Chapter 21

Recommendations for programming

The model setup requires that FORTRAN 90 program code is inserted in several *Usrdef_*files. Below are a few recommendations and hints.

declaration of variables All variables used within a routine must be declared. For local variables this is easily done within the routine. Although not required explicitly, the COHERENS code is entirely programmed without "implicit typing rules". This means that the local variable declaration part should start with the line

```
IMPLICIT NONE
```

Most variables within a usrdef_ routine have a non-local scope and are already declared in a MODULE file. They are made accessible in the usrdef_ routine via a USE X statement at the head of the routine where X is the name of the module. The name of a module always appears on the first line of the a module file. These are the files in the **source** directory whose names start with a lower case character and contain no "_" (underscore) character. The file names have a close relation with their contents (like *currents.f90* which contains the modules with all declarations concerning currents). A description of all program variables is given in Chapter 33. Module routines used within a usrdef_ routine need to be declared with a

```
USE X, ONLY: Y
```

statement where X is the name of the module and Y the name of the module routine. Module routine file names start with a lower case

letter and have an "_" character in their name. The routine name can be found within the routine. A full description of modules and module routines is found in Chapters 33 and 31. The following is an example of USE statements

```
USE currents
USE inout_routines, ONLY: close_file, open_file
```

file operations Important to know is that files cannot be opened or closed with the standard FORTRAN OPEN and CLOSE statements. There are two alternatives, based upon routines declared in *inout_routines.f90*:

1. Use open_filepars to open and close_filepars to close files. The first argument of these routines is a derived type variable of type FileParams. Besides opening and closing, the routines set or reset a number of file attributes.
2. Use open_file and close_file. The routines are similar to the classical OPEN and CLOSE FORTRAN routines.

The syntax of these routines is explained in Chapter 31.

time series data A special procedure needs to be followed within a usrdef_ routine which reads time series data (i.e. the routines having ciodatetime as an argument).

1. On first call, the file is opened and the iostat attribute is reset from 0 to 1. No data are read.
2. A second call is made, immediately after the first one (i.e. at the same model time step) to read the date/time and first series of data. These calls are repeated until the date/time of the last input is later than the actual model time.
3. When during program execution, the model date equals or becomes later than the one obtained from the last input, the usrdef_ routine is called again, until the date/time in the file becomes later than the actual model time.
4. If an end of file condition occurs or the user knows that this will occur on a next read, the user must set the iostat number to 2. The program will no longer call the routine. Further action (decided by the program) now depends on the value of the endfile attribute (see Section 14.7.2).

5. At the end of the simulation the program automatically closes all open files whose attributes are stored in an element of the array modfiles. Otherwise, the user has to close the file when the last data record has been read.

The first point can be programmed as

```
IF (modfiles(iddesc,ifil,1)%iostat.EQ.0) THEN
   CALL open_filepars(modfiles(iddesc,ifil,1))
   RETURN
ENDIF
...
```

if the file exists, or

```
IF (modfiles(iddesc,ifil,1)%iostat.EQ.0) THEN
   modfiles(iddesc,ifil,1)%iostat = 1
   RETURN
ENDIF
...
```

if the file does not exist and the data are defined without reading from an external data file. The third point could be implemented as follows

```
   iunit = modfiles(iddesc,ifil,1)%iunit
   READ (iunit,..,END=99)
   ...
99 modfiles(iddesc,ifil,1)%iostat= 2
   ...
```