U. S. Department of Commerce
National Oceanic and Atmospheric Administration
National Weather Service
National Centers for Environmental Prediction
5200 Auth Road Room 207
Camp Springs, MD 20746

Technical Note

WAVEWATCH III ® development best practices [†].

Hendrik l. Tolman[‡]
Environmental Modeling Center
Marine Modeling and Analysis Branch

Version 0.1, May 2010

[†] MMAB Contribution No. 286.

[‡] e-mail: Hendrik.Tolman@NOAA.gov

This page is intentionally left blank.

Abstract

This report describes best practices for code development of WAVEWATCH III ®. This includes guidelines for packaging of codes delivered by general users to NCEP according to the WAVEWATCH III license, as well as instructions for co-developers on the use of the subversion depository at NCEP. The report addresses codes, documentation and manuals.

Change log

version	svn rev.	date	comment
0.1	7868	May 14, 2010	Initial MMAB No. 286. No regression testing yet.

Acknowledgments. Code management for WAVEWATCH III is provided by NCEP. Arun Chawla provided a first filter for this report.

This report is available as a pdf file from

http://polar.ncep.noaa.gov/waves

Contents

This page is intentionally left blank.

1 Introduction

The WAVEWATCH III ® wind wave model has a history dating back to the second half of the 1980s. It's history started with the development of the WAVEWATCH model at Delft university of technology (Tolman, 1989, 1990, 1991). The next step of development occurred at NASA, Goddard Space Flight Center in the early 1990s, with the development of WAVEWATCH II. This model was explicitly designed for (vector) super computing, and focused on improved numerics (Tolman, 1992a,b). Development of WAVEWATCH III at NCEP started in 1993. Compared to previous WAVEWATCH models, this model uses modified basic equations, and introduces the present model architecture. This model utilizes vector optimization, together with OpenMP or MPI parallel optimization, and hence can be run efficiently on most modern computer architectures. With model version 3.14, WAVEWATCH III has been trademarked and copyrighted, and has been distributed under an open-source style license (see section 1.2 of Tolman, 2009, or the web site[1]). Henceforth, WAVEWATCH III will be denoted as WW3.

Three public releases of WW3 have been made available (Tolman, 1999, 2002, 2009). This best practices report was first provided with model versions 3.14 for two reasons. First, coding standards are needed to foster and support community model development. This has become particularly important with the National Oceanographic Partnership Program (NOPP) project to improve all basic wave model source terms, which will run from 2010 through 2014, and which will use WW3 as a main development vehicle. This means that several teams will work simultaneously on the WW3 code. Second, there is a need for unifying coding approaches within NCEP. A first set of standards has been developed for the Community Radiative Transfer Model (CRTM) as presented in Van Delst (2008). Whereas it is unrealistic to retrofit all NCEP codes to a completely homogeneous coding standard due to the shear size of legacy codes, all basic precepts should be the same, and are consistent between the present report and Van Delst (2008).

From the beginning, WW3 has been envisioned as a modeling framework, with various options for numerical and physical approaches, both for operational and research applications. With a focus on operations at NCEP, selections of numerical and physical approaches are done at the compile level of the code. This limits the complexity of the source code that is used in operations. For example, complex exact interaction codes are not compiled into the operational models at NCEP, and hence do not need to be maintained in the operational code versions. Compile level code selections are made using the native WW3 preprocessor and 'switches' in the source code, as described in full in the WW3 manual (Tolman, 2009, or more recent versions).

Starting with the release model version 3.14, we are maintaining the code

[1] http://polar.ncep.noaa.gov/waves/wavewatch/license.shtml

using subservion (Collins-Sussmann et al., 2004). The master version of WW3 will be maintained and supported at NCEP. With the licensing of model version 3.14, users that develop code and/or modifications for WW3 are obligated to offer these back to NCEP, relative to the most recent version of WW3 available to them[2]. NCEP will then decide if such modifications and additions will be included in the master version of WW3, and will be responsible for including it in the subversion depository. Alternatively, collaborators on well defined projects and with proven development capability will be considered as 'co-developers', and will be given direct access to our subversion server, and thus to developmental version of WW3. NCEP will invite collaborators to become co-developers, and will considered requests to become co-developers.

In this context, coding standards, best practices for upgrading parts of WW3 and for adding new pieces to WW3, regression testing, and maintenance of documentation are essential. These issues are approached in Sections 2 (also including copyright statements), 3 4, and 5, respectively. Finally, Section 6 discusses standards of code management using the subversion server at NCEP.

Some formatting practices for the WW3 manual are used in this report. The file font will be used to identify files, scripts and command line entries. The `CODE` font is used to identify source code. Previous experience with WW3 is expected, and the use of, for instance, optional switches in the code, will not be explained here in detail.

[2] Public release or research version on svn server, depending on user access.

2 Programming style

WW3 is written in ANSI standard Fortran 90, fully modular, and with an internal dynamic data structure exclusively using use-associated data modules. All modules are internally documented with a style of documentation as illustrated in Fig. 2.1 and 2.2 for subroutines and modules, respectively. Examples of this can be found throughout the source code, and templates are also provided in 'user slot' routines for source terms and propagation schemes.

Is should be noted that WW3 consists of incomplete ('.ftn') FORTRAN files that require standard WW3 preprocessing. All changes and additions should be made in these files, not in extracted true FORTRAN files (for details see the manual).

The following is expected of codes provided to NCEP for inclusion in the official version of WW3:

i) Fully document the code following the outline described above.

ii) Follow the coding style of WW3, in particular :

- For readability, code is written following the use of columns as in fixed format Fortran, even though codes are technically written in free format. Use typical indent strategies for loops and logical structures.
- Code intended as permanent code is written in upper case, temporary (test) code is written in lower case. Note that we encourage the inclusion of permanent test output to be activated at compile time using the WW3 compile switches[3] like '`!/T`'. The latter test output should be coded in upper case as a permanent part of the code.

iii) Maintain an update log at the top of each module and for each individual routine or function, and update the last update date in the header of each module, function and routine, as has been done in the distribution version of WW3. If a module only contains one program element, only a single update log needs to be maintained. This is a legacy from code management before using subversion, but will be retained until further notice.

iv) Each subroutine, function or groupings should be embedded in a module to allow for full use association and internal automatic interface checks in Fortran compilers. File naming conventions include:

[3] See manual for details on use of compile level switches.

- File names for elements of the basic wave model should start with w3.
- Program elements related to the multi-grid capability should start with wm.
- Module file names should end in md (before the file extension).
- Files with main programs should be stored in file names starting with ww3_.
- The file extension.ftn identifies code elements that need to be preprocessed by the WW3 preprocessor w3adc to activate switches.
- Files with ready-to-use source code (no need for the WW3 preprocessor) are identified by the extension .f90. This includes external packages interfaced to WW3.

As examples w3snl2md.ftn is a module of the basic wave model (one of the S_{nl} source term options) that needs to be preprocessed by the WW3 preprocessor. ww3_grid.ftn contains the main program for the grid preprocessing. mod_constants.f90 is a part of a user-supplied package that does not require WW3 code preprocessing. Note that the only file not following the WW3 naming convention is constants.ftn, which contains a module with physical constants.

v) For now, we have been using the Fortran 90 standard. Required coding practices include:

- Use free format with style as described above.
- Use `IMPLICIT NONE` in each module.
- Do not use `COMMON` declarations. Eventually all major data structures should become part of the WW3 dynamical data structures (see manual), which are all contained in separate modules, and can be used by use association. See section 3 for suggestions on how to deal with these data structures during (initial) code development.
- Each module used in a given program element will need to be use associated with a `USE` statement. Where feasible, use

  `USE` *module_name*`, ONLY:` *used names*

  to avoid unintended use of variables in modules.
- For the same reason, use `PRIVATE` for general declarations in modules.
- Declare `INTENT` on all dummy argument list items.
- Do not use tab characters in the code (not in Fortran character set).

- Name `END`s fully both for readability and because several compilers will require this.
- Do not use numbered `DO` loops.
- Use `CYCLE` and `EXIT` instead of `GOTO`.
- Use `CASE` statements with a default rather than `IF` statements for multiple selection tests.
- As a holdover of days long gone, short variable names have been used throughout the WW3 code. Although this makes it easy to keep documentation readable, it does not necessarily make it easy to understand the code at a glance. Feel free to use longer variable names to make the code more easily understandable.
- Up to now, there has been no need for explicit `KIND` declarations in WW3. If such declarations are needed, follow the standard set in Van Delst (2008).

vi) Provide documentation for the modules to be included in the WW3 manual. The manual is written in LaTeX. Required manual elements to be provided are

- Description or update of basic equations / physical parameterizations as needed.
- Description or update of numerical approaches as needed.
- Update of system documentation including description of parameters in the dynamical data structure of WW3.

The coding style does not imply that existing packages that are attached to WW3 need to be re-written in this style. However, it is strongly recommended that any such package should be fully documented inside the code. Typically, a user provided package will require an interface routine to WW3. Such an interface routine is expected to conform to the WW3 coding practices.

Note that the NWS claims copyright for all main elements of WW3, and generally will claim copyright for interface routines. Providers of packages to be included with the distribution of WW3 are encourage to provide copyright statements and disclaimers in these packages as appropriate (as NWS will not claim copyright of such packages).

```
!/ ------------------------------------------------------------------ /
      SUBROUTINE W3XXXX
!/
!/                  +----------------------------------+
!/                  | WAVEWATCH III           NOAA/NCEP |
!/                  |             John Doe             |
!/                  |                         FORTRAN 90 |
!/                  | Last update :         01-Jan-2010 |
!/                  +----------------------------------+
!/
!/    01-Jan-2010 : Origination.                        ( version 4.xx )
!/
!  1. Purpose :
!  2. Method :
!  3. Parameters :
!
!     Parameter list
!     ----------------------------------------------------------------
!     ----------------------------------------------------------------
!
!  4. Subroutines used :
!
!      Name      Type  Module   Description
!     ----------------------------------------------------------------
!      STRACE    Subr. W3SERVMD Subroutine tracing.
!     ----------------------------------------------------------------
!
!  5. Called by :
!
!      Name      Type  Module   Description
!     ----------------------------------------------------------------
!     ----------------------------------------------------------------
!
!  6. Error messages :
!  7. Remarks
!  8. Structure :
!  9. Switches :
!
!     !/S  Enable subroutine tracing.
!
! 10. Source code :
!
!/ ------------------------------------------------------------------ /
!/S      USE W3SERVMD, ONLY: STRACE
!/
      IMPLICIT NONE
!/
!/ ------------------------------------------------------------------ /
!/ Parameter list
!/
!/ ------------------------------------------------------------------ /
!/ Local parameters
!/
!/S      INTEGER, SAVE           :: IENT = 0
!/
!/ ------------------------------------------------------------------ /
!/
!/S      CALL STRACE (IENT, 'W3XXXX')
....
!/
!/ End of W3XXXX ---------------------------------------------------- /
!/
      END SUBROUTINE INSBTX
```

Fig. 2.1 : Documentation template for subroutines. Note that each subroutine is expected to include a call to the `STRACE` *subroutine to enable subroutine tracing inside WW3,*

```
!/ ------------------------------------------------------------------ /
      MODULE W3XXXXMD
!/                  +----------------------------------+
!/                  | WAVEWATCH III           NOAA/NCEP |
!/                  |             John Doe             |
!/                  |                         FORTRAN 90 |
!/                  | Last update :         01-Jan-2010 |
!/                  +----------------------------------+
!/
!/    01-Jan-2010 : Origination.                        ( version 4.xx )
!/
!/    Copyright 2010 National Weather Service (NWS),
!/       National Oceanic and Atmospheric Administration.  All rights
!/       reserved.  WAVEWATCH III is a trademark of the NWS.
!/       No unauthorized use without permission.
!/
!  1. Purpose :
!  2. Variables and types :
!
!      Name      Type  Scope    Description
!     ------------------------------------------------------------
!     ------------------------------------------------------------
!
!  3. Subroutines and functions :
!
!      Name      Type  Scope    Description
!     ------------------------------------------------------------
!      W3XXXX    Subr. Public   ........
!     ------------------------------------------------------------
!
!  4. Subroutines and functions used :
!
!      Name      Type  Module   Description
!     ------------------------------------------------------------
!      STRACE    Subr. W3SERVMD Subroutine tracing.
!     ------------------------------------------------------------
!
!  5. Remarks :
!  6. Switches :
!
!     !/S  Enable subroutine tracing.
!
!  7. Source code :
!/
!/ ------------------------------------------------------------------ /
!/
      PRIVATE
!/
      CONTAINS
!/ ------------------------------------------------------------------ /
      SUBROUTINE W3XXXX
.....
!/
!/ End of w3XXXX ---------------------------------------------------- /
!/
      END SUBROUTINE W3XXXX
!/
!/ End of module W3XXXXMD ------------------------------------------- /
!/
      END MODULE W3XXXXMD
```

Fig. 2.2 : Documentation template for modules. Copyright statement to be adapted as appropriate.

This page is intentionally left blank.

3 Adding to the model

WW3 is designed as a highly plug-compatible code. Source term and propagation approaches can be included as self-contained modules, with limited changes needed to the interface of routine calls in `W3SRCE`, `W3WAVE`, and in the point post-processing programs only. General users can experiment with new approaches in user slots that are provided as dummy model slots like `W3SNLX` in the file w3snlxmd.ftn for the nonlinear interactions. General users are expected to provide these 'user slot' routines to NCEP for inclusion in subsequent versions of WW3, following the instruction in this report and in the documentation of routines like `W3SNLX`. Such codes should be self-contained in the way described below.

When providing a module for a source term like `W3SNLX` or for a propagation scheme the following programming guidelines should be followed:

i) Follow coding guidelines as outlined in the previous section.

ii) Provide a file with necessary modifications to `W3SRCE` and all other routines that require modification.

iii) Provide a test case with expected results.

Furthermore, the module needs to be self-contained in the following way.

i) All saved variables connected with this source term need to be declared in the module header. Upon acceptance as permanent code, they will be converted to the WW3 dynamic data structure.

ii) Provide a separate computation and initialization routine. In the submission, the initialization should be called from the computation routine upon the first call to the routine. Upon acceptance as permanent code, the initialization routine will be moved to a more appropriate location in the code (i.e., being absorbed in ww3_grid or being moved to `W3IOGR`).

When such packages are provided to NCEP, NCEP may choose to not include the package, or to provide the package as a 'user slot routine' like `W3SNLX`, with some minor work of users required to install these routines, or may choose to fully integrate the routines as a standard option in WW3.

Co-developers of WW3 with access to the subversion server are expected to fully integrate the new modules in the experimental versions of WW3, using software selection switches as provided by the NCEP code managers. It is, nevertheless, strongly recommended that initially data structures are kept internal to the modules that are being developed, and that data for the modules are only included

in the dynamic data structure of WW3 when the module is mature. This will make code development and unification much easier when multiple developers are working on the code simultaneously.

The above approach are applicable to inherently modular elements of WW3 such as source terms or propagations schemes. For more intricate changes to the code, please consult the WW3 code managers[4] on how to proceed with developing and providing code upgrades.

[4] Mail to NCEP.EMC.wavewatch@NOAA.gov

4 Regression testing

Regression testing standards have not yet been defined, and this section is presently a placeholder only.

This page is intentionally left blank.

5 Manual and documentation

The WW3 manual and other WW3 documents like this report are written in LaTeX. Since these are dynamic documents, the corresponding files are maintained in svn, together with the WW3 source code, script and auxiliary files. Because the manual is rather large, it has been stored in several .tex files. The main files making up the manual are

manual.tex	Main .tex file, mainly combining the .tex files below into the complete manual.
defs.tex	User defined LaTeX constructs used in the manual.
start.tex	Title page and table of contents set up.
intro.tex	Chapter: Introduction.
eqs.tex	Chapter: Governing equations.
num.tex	Chapter: Numerics.
run.tex	Chapter: Running the model.
impl.tex	Chapter: Installing the model.
sys.tex	Chapter: System documentation.
more.tex	Appendix: Managing multiple model versions.
tstep.tex	Appendix: Setting time steps.
nest.tex	Appendix: Nesting.
mpi.tex	Appendix: Compiling MPI versions of the model.
move.tex	Appendix: Moving grid options.
fig_*XXXX*.tex	Various figures made directly using LaTeX.
inp_*XXXX*.tex, inpg_*XXXX*.tex	Example input files for WW3 programs used in run.tex.
XXXX.eps	Various encapsulated postscript graphics.
manual.bib	BibTex database with references used in the manual.
jas.bst	Bibliography style file used for the manual.

Apart from the files making up the manual, a support script is provided:

make_inps.sh	Convert input files for WW3 programs to LaTeX file for use in the manual (e.g., convert the model input file ww3_grid.inp to inp_grid.tex). This script assures that the example input files provided with the code are the files displayed in the manual.

Note that the manual consist of both a conventional manual and a basic system documentation. The following standards should be used in writing LaTeX contributions to the manual:

- Use American spelling and grammar.
- Use dynamic references to equation, chapter and section numbers, etc. Do not use any hardwired reference numbers when referring to equations, sections etc.
- Use BibTex exclusively for references to other work. Do not write any references directly into the text.
- Do not use excessive line lengths in the .tex files. We typically use a maximum line length of 78 characters and 'auto-fill-mode' when writing or updating .tex files using emacs.
- When adding contributions to the manual, add a note of the update to the introduction, so that users of the public releases have a concise log of upgrades since the previous model release.
- If you have no LaTeX capability or experience, contact the WW3 code managers to determine an acceptable method of delivering contributions to the manual.

For general users we will provide a recent manual package when they are ready to provide their manual contributions. For co-developers, the most recent version of the manual will be available on the svn server.

WARNING

This guide and other LaTeX WW3 documents like the manual use the svn package for LaTeX. This package is generally not automatically installed with LaTeX and therefore might result in failure of compiling the .dvi files. The package is available from the CTAN web site (http://www.ctan.org).

WARNING

6 Subversion repository

Starting with model version 3.14, WW3 in maintained using subversion (Collins-Sussmann et al., 2004). All EMC codes are either on,or are being transferred to the EMC subversion server

https://svnemc.ncep.noaa.gov

Access to this server requires an account and password. The WW3 model is maintained in

https://svnemc.ncep.noaa.gov/projects/ww3

In the WW3 directory, the conventional trunk, branches and tags directories have been created, The trunk directory contains the main model development, tags contains model releases (formal, internal and beta testing), and branches contains work space for individual developers (as well as maintenance of released versions). For instance, Hendrik's work space is identified as

https://svnemc.ncep.noaa.gov/projects/ww3/branches/hendrik

Co-developers will get read access to the trunk and tags, and write access to their designated directory in branches. In trunk (and tags), we have set up directories

docs	Work space for shared manuscript writing.
model	Model files as previously distributed as .tar files.
manual	LaTeXfiles and graphics files for the manual.
guide	LaTeXfile for this guide.
utilities	Additional utilities such as the grid generation package.

In the model directory, subdirectories aux, bin, ftn, inp and test are created, containing the same files as in the previous model distributions. To install WW3 from the subversion repository, a new script

install_ww3_svn

has been created, which replaces the script install_wwatch3 used to install he model from .tar files. This script will first fill the five directories svn/aux through svn/test with all the files from the svn repository, and will then continue to set up the model identical to the model setup from the tar files. The only difference is that distribution files in, for instance the bin directory, now are linked to the actual files in the svn/bin directory. Thus,

called from the svn directory will update the repository. Note that this implies that new files need to be added in the svn directory first, and then need to be added as links in the conventional directory. The script install_ww3_svn can be used for either an initial install, for updating the working copies from the svn repository, and for updating link to the local svn directory.

The script install_ww3_svn needs to reside in the main WW3 directory. Upon first install, this script can be pulled from

trunk/model/bin/install_ww3_svn

If this script is found to be identical to the local work copy in the local svn/bin directory, it will be replaced with a link to the latter, so that modifications to the script will also be version controlled with svn.

Co-developers will have read access to the trunk and tags, and will have read and write permission to their workspace in the branches. Read access to the latter work space will be set as requested by the co-developer. NCEP code managers will have read access throughout, and will be responsible for merging mature codes into the trunk. As mentioned above, co-developer will be responsible for providing mature upgrades relative to the most recent upgrade of the trunk.

To fully use the potential of subversion, it is critical that detailed commit logs are maintained. As with the CRTM, WW3 commit logs should follow the GNU ChangeLog format[5]. An example of a log entry is given in Fig. 6.1. The log entry should mention each file that has been changed. The first line of each block should contain the subdirectory name. The log entry should mention every file that has been changed. For every file, every procedure changed should be named in full (no wildcard) to enable searching the log. We are aware that some duplicity is introduced by also asking for a change log entry in the actual source files. The latter is typically only a one-line cryptic description. For now, this change log will also be maintained because it refers to the WW3 version number. If the commit logs are properly maintained, the ChangeLog will be provided with future releases, at which time we may discontinue the habit of providing simple change logs in the source files.

Finally, we are using Trac[6] as a web-based management tool for the development of WAVEWATCH III. The Trac pages are found at

https://svnemc.ncep.noaa.gov/trac/ww3

[5] http://www.gnu.org/prep/standards/standards.html#Change-Logs
[6] http://trac.edgewall.org/

```
ww3/branches/hendrik/workcopy/model/aux subdirectory
 * spec_ids: File removed, as it was eroneously added to the repository with
             the initial import of model version 3.14.

ww3/branches/hendrik/workcopy/model/bin subdirectory
 * install_ww3_svn: New script to install wave model from svn reporitory.

ww3/branches/hendrik/workcopy/guide subdirectry (new subdirectory)
 * report.tex: the latex file with the guide
 * report.bib: bibtex bibliography information for guide

ww3/branches/hendrik/workcopy/model/ftn subdirectory
 * w3sbtxmd.ftn: added copyright statement
   (w3sbtx): cosmetic changes
   (insbtx): cleared typos from documentation
```

Fig. 6.1 : Example of commit log entry following the GNU ChangeLog format.

and are accessed with the user name and password of the svn pages. The front page is a wiki page. Trac gives a web-based way to access files in subversion, including a time line of submissions to subversion. Additional tools include a road map with milestones, and a ticket system. We intend to use this system to manage code development for WAVEWATCH III, and possibly as the beginning of a user forum. Trac will be accessible to all those with accounts for our subversion server, and is presently being set up by the NCEP managers.

This page is intentionally left blank.

References

Collins-Sussmann, B., B. W. Fitzpatrick and C. M. Pilato, 2004: *Version control with subversion*. O'Reilly, 320 pp.[1]

Tolman, H. L., 1989: The numerical model WAVEWATCH: a third generation model for the hindcasting of wind waves on tides in shelf seas. Communications on Hydraulic and Geotechnical Engineering 89-2, Delft Universtity of Technology, ISSN 0169–6548, 72 pp.

Tolman, H. L., 1990: Wind wave propagation in tidal seas. Communications on Hydraulic and Geotechnical Engineering 90-1, Delft Universtity of Technology, ISSN 0169–6548, 135 pp. (Doctoral Thesis).

Tolman, H. L., 1991: A third-generation model for wind waves on slowly varying, unsteady and inhomogeneous depths and currents. *J. Phys. Oceanogr.*, **21**, 782–797.

Tolman, H. L., 1992a: Effects of numerics on the physics in a third-generation wind-wave model. *J. Phys. Oceanogr.*, **22**, 1095–1111.

Tolman, H. L., 1992b: Effects of the Gulf Stream on wind waves in SWADE. in *Proc. 23rd Int. Conf. Coastal Eng., Venice, Italy*, pp. 712–725. ASCE.

Tolman, H. L., 1999: User manual and system documentation of WAVEWATCH III version 1.18. Tech. Note 166, NOAA/NWS/NCEP/OMB, 110 pp.

Tolman, H. L., 2002: Testing of WAVEWATCH III version 2.22 in NCEP's NWW3 ocean wave model suite. Tech. Note 214, NOAA/NWS/NCEP/OMB, 99 pp.

Tolman, H. L., 2009: User manual and system documentation of WAVEWATCH III $^{\text{TM}}$ version 3.14. Tech. Note 276, NOAA/NWS/NCEP/MMAB, 194 pp. + Appendices.

Van Delst, P., 2008: CRTM: Fortran95 coding guidelines. Technical report, Joint Center for Satellite Data Assimilation.

[1] Updated versions available online at http://subversion.tigris.org/.

This page is intentionally left blank.