al, The Southeastern University Research Association Coastal Ocean Observing and Prediction Program: Integrating Marine Science and Information Technology," *Proceedings of the OCEANS 2005 MTS/IEEE Conference*. Sept. 18-23, 2005, Washington, D.C.

17. D. Huang, G. Allen, C. Dekate, H. Kaiser, Z. Lei and J. MacLaren "getdata: A Grid Enabled Data Client for Coastal Modeling," Published in HPC06.

18. P. Bogden et al., "The SURA Coastal Ocean Observing and Prediction Program (SCOOP) Service-Oriented Architecture," *Proceedings of MTS/IEEE 06 Conference in Boston*, September 18-21, 2006 Boston, MA, Session 3.4 on Ocean Observing Systems.

19. J. Bintz et al., "SCOOP: Enabling a Network of Ocean Observations for Mitigating Coastal Hazards," *Proceedings of the Coastal Society 20th International Conference*, May 14-17, 2006; St. Pete Beach, FL.

20. SCOOP Website http://scoop.sura.org/, 2006.

21. D. A. Reed, et al., "Building the Bioscience Gateway," *Global Grid Forum Technical Paper*, June 2005.

22. North Carolina Forecasting System. http://www.renci.org/projects/indexdr.php

23. S. Graves, K. Keiser, H. Conover, M. Smith. “Enabling Coastal Research and Management with Advanced Information Technology,” 17th Federation Assembly Virtual Poster Session, July 2006.

24. G. von Laszewski, I. Foster, J. Gawor, and P. Lane, "A Java Commodity Grid Kit," *Concurrency and Computation: Practice and Experience*, vol. 13, no. 8-9, pp. 643-662, 2001, http://www.cogkit.org/.

25. D. Angulo, R. Aydt, F. Berman, A. Chien, K. Cooper,H. Dail, J. Dongarra, I. Foster, D. Gannon, L. Johnsson, K. Kennedy, C. Kesselman, M. Mazina, J. Mellor-Crummey, D. Reed, O. Sievert, L. Torczon, S. Vadhiyar, and R. Wolski. Toward a framework for preparing and executing adaptive grid programs. In *Proceedings of International Parallel and Distributed Processing Symposium (IPDPS)*, 2002(41).

26. K. K. Droegemeier et al, “Service-Oriented Environments In Research And Education For Dynamically Interacting With Mesoscale Weather,” *IEEE Computing in Science and Engineering*, November-December 2005.

27. R. Wolski, N.T. Spring, J. Hayes, “The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing,” Future Generation Computer Systems, 1998.

28. K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, “Grid Information Services for Distributed Resource Sharing,” *Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press*, August 2001.