Modular structure of WAVEWATCH III and general features

Arun Chawla

The WAVEWATCH III Team + friends
Marine Modeling and Analysis Branch
NOAA / NWS / NCEP / EMC

NCEP.list.WAVEWATCH@NOAA.gov
NCEP.list.waves@NOAA.gov

Outline

Covered in this lecture:

- Modifying code.
- Internal data structure.
- Best practices.

Modifying code

Code may need to be updated for bug fixes, or as part of systematic model development.

- For simple edits our preferred way to work is:
  - Use ***ln3*** to make a link to the file in the ./work directory under the main wave model directory.
  - Edit the link in the ./work directory, and test there with ***w3_make*** and by running standard tests.
  - Note: there is a link to the switch file in this directory to modify the model configuration.
  - After the modification is satisfactory, remove the links from the ./work directory.
- HINT: use ***arc_wwatch3*** to make archive files before and after code modification, if no other management tool like subversion is used. The resulting ***.tar*** files can be re-installed with ***install_wwatch3***.

Manual section 5.5

Modifying code

- If systematic modifications or additions to the code are made, there will likely be a need for:
  - Adding subroutines in existing modules.
  - Adding subroutines in new modules.
  - Adding old switches to existing subroutines.
  - Adding new switches to the model.

- These actions will be discussed in the following slides, and are also described in section 5.5 of the manual.
- Note that if a new module with new switches is included, instructions for both modifications need to be followed.

- See HINT on previous slide …..

Manual section 5.5

Modifying code

Adding subroutines in existing modules.

- This is in principle simple. Add the code and recompile using ***w3_make***.

- A complication may occur if the subroutine is used by other modules. In that case, the proper "use" statement needs to be added to the calling module.
  - This may modify relations between files in the makefile and ***make*** commands.
  - Run ***make_makefile.sh*** manually to assure that the makefile is updated, before ***w3_make*** is run.
  - This only needs to be done if "use" statements are modified.

Manual section 5.5

Modifying code

Adding subroutines in new modules.

- This typically adds a new file like w3coolmd.ftn or mypackage.f90 to the model files.
- To assure that the new files are included in the compilation, ***make_makefile.sh*** needs to be modified as follows:
  - Add module name to sections 2.b and 2.c to assure inclusion in the makefile under proper conditions.
  - Add module name and object file names to section 3.b to assure proper dependencies in makefile.
  - Run ***make_makefile.sh*** manually and check makefile in ./ftn directory for proper inclusion of new file.

- NOTE: ***make_makefile.sh*** checks use statements in .f90 (preprocessed) files to determine file dependencies.

Manual section 5.5

Adding old switches to existing subroutines.

- Relationships of switches to model files are maintained in the ***w3_new*** script.
- If old switches are added to new files the following actions are needed:
  - Add the new file to the lists of files to be touched in section 2 of ***w3_new***.
  - If the switches include use statements, interactively run ***make_makefile.sh*** to assure that the makefile is updated as needed.

Manual section 5.5

Adding new switches to the model.

- After a new switch is added to an existing file, the following action is required.
  - If the switch is part of a new group of switches of which one is to be selected, add a new ‘keyword’ ($key) to section 2 of ***w3_new***.
  - Update files to be touched in section 2 of ***w3_new*** as necessary.
  - Add ‘keyword’ and/or switches to section 2 of ***make_makefile.sh***.
  - Run ***make_makefile.sh*** and check consistency of ./ftn/makefile.

Manual section 5.5

Data structures

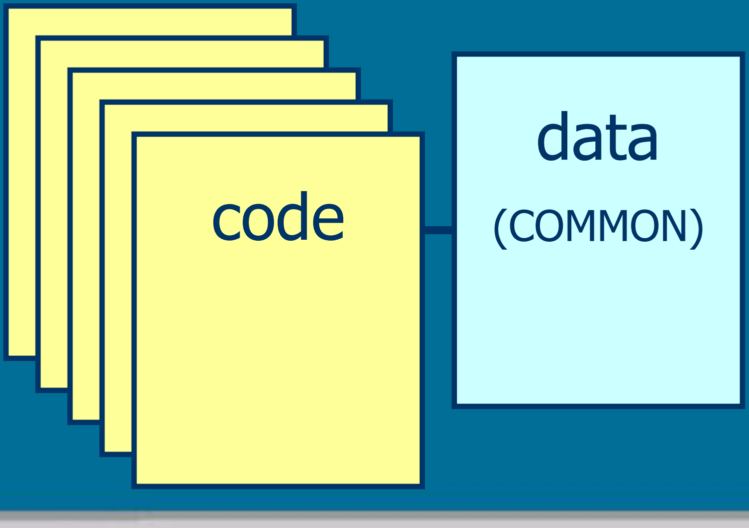
When adding to the wave model, it is essential to understand how data is stored.

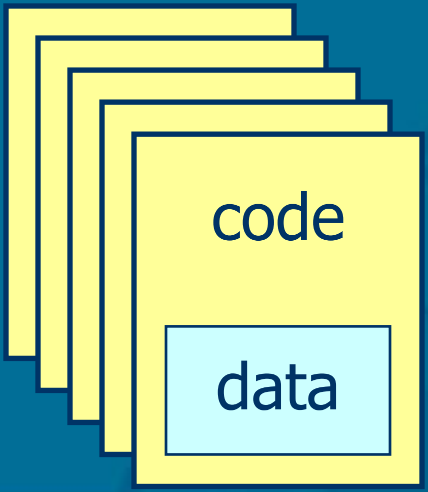
Model version 1.18 (1999)

- FORTRAN 77
- COMMON data structure
- Single static data structure.

Model version 2.22 (2002)

- FORTRAN 90
- Modular
- Object oriented, static data structure bundled with code

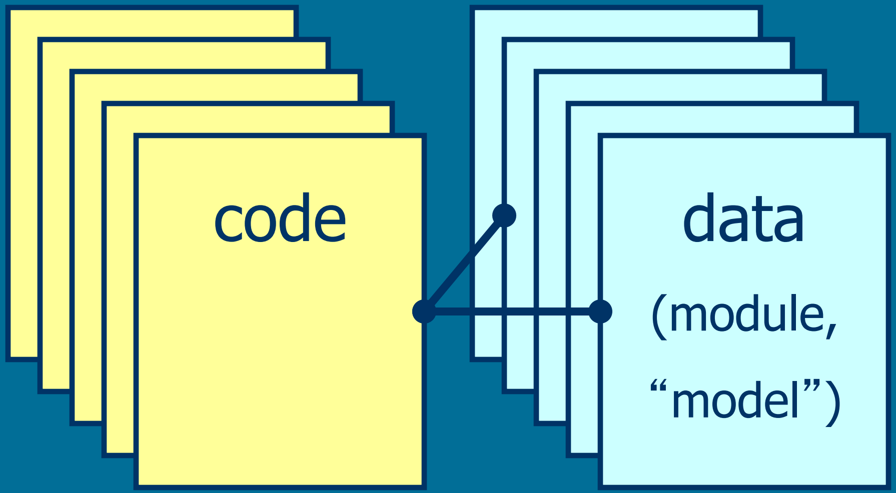




Data structure

Model version 3.06 (2005)

- Modular FORTRAN 90
- Dynamic / multiple data structure (modular)
- Small overhead (7% on Linux, 2% on IBM SP)



Present status :

- F77 and COMMON data structures are obsolete.
  - Exceptions are aux codes like w3adc.f.
- Data embedded in modules largely obsolete.
  - Use in model development, see best practices.
- Data in data modules now the norm in 3.14.
  - Exception: file constants.ftn.

Data structures

How is this done?

- Inside the code variables look like they are defined for a single grid, for instance, the grid dimensions `NX,NY`, and a bottom depth array `ZB`.
- However, these variables are declared as pointers.
- The actual data is stored in a user-defined type `GRID`.
- An array of `GRIDS` of this type allows for data of multiple grids to be stored simultaneously.
- The pointers are then set to represent values of the grid currently under consideration.

```
!/
!/ data structure
!/
      TYPE GRID
        INTEGER       :: NX, NY
        REAL, POINTER :: ZB(:,:)
      END TYPE GRID
!/
!/ Data storage
!/
      TYPE(GRID), TARGET,                  &
                 ALLOCATABLE :: GRIDS(:)
!/
!/ Pointers
!/
      INTEGER, POINTER :: NX, NY
      REAL, POINTER    :: ZB(:,:)

!/ ***********************************

      NX => GRIDS(I)%NX
      NY => GRIDS(I)%NY
      ZB => GRIDS(I)%ZB
```

Data structures

- There are many data structures defined in the model.
- All essential model data for model setup as well as dynamic wave conditions is stored in five data modules:

w3gdatmd.ftn	Grid and model setup data.
w3adatmd.ftn	Auxiliary data used and stored internal to the model only.
w3idatmd.ftn	Model input data.
w3wdatmd.ftn	Basic wave model state.
w3odatmd.ftn	Model output data.

- Each module contains data for as many grids as identified in the mosaic (including model input and spectral point output grids).

Manual sections 6.5 & 6.6

Best practices

- For those who want to modify / contribute to WAVEWATCH III, a best practices guide is available.
- Note that as a part of the license, additions made to the model have to be offered to NCEP for inclusion in future model distributions.
- Best practices cover :
  - Programming style
  - Adding to the model.
  - Manual and documentation.
  - Subversion repository.
  - Regression testing.
- These issue will be touched upon briefly here, but the guide will be the authoritative source.

Best practices guide

Programming style:

- Use WAVEWATCH III documentation style (see templates).
- Use coding style:
  - Free format but layout as in old fixed format.
  - Upper case for permanent code, lower case for temporarily code. Latter can be included as permanent testing using `!/Tn` switches.
- Maintain updated log at header of documentation.
- Embed all subroutines in modules or main programs, using naming convention outlined before.
- Follow FORTRAN 90 standard, with best practices as outlined in section 2 of the guide.
- Provide appropriate documentation in LaTeX format for inclusion in the manual.

Best practices

subroutine template

```
!/ ------------------------------------------------------------------- /
      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
!     ------------------------------------
!
!  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
```

Best practices

module template

```
!/ ------------------------------------------------------------------- /
      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
!     ------------------------------------
!
!  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
```

Programming style cont’ed:

- If existing packages are added to the wave model, then such packages do not need to be re-coded to conform to our standards.
- Such packages will require interface routines, which are expected to confirm to the standards.
- Copyright of NWS may extend to interface routines, but obviously not to linked in packages.

Adding to the model
(no NCEP subversion access)

- Propagation schemes and source terms:
  - Use available “user-defined” dummy modules.
  - Follow coding guidelines.
  - Provide file with necessary modifications to w3srcemd.ftn, w3wavemd.ftn, and all other model files that need to be updated.
  - Provide (previous) test cases with expected results.
  - Make each module self-contained.
    - Define all variables in the module header. We will integrate them in the full data structure.
    - Separate initialization and computation as outlined in the dummy modules.

Adding to the model (no NCEP subversion access)

- For more intricate modifications to the code, consult with NCEP code managers on how to do this and on how to provide this to NCEP.

- New pre- or postprocessors should be provided in their entirety, included in the compile and link system.

- HINT: when developing new source terms, include and test them in the postprocessors ***ww3_outp*** and ***gx_outp*** first, before including/testing them in actual wave model integration.

Adding to the model (with NCEP subversion access)

- Same rules apply as for those without svn access with following exceptions:
  - NCEP code managers will assign switches to new sources and propagation scheme to be used instead of the ‘X’ switches.
  - Developers will integrate the data structure:
    - Only after rigorous testing of self-contained system.
  - Changes to be provided relative to most recent `TRUNK`.
  - NCEP code managers will add new code to the `TRUNK` of the repository.
    - E-mail notification to co-developers.

Manual and documentation.

- Provide full LaTeX documentation for inclusion in the manual:
  - NCEP svn users have access to manual, and are expected to add to it directly.
    - NCEP will provide editing.
  - Others provide separate files.
    - NCEP will integrate.
  - Use BibTEX.
  - Use dynamic references to equations, figures and tables only.

The end

End of supplemental material