Information for Code Contributors

WRF is a community modeling system, reflecting contributions by scientists and software engineers for a spectrum of codes and capabilities. It is through such contributions that the system advances. To facilitate this contribution process, the procedures for the addition of codes are described here. Prospective contributors should become familiar with these procedures.

NCAR is responsible for the support to the user community of WRF code and for the administration of the code repository and of code releases. This encompasses oversight of the processes for the testing of new code and its addition to the repository. To guide NCAR's support activities, the document "WRF Code Repository and Release Administration" [http://www.mmm.ucar.edu/wrf/docs/code_admin.pdf] lays out the procedures and policies. That document may be consulted for details on the mechanisms summarized here.

Code Contribution– General Information

Developers' Committee Oversight

WRF repository management is handled by the Developers' Committee, while release oversight is handled by the Release Committee. Both of these committees are described in detail in the "WRF Code Repository and Release Administration" document.

The Developers' Committee (DC) oversees the management and maintenance of the repository of WRF system software. The DC keeps the WRF code in order and in a state of readiness through testing and through reviewing proposed modifications to the code trunk. Code modifications or additions are validated with an automated set of tests that are run independently of the contributor. Changes that modify scientific results must be verified by the contributor and be available for review. The DC has responsibility for the timely testing and inclusion of code into the repository.

The DC meets on a regular basis to consider proposals for adding code to the WRF repository. Such proposals are submitted in an agreed-upon time period prior to the meeting and distributed to the DC.

Basic Responsibilities of Prospective Contributors

Prospective contributors of code should contact a member of the Developers' Committee (DC) on their desire to provide software to the WRF system. At this time, wrfhelp (wrfhelp@ucar.edu) can be consulted for information on contacting the DC. Prospective contributors will be assigned a member of the DC to work with in the preparation and integration of their code into the repository.

It is the contributor's responsibility to make reasonable efforts to ensure that a proposed change is correct and that its impact on other parts of the model, if any, are described in adequate detail to any other developers who might be affected. Any impacted contributor may request a code review and may work with the originating contributor to run tests to validate that the change does no harm.

Disagreements about changes to scientific results must be resolved before changes can be committed. Developers have effective veto power over changes that affect aspects of the WRF system for which they have primary responsibility.

Before each DC meeting, each contributor will have already run the regression tests on their working copy, using code updated to the top of the repository trunk. During the meeting, proposals will be reviewed, and those that are approved will be placed in a *merge-test-commit* queue. The contributors in the queue will the use the *WRF Merge-Test-Commit* procedure to commit their changes into the WRF source code repository. The Merge-Test-Commit procedure is described at https://wiki.ucar.edu/display/mmm/WRF+merge+test+commit+procedure.

Code Contribution– Procedures

The processes involved in code contributions are described here in terms of periods related to the approximately annual schedule of major releases for WRF. Further details on these periods are provided in "WRF Code Repository and Release Administration".

Contributors should coordinate with DC members to make sure their code intentions for future releases are known. This can be through an informal collaboration with members of the DC or other communication of plans to members. The WRF users' web site provides information on release procedures and timetables and on the responsibilities of contributors during the release preparation period.

Normal Status/Post-Release Period

New developments, existing code improvements, and contributions of bug fixes take place between releases. Items are developed either by DC members or by contributors who provide them to a DC member. In either case, it is up to such DC members to confirm that the provided code works with the top of the repository and passes the basic regression tests. Code contributors are responsible for:

(1) verifying and validating the single-processor code;
(2) ensuring that the source code conforms to the WRF software architecture requirements;
(3) warning the DC about limiting underlying assumptions or possible code conflicts;
(4) working with DC members to develop the necessary tests to verify that future modifications do not adversely impact their code;
(5) incorporating these tests into the WRF regression test suite; and
(6) documentation.

A proposed modification to the repository is circulated among the DC via email. The proposing developer classifies the modification (e.g., bug fix, enhancement, new feature), provides a motivation for the modification, describes the changes to the code required, and lists the touched files. If the proposed change modifies scientific results, the proposer must show how results are changed, and additional testing may be required to obtain this information. Other members of the

DC may ask for a hold to be placed on some or all of a set of proposed commits to allow for further review.

Testing of contributions is coordinated with the regular meetings of the DC. Every proposed commit is initially vetted by a short regression test conducted by the proposer. After each meeting's accumulated modifications have been committed to the repository, a large regression test is conducted on the primary supercomputers at NCAR. The purpose of the large regression test is to identify software errors (e.g., bit-for-bit differences) and failures to compile or run. These tests do not attempt to detect or analyze variances in the forecast skill.

Once the large regression test has been conducted and the results obtained, any unexpected failures are investigated to identify which of the previous commits introduced a fail condition in the testing package. The contributor is informed of the testing status. If a fix is easily integrated into the code, it is, and the fixed code is re-tested. If no simple fix is available, or if the supplied fix cannot rectify the testing suite to the baseline passing status, then the modification is backed out of the repository. A final large regression test is always required on the current top of the repository to validate the code.

Contributors are responsible for supplying documentation on the code they provide. This may be in the form of a web page or adequate (as deemed by the DC) README files or inline documentation.

Pre-Release Period

Prospective contributors of code targeted for the next release who are not on the DC are assigned a DC member to coordinate testing of their components, and they shall work with their DC liaisons to get the source code committed to the repository. The contributors are responsible for following all contribution policies and DC requests and for providing any necessary accompanying data needed for the testing.

(i) Release minus 4 months (R–4m)

By four (4) months before a release target date developers must provide their code to the relevant points of contact (POCs) on the release committee. *R–4m is the point at which code must be provided to the POCs, and it is the responsibility of the code developers to meet this deadline.*

(ii) Release minus 3 months (R–3m)

At this point (R–3m) all new code and features to be included in the release must be in the repository. The repository is frozen except for those changes that are required for the release development. Limited final modifications, such as for bugs found in the new features or for difficulties with feature interaction, are permitted within the 3-month period. Large code changes unrelated to the main elements or goals of the release may be denied and postponed until after the release.

(iii) Release minus 2 months (R–2m)

The code remains frozen. The DC only entertains proposed repository modifications that are related to deficient features slated for the release.

(iv) Release minus 1 month (R–1m)

The only acceptable changes are those pertaining to testing of the frozen code. Improvements or bug fixes uncovered in testing and documentation updates are allowed into the repository, but only with approval of the DC, and only by the person designated by the DC to update the repository in this period.

Repository Information

The WRF code repository is the store of files making up the WRF model and software infrastructure (code and meta-code, build scripts, testing mechanisms and data sets, documentation, etc.) maintained under a version control system (currently Subversion). The repository is maintained such that it always contains the most current, working, and theoretically-releasable revision of the WRF model, plus a fully-recoverable history of past revisions and developer notations. The physical location of the repository is a server at NCAR.

The Subversion directory structure contains the *trunk*, which is the repository itself; *tags*, which are a series of development snapshots of the trunk; plus a number of *branches* managed and maintained by individual or groups of developers independent of the trunk. (As such, branches are neither under the control of, nor the responsibility of, the DC, but are maintained under Subversion to provide revision control for the projects while they are being developed and to ease integration of new developments onto the trunk when they are ready.) Each tag on the trunk is named with a date stamp and a developer ID, and every set of code modifications is assigned a tag.

Access to the Subversion repository is by agreement of the DC. Write access to the repository is limited to the members of the DC. Read-only access to the repository is readily available upon request, with the understanding that repository versions of the code are not releases and therefore not supported.