A WRF Portal for Model Test and Verification

Mark W. Govett
National Oceanic and Atmospheric Administration
Forecast Systems Laboratory
Boulder, Colorado 80305
May 7, 2003

ABSTRACT

Testing has begun on two variants of the WRF model to determine the best candidates that will be part of a six member ensemble slated to become the operational model in October 2004. A testing procedure has been identified in the WRF test plan that will require 1255 tests. Additional WRF variants have been proposed for future comparisons that will likely require thousands of additional tests. The management of software, data and tests will require significant resources at multiple super-computing sites both during initial WRF testing and successive research and development. To simplify this process, the Forecast Systems Laboratory has begun working on the design and development of a WRF portal that will be used for the test, evaluation and verification of changes made to the WRF model. The goal of the portal is to simplify the testing and evaluation of the WRF model and ancillary components and changes that are made to them. A portal is broadly defined to be a gateway in which users can access and utilize distributed compute resources without regard to what or where they are. Technologies that can be used include web services, grid computing, clusters, data storage systems and a networking infrastructure. In designing a WRF portal, we wish to incorporate these technologies to the extent they are feasible.

1 INTRODUCTION

The WRF test plan identifies a process by which testing of two variants of the WRF model (NMM and Mass Core dynamics) will be evaluated. End-to-end testing designated by the test plan is a complicated process currently involving four major components: initialization, forecast, post-processing, and verification. These components, illustrated in Figure 1, are represented by ten executables and many thousands of lines of source code that must be managed and ported to several compute platforms. Scripts are also used to both link these components together and to invoke mundane but essential operations including file and directory creation, removing files, copying files to mass storage systems, determining process quotas, setting file permissions, coordinating execution, handling process management, and error recovery. In all, the WRF test plan identifies 1255 tests that must be run to compare these two variants.

The WRF test plan further states that "the initial NCEP WRF system will consist of a six member mesoscale model ensemble". This will require an unspecified number of additional tests to determine the proper mix of ensemble members. Management of the tests and the software will require great care to insure they are setup and run correctly, output results are stored in a logical way, error handling is managed, and details about each test (eg system, input specifications, compiler options) is recorded so tests can be duplicated when necessary. Scripts are being developed to simplify the testing procedure but the tests are still expected to require significant time and effort.

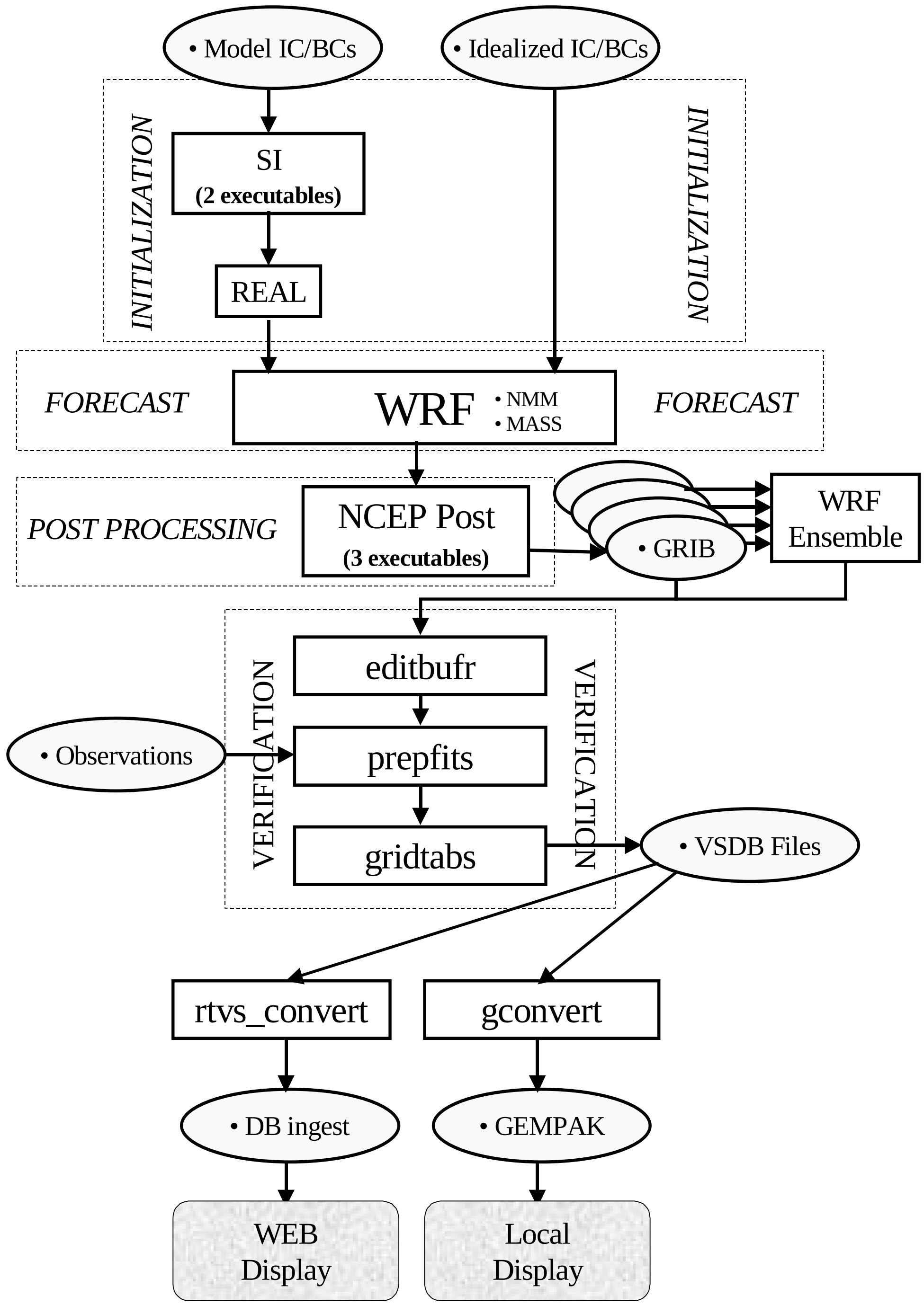

Figure 1: Components and sequence of processes required for testing under the WRF test plan.

These tests must be completed by October 2003 so that operational testing and evaluation can begin and the model can reach operational status by October 2004. During initial and successive operational testing it is inevitable that changes to the model and related ancillary components will occur. It is also unclear how these changes will be made available to other participants of WRF testing and the larger WRF community. How this is accomplished will greatly affect the success of future model tests and development.

While this testing is occurring, the development of WRF and the ancillary components will continue at government and research institutions. In addition to ongoing software upgrades, many new variants of WRF have been proposed as candidates in the operational system; some of these are listed in Figure 2. This list represents a few of the model variants that could be developed, tested and verified during future WRF model development. There are a huge number of possible combinations that could be evaluated resulting in many thousands of model runs, requiring peta-bytes of data and massive compute resources.

Model Initialization
- ETA
- RUC
- GFS

Data Assimilation
- EDAS
- WRF 3DVAR
- WRF 4DVAR

Dynamical Core
- MassCore
- NMM
- NCEP Semi-Lagrangian
- COAMPS

Physics: Convection
- Betts-Miller-Janic
- Kain-Fritsch

Post Processing
- Eta (GRIB, BUFR)
- Visualization (FX-NET, AWIPS)

Verification
- NCEP package
- RTVS

Ensemble
- Arithmetic Mean
- other

Figure 2: Model components that have been proposed as candidate packages for WRF testing.

While the initial testing described by the WRF test plan is manageable at a single institution, additional model variants will involve many researchers in the meteorological community at government facilities and research institutions around the world. Mechanisms must be in place to allow their research to be evaluated by the operational community in a coordinated and efficient manner. The development of a WRF portal for model development, test and verification is intended to address these needs. In the next two sections, the goals of this portal development and the design of a prototype portal will be further described.

2 GOAL OF THE WRF PORTAL

The goal of a WRF portal is to simplify the testing and evaluation of the WRF model and its ancillary components and changes that are made to them. The portal will be defined in a way that will permit grid and web technologies to be used where appropriate. Grid technologies will be employed to provide a powerful and flexible way to access and combine resources from multiple institutions so that tests can be run or data results can be accessed more efficiently. Web technologies will be employed in order to simplify the access to the WRF model for tests and evaluation by the larger meteorological community. The portal will incorporate all the elements of end-to-end testing including initialization, model forecast , post-processing, and verification.

In designing the portal, accessibility, extensibility, software management, data management, user interface, and security issues must be considered. Access to the portal must be broad so that anyone can evaluate their own model developments and compare them to the operational or any other version of the code. Extensibility is important to allow new models and methodologies to be tested. Models are always being enhanced with new dynamics, physics, different resolutions that must be included in any portal design and related Graphical User Interface (GUI) development. WRF was designed with Application Programming Interfaces (APIs) that permit software to easily be plugged into the model for test and evaluation, allowing users to easily test new science. Effective software management is required to track, test and evaluate different variants of the WRF components. In addition, the software used to run the test must be stored so that tests can be rerun and results duplicated when required. Data results from model runs must be stored in a way that permits easy access for comparison and evaluation. All of these considerations must be combined into an effective, well-designed user interface that will allow users to take advantage of the full capabilities of the portal.

Finally, security issues must also be considered. Software to support grid computing are maturing rapidly, but NOAA has yet to develop a policy that must address security concerns that are inherent in intra-institution access to resources. However, the portal development need not be constrained by these issues. Current testing in the WRF test plan relies on the ability of each participating institution (currently NCAR, FSL NCEP, AFWA, FNMOC) to run independent tests on individual computers over the same data sets. The verification results will be made available via a web server so all participants can access results for each test run by each institution. Similarly, the WRF portal could permit participating institutions to run their own tests and make available verification results via web files. So while security issues must be considered and a WRF portal would benefit by incorporating grid technologies into the development, the portal need not be limited by inter-institutional issues in the short term. In the longer term, as grid computing becomes more mainstream, security and control issues will become less problematic.

3 A WRF PORTAL CONCEPT

3.1 Accessibility

The portal software will be designed to be flexible so all of the meteorological groups can obtain, access and use its capabilities. There are two main executing environments for web-based applications: by the client and by a server. For server-based applications (an applet), the client enters the portal via a web page that is served by a remote system. For client-based applications the software is installed by the user on the client system and can be run independently thereafter. When new versions of the portal software become available, applets will provide access to them as soon as they are available. Since client-based applications download the portal manually, they could be required to reinstall the software. However, at least one language, Java has solved this problem. Java permits an add-on package, called Java WebStart, to automatically update the application, if required, each time it is run by the user. This allows users to take advantage of new portal developments as they become available.

Both environments would be appropriate for the WRF portal, however client-based applications are superior to an applet for several reasons: (1) Applets are run within the context of a client browser and differences in these browsers (new versions or different browsers) can cause portability problems. (2) Applets require the application to be downloaded by the client each time the portal web page is referenced; if the application becomes large, the download penalty could become significant. (3) Failure of the network or systems where the server(s) resides would prevent anyone from running the applet.

3.2 Extensibility

The WRF portal must be extensible and will be designed: to permit new model components or new tests to be added, to easily configure new compute or data storage resources as grid resources, and to allow multiple portals to exist collaboratively or independent of one another. It is expected that continuing research into various aspects of weather prediction will lead to a huge number of model variants that will need testing. The portal software must be designed to easily add these variants into the WRF test portal. In addition, compute and data storage requirements are expected to be heavy. As new compute and storage resources become available, the portal must be designed to easily incorporate them.

It is expected that more than one WRF test portal will exist. In Figure 3, for example, two portals are shown: a "Reference test facility" contains primary participants (filled box) and some collaborators (striped box), while a second "Research test facility" contain members of the university research community who could build their own testing infrastructure. Both of these portals could interact in a way that promote the sharing of test results if desired. For example, FSL might be a collaborator on one portal and a contributor on another. As a collaborator, FSL could track developments in a research environment and perhaps incorporate elements of this work into further tests on the Reference test facility when appropriate. Further discussion of data management issues follow in section 3.4.

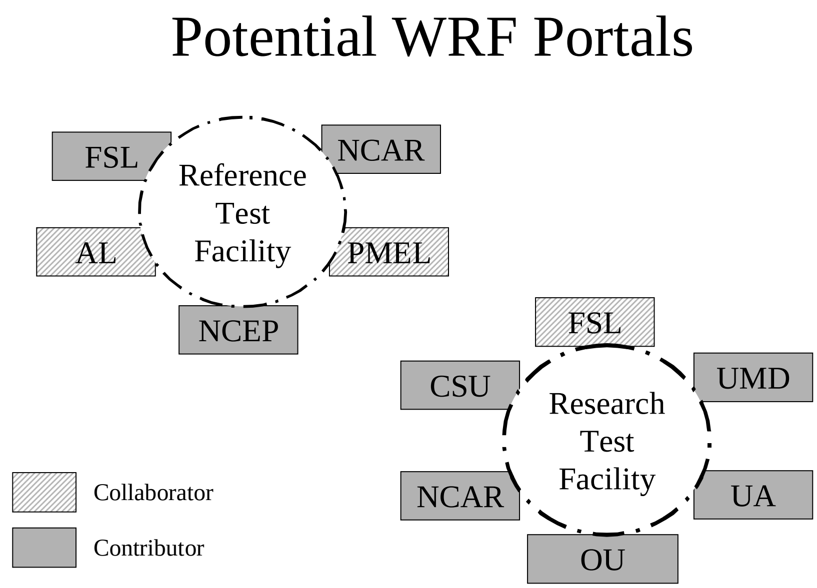

Figure 3: An illustration of two WRF portals. Users of contributing institutions are allowed to run tests, store data and view results. Users of collaborating institutions are only permitted to view results.

3.3 Software Management

WRF model tests must be conducted in a way that will tightly control both software and data to ensure test results can be duplicated. Software management comprises such details as the modeling software, run scripts, model configurations (eg. namelist input files) for each component, type and location of output files generated by the run, location of statistical results of any model comparison, metrics describing the model run (eg. execution time), and details about the system where the run was executed.

Software source code control is very important. Reference source code must be identified explicitly (via version numbers) and accessible so tests can be rerun and results verified when necessary. Software management tools such as CVS (Concurrent Versions System), at a minimum, should be used to track, store and retrieve the WRF source code. This information must be stored when tests are run, results verified and reference code established.

3.4 Data Management

The WRF test plan identifies 1255 tests that must be run to compare two different dynamics packages and related physics. As the number of different components increase, the number of tests is expected to require tens of thousands of model runs. The ability to access and save both input and output files is critical. Multiple sites containing mass storage systems will be required. While input files are needed to reproduce results, some output files may not need to be stored. Model verification results will be required; however, intermediate files (between processes) and

GRIB output files may not be accessed frequently enough to warrant quick access to them. It may be simpler to rerun the test cases to generate the data in the event they are needed. Whether these data are stored really depends on how frequently they are required.

Once the tests are run, information about the tests must be stored and made accessible for broad access by the WRF research community. This will allow others to determine what tests have already been run and where to find the results. Information about the tests (some or all) can be described in web accessible file whose contents can be searched, sorted, analyzed and displayed as a web file. An XML formatted web file provides a standardized format to meet these requirements. XML (eXtensible Mark-up Language) is a more general variant of the widely used HTML web format. It is rapidly becoming the format of choice for data exchange on the web. While HTML is only used for displaying data on web pages, XML can describe the content of the data files and allow standard operations on the data such as searching or sorting of any criteria contained within the file. Criteria contained within the XML file should include the following information:

<u>Test case details</u>

- Model run dates
- Description of each model component (eg. RUC background fields)
- Model resolution & domain
- Software versions (CVS) of each model component
- Description of hardware used to run the test
- Performance results (eg run-time, elapsed time) for each component
- Institution where test was run (eg. NCEP)
- Date when test was run and by whom
- Specific run-time component selections (eg. namelist files)
- Type and location of background files used for model initialization
- Did the run complete and are results available?

<u>Test case data</u>

- Verification results
- Observational data used to verify tests*
- Model input (for initialization)*
- Model output (eg. GRIB, BUFR, GEMPAK, FX-NET files)*

  * non-local data – could be links to the location of the data (eg. point data could be available via the MADIS project, model output via the NCAR Mass Store), and input via the FSL Mass Store)

Sufficient information about the tests would be provided for access by users of the portal where the tests were run, and to the broader meteorological community. A schema defining the structure and format of these XML files would be defined in a way that promote searching of cases using specific testing criteria. A search engine, somewhat analogous to "google", would be built as part of the portal, that understands the structure and information contained within the XML files. For example, a user could request all tests that have been run for January 1st 2003 using the Mass-Coordinate dynamics, NCAR physics, and RUC background initialization that were run on the ijet computer. The user could then avoid repeated runs, and save valuable compute resources by simply getting the results instead of being forced to rerun the case.

Figure 4 illustrates a possible configuration that incorporates these technologies. Four main components are shown: institutions, data storage, data server, and an XML crawler. Three of the institutions (filled ovals) are permitted full access to compute and data resources, the other (striped oval) is only permitted access to data results. However, each institution would have their own data storage used to retain input and model output data. These data could restrict access in a number of ways: (1) as local data, (2) accessible only by portal participants, and (3) available to all users including those outside the portal. Data would be made available to the portal participants using a registry which identifies a location where the XML case data is stored. This registry would be accessed by an XML crawler process that periodically searches these locations for new or updated tests. This information is then used to populate an XML data base which is queried by portal users via a data server.

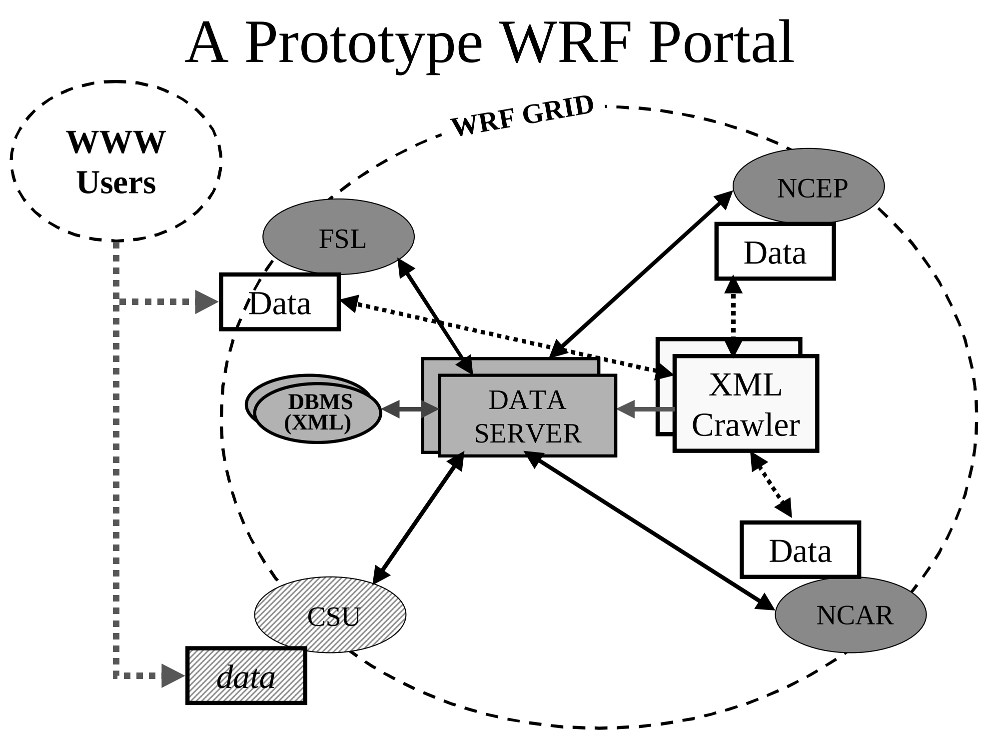

Figure 4: A WRF portal test facility. Each institution on the grid (dashed circle) would permit portal searches to query the XML data base via a data server. This server would access a database which stores the contents of XML files that were "discovered" and populated via an XML web crawler process. The XML Crawler would look for WRF XML files on any system where WRF runs are registered. WRF runs are registered in the data server (illustrated) by identifying a location on the local file system (eg. at NCAR) where XML data representing the WRF runs will be stored.

3.5 User Interface

Users will interact with the portal via a graphical user interface (GUI). The GUI must be flexible such that different instances of the WRF model can be tested. It is the single most important area where a thorough requirements analysis and input from users is needed. The portal should both incorporate the requirements of the WRF test plan and be designed so that both new and different models can be evaluated using the portal software.

Users should be able to set-up, run and re-run test cases with minimal effort. Runs constructed by the user should be easy to build, be simple to save so cases can be rerun, and be easily morphed into new test cases. The user should also be able to construct ensemble runs containing multiple source versions and models that can individually produce results or be combined collectively into a single output file that is verified. In addition to constructing single runs, multiple test cases should also be permitted to be built and simultaneously launched. For example, a user should be able to launch a suite of tests for comparison that would be run concurrently on the portal computers.

Once these tests are launched users should be able to track their progress while they are running, be informed when they have completed, and know jobs will automatically restart in the event they fail. Software packages such as Condor have been successfully used to provide this job monitoring, resource scheduling, and job restart capability. It is likely that such technology will be leveraged in the development of the WRF portal to handle these requirements.

The portal should also users to view system and job queue loads to determine where the best resources are to run the test. The application must be simple to setup and configure new resources and possibly discover new resources as they become available. In end-to-end testing, some components (such as initialization) could require fewer processes / memory than other more compute intensive processes. The portal and related scheduling mechanisms should be able to move jobs and related data to the machine where it will run most efficiently.

Figure 5 describes initial GUI requirements that can serve as a starting place for discussion of the user interface development. It is currently composed of three main areas of development: Data Management, Test Case Management and Resource Management.

A. <u>Data Management</u>
    1. input data case management
        i. idealized ICs w/physics
        ii. real case ICs w/o physics
        iii. real case ICs w/physics
        iv. real case seasonal test
        v. 3D convection
        vi. resolution (6, 8, 10, 12km)
        vii. real-time testing
    2. output data file management
        i. local / national domain
        ii. hi/low resolution window
        iii. file format (NetCDF, AWIPS, FX-NET)

3. Verification output
    i. NCEP Verification system
    ii. Real-Time Verification System
        - % bins / std dev
        - Ebert and McBride system
    iii. RUC Verification
    iv. Hovmoller diagram perspective
    v. AFWA 3-D cloud analyses

B. <u>Test Case Management</u>
1. software version for each component / scripts
    i. CVS tag that identifies the unique software used
    ii. CVS software repository location
2. Model / Component selections
    i. model initialization type
    ii. data assimilation package
    iii. model forecast variant
    iv. post processor package
    v. verification package
3. Ensemble selections
    i. how the ensemble is populated
4. type of test
    i. idealized
    ii. retrospective data
    iii. real data
5. hardware requirements / job queue preferences given
    i. number of compute nodes required
    ii. minimum memory requirements
    iii. specific machine architecture
    iv. network requirements (minimum bandwidth, etc)
6. input, output and scratch directory path names
    i. there could be multiple paths for both input and output data
7. user notes about this test
    i. motivation for test
    ii. expected results
    iii. ensemble details

C. <u>Resource management</u>
1. Job monitoring
    i. current model state
    ii. execution time
2. Checkpoint / restart
    i. job migration
    ii. generation of restart files

3. Grid Resource discovery / configuration
    i. determine resources available
    ii. add new resources
    iii. determine the sub-set of resources for each job component
4. Data storage
    i. storage available or used on Mass Store, scratch file system
    ii. requirements of each component for local storage
    iii. migration of intermediate data to systems where "next" WRF component will run
5. Job Dependence Relations
    i. identify processes that depend on the completion of others
    ii. migration of data to the system where the next job will run
6. Security
    i. single sign-on
    ii. authentication
    iii. user-based access to resources

Figure 5: A few of the requirements that could be incorporated into the development of the GUI for the WRF portal. Major functionality would be divided into the management of resources, data, and test cases.

3.6 Security

There are two aspects to security. One aspect is security within a single institution. Gateways or firewalls are typically provided to limit access to compute and data resources within a single physical location. These mechanisms are constantly being improved to protect the organization from electronic attack. A second aspect to security occurs when users wish to access resources outside of a single institution. Portals often span locations or institutional boundaries in order to make more efficient use of resources which can present more significant security issues: both real and perceived.

Grid software has been developed using the concept of a single sign-on which handles both user authentication and access. When the user logs on to a grid they are authenticated to insure their identity. This identity is then used to determine what resources they are permitted to access and what resource quotas are enabled. When the grid account is established, they are assigned a user certificate which handles the access to grid resources. The certificate is passed to grid resources as they are needed to broker what access is permitted. This eliminates the need to re-authenticate (log on). It further allows users to run tests on a variety of resources without regard to where or what they are. This is a founding principal of grid computing and contains many benefits for a WRF test portal.

Perceived security issues stem from reluctance of individual institutions to give up access and control of resources to multiple institutions (to the grid). The concept of a single sign-on allows users to use resources in another organization without having to re-authenticate. This idea is similar to using a secure shell commands (eg. ssh, scp) between systems and that has been widely accepted. Using ssh for example, authorized keys are defined for each system users wish to access; hosts are trusted in the same way as grid-based security certificates handle access to resources.

In time, policy decisions will need to be made by participating institutions to the extent grid computing will be adopted, however implementation of a grid portal need not be limited by these issues. Instead the WRF portal can focus in the short term on (1) facilitating WRF testing on a single institution's computer systems thus avoiding security issues, and (2) enabling the viewing of WRF test results via the web. In the longer term, it is felt that many of the security, access and control issues will be resolved, resulting in better collaboration between institutions and more efficient utilization of compute and data resources.