



OASIS3-MCT tutorial

All the documentation about the coupler OASIS3-MCT can be found on the OASIS web site at <http://oasis.enes.org> and in the OASIS3-MCT sources in the `oasis3-mct/doc` directory.

The current OASIS3-MCT coupler uses internally the Model Coupling Toolkit (MCT) developed by the Argonne National Laboratory (<http://www.mcs.anl.gov/mct>) to perform parallel regridding and parallel exchanges of the coupling fields.

A. Extracting OASIS3-MCT sources

Go to the OASIS web site to register and download the sources at:
<https://verc.enes.org/oasis/download/oasis-registration-form>

- If you download directly the `oasis3-mct.tar.gz` file from the web, just gunzip it and untar it
- If you use the command `svn checkout` from the web site, create a directory (`mkdir oasis3-mct`), go into this directory and execute the `svn checkout`

B. Compiling OASIS3-MCT and running an empty "tutorial" toy model

1. To compile the OASIS3-MCT coupler:
 - Go into directory `oasis3-mct/util/make_dir`
 - Adapt the "make.inc" file to include your platform header makefile
 - Adapt the value of `$COUPLE` and `$ARCHDIR` in your platform header makefile
 - Type "`make realclean -f TopMakefileOasis3`" and then "`make -f TopMakefileOasis3`"
 - The libraries "`libmct.a`", "`libmpeu.a`", "`libpsmile.MPI1.a`" and "`libscrip.a`" that need to be linked to the models are available in the directory `$ARCHDIR/lib`
2. To compile the tutorial models:
 - Go into directory `oasis3-mct/examples/tutorial`
 - Type "`make clean; make`" (note that the Makefile in this directory automatically includes your OASIS3-MCT header makefile – see the first line in the Makefile)
 - The executables `model1` and `model2` are available in the current directory.
3. To run the two tutorial component models:
 - Edit the script `run_tutorial` to adapt it to your platform and execute it
`> ./run_tutorial`
 - The results of the component models are now in subdirectory `$rundir` defined in `run_tutorial`.
 - In their current form, "`model1.F90`" and "`model2.F90`" are not interfaced with the OASIS3-MCT library and do not perform any exchanges. They just run for 6 time steps of 1 hour each without doing anything specific (see the files "`model1.out_100`" and "`model2.out_100`" in the output directory `work_tutorial`)

C. Interfacing the "tutorial" toy model with OASIS3-MCT

To interface the `model1` and `model2` codes, you must understand the coupling algorithm that this toy should reproduce.

The "tutorial" toy model should reproduce the coupling between two simple codes, `model1` and `model2`, with the OASIS3-MCT coupler.

The total run is 6 hours (21600 seconds). `Model1` has a time step of 3600 seconds and runs on the logically-rectangular ORCA2T grid (182x149). `Model2` has a time step of 1800 seconds and runs on the logically-rectangular LMDz grid (96x72).

The coupling exchanges to implement are as follows: a field with **symbolic name** "FSENDOCN" in model1 and "FRECVATM" in model2 is sent from model1 to model2 every 2 hours and a field with **symbolic name** "FSENDATM" in model2 and "FRECVOCN" in model1 is sent from model2 to model1 every 3 hours:

- At the beginning of the run, model1 receives "FRECVOCN" coming from a restart file; at the end of the coupling period of 2 hours, model1 sends the coupling field "FSENDOCN" to model2 that receives it as "FRECVATM" at the beginning of the next coupling period (see **figure fsendocn** below).
- In addition, model1 also outputs to a file the **time averaged field** "FSENDOCN" every 2 hours (see the **figure fsendocn_file**).
- At the beginning of the run, model2 receives "FRECVATM" coming from a restart file; at the end of the coupling period of 3 hours, model2 sends the coupling field "FSENDATM" to model1 that receives it as "FRECVOCN" at the beginning of the next coupling period (see the **figure fsendatm** below).
- The interpolation from "FSENDOCN" into "FRECVATM" uses a pre-existing weight and addresses file called "my_remapping_file_bilinear.nc". The interpolation specified to transform "FSENDATM" into "FRECVOCN" is the SCRIPR/BILINEAR one (i.e. the weight and addresses file used for the interpolation are calculated at the first coupling time step with the SCRIP library).

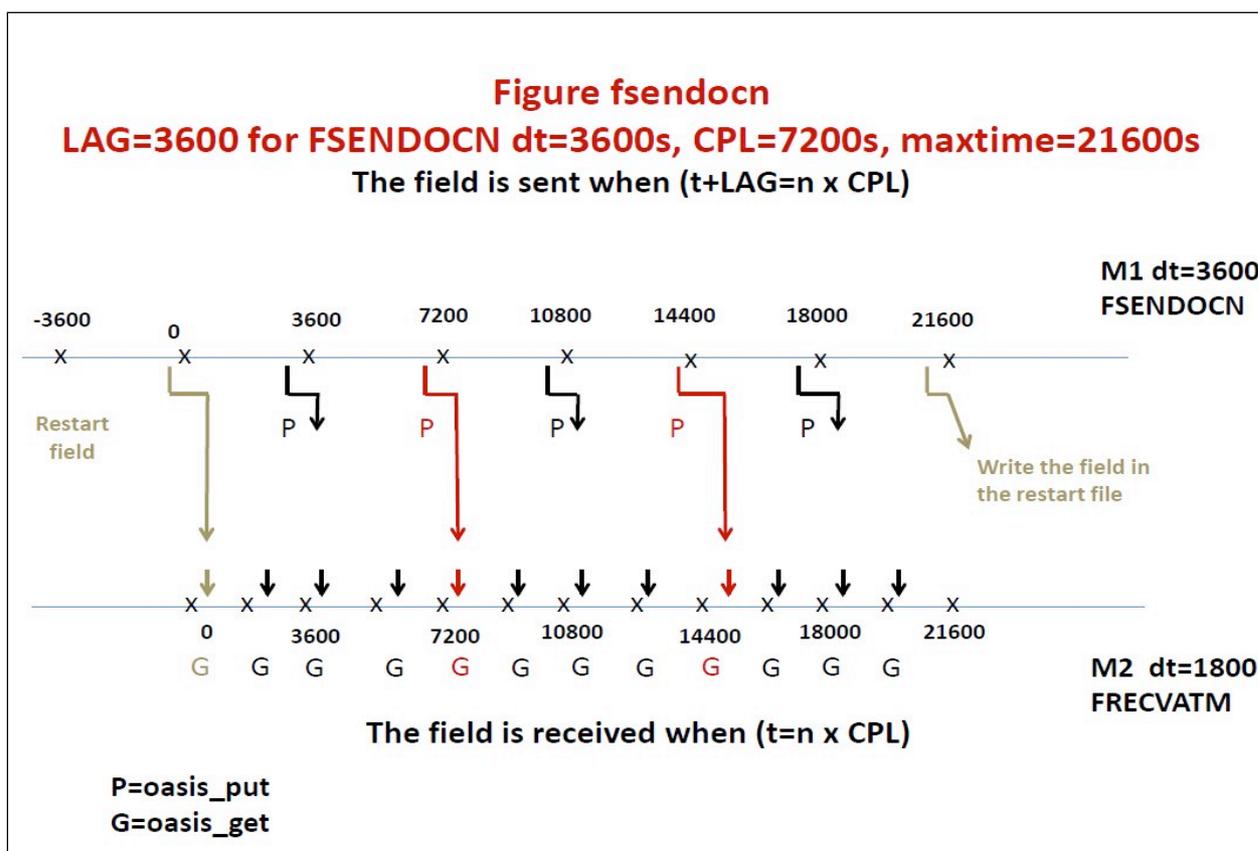


Figure fsendocn_file
FSENDOCN sent to a file CPL=7200s, maxtime=21600s
 The field is sent when $(t=n \times \text{CPL})$

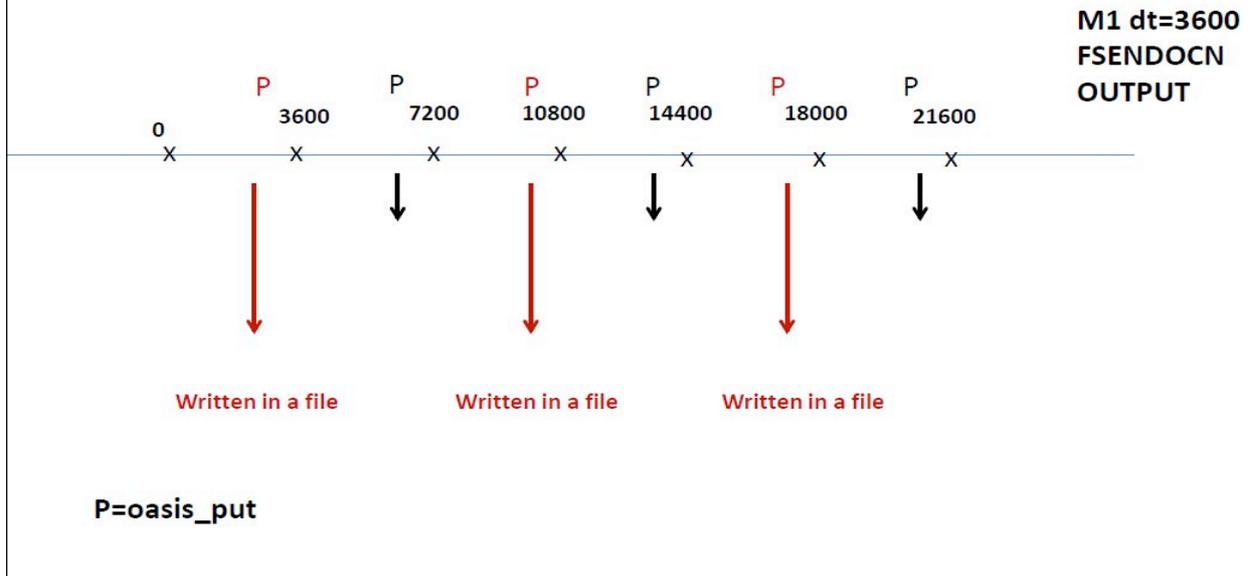
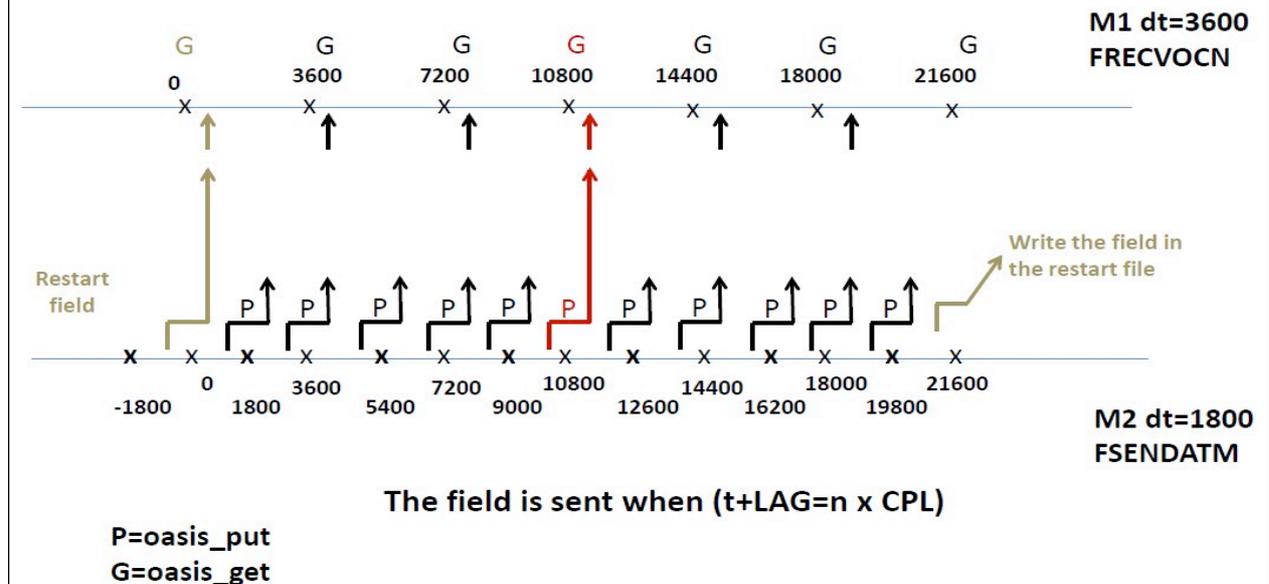


Figure fsendatm
LAG=1800 for FSENDATM dt=1800s, CPL=10800s, maxtime=21600s
 The field is received when $(t=n \times \text{CPL})$



1. Modify "model1.F90" and "model2.F90" so to implement the initialisation, definition and declaration calls of OASIS3-MCT library. The lines where to introduce the OASIS3-MCT specific calls are marked with ""!! TOCOMPLETE ...". As a first step, suppose that each model will run on only one process. Most of the variables that will be used in the oasis routines calls are already defined at the beginning of the programs.
2. Modify "model1.F90" and "model2.F90" so to implement the coupling exchanges, as described in the paragraph C. In the time step of the models, implement the reception of the input coupling fields ("oasis_get") at the beginning of the time step and then, the sending of the output coupling field ("oasis_put"), at the end of the time step (see **figures fsendocn and fsendatm**).

D. Creating the namcouple with the OASIS3-MCT GUI

The objective is to create the *namcouple* text file used of the *tutorial* toy coupled model with the OASIS3-MCT GUI. The *namcouple* contains all the user-defined information necessary **to configure a particular coupled run**. The OASIS3-MCT GUI is an application of OPENTEA, a graphical interface used by different codes, and developed at CERFACS. Its sources are located in the repository oasis3-mct/util/oasisgui. A short description of the OASIS GUI is given in the **README file** in oasis3-mct/util/oasisgui.

Define the following environment variables:

- OASISGUIHOME = \$HOME/oasis3-mct/util/oasisgui
- PYTHONPATH = \$OASISGUIHOME/XDRpy

Define the alias command:

- alias oasisgui 'wish \$OASISGUIHOME/opentea/opentea.tcl -config \$OASISGUIHOME/myconfig.xml -code oasis3-mct &'

Launch the GUI using the alias you just created:

- oasisgui

You must fill up the different tabs from the left to the right. Each window must be validated (button "process" on the bottom on the right in each window). An orange bullet in the tab means that no information was stored yet in the window, a red bullet in the tab means that there is an error in the window and a green bullet in the tab means that the data entered in the window is correct. You can find information on OASIS3-MCT by clicking on the blue bullets (?) "Learn more" in each window.

Validated data will be progressively stored in an xml file when filling the GUI. This file must be explicitly saved when leaving the GUI (you will be automatically asked for this).

Once you will have saved your xml data, you will be able to reload it using the (File/Load as new) menu at the left top of the GUI (**do not use the (File/Load as part) menu**).

Go through the following steps to construct the namcouple:

1. Read the description of the GUI in the first window and validate it by processing it using the "Process" button on the bottom right.
2. Window "Gen" (Generalities):
 - Enter the total simulation length corresponding to the total run of the tutorial of 6 hours.
 - Put the debug level at 1, which corresponds to debug information written by OASIS3-MCT only by the master of each model (see the blue bullet for more details) in files "debug.root.01" and "debug.root.02".
 - Put the Time statistics option to 0 so to not perform any time statistics for the tutorial toy.
 - Do not activate the "Enable grouped fields" as there will be no fields sent together in the tutorial toy.
 - Validate the window by processing it using the "Process" button.
3. Window "Grids":
 - Enter the names of the grids associated to the coupling fields of the different models, clicking on the button "add Grid in use", double clicking on the new row created and filling the part "no label" under "Grid names". In your case, the grids defined in the models are **torc** for model1 and **lmdz** for model2.
 - For each grid you must specify if the grid is global (periodical) or regional and the number of overlapping points. ORCA2T (torc, 182x149) is periodical with 2 overlapping points and LMDz (lmdz, 96x72) is also periodical but without any overlapping point.
 - You may define the dimensions of each grid or not, but if they are specified for one grid, they must be specified for all grids. It is easier afterwards to visualize the results with Ferret if you put the dimensions.
 - When processing the window, you are asked to save the data in an xml file. Save the new xml file in the data directory of the tutorial toy: *oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial.xml*.
 - The namcouple text file will be saved, when all the windows will be processed, in a sub-directory named after the xml file in directory : *oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial*
4. Window "Fld name" (Field name):

- For each coupling field, a "user reference name" must be given, clicking on the button "add Coupling field", double clicking on the new row created and filling the part "no label" under "Field label". This "user reference name", as the GUI Id (generated automatically in the second column when processing), is only used by the GUI and will not appear in the namcouple.
Then for each coupling field, the symbolic name used in the source and target models must be specified as well as the source and target grids. These symbolic names in the namcouple are the ones that will be used when interfacing with OASIS3-MCT using oasis_def_var.
To define the coupling fields, please refer to section C.
- When processing the window to validate the data, an automatic "GUI Id" is associated to each field by the GUI. This Id will be used in the last tabs to define the different transformations associated to each field. If the coupling fields are not grouped the Id is the Field label.

5. From window "Fld status" to window "Cons" (for Conservation for global transformations):

- Define all characteristics and transformations associated to each coupling field identified by its Id under the tab Fields in each window. You have some documentation available by clicking on the blue (?) "Learn more".
 - ✓ The name of the restart file for the field **FSENDOCN** sent by model1 already exists and is named **fdocn.nc**, the name of the restart file for the field **FSENDATM** sent by model2 also already exists and is named **fdatm.nc**. The field **FSENDOCN** will additionally be time averaged and written to a file; the name of a restart file (not used initially but used between two runs), for example **f2avg.nc**, must be also defined for this field. Define the status so to be able to visualize and debug the results. For the field FSENDOCN written to a file, the status must be **OUTPUT**. In this case the output file and the debug file are the same.
 - ✓ As each model reads a restart file, define the corresponding LAG for each field (see section 2.10.1 of User Guide for more details).
 - ✓ To be able to verify quickly the results and the remapping, we suggest you to activate the CHECKIN and CHECKOUT options for each field (*the results will appear below the attribute diags in the debug files*).
 - ✓ The interpolation specified to transform "FSENDATM" into "FRECVOCN" should be the **SCRIPR/BILINEAR** one (i.e. the weight and addresses file used for the interpolation are calculated at the first coupling time step with the SCRIP library) under the SCRIP tab. See the User Guide section 4.3 for more details on the BILINEAR interpolation.
 - ✓ The interpolation from "FSENDOCN" into "FRECVMATM" should use a pre-existing weight and addresses file called "my_remapping_file_bilinear.nc", to be specified under the MAP tab.
 - ✓ Do not forget to process each window one after the other.

6. Window "Sum" (for Summary):

- Process the window to obtain a summary of all the coupling fields and their transformations.

7. Window "Config" (for Configuration file):

- Process the window to obtain the namcouple; the textfile will be saved in the sub-directory named after the corresponding xml file, e.g. *oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial*, if, as suggested above in D3, you saved the xml file in *oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial.xml*. Copy the namcouple file from */data_oasis3/namcouple_tutorial* to */data_oasis3*.

If you close and reopen the oasisgui application, upload the xml file located in *oasis3-mct/examples/tutorial/data_oasis3/namcouple_tutorial.xml* using the (File/Load as new) menu.

E. Running the tutorial toy model

Run the toy coupled model with the script "run_tutorial" (after modifying the part "4. Execute the model"). In this configuration, where both models call their oasis_get before their oasis_put, a deadlock may occur in the coupled toy model if the coupling fields received at the beginning of the run are not automatically read from the coupling restart files. If this happens, find which modifications are needed in the toy to remove this deadlock without changing the order of the oasis_get and oasis_put (see section 2.3.1 of User Guide for more details).

- The results of the tutorial coupled model are now in your work subdirectory: the file "nout.000000", written by the master process of one model, contains the information read in the configuration file *namcouple*. The OASIS3-MCT debug files debug.root.01 and debug.root.02 are written by the master process of each model. The level of debug information in these files (which corresponds to the debug level defined in the GUI in the second window "Gen") depends on the value of the first number on the line below \$NLOGPRT in the *namcouple* (see the section 3.2 of the User Guide for more details).
- If you have put "EXPOUT" for the coupling field status, you can visualize the content of the netCDF debug files with ferret and the scripts script_ferret_*.jnl. Type "ferret" and then "go xxx.jnl" where xxx.jnl is the name of the ferret script you want to run. (Note that to run another ferret script, you have to restart ferret.)
- Explain why the number of time steps in the file FSENDATM_model2_02.nc is not the same than in the file FRECVOCN_model1_03.nc.

- Keep your results by renaming your working directory

In this configuration, model1 and model2 run with one process each. This is indicated in the launching script "run_tutorial" (see "nproc_exe1" and "nproc_exe2").

To run and couple model1 and model2 in parallel, one has to ensure that the partition definition transferred to OASIS3-MCT via the call to oasis_def_partition is properly done. In model1 and model2, the information to provide as arguments of the oasis_def_partition is calculated by the subroutine decomp_def that you can check in more detail. Then, the number of processes has to be modified in the launching procedure.

- Specify for example "nproc_exe1=3", "nproc_exe2=3" in the script run_tutorial .
- In oasis3-mct/examples/tutorial, execute "run_tutorial", which then launches the models on 3 processes each.
- Keep your results by renaming your working directory
- Visually compare the results with the non-parallel case

OASIS3-MCT supports different parallel decompositions for the models (see the section 2.4 of the User Guide). Tutorial routine "decomp_def.F90" can define the local partition for the BOX or the APPLE decompositions. By default, the APPLE decomposition is used. To test the BOX decomposition, recompile (make clean ; make) the tutorial models with, in Makefile: "CPPKEYDECOMP_M1=DECOMP_BOX" or "CPPKEYDECOMP_M2=DECOMP_BOX".