

PRISM
Program for Integrated Earth System Modelling



PRISM Support Initiative
NEMO adaptation to PRISM Standard
Compiling Environment

J. Ghattas
CNRS/CERFACS

PRISM Report Series # 22

November, 4th 2005

Contents

1	Overview	1
2	Original structure for NEMO	1
3	First method : Without submodels	1
4	Second method : With submodels	4
5	Bibliography	6

1 Overview

This document describes two methods of adapting the ocean model NEMO (alias OPA9) consisting of OPA, LIM and TOP into the PRISM Standard Environment, SCE.

Major changes for both methods are the way of creating the compile scripts and makefiles. In both cases the configurations, consisting of different cpp key combinations can be set in one file.

- In the first method NEMO is implemented as one model containing OPA, LIM and TOP. Due to tools for creation of makefiles the middle directory level has been removed. Include files used by sources in more than one directory have been removed to a new include directory. No changes in the model source code is necessary.
- For the second method NEMO is separated into the main model OPA and the submodels LIM and TOP. Major changes in the source code have to be done.

2 Original structure for NEMO

```
- NEMO/LIM/  
  /OPA/  
    DIA/  
    DOM/  
    DTA/  
    DYN/  
    FLO/  
    LDF/  
    OBC/  
    SBC/  
    SOL/  
    TRA/  
    TRD/  
    ZDF/  
  /TOP/  
    SMS/  
    TRC/
```

Each directory contains both fortran sources *.F90/*.F and include files *.h90/*.h. The directories LIM, OPA and TOP contain source code as well as directories.

3 First method : Without submodels

The first method describes how to add NEMO to the PRISM SCE as one model without any submodels. The sea ice and the tracers, LIM and TOP, are included directly in the model source code. To use this method a script *install_opa2prism* was written.

The source directory structure in SCE

```
- prism/src/mod/NEMO/
    LIM_SRC/  OPA_DIA/  OPA_DOM/  OPA_DTA/
    OPA_DYN/  OPA_FLO/  OPA_LDF/  OPA_OBC/
    OPA_SBC/  OPA_SOL/  OPA_OCE/  OPA_TRA/
    OPA_TRD/  OPA_ZDF/  TOP_SMS/  TOP_TRC/
    TOP_TRP/  include/
```

Each source directory contains the source files *.F90/*.F and the include files *.h/*.h90. Changes from the original structure are mainly removal of the second directory level (OPA, LIM, TOP) and renaming of directories. The prefix LIM_, OPA_ and TOP_ are added in front of the directory names to indicate which original tree branch the directories come from. Three new directories have been created, LIM_SRC, OPA_OCE and TOP_TRP containing the sources who reside originally directly in the directory LIM, OPA and TOP respectively. An include directory has been created for the files included by more than one directory.

All include files must reside in the same directory as the code from where they are included. If an include file is included by modules from more than one directory the include file must be moved to the directory prism/src/mod/NEMO/include, see table 3 for files concerned. If desired an include directory can be created below each source code directory. In this directory all include files used only in the source code above should reside. All include files must end with the suffix .h or .h90. These restrictions for the include files are due to the tool Append_dependencies which is used to find and append the rules of dependencies for the makefiles. If an include file is not in the correct directory or it is named with a bad suffix it will not be found by the dependency check.

It is not possible to have a deeper level, a third level, of code directories, also due to the tool Append_dependencies. Another tree level would implicate a more complex dependency check which is not developed in PRISM. Therefore in the original NEMO structure one tree level is aborted.

Creating and running compile scripts

For creation of compile script some new files were created, see table 1, and some files already in the SCE needed to be modified, see table 2.

A Makefile_1 is added to each source code directory. The Makefile_1 can be a copy of the same file for all directories except OPA_OCE where the main program exists. With the script Append_dependencies a Makefile is created for each source directory. The first part in Makefile will be a copy of Makefile_1 and a second part with all dependency rules will be added. The user do not need to run Append_dependencies if he doesn't add any new code to the model. In the later case the Makefiles have to be recreated with the call to Append_dependencies in each subdirectory. The Makefile_1 is not changed.

The compile scripts are created for the libraries and the model respectively by launching:

```
./Create_COMP_libs.frm
./Create_COMP_models.frm nemo "NONE" "" "" "" "" expid "nemo"
```

The user chooses the name for the experiment expid which will be a part of the executable's name. NONE implicates that no coupler will be used (can be set as default value). If the user wants to change in the default setup he can enter the new compile script COMP_nemo_expid.rhodes. The options of configurations with the corresponding cpp keys are ORCA2_LIM and GYRE (more configurations can be added) or the user can specify her own configuration with corresponding cpp keys. To start the compilation COMP_nemo_expid.rhodes is launched. This script will first do a call to the compilation script for the libraries and then start compiling NEMO. The executable will have the name nemo_expid.x.

File name	Description
prism/util/compile/frames/include/ Libraries_NEMO.h	List of source directory names, libraries and main program
prism/util/compile/frames/include/ Print_par_NEMO.h	Check for error in setup specific to NEMO
prism/util/compile/frames/include/ Guispecif_NEMO.h	Choice of model configuration, compile option, list of possible coupled models
prism/util/compile/frames/include/ Cppflags_NEMO.h	List of cpp flags for different configurations
prism/util/compile/frames/include_rhodes/ Compile_mode_NEMO_rhodes.h	List of compile options
prism/src/mod/NEMO/src_DIR/ Makefile_1	First part of the Makefile for all source directories (<i>src_DIR</i>) except OPA_OCE
prism/src/mod/NEMO/OPA_OCE/ Makefile_1	First part of the Makefile for directory OPA_OCE, contains linking and creation of executable

Table 1: List of new files needed in PRISM SCE

File name	Description
prism/util/compile/frames/include/ Guispecif_all.h	User defaults for keys message_passing, coupler and use_key_noIO
prism/util/compile/frames/include/ Guispecif_models.h	Changed target mode to forced
prism/util/compile/frames/ Create_COMP_models.frm	Added NEMO in list of PRISM models
prism/util/compile/ Append_dependencies	Modified to treat include files with suffix .h90

Table 2: List of files needed to be modified in PRISM SCE

domzgr_substitute.h90	trctl.npzd.h	trclsm.hamocc3.h
ldfdyn_substitute.h90	trctl.p3zd.h	trclsm.lobster1.h
ldfeiv_substitute.h90	trctl.pisces.h	trclsm.npzd.h
ldftra_substitute.h90	trcini.hamocc3.h	trclsm.pisces.h
obc_vectopt_loop_substitute.h90	trcini.lobster1.h	vectopt_loop_substitute.h90
passivetr_substitute.h90	trcini.npzd.h	zdf.matrixsolver.h90
trctl.hamocc3.h	trcini.pisces.h	zdf.matrixsolver.vopt.h90
trctl.lobster1.h	trclsm.age.h	zdfddm_substitute.h90

Table 3: Include files moved to directory prism/src/mod/NEMO/include/

IPSL comment

This method was not chosen by IPSL global climate modelling group for at least the following reasons :

- The list of files to be moved into directory prism/src/mod/NEMO/include is a mix of routines describing the domain (domxxx), routines describing lateral and vertical diffusion parametrisation (ldfxxx, zdfxxx), routines describing open boundary condition (obcxxx), and routines describing tracer transport and biogeochemistry. For NEMO developers, it is essential to separate the code used by the different physical processes in the model.
- The system of include files is too complex; it is difficult to include a new model or a new configuration of a model because it is not straightforward to identify the files that have to be modified or added.

4 Second method : With submodels

The second method describes how to add NEMO into PRISM separated into 1 main model and 2 submodels named OPA, LIM and TOP.

Source directory structure in SCE

```
- prism/src/mod/OPA/DIA/
    DOM/
    DTA/
    DYN/
    FLO/
    LDF/
    OBC/
    SBC/
    SOL/
    OCE/
    TRA/
    TRD/
    ZDF/
    include/

- prism/src/mod/LIM/SRC/

- prism/src/mod/TOP/TRP/
    SMS/
    TRC/
    include/
```

As in method 1 each source directory contains the source files *.F90/*.F and the include files *.h/*.h90. No include files used by more than one model or submodel is allowed. The files in OPA/include/ can be used by the whole OPA model but not in LIM nor in TOP. The files in TOP/include can only be used within TOP.

The compilation starts with the libraries, thereafter the submodels LIM and TOP. Finally the main model OPA will be compiled and an executable created. The main difference when defining submodels is that the submodels must be compiled first and must not depend on modules in the main model. All information shared between the submodels and the main module must be passed as arguments in subroutines and not by use of modules.

Creating and running compile scripts

As in method 1 a Makefile_1 is added to each source code directory and the Append_dependencies is used to create the Makefiles for each source directory.

To compile, one compile script for each submodel is needed and one for the libraries. The compile scripts are created by

```
./Create_COMP_libs.frm "" "" ""
./Create_COMP_models.frm OPA "NONE" "" "" "" expid "OPA LIM TOP"
./Create_COMP_models.frm LIM "NONE" "" "" "" expid "OPA LIM TOP"
./Create_COMP_models.frm TOP "NONE" "" "" "" expid "OPA LIM TOP"
```

File name	Description
prism/util/compile/frames/include/ Libraries_OPA.h Libraries_LIM.h Libraries_TOP.h	List of source directory names, libraries and main program for OPA, LIM and TOP respectively
Print_par_OPA.h Print_par_LIM.h Print_par_TOP.h	Check for error in setup specific to OPA, LIM and TOP respectively
Guispecif_OPA.h Guispecif_LIM.h Guispecif_TOP.h	Choice of compile option, list of possible coupled models
Cppflags_OPA.h Cppflags_LIM.h Cppflags_TOP.h	Include, in all 3 files, of Cppflags_opalimtop.h
Cppflags_opalimtop.h	Choice of configuration, list of cpp flags for different configurations. NB! This is a common file for the 3 models OPA, LIM and TOP
prism/util/compile/frames/include_rhodes/ Compile_mode_OPA_rhodes.h Compile_mode_LIM_rhodes.h Compile_mode_TOP_rhodes.h	List of compile options
prism/src/mod/OPA/Makefile_1 prism/src/mod/LIM/Makefile_1 prism/src/mod/TOP/Makefile_1	First part of the Makefile for all source directories. Will differ slightly for OPA, LIM and TOP

Table 4: List of new files in PRISM SCE

To start the compilation the newly created script COMP_OPA_expid.rhodes is launched. This script will then launch the 3 other compile scripts before it will compile OPA. When compiling OPA, linking will be done and the executable created.

If the user wants to change the configuration, the options ORCA2 LIM and GYRE will be available or the user can define her own configuration. This is done in the file Cppflags_opalimtop.h which is a common file for the OPA, LIM and TOP. If the configuration is changed the compilation scripts must be recreated.

To be done for separation

To separate LIM and TOP as 2 submodels there must be some changes done in the source code. No use of modules from OPA is possible in LIM and TOP. Instead everything needed in the submodels must be passed as arguments in subroutines. A way to proceed would be to create a startup module in LIM and TOP respectively which passes all information by arguments from OPA to the submodel, everything needed by the submodel as for example dimensions and fields. Inside the submodel this module will be used instead of the OPA modules.

The issue of submodels has been studied by C. Le Quere, and others, to prepare the PRISM demo based on OPA+TRC+PISCES. The work has so far not been finished.

IPSL comment

This method was not chosen by IPSL global climate modelling group because it is important for us to keep dynamic part of ocean modelling and transport of oceanic tracers into one model.

Furthermore, another example of the complexity of the SCE include file system is illustrated here: the addition of a new computer implies the addition of one file per model even if nothing specific to the

model needs to be specified (e.g. in directories prism/compile/frames/include_rhodes, the files Compile_mode_OPA_rhodes.h, Compile_mode_LIM_rhodes.h and Compile_mode_TOP_rhodes.h are identical and not specific to OPA, LIM or TOP).

5 Bibliography

Legutke, S., V. Gayler, 2005: The PRISM Standard Compile Environment Handbook. PRISM Report Series, No 4

Demory, M-E., 2004: The IPSL_CM4 coupled model adaptatiion guide. PRISM Report Series, No 9