

## WRF Code Repository and Release Administration

This document supplements the “WRF Code Management and Support” document by providing details on the policies and procedures for NCAR’s maintaining the WRF code repository and overseeing releases.

### A. Committees

#### 1) Overview

WRF repository and release management is primarily handled by two committees: the Developers’ Committee and the Release Committee. The Developers’ Committee oversees regular additions to, and maintenance of, the repository, while the Release Committee oversees the periodic issuance of new code to the user community.

The Developers’ Committee has responsibility for maintaining the WRF system software through the implementation of procedures, and oversight of the process, for source code modification, source code testing, and repository upkeep. It determines whether modifications are acceptable for inclusion in the repository and manages the repository. The Release Committee has responsibility for managing the process of the preparation and issuance of WRF major releases. It reviews input from, and provides information to, the user community, and oversees the release process.

#### 2) Structures and Functions

##### *a) Developers’ Committee*

The Developers’ Committee keeps the WRF system code in order and in a state of readiness through testing and reviewing proposed modifications to the code trunk. All changes are validated with an automated set of tests on a code modification or addition that are run independently of the contributor. In addition, changes that modify scientific results must be verified by the contributor and be available for review. The Developers’ Committee has responsibility for the timely testing and inclusion of code into the repository prior to releases.

The Developers’ Committee is constituted of those active in developing or maintaining the WRF system. To date, committee membership requirements have been informal, requiring only that one participate in the regular meetings; that a developer have some code either already in the model or proposed to go into the model; and that one agrees to abide by the rules for making *commits* (changes) to the WRF repository.

##### *b) Release Committee*

The Release Committee oversees the scheduling, preparation, and issuance of updated WRF modeling system code in the form of major releases. It reviews the priorities on the list of user-requested capabilities and recent and planned code contributions to identify candidates for release inclusion. For new elements, it determines which items could meet the release schedule given their degrees of development and testing and their expected progress.

The Release Committee sets the specific release schedule, including the timetable for testing and preparation. The Release Committee provides information on releases to the WRF user community through postings under the WRF model web page and at the annual WRF workshop.

The Release Committee is constituted primarily of NCAR personnel involved in WRF code development and user support; it typically, however, includes others representing areas of code contribution for a given release. Some of its members serve as liaisons with the Developers' Committee. The Release Committee is chaired by a person in the Mesoscale and Microscale Meteorology (MMM) Division of NCAR (appointed by the MMM director), as MMM is responsible for support of the code to the community. Committee decisions are made by consensus, with the chairperson arbitrating final decisions if necessary.

## B. Repository and Release Management

### 1) Repository

#### *a) Definition, structure, and access*

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 datasets, documentation, etc.) maintained under a version control system (currently Subversion). The repository is managed and maintained by the WRF Developers' Committee 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 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 Developers' Committee, 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.

The current physical location of the repository is a server at NCAR, and the address of the repository is <https://svn-wrf-model.cgd.ucar.edu>. This site reflects a cost-effective method to provide easy access to WRF developers outside the NCAR network firewall, while providing a secure environment for the source code.

Access to the Subversion repository is by agreement of the Developers' Committee. Write access to the repository is limited to the members of the Developers' Committee. 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.

### *b) Responsibilities of the Developers' Committee*

The Developers' Committee oversees the management and maintenance of the WRF repository. The purposes and guiding principles of the Developers' Committee are as follows.

- *Shepherd new development*  
The Developers' Committee provides an interface to outside developers and facilitates incorporation of new or enhanced features and functionalities in WRF subject to the constraints of appropriateness, correctness, impact on existing features and functionalities, and available resources for implementation and testing.
- *Quality assurance and investment preservation*  
The Developers' Committee ensures that the current revision of WRF— the *top of the repository*— is correct and theoretically releasable, subject to the release procedures described in this document. New or enhanced functionality should have minimal impact on existing features and functionalities.
- *Process management*  
The Developers' Committee is responsible for establishing and executing processes for maintaining the WRF repository and for evaluating, incorporating, and assuring the quality of new features and functionalities.
- *Release recommendation*  
The Developers' Committee, after adequate testing has been performed, reports to the Release Committee to certify the repository as suitable for release.

The Developers' Committee meets on a regular basis (e.g., weekly, biweekly) to consider proposals for adding new developments to the WRF repository. Such proposals are submitted in an agreed-upon time period prior to the meeting and distributed to the Developers' Committee.

It is the proposing developer's responsibility to make reasonable efforts to ensure that the 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 developer may request a code review and may work with the originating developer 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 Developers' Committee meeting, each developer 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 developers in the queue will 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>.

## 2) Releases

### *a) Responsibilities of the Release Committee*

The WRF Release Committee oversees the major release process. This includes review of information on desired features; review of the status of new code/features already contributed or being planned; deciding on the content and specific timing of major releases; and coordination of release preparation, testing, and communication. The Release Committee may determine that a major release is warranted based on the approximate annual release cycle or the needs of the user community. Also, the WRF Developers' Committee may make a recommendation for a release to the Release Committee, reflecting significant accumulated changes or major updates.

After the Release Committee decides upon a release, it sets a release timeline. The Release Committee communicates the release decision and timeline information to the WRF community. Information for contributors and on release procedures and the release timetable is posted under the WRF model web page (<http://www.wrf-model.org>). Release news is presented at the WRF Users' Workshop.

### *b) Types*

#### (i) Major

Major releases are made on an approximately annual basis. Their frequency may vary slightly depending on the status of key desired developments and of major development testing. Major releases generally reflect new and improved capabilities to satisfy noted priorities and strategic goals. The Release Committee makes a plan and schedule for the major release and posts release information and a timetable on the web.

#### (ii) Minor

Minor releases mostly address bug fixes. Minor releases are determined and overseen by the director of wrfhelp, and their preparation and (after issuance) support are the responsibility of wrfhelp. Minor release material usually consists of the modified routines or files and accompanying documentation and/or instructions posted on the "Known Problems" pages maintained by wrfhelp. For minor release material, only limited tests are performed: the affected code and the broader system using the code do not go through the exhaustive testing conducted for a major release. Furthermore, for minor release material, the user is informed of the limitations on the code issued and the potential for unknown interactions with the rest of the WRF system.

Information on bug fixes is posted on the web. The director of wrfhelp shall inform the Release Committee of the decision to make a minor release.

### *c) Nomenclature*

Releases are generally numbered as Release a.b or a.b.c, or, in rare, emergency situations, a.b.c.d. The first digit, or changes to it, reflects major restructuring, development, or upgrades. The second digit reflects new capabilities and backward compatibility with older input files, and is usually associated with a major release. The third digit reflects bug fixes/minor releases. On occasion, an emergency minor release might be issued to correct quickly a recently-discovered, significant bug, and it is designated by a fourth digit.

### 3) Activity status and procedures

The levels of code maintenance and testing activity vary with the calendar for a major release. The two basic status designations are *normal* and *pre-release*.

The first designation— status *normal*— refers to a baseline level of activity, that occurring in the period immediately after a given release and before the ramp-up to the next. Baseline level activity is predominantly software engineering testing. The testing is performed with an automated system and is done to ensure accuracy and confidence in parallel results. The second status— *pre-release*— refers to the three (3) months prior to a scheduled major release, where code testing covers additional architectures, performance of case studies, regression testing with varied input data, and broader testing suites including pre- and post-processors.

#### a) *Baseline activity— Status: Normal*

##### (i) Post-release

During the period following a WRF release, there is a relatively relaxed testing schedule. The testing is coordinated with and tied to the regular meetings of the Developers' Committee. Every proposed commit is initially vetted by a short regression test conducted by the contributor. After the test cycle's accumulated modifications have been committed to the repository, a larger regression test is conducted on the primary supercomputers at NCAR. The purpose of this regression testing is to identify software errors (e.g., bit-for-bit differences) and failures to compile or run. These tests are short and do not attempt to detect or analyze variances in the forecast skill.

A proposed modification to the repository is circulated among the Developers' Committee via email. The proposing contributor 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 contributor must show how results are changed, and additional testing may be required to obtain this information. Other members of the Developers' Committee may ask for a hold to be placed on some or all of a set of commits to allow for further review.

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 is not able to 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. The Release Committee solicits new features (e.g., via notices to *wrf-news* and web postings) and the code for them from the WRF user community

(ii) Release minus 6 months (R-6m)

The designation of the testing level as normal extends into the first third of the R-6m phase. The Release Committee targets a major release as a combination of a proposed date and set of new code or features. The Release Committee reviews ongoing WRF development and considers for the next major release those capabilities that would be important for the system or the user community and that can be integrated into the repository within the release timeline. The *release picture* is the list of features that can be considered for release inclusion and any features that are considered mandatory for a release. The Release Committee decides upon the preliminary make-up of the released code and lists the candidate release elements and the release schedule on a web page on the WRF site.

Prospective code contributors who are not on the Developers' Committee are assigned a Developers' Committee member to coordinate testing of their components and to support the commit process. The contributors work with their committee liaisons to commit the source code into the repository. The contributors are responsible for following all contribution policies and Developers' Committee requests and for providing any necessary accompanying data needed for the testing. The Developers' Committee updates the Release Committee on the status of the source code testing in general and the commit progress for the targeted new features in particular.

b) *Release activity— Status: Pre-release*

(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.*

(ii) Release minus 3 months (R-3m)

A check on new features to be included in the release is discussed by the Release Committee and by the Developers' Committee. The decision to either (i) excise a feature which will not be ready within the schedule or (ii) change the release date so that the feature is included in the release is made by the Release Committee.

*In this period (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. Tests for new features are included in the regression suite.

During this period (R–3m), meetings of the Release Committee are held approximately every 2–3 weeks. The status of the new source code features, as they relate to meeting the release schedule, are discussed by a member who also sits on the Developers’ Committee.

(iii) Release minus 2 months (R–2m)

Members of the Release Committee may define a list of non-automated tests to review the forecast skill of the model. These may include case study analyses or linking to other software packages. As such tests are run, the results are discussed at the Release Committee meetings.

During this period the regression test data sets are finalized. Links to the pre- and post-processors are tested with the top of the repository (e.g., naming conventions, directory structure). For data assimilation code testing, cycled runs for the data assimilation period are typically done. The status of the links to the forecast component of the model is reported to the Release and Developers’ Committees. Deficiencies in documentation are identified and update tasks are assigned to the developers responsible.

Within this period, the Developers’ Committee at its discretion may make a beta-version of the code available to vendors and friendly users for their own pre-release testing. Also within this period, the Developers’ Committee may from time to time implement moratoria on changes/commits to the repository to facilitate the resolution of problems or testing by various users and contributors. Such moratoria and their timing shall be at the discretion of the Developers’ Committee.

(iv) Release minus 1 month (R–1m)

Except for the accommodation of bug fixes, in this period the repository is frozen; only fixes for bugs identified during testing on the code to be released may be introduced. Changes to documentation files, however, are freely accepted. Changes to the repository are announced, and a designated member of the Developers’ Committee updates the repository, with the Developers’ Committee identifying a backup member responsible for the updating in case the primary designee cannot. The Release Committee and the Developers’ Committee each meets weekly, where the testing status is reviewed.

The full set of regression tests is run on the relevant machines at NCAR. The pre-release version of the code may be used in real-time slots in additional configuration tests. The code is re-run through cycling tests for WRF-Var. Specific tests involving forecast skill (e.g., case studies) and final tests for funding agency and operational suite needs may be conducted.

(v) Release minus 3 weeks (R–3w)

Software-related issues uncovered in the first round of broad testing are assigned to members of the Developers' Committee. The task of those members is to return with a solution. If the problem cannot be fixed, then the Release Committee is given one of three recommendations: (i) accept the code as is, (ii) remove the offending code from the release, or (iii) delay the release date. The Release Committee makes the decision with respect to the release.

(vi) Release minus 2 weeks (R-2w)

The final round of large testing begins, with the modifications of the Developers' Committee members' fixes. The weekly meetings of the Developers' Committee and the Release Committee focus on the status of the code and the testing.

(vii) Release minus 1 week (R-1w)

The Developers' Committee determines the *certification*, or final approval, of the code. The repository is turned over to the control of the WRF user support group. The final modifications for code identification, documentation, and release readiness are committed. The code is packaged and staged for final release. The countdown list of WRF user support includes final tar file generation, web page updates, user notification, and documentation updates.

### c) *Release procedures*

(i) Publication of release plans

After they are developed, the plans and schedule for the next major release are posted on the web (under the WRF model site). The information includes a list of the significant candidate items for the release and a release preparation timetable.

At the WRF Users' Workshop, the plans for the next major release are included in a presentation. This includes information on the schedule and candidate features. Code contributions for the release are solicited.

(ii) Coordination with contributors and developers

Contributors should coordinate with Developers' Committee members to make sure their code intentions for future releases are known. This can take the form of an informal collaboration with members of the Developers' Committee or other communication of plans to members of the committee. Usually the Developers' Committee members can gauge whether developments will be ready for inclusion in the next release.

(iii) Publication of procedures and responsibilities

The Release Committee maintains a page on the WRF model web site to provide information on major release procedures and timetables. The page includes a description of, or a link to a page

with, the responsibilities of contributors and the procedures for contribution. Information on the Release Committee is also provided.

#### 4) Code Contribution

##### *a) Normal/post-release periods*

New developments, existing code improvements, and contributions of bug fixes take place between releases, including the addition to the repository of modifications that might have been put on hold during the ramp-up to the last release. Such items are developed either by Developers' Committee members or by other contributors who provide them to a member of the Developers' Committee. In either case, it is up to such Developers' Committee members to confirm that the provided code works as advertised 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 Developers' Committee about limiting underlying assumptions or possible code conflicts;
- (4) working with Developers' Committee 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.

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

The Developer's Committee determines the procedures for repository access. Denial of addition to the repository is a decision by the Developers' Committee. The procedure reflects a denial by veto rather than a majority rule decision. Upon any member's veto of the change, the proposed code modification will not occur. In practice, this situation is rare.

##### *b) Pre-release periods*

- (i) Release minus 4 months (R-4m)

At the R-4m point, all code sought to be in the release by developers must be provided by such developers to the release committee point(s) of contact. It is the responsibility of the developers to be in contact with the POCs prior to this point to make sure that the developers meet this deadline.

- (i) Release minus 3 months (R-3m)

At the R-3m point, all of the proposed new features for the major release must be in their initial form in the repository. The Release Committee and the Developers' Committee work with contributors to make sure that the release schedule does not slip to accommodate laggard development efforts. As the release gets nearer, general bug fixes or modifications to the repository to address issues uncovered in testing are still eligible for inclusion. However, large code changes unrelated to the main elements or goals of the release are postponed until after the release.

For items arriving after the 3-month freeze date, the Release Committee makes the determination regarding the late-arriving modifications: delay the release date, remove the feature as candidate for the release, or modify the feature to the satisfaction of the Developers' Committee. The Developers' Committee only entertains proposed repository modifications that are related to features slated for the release that are deficient.

(iii) Release minus 1 month (R-1m)

In this stage the only acceptable changes are those pertaining to testing of the frozen code. Improvements to the frozen code or bug fixes uncovered in testing are allowed into the repository, but only with approval of the Developers' Committee, and only by the person designated by the committee to update the repository in this period. The automated regression tests and the repository updates are handled by this single member of the Developers' Committee.

The Release Committee posts information on the status of the major release. During this last month of testing, final results for additional architectures, case studies, timing studies, and vendor reports are received. The Release Committee reviews the information to ascertain that the release is on schedule or that remedial action is required.