RegESM (Regional Earth System Model) User Guide

U. U. Turuncoglu, Istanbul Technical University, Informatics Institute, Turkey

1. Model Design

Regional Earth System Model (RegESM) is designed to be a state-of-art coupled modeling system that allows using variety of different earth system models. It also supports easy to plug new sub-components by using its simplified interface. In this case, developers from different disciplines might easily adapt their sub components (i.e. wave, ice, land surface) to the modeling system by following the common conventions, which is used in RegESM.

The designed modeling system currently includes three different model components, which are connected via ESMF[1] (Earth System Modeling Framework):

- **Atmosphere:** ICTP's RegCM[2] regional climate model
- **Ocean:** In this case modeling system supports two different ocean models. Rutgers University's ROMS[3] ocean model (s-coordinate model) or MITgcm[4] (z-coordinate model)
- **River Routing:** Max-Planck Institute's HD[5] river routing model.

The main aim is to use such kind of coupling library (like ESMF) is to standardize the coupling interfaces and having efficient interaction among the model components. The key component of the coupled model is the "driver" (or "coupler"), which is responsible to synchronize the model components and the interaction (via exchange fields) among them. In general, the transferred exchange fields depend on the interaction between the components and the application itself. The information about the definition of the exchange fields between the modeling components can be seen in the Section X.

The interaction of the main components can be seen in Fig. 1.

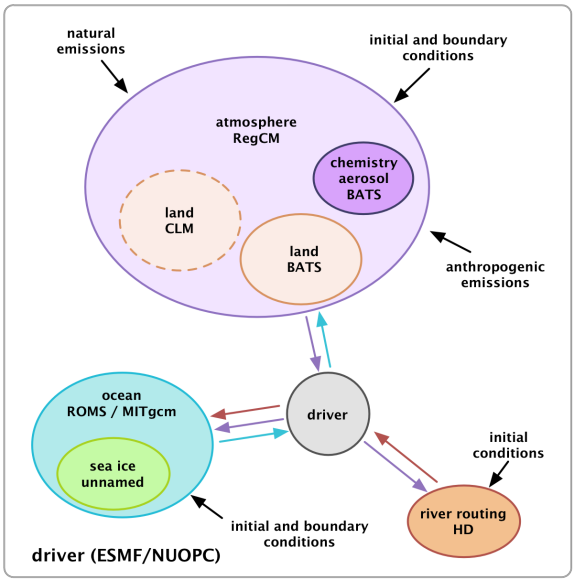

Figure 1 Components of RegESM modeling system. The arrows indicate the interaction direction between the sub-components.

[1] http://www.earthsystemmodeling.org/

[2] http://gforge.ictp.it/gf/project/regcm/

[3] http://www.myroms.org/

[4] http://mitgcm.org/download/

[5] http://http//www.mpimet.mpg.de/en/science/the-land-in-the-earth-system/terrestrial-hydrology/hd-model.html

The design of the RegESM follows the common conventions about the multi-component earth system model. It is designed as an orchestrator to control the plugged sub-components. The RegESM itself is also called as "driver" or “coupler” and it basically has no code related with the physical sub-models. It just holds the definition of the "Initialize", "Run" and "Finalize" routines, component grid structure information (masking, grid coordinates and decomposition properties) and time information to achieve synchronization among the components.

The modeling system uses Earth System Modeling Framework (ESMF) as a coupler library to connect different variety of standalone earth system models. In this case, each component is assigned as a gridded component and the interaction between them is defined using connectors - The National Unified Operational Prediction Capability (NUOPC[6]) - interface. In the future, we are planning to use mediator instead of connectors to have much more flexible and efficient design.

For instance, the following figure (Fig. 2) shows the interaction between three-component earth system models. It basically summarizes the interaction between RegCM, ROMS and HD models.

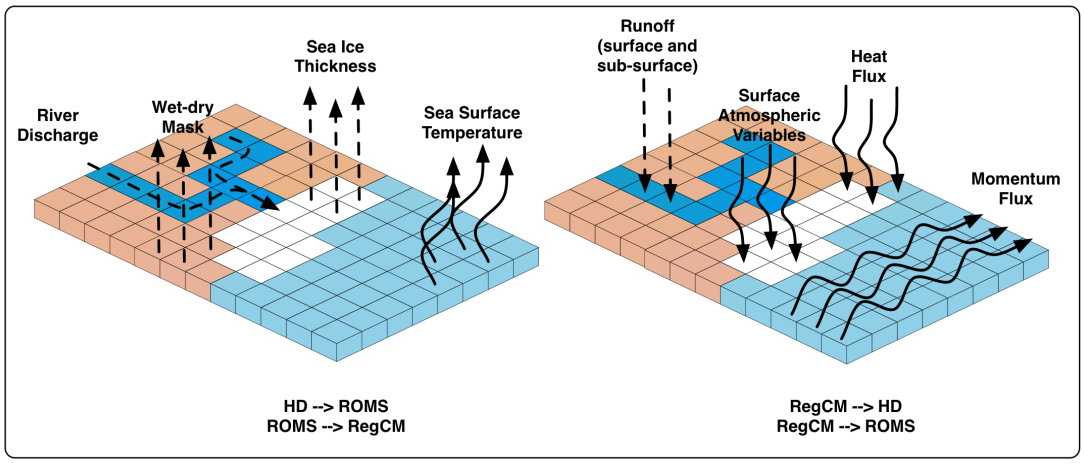

Figure 2 Example of interaction between components. Arrows shows the exchange fields.

The current design allows us to define different coupling intervals among the sub-components (i.e. fast and slow time steps). This is crucial because the response time of the components and the physical processes might be in different time interval. So, it is better to have a flexible modeling system to define different coupling interval among the model components. In Fig. 3, the river routing component (RTM) is interacting with atmosphere (ATM) and ocean (OCN) components with 1-day interval but OCN and ATM components exchange data (i.e. sea surface temperature, heat and momentum fluxes) in 3-hour interval. In this case, the coupler component (RegESM) is responsible from the synchronization.

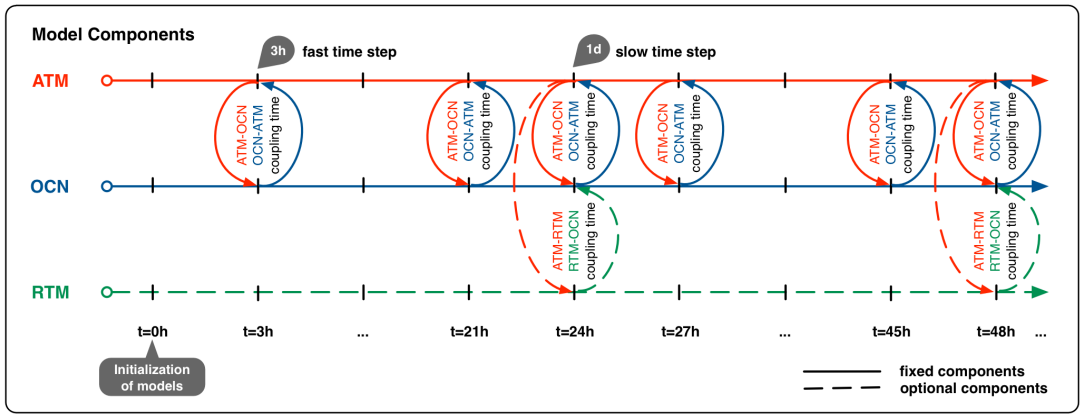

Figure 3 The run sequences of the components.

6 http://www.earthsystemmodeling.org/conventions/nuopc.shtml

The components of the modeling system can be activated or disabled easily via driver's configuration file (namelist.rc, more information can be found in Section X). The driver configuration file is also responsible for distributing CPU (or cores) to the model components. In this case, model components can run in different number of processor (or core). The current design of the modeling system supports both **sequential** and **concurrent** execution of the model components (Fig. 4).

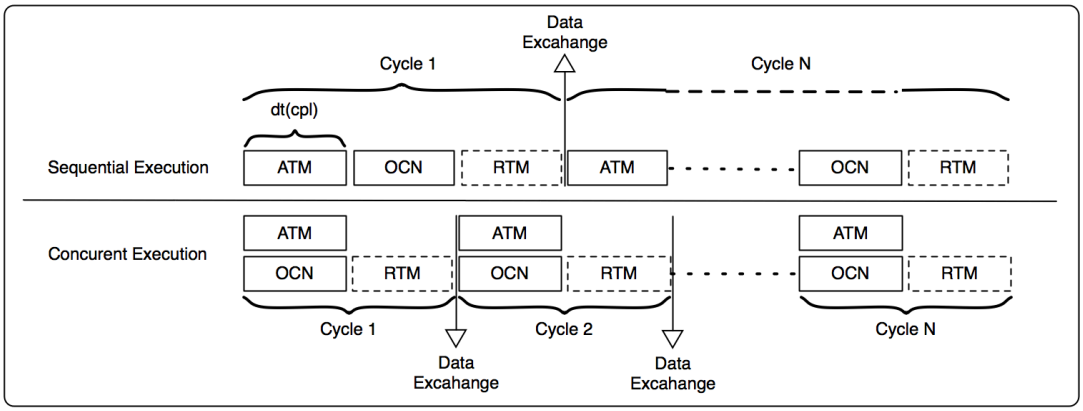

Figure 4 Sequential vs. concurrent execution mechanism

In sequential execution mode, all model components and also driver use all the available cores except RTM component. RTM component shares last core with the OCN component because it is not parallelized using MPI and it uses only one core. In this mode, models are executed in an order and each one of the model components waits others to run again. After running all model components (just only for coupling time step), the models exchange the data and start to run again in an ordered fashion until next data exchange time. This can be seen in the upper part of the Fig. 4. In this case, using RTM component is optional and it is shown as dashed lines.

The coupled model also supports concurrent execution mode. In this case, each model component uses its own set of processor (=cores) except RTM component and driver. The RTM component again uses last available processor (shared with OCN component) due to the sequential design of the model component. The driver uses all cores to perform interpolation and data exchange. To assign different number of core for each model component can be done by modifying driver configuration file.

The following table (Table 1) shows the tested coupled model configurations along with the tested model versions.

Table 1 Matrix for tested model configurations.

# Model Component	Domain	ATM	OCN		RTM
		RegCM	ROMS	MITgcm	HD
2	Mediterranean Sea	+	+		
	Mediterranean Sea	+		+	
	Caspian Sea (Turuncoglu et al., 2013, GMD)	+	+		
3	Caspian Sea	+	+		+

2. Installation

The basic requirements of the installation processes of the RegESM can be divided into three main sections:

- In the first step, user needs to prepare required working environment for the RegESM installation. In this case, a set of third-party software libraries (i.e. netCDF, ESMF etc.) must be installed or available in the host system that will be used to run the coupled model
- Then, each model components (i.e. RegCM, ROMS, MITgcm) must be installed with coupling support. Due to the various design of the standalone model components, the coupling support are achieved by different ways. For example, the latest version of RegCM (version 4) can be used as a model component in RegESM without doing any extra modification or applying patch. In ROMS case, user needs to apply a lightweight patch to use the model as a RegESM component. Unlike, RegCM and ROMS, MITgcm has an additional module that allows coupling.
- The last step is the installation of RegESM by using external libraries and the standalone installation of model components.

In this section, it is assumed that the user has basic information about usage and installation of the individual model components. So, this document does not include detailed information about the installation of the every libraries and the configuration of the individual model components, which is mainly depend on the application and used model configuration.

The following sections mainly aim to give detailed information about the installation procedure about the given steps.

2.1. Installation of Third-Party Libraries

This section includes a set of example commands to install required libraries and tools on a Linux (ICTP's Argo Cluster, Centos 6.4 Final) based system by using Intel Compiler (13.1.0) and OpenMPI (installed with same version of Intel Compiler). The users also note that the installation procedure might change in other systems and compiler combinations.

The **PROG** environment variable, which is used in this section, is mainly indicates the directory for tools and library installations.

2.1.1. Hierarchical Data Format (HDF5)

Before installation of HDF5 library (1.8.11), it is necessary to install a compression library. The HDF5 supports both zlib[7] and szip[8] libraries but in this document we choose to install zlib (1.2.8) instead of szip library.

```
cd $PROGS
wget http://zlib.net/zlib-1.2.8.tar.gz
tar -zxvf zlib-1.2.8.tar.gz
cd zlib-1.2.8
export CC=icc
export FC=ifort
./configure --prefix=`pwd`
make
make install
```

Installation of HDF5 library:

```
cd $PROGS
wget http://www.hdfgroup.org/ftp/HDF5/releases/hdf5-1.8.11/src/hdf5-1.8.11.tar.gz
tar -zxvf hdf5-1.8.11.tar.gz
./configure --prefix=`pwd` --with-zlib=$PROGS/zlib-1.2.8 --enable-fortran --enable-cxx CC=icc FC=ifort
CXX=icpc
make
make install
```

[7] http://www.zlib.net

[8] http://www.hdfgroup.org/doc_resource/SZIP/

2.1.2. Network Common Data Form (netCDF)

The netCDF library is distributed separately for each programming language. Because of this restriction, it is necessary to follow specific order to install netCDF C (4.3.0), C++ (4.2) and Fortran (4.2) libraries.

```
cd $PROGS
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-4.3.0.tar.gz
tar -zxvf netcdf-4.3.0.tar.gz
cd netcdf-4.3.0
mkdir src
mv * src/.
cd src
./configure --prefix=$PROGS/netcdf-4.3.0 CC=icc FC=ifort LDFLAGS="-L$PROGS/zlib-1.2.8/lib -
L$PROGS//hdf5-1.8.11/lib" CPPFLAGS="-I$PROGS/zlib-1.2.8/include -I/$PROGS/hdf5-
1.8.11/include"
make
make install
export LD_LIBRARY_PATH=$PROGS/netcdf-4.3.0/lib:$LD_LIBRARY_PATH
```

```
cd $PROGS
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-cxx-4.2.tar.gz
cd netcdf-cxx-4.2
mkdir src
mv * src/.
cd src
./configure --prefix=$PROGS/netcdf-cxx-4.2 CC=icc CXX=icpc LDFLAGS="-L$PROGS/zlib-1.2.8/lib -
L$PROGS/hdf5-1.8.11/lib -L$PROGS/netcdf-4.3.0/lib" CPPFLAGS="-I$PROGS/zlib-1.2.8/include -
I/$PROGS/hdf5-1.8.11/include -I$PROGS/netcdf-4.3.0/include"
make
make install
```

```
cd $PROGS
wget ftp://ftp.unidata.ucar.edu/pub/netcdf/netcdf-fortran-4.2.tar.gz
cd netcdf-fortran-4.2
mkdir src
mv * src/.
cd src
./configure --prefix=$PROGS/netcdf-fortran-4.2 CC=icc FC=ifort LDFLAGS="-L$PROGS/zlib-1.2.8/lib -
L$PROGS/hdf5-1.8.11/lib -L$PROGS/netcdf-4.3.0/lib" CPPFLAGS="-I$PROGS/zlib-1.2.8/include -
I/$PROGS/hdf5-1.8.11/include -I$PROGS/netcdf-4.3.0/include"
make
make install
```

```
cd $PROGS/netcdf-4.3.0/lib
ln -s ../../netcdf-cxx-4.2/lib/* .
ln -s ../../netcdf-fortran-4.2/lib/* .
ln -s ../../netcdf-cxx-4.2/include/* .
ln -s ../../netcdf-fortran-4.2/include/* .
export NETCDF=$PROGS/netcdf-4.3.0
export PATH=$NETCDF/bin:$PATH
```

The last section of commands is required for reaching to all netCDF interfaces or APIs (C, C++ and Fortran) from the single directory.

2.1.3. Parallel netCDF (optional)

Actually, this step optional but if there is a plan to use ESMF netCDF I/O capabilities to write exchange fields to disk, then parallel netCDF library (1.3.1) is required.

```
cd $PROGS
wget http://ftp.mcs.anl.gov/pub/parallel-netcdf/parallel-netcdf-1.3.1.tar.gz
tar -zxvf parallel-netcdf-1.3.1.tar.gz
cd parallel-netcdf-1.3.1
./configure --prefix=`pwd` --with-mpi=/opt/openmpi/1.6.5/intel/2013psm FC=mpif90 F77=mpif90
CXX=mpiccpc
```

```
make
make install
export PNETCDF=$PROGS/parallel-netcdf-1.3.1
```

Note that the path for MPI installation might be change. In this case, user must supply correct path to "--with-mpi" configuration option.

2.1.4. Apache Xerces C++

This library is required for ESMF installation. It is basically responsible from reading/writing grid definitions and attributes (field, component and state level) as XML format.

```
cd $PROGS
wget http://apache.bilkent.edu.tr//xerces/c/3/sources/xerces-c-3.1.1.tar.gz
tar -zxvf xerces-c-3.1.1.tar.gz
cd xerces-c-3.1.1
./configure --prefix=$PROGS/xerces-c-3.1.1 CC=icc CXX=icpc
make
make install
```

2.1.5. Earth System Modeling Framework (ESMF)

The one of the main component of the coupled model is the coupling library, which is used to create driver to control the standalone model components. The detailed and up-to-date information of the installation procedure can be found here[9]. The users need to pay attention to the following issues;

- The coupled model might need special features of ESMF based on the selected options in the driver configuration (namelist.rc) file. In this case, the debug level option is important. If user try to debug the exchange fields by writing them into disk in netCDF format (debug level 3 must be selected in this case), then it is necessary to install ESMF library with parallel I/O and netCDF support (for more information, look at the example environment variables defined in ESMF installation related with NETCDF and PNETCDF). The installation of netCDF
- After netCDF version 4.3.0 the C++ interface is changed and ESMF 6.2.0 (tested with coupled model) is not compatible with it. So, it is better to use the <= 4.3.0 version of netCDF in this case.

Example environment variable definitions for ESMF installations:

ARGO[10] (ICTP's Cluster System, Italy):

Note that the list of environment variables (given as tcsh or csh shell syntax, for bash type shell the *setenv* command must be replaced by *export* command along with the usage of = sign) is specific to **ARGO** cluster, its architecture **(x64_64)**, operating system **(Centos 6.4 final)** and installed compiler **(Intel Compiler, 13.1.0)** and MPI **(OpenMPI, 1.6.5)** versions. They must be modified to install ESMF library to the other computing systems or clusters. The detailed information about definition of environment variables for ESMF library itself can be found in ESMF Web site[11].

[9] http://www.earthsystemmodeling.org/esmf_releases/last_built/ESMF_usrdoc/

[10] http://argo.ictp.it

[11] http://www.earthsystemmodeling.org/download/platforms/

```
setenv ESMF_OS Linux
setenv ESMF_TESTMPMD OFF
setenv ESMF_TESTHARNESS_ARRAY RUN_ESMF_TestHarnessArray_default
setenv ESMF_TESTHARNESS_FIELD RUN_ESMF_TestHarnessField_default
setenv ESMF_DIR $PROGS/esmf_6.2.0
setenv ESMF_TESTWITHTHREADS OFF
setenv ESMF_INSTALL_PREFIX ${ESMF_DIR}/install_dir
setenv ESMF_COMM openmpi
setenv ESMF_TESTEXHAUSTIVE ON
setenv ESMF_BOPT O
setenv ESMF_SITE default
setenv ESMF_ABI 64
setenv ESMF_COMPILER intel
setenv ESMF_PIO internal
setenv ESMF_PNETCDF "standard"
setenv ESMF_PNETCDF_INCLUDE $PROGS/parallel-netcdf-1.3.1}/include
setenv ESMF_PNETCDF_LIBPATH $PROGS/parallel-netcdf-1.3.1}/lib
setenv ESMF_NETCDF "split"
setenv ESMF_NETCDF_INCLUDE $PROGS/netcdf-4.3.0/include
setenv ESMF_NETCDF_LIBPATH $PROGS/netcdf-4.3.0/lib
setenv ESMF_XERCES "standard"
setenv ESMF_XERCES_INCLUDE $PROGS/xerces-c-3.1.1/include
setenv ESMF_XERCES_LIBPATH $PROGS/xerces-c-3.1.1/lib
```

Then following command must be issued to install ESMF,

```
cd $ESMF_DIR
make >&make.log
make install
```

After installation of the ESMF library, user can create a new environment variable (ESMF_LIB and ESMFMKFILE) to help RegCM configure script to find the location of the required files to build coupled model. The example environment variables for C shell,

```
setenv ESMF_LIB
"${ESMF_INSTALL_PREFIX}/lib/lib${ESMF_BOPT}/${ESMF_OS}.${ESMF_COMPILER}.${ESMF_ABI
}.${ESMF_COMM}.${ESMF_SITE}"
setenv ESMFMKFILE "${ESMF_LIB}/esmf.mk"
setenv PATH
"${ESMF_DIR}/apps/apps${ESMF_BOPT}/${ESMF_OS}.${ESMF_COMPILER}.${ESMF_ABI}.${ESM
F_COMM}.${ESMF_SITE}:${PATH}"
```

At this level it is always better to install ESMF library also with debug support (change ESMF_BOPT environment variable as *g* and defined the variable again before installation). To have both optimized and debug version of ESMF library might help to find the source of the possible errors caused by the coupled model. The production runs can be done with the optimized version but for debugging use could install the model with debug version of the ESMF library.

2.2. Installation of RegESM

After installing the required libraries, it is necessary to install individual model components first. In this case, user might install the individual model components to system and defines the installation directories in the configuration phase of the coupled model to build RegESM executable. At this stage, it is better to follow a common convention about the installation and create hierarchical directory structure for installation of the model components, their input and output files. This is crucial because it helps to find the source of the possible errors easily and it also helps to keep clean the working directory. In this case, the suggested directory structure to install RegESM can be seen in Figure 5. As it can be seen from the figure, each model components (including “driver” itself – called as “esm”) use its own directory for the source files, input and output. Due to the some limitations of the MITgcm model and its design, it is impossible to write/read the model files to/from specific directory and those files must be stored in the **BASE_DIR** (main RegESM working directory – it is defined by user).

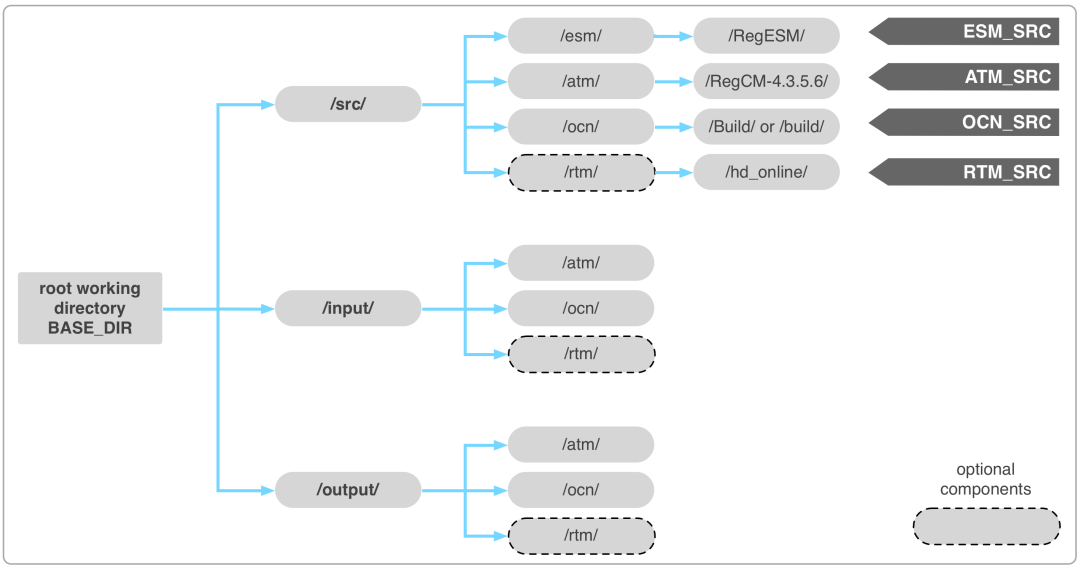

Figure 5 Example directory structure for RegESM installation

In addition to the directory structure, the configuration files, run script (OpenPBS, LSF etc.), input and output files (just for MITgcm model) must be placed in the main working directory (**BASE_DIR**). Also, the RegESM executable is placed in the working directory (the soft link must be created (i.e. following command can be used for this purpose - *cd $BASE_DIR; ln -s $ESM_SRC/regesm.x .*).

2.2.1. Regional Climate Model (RegCM)

To install RegCM with coupling support user must issue following commands:

```
mkdir $ATM_SRC
cd $ATM_SRC
wget https://gforge.ictp.it/gf/download/frsrelease/161/983/RegCM-4.3.5.6.tar.gz
tar –zxvf RegCM-4.3.5.6.tar.gz
cd RegCM-4.3.5.6
autoreconf –f –i (or bootstrap.sh) ← issuing this command could be unnecessary in some cases
./configure –prefix=`pwd` --enable-cpl CC=icc FC=ifort
make
make install
```

The ATM_SRC environment variable is used to point the installation path of the atmospheric model and user might replace it by any valid directory.

The command that is given here is for Intel Compiler and Linux operating system and the commands might change in usage of different compiler and computing environment. The main important issue is that the model components does not need ESMF library anymore (all the ESMF related code is moved to the RegESM (namely 'driver'). So, the installation of the model is almost same as the standalone version except given extra configure option (--enable-cpl) to enable coupling support.

The detailed information about RegCM, usage and its configuration parameters can be found in the user guide[12] and the reference manual[13].

2.2.2. Regional Ocean Model (ROMS)

To install ROMS with coupling support, the user need to patch the original version of the model. The patch includes set of minor modifications to prepare ROMS ocean model for the coupling. The reader also notes that there is no any generic patch for the all versions ROMS due to the existence of the different versions and branches (i.e. Rutgers University's ROMS, ARGIF ROMS, UCLA ROMS, ROMS with sea-ice) of the model.

[12] https://gforge.ictp.it/gf/download/docmanfileversion/63/1152/UserGuide.pdf

[13] https://gforge.ictp.it/gf/download/docmanfileversion/64/1153/ReferenceMan.pdf

The current version of the RegESM comes with a patch that is created by using a snapshot of the ROMS branch[14] with sea-ice support and it can be found in the "tools/ocn". Doing modifications and applying the patch to the model are the responsibility of the user but the given version of the patch can be used as a reference to modify the any ROMS version.

Then ROMS official documentation[15] can be used to install and create a realistic case for the ROMS ocean model.

2.2.3. MITgcm Ocean Model

To install MITgcm with coupling support, the user must need to add the ESMF capability to the MITgcm firstly. Please contact with the developer for the MITgcm package to activate coupling. Then MITgcm official documentation[16] can be used to install and create a realistic case.

2.2.4. HD River Routing Model

The HD river routing model is the property of Max Planck Institute, Germany and it is distributed via a user agreement. So, it is not distributed freely. In our design, we modified the HD model to allow easy to use and plug into RegESM driver. The following list summarizes the modification done by ITU.

- The file format of the model output and restart files are changed from binary (SRV format) to netCDF.
- The model main code is split into three parts: initialization, run and finalize by following common convention of the ESMF library.
- The necessary modifications are done for the coupling. In this case new preprocessor flag is added to activate coupling.
- A set of NCL functions is created to drive the standalone model using RegCM output and to create SRV formatted input files.

Because of the license restriction that is mentioned previously, the HD model and the modifications for the coupling are not distributed publicly via GitHub repository. If there is plan (and need) to use the three component modeling system, then user should get the license from the Max Planck Institute for the standalone version of the HD model and contact with ITU/ICTP to get the modified version of the HD model to use in the coupled model configuration. The user also port his/her own RTM component to the coupled modeling system by following same methodology given the list and the help of the RegESM source code (mod_esmf_rtm.F90).

To install modified version of HD model with coupling support, user should edit the Makefile and issue following commands:

```
mkdir $RTM_SRC
tar –zxvf hd_online.tar.gz
cd hd_online
make
make install
```

The Makefile has a set of macro to define the compiler, netCDF library and coupling support. –DCPL option must be added to FC macro along with the compiler command to activate the coupling support.

2.3. Installation of RegESM Driver

After installation of the individual model components (i.e. RegCM, ROMS or MITgcm), the RegESM can be installed. The RegESM basically uses the object (*.o), module (*.mod) and header (*.h) files of the model components to create static library file for each model (i.e. libatm.a for ATM, libocn.a for OCN and librtm.a for RTM component) and uses these static

[14] https://github.com/kshedstrom/roms

[15] https://www.myroms.org/wiki/index.php/Documentation_Portal

[16] http://mitgcm.org/public/docs.html

libraries to create the final RegESM executable. Due to this restriction, the RegESM installation requires the installation of the model components.

The ESMF library is also required to compile ESMF related component codes in “driver” side. The configuration script basically tries to find the installation directory of ESMF by looking into a specific environment variable (**ESMF_LIB**). If **ESMF_LIB** environment variable points the directory of ESMF shared library (libesmf.so) and configuration file (esmf.mk) exist and valid then it could be used to compile the RegESM. In case of non-defined **ESMF_LIB** environment variable, user might specify the ESMF library directory by using **--with-esmf** configure option.

Currently, RegESM project is maintained by using a Git repository[17] (GitHub). The model can be installed using following commands ($ESM_SRC is the director of RegESM installation):

```
cd $ESM_SRC
git clone https://github.com/uturuncoglu/RegESM.git
cd RegESM
./configure --prefix=`pwd --with-atm=$ATM_SRC --with-ocn=$OCN_SRC --with-rtm=$RTM_SRC
CC=icc FC=ifort
make
make install
```

The configure options **--with-atm**, **--with-ocn** and **--with-rtm** (optional) is used to point the installation directories of the model components. For ROMS case, **--with-ocn** option must point “**Build”** directory that holds the compiled source files of the ROMS installation. This is also similar for the other components (i.e. “**build**” directory for MITgcm).

The configure script is smart enough to check the some key files to find the types of the ocean model component (ROMS or MITgcm) and compiles the required files suitable for selected model components. In addition, the configure script also checks the given ROMS model installation directory (if this is the case) to enable the sea-ice related part of the data exchange routines in the “driver” side. So, user does not need to set any other option when using sea-ice enabled ROMS version.

[17] https://github.com/uturuncoglu/RegESM

3. Usage

To run the RegESM model, user needs to create or modify two files:

- Exchange field table
- Driver (or high level) configuration file

Then, user is able to run the coupled modeling system. It is also note that both exchange field definition and the model configuration file must be placed with the same directory as RegESM executable file.

3.1. Exchange Field Table (exfield.tbl)

The exchange field table is the main component of the coupled model and it basically keeps the definition of the exchange fields and their attributes (i.e. description, units, grid location etc.). The main structure of the exchange field table can be seen in Table 2.

Table 2 Main structure of the exchange filed table.

[Number of Fields - N] [Direction of Coupling]
[Exchange Field 1]
[Exchange Field 2]
…
[Exchange Field N]
[Number of Fields] [Direction of Coupling]
[Exchange Field 1]
[Exchange Field 2]
…
[Exchange Field N]
[Number of Fields] [Direction of Coupling]
…
…

As it can be seen from the table, the file has header section for each coupling direction. In this case the number of field must be same with the number of field added just after the header section. Currently, the direction of the coupling can be defined as:

- atm2ocn
- ocn2atm
- atm2rtm
- rtm2ocn

but this section will be expanded when new modeling components (i.e. ice, wave etc.) will be added.

After definition of the header section, the user needs to define the exchange fields and their attributes based on the table (see Appendix, Table 3). In this case, attributes of the exchange fields are separated ":" (double-comma) symbol.

The example exchange field tables for different model setups and detailed definition abut them can be found under "external" folder of the RegESM source. Currently, following tables are stored in this directory:

- **000** – Two-component configuration (ATM-OCN; RegCM+ROMS with ice): The ROMS uses BULK_FLUX parameterization in this case.
- **001** – Three component configuration (ATM-OCN-RTM; RegCM+ROMS with ice+HD): Again ROMS model uses BULK_FLUX parameterization and river discharge is activated by using specific CPP options in ROMS model.
- **002** – Two component configuration (ATM-OCN; RegCM+MITgcm)

The definition of the exchange field table mainly depends on the application and activated individual model components itself and it is not east to create a generic definition for all cases.

3.2. RegESM Configuration (namelist.rc)

It is simply followed ESMF convention to support generic configuration file for RegESM, The configuration file is manly responsible from PET assignment to model components (both

number of CPU or core and the type of execution – sequential vs. concurrent), definition of high-level simulation period (start, stop, restart time and calendar), coupling time step (slowest one), matrix of coupling time step multiplier to calculate fast and slow time steps for data exchange among the components, debug level and list of rivers (along with coordinates, number of source point and monthly correction factors) that is used for the RTM coupling. The detailed explanation of each configuration option can be found in Appendix section (Table 4).

3.3. Running RegESM

Running of the coupled model is very similar to running any other standalone model component. The only difference is that the user might need to give two different configuration files for each model component (only for ROMS coupling) to the RegESM executable. The all configuration files of individual model components must be in the same director with the executable.

The following sample scripts can be used as a base.

RegCM+MITgcm using OpenPBS job scheduler (on ICTP's ARGO cluster): regesm.job

```
#!/bin/bash

#PBS -N test
#PBS -l walltime=24:00:00
#PBS -l nodes=72
#PBS -q esp

# load required modules
. /etc/profile.d/modules.sh
module purge
module load intel/2013
module load openmpi/1.6.5/intel/2013

setenv PROGS /home/netapp/clima-users/users/uturunco/progs
setenv XERCES $PROGS/xerces-c-3.1.1
setenv PNETCDF $PROGS/parallel-netcdf-1.3.1
setenv NETCDF $PROGS/netcdf-4.3.0
setenv HDF5 $PROGS/hdf5-1.8.11
setenv PATH $NETCDF/bin:$PNETCDF/bin:$PATH
setenv LD_LIBRARY_PATH $NETCDF/lib:$PNETCDF/lib:$XERCES/lib:$HDF5/lib:$PROGS/zlib-
1.2.8/lib:$LD_LIBRARY_PATH

setenv ESMF_OS Linux
setenv ESMF_TESTMPMD OFF
setenv ESMF_TESTHARNESS_ARRAY RUN_ESMF_TestHarnessArray_default
setenv ESMF_TESTHARNESS_FIELD RUN_ESMF_TestHarnessField_default
setenv ESMF_DIR $PROGS/esmf-6.2.0
setenv ESMF_TESTWITHTHREADS OFF
setenv ESMF_INSTALL_PREFIX ${ESMF_DIR}/install_dir
setenv ESMF_COMM openmpi
setenv ESMF_TESTEXHAUSTIVE ON
setenv ESMF_BOPT O
setenv ESMF_OPENMP OFF
setenv ESMF_SITE default
setenv ESMF_ABI 64
setenv ESMF_COMPILER intel
setenv ESMF_PIO internal
setenv ESMF_PNETCDF "standard"
setenv ESMF_PNETCDF_INCLUDE ${PNETCDF}/include
setenv ESMF_PNETCDF_LIBPATH ${PNETCDF}/lib
setenv ESMF_NETCDF "split"
setenv ESMF_NETCDF_INCLUDE ${NETCDF}/include
setenv ESMF_NETCDF_LIBPATH ${NETCDF}/lib
setenv ESMF_XERCES "standard"
setenv ESMF_XERCES_INCLUDE ${XERCES}/include
setenv ESMF_XERCES_LIBPATH ${XERCES}/lib

setenv ESMF_LIB
```

```
"${ESMF_INSTALL_PREFIX}/lib/lib${ESMF_BOPT}/${ESMF_OS}.${ESMF_COMPILER}.${ESMF_ABI
}.${ESMF_COMM}.${ESMF_SITE}"
setenv ESMFMKFILE "${ESMF_LIB}/esmf.mk"
setenv PATH
"${ESMF_DIR}/apps/apps${ESMF_BOPT}/${ESMF_OS}.${ESMF_COMPILER}.${ESMF_ABI}.${ESM
F_COMM}.${ESMF_SITE}:${PATH}"

setenv LD_LIBRARY_PATH ${ESMF_LIB}:${LD_LIBRARY_PATH}
setenv PATH
${ESMF_INSTALL_PREFIX}/bin/bin${ESMF_BOPT}/${ESMF_OS}.${ESMF_COMPILER}.${ESMF_AB
I}.${ESMF_COMM}.${ESMF_SITE}:${PATH}

# run coupled model
cd /home/netapp/clima-users/users/uturunco/MED/RegESM/run2
ulimit -s unlimited
mpirun -v ./regesm.x regcm.in_MED50km >& regesmout.txt
```

RegCM+ROMS using LSF job scheduler (on UHEM, Turkey): regesm.lsf

```
#!/bin/bash
#BSUB -P avktis
#BSUB -J cpl
##BSUB -q mid
#BSUB -q deci9
#BSUB -m karadeniz_prace
#BSUB -o %J.out
#BSUB -e %J.err
#BSUB -a intelmpi
#BSUB -n 32

mpirun.lsf ./regesm.x regcm.in_MED50km med12.in > regesmout.txt
```

Also not that in this case RegESM executable gets two-configuration file (one for RegCM and one for ROMS).

4. Known Limitations

4.1. Conservation of the exchange fields

In the current version of the coupled modeling system, a prototype version of the conservation algorithm is implemented. In this case, the conservation algorithm ensures that the global integral of the source and destination fields (over the matched regions) will remain same but it does not guarantee local conservation of the exchange fields. It just applies the difference of the global integral of source and destination fields to the destination field across the domain.

The implemented conservation algorithm works as following;

1. First, the algorithm finds the matched grid points between the model components (atmosphere and ocean models).
2. It calculates the global integral of the source and destination fields
3. It calculates the difference of the global integrals (destination – source)
4. Then, it calculates the unit difference (by dividing the difference to the total surface area in the destination grid)
5. Then, it applies the unit difference to each grid cell

The local conservation feature might be implemented in the future version of the modeling system with availability of the high-order conservative type interpolation in the ESMF side (expected in 2014). Also, user might also aware that the first-order conservative interpolation technique, which is currently supported by ESMF, might create artifacts (square like shapes in the destination field) in the interpolation when the grid resolution difference is high (i.e. 50 km ATM and 5-7km in OCN) between the components.

The user also notes that the current conservation algorithm is only works with bilinear type interpolation (It can be defined in the exchange field table for each variables separately) and also only supports data exchange between atmosphere and ocean components (ATM-OCN and OCN-ATM)

APPENDIX

Table 3 Definition of the fields (or columns) of the exchange field table

#	Column Name	Description
1	**Short Name**	**The sort name of the exchange field. The list of the available fields:** taux - zonal surface wind stress (N/m^2 or Pa) tauy - meridional surface wind stress (N/m^2 or Pa) wndu - zonal wind component (m/s) wndv - meridional wind component (m/s) wspd - wind speed (m/s) psfc - surface pressure (hPa or mb) tsfc - 2 meter surface temperature (K) qsfc - 2 meter specific humidity (kg/kg) lwrd - net longwave radiation (W/m^2) swrd - net shortwave radiation (W/m^2) dlwr - downward longwave radiation (W/m^2) dswr - downward shortwave radiation (W/m^2) lhfx - latent heat flux (W/m^2) shfx - sensible heat flux (W/m^2) nflx - net heat flux, latent+sensible+longwave-shortwave (W/m^2) prec - total precipitation, P (m/s) evap - evaporation, E (m/s) sflx - net freshwater flux, E-P (m/s) rnof - surface runoff (m/s, just over land) snof - sub-surface runoff (m/s, just over land)
2	**Standard Name**	The standard name of the exchange filed. The user can use anything in here but using CF conventions for the field is suggested.
3	**Interpolation Type**	The type of interpolation that is used to transfer data form source grid to destination. It can be defined as “bilinear”, “conserv”, “nearstod”, “neardtos” and “none”. **Restrictions:** • When transferring data from RTM component to the ocean model the interpolation type must be selected as “nearstod” The “bilinear” type interpolation type must be use to apply conservation correction to the field
4	**Source Grid Point**	This field basically used to define the location of the source filed in the grid definition (Arakawa type grids). It can be defined as “cross”, “dot”, “u” and “v”.
5	**Destination Grid Point**	It is same with the previous filed but in this case it defined the destination point in the destination grid.
6	**Input Unit**	It is just for information. The model does not use this information. The space character is not allowed.
7	**Output Unit**	Same as input unit
8	**Scale**	Used for the unit conversion of the exchange field. The source field is multiplied by this scale factor after interpolation. The user can use scale factor as a number or can use following shortcuts. cf1 – rho0*cp (rho0 = 1025.0 kg/m^3, water density and cp = 3985.0 J/kg/K, heat capacity) cf2 – 1/cf1 cf3 – 1/rho0 It is also possible to use – symbol in the definition of the shortcuts (i.e. –cf1).
9	**Offset**	Used for unit conversion of the exchange filed. The value is added to the source field after applying scale factor. So, the combination of the Scale and Offset parameters might help to convert the units.
10	**Flag for Conservation**	It can be “T” or “F”. If it is defined as true (T), then coupler component applies the difference of source and destination field integral to the destination field. It can be applied in field basis. In general, heat flux components can be defined as true to conserve the fields globally. **Restrictions:** • It just work between ATM and OCN components The interpolation type of the field must be defined as “bilinear”

Table 4 Configuration options and their descriptions

#	Option	Description
1	**PETLayoutOption**	It is used to define the execution mode of the coupled model (see Fig.4). The value can be **"sequential"** or **"concurrent"**. • If **"sequential"** is selected then model components are triggered by sequential fashion (one after another). In this case, set same number of processor to all components using "PETs" parameter of the configuration file except RTM (it must be 1) component and physically same PETs will be used by the components with order. • If **"concurrent"** option is selected, then model component run in parallel (each component uses distinct PETs). In this case, PETs are not overlapped and it allows parallel execution of the components. The user also note that the informative output of the model components (basically standard output or stdout) can be mixed due to the race conditions among the independent PETs.
2	**PETs**	It defined the PET distribution of the model components. The number of PETs must be given in a specific order with a space between them. Currently, the order is ATM, OCN and RTM respectively. • If **"sequential"** PET layout selected and same number of PET must be given to all components then driver uses same number of PETs in the job execution. • In contrast to **"sequential"** execution mode, user might select the **"concurrent"** type execution. In this case, the number of PETS is assigned to the components based on the given number of processor. The total usage of the PETs will be the sum of PETs defined for components except RTM. Again, RTM component is automatically assigned to the last PET.
3	**UnmappedFill**	Activates extrapolation due to the mismatch among the land-sea mask of the model components and wrong heat flux artifacts near the coastlines. Currently, only supports interpolation between ATM-OCN and OCN-ATM components. The value can be **1 or any other value grater than 0 (enabled)** or **0 (disabled)**.
4	**DebugLevel**	The debug level is used to find the source of the possible errors (i.e. wrong grid representation, artifacts in the exchange fields, upper and lower limits of the decomposition elements etc.). Actually, there are four level of debug option: **1.** It enables no debug output. This is not suggested generally. **2.** It enables minimal debug output from driver component. In this case, model only prints informative messages related with the upper and lower bounds of the decomposition elements, name of running components etc. **3.** It enables writing grid information of the components in VTK format. The VTK files can be used to create visual output of the grid structures. In this case, Visit or another similar visualization tool that supports VTK format. **4.** It enables writing exchange fields to a netCDF file. In this case, files are written to disk in each coupling time step. This option must be use for short runs to check the correctness of the exchange fields. It must be used with caution because it creates lots of file (depends on coupling time step, number of component and number of exchange field). For this option, ESMF must be compiled with parallel-netcdf support. The user also notes that the exchange fields might be in wrong dimension order (transposed). This is not a bug or error. It is normal and just way of storing arrays in ESMF side (ESMF uses right-hand coordinate system). **5.** This level is same with the previous level but in this case the exchange fields are written to disk in ASCII format. Due to the limitation of the keeping track of the unit numbers in Fortran, this level might produce corrupted files and it must be used with caution. The user also notes that all the levels (< 4) also include the previous level of debug output.
5	**Calendar**	This configuration option defines the global calendar type used in the time synchronization among the components. The coupled model currently supports three different calendar option: • gregorian • 360_day • julian The "driver" basically uses this information to check the components calendars to have a consistent view of the time dimension among the model components.

		In the future, we are planning to support more calendars in the driver side but this is also depending on the model components.
6	**StartTime**	Sets simulation start time. It must be consistent among the model components otherwise “driver” triggers an error message and kills the all processes.
7	**RestartTime**	Sets simulation restart time. If start time and restart time differs then “driver” try to run the model components in restart mode. It must be consistent among the model components otherwise “driver” triggers an error message and kills the all processes. There is also **“restart.sh”** script under tools/ directory to help to organize the configuration files of the model components in case or restarting. Also note that this script does not modify the **namelist.rc** and user must need to modify it manually.
8	**StopTime**	Sets simulation start time. It must be consistent among the model components otherwise “driver” triggers an error message and kills the all processes.
9	**TimeStep**	This configuration of option is used to set the coupling interval (the slowest one) among the components. The model components might interact with each other with different coupling interval (fast vs. slow time step – asynchronous coupling). In this case, **DividerForTStep** configuration option (basically it holds the divider matrix) is used to calculate coupling interval among the components.
10	**DividerForTStep**	It is used to calculate coupling interval among the model components. By using this option, user might define different coupling interval for each coupling direction. This is basically required for asynchronous coupling due to the different response time of the model components and their time resolution limitations. For example, HD river routing model runs in daily time scale and there is no any mean to couple RTM component with ATM and OCN with time scale less than day but ATM and OCN component might be coupled with 3-hour interval. So, this is the possible case to use this configuration as a divider matrix. It basically divides the time step (defined in **TimeStep** option) to get the coupling direction time step. For example, if the Time Step is defined as 1-day and divider matrix is set to 8 in ATM-OCN and OCN-ATM direction then the coupling step for ATM-OCN and OCN-ATM direction will be 3-hours. The **TimeStep** must be defied as a slowest time interval.
11	**RiverList**	This configuration is basically required for coupling with RTM component. It holds: • The list of the river mouth positions (the “driver” queries river discharge from RTM component grid, which is the closest grid to these coordinates) • River direction (used in ROMS coupling) • Number of source point for specific mouth. In some cases it is better to distribute the river discharge along the source point for same river. The “driver” basically divides the river discharge to this number to find the discharge of the each individual source point. • Monthly correction factors or weights (one for each month – total count is 12) for simplified bias correction. The weights must be between 0 and 1 to correct seasonal distribution of river discharge. The correction factors can be calculated by comparing RTM model generated river discharges with the observations and set in here. This option currently only works with the RegCM+ROMS+HD configuration but we have a plan to extend it MITgcm case also.