Community Coupler C-Coupler1
User's Guide

Li Liu, Ruizhe Li, Cheng Zhang, Guangwen Yang, Bin Wang

liuli-cess@tsinghua.edu.cn

Ministry of Education Key Laboratory for Earth System Modeling,

Center for Earth System Science (CESS),

Tsinghua University, Beijing, China

1st, February, 2014

Content

1 Introduction

1.1 Brief introduction to coupler

Climate system model or earth system model is an important tool for global climate change study. It is always a coupled model consisting of several component models, which are coupled together with a coupler, to simultaneously simulate the variation of atmosphere, ocean, land surface, sea ice, etc. With the fast development of science and technology, there are more and more coupled models as well as component models in the world. For example, there are more than 50 coupled model versions participating in the Coupled Model Intercomparison Project Phase 5 (CMIP5).

Coupler (Valcke et al., 2012) is a key component for model development. It achieves effective interaction and parallel computation among multiple component models, and controls the integration of the whole coupled model. It also provides a platform to make scientists and engineers cooperate together. Most of state-of-the-art climate system models and earth system models are constructed with couplers. With more and more component models (e.g., land ice model, chemistry model, and biology model) to be added into earth system model, coupler becomes more and more important to the development of earth system models.

There are several existing couplers, e.g., OASIS (Redler et al., 2010; Valcke, 2013), MCT (Larson et al., 2005; Jacob et al., 2005), ESMF coupler (Hill et al., 2004), FMS coupler (Balaji et al., 2006), Scup (Yoshimura and Yukimoto, 2008), CPL6 (Craig et al., 2005), CPL7 (Craig et al., 2012), etc.

1.2 C-Coupler project

C-Coupler with its platform (also known as runtime environment) is an open source community coupler developed in China. There was no coupler developed for earth system model in China before. The coupled models developed in China always

use CPL6, MCT and OASIS for model coupling. For example, almost all coupled models for CMIP5 developed by Chinese institutions use CPL6. To help the development of earth system models in China, China initiated the C-Coupler project in 2010.

The most important mission of developing C-Coupler is to provide various coupling functions for various coupled models in a user-friendly way and to integrate various models on the same model platform for sharing. Under this mission, C-Coupler is designed to be a coupler framework which can generate various coupler instances for different kinds of coupled models, with a series of objective coupling functions as follows:

1) Flux computation. C-Coupler will integrate flux algorithms, such as the algorithms to calculate flux variables between atmosphere and ocean, to provide the function of flux computation.
2) 3D coupling. Besides traditional 2D coupling, C-Coupler will provide 3D data communication and 3D data remapping to enable the coupling of 3D variables.
3) Model nesting. C-Coupler will support one-way and two-way model nesting to facilitate the work of nesting regional models.
4) Ensemble. C-Coupler will be able to integrate ensemble packages or ensemble algorithms. Multiple instances of model simulation can be run simultaneously on the same platform.
5) User friendliness. C-Coupler will provide various coupling functions in a user friendly way. C-Coupler will try to facilitate the work in integrating component models, constructing coupled models, integrating external algorithms and running models.
6) Modularity and extendibility. C-Coupler will provide modular and extendible software framework for coupling various component models and integrating various algorithms, such as flux algorithms and remapping algorithms.
7) Efficient parallelization. C-Coupler itself will be parallelized. It will provide parallel I/O for models. With the development of models as well as

C-Coupler, we will continuously improve the parallel efficiency, especially when the resolution gets higher.

8) Standardization. C-Coupler will provide uniform standards for integrating component models and external algorithms, to facilitate the sharing of component models and algorithms among various coupled models.
9) Automatic error detection. C-Coupler will provide various kinds of automatic error detection to help users to detect and fix bugs in constructing a coupled model.
10) Reliability. We will build an increasing number of test cases for C-Coupler based on the various coupled models using it. Before releasing a new version, C-Coupler must pass all test cases.
11) Reproducibility of model simulation results. C-Coupler platform will provide the function of managing model simulation. It will formulate the rules for reproducing simulation results. The information for reproducing simulation results will be automatically generated by the C-Coupler platform.
12) Free and good service. As the team of developing C-Coupler, we will try our best to provide free and good service for various users. Any new requirements are possibly included in the future plan of C-Coupler development.

1.3 C-Coupler1.0

C-Coupler1.0 is the first version of C-Coupler for public use. It is mainly programmed by C++, with more than 23000 lines of source code programmed by C-Coupler team. Guided by the objective coupling functions of C-Coupler project, C-Coupler1.0 achieves the following functions:

1) Flux computation. C-Coupler1.0 provides flux computation function for climate system models through integrating existing flux algorithms, such as the flux algorithms in NCAR CPL6 (Craig et al., 2005).
2) 3D coupling. Besides 2D coupling, C-Coupler1.0 provides 3D data

communication and 3D data remapping to support the coupling of 3D variables.

3) User friendliness. C-Coupler1.0 has not achieved all targets in user friendliness of the C-Coupler project. It tries to facilitate the work in running models on the C-Coupler platform.
4) Modularity and extendibility. C-Coupler1.0 is a modularized library with object-oriented code structure. It provides Fortran interfaces for coupling component model and interfaces for integrating external algorithms. External algorithms can be either in Fortran or C++.
5) Parallelization. C-Coupler1.0 is a parallel library based on the MPI parallel programming model, which achieves the same (bit-identical) results when using any number of processor cores. It supports model coupling in parallel, and also supports the parallel execution of component models, where 1D and 2D parallel decomposition on horizontal grid is supported. It uses multiple executables for coupled models. It enables direct coupling between component models for better parallel performance, where C-Coupler does not take unique executable and processes/processor cores.
6) Automatic error detection. C-Coupler1.0 provides several kinds of automatic error detection. It can check the order of calling C-Coupler interfaces in component models, check the consistency of grids between component models and remapping weight files, check whether component models provide sufficient coupling variables, etc.
7) Reliability. C-Coupler1.0 has passed various test cases with more than 600 diagnostic statements intra its source code. There are several standards for testing C-Coupler, e.g.: C-Coupler must achieve the same (bit-identical) result no matter how many processor cores are used; C-Coupler must achieve the same (bit-identical) result in restart run; a coupled model can achieve the same (bit-identical) result after replacing the original coupling approach with C-Coupler; etc.
8) Reproducibility of model simulation results. C-Coupler1.0 provides the

function of managing model simulation. It formulates rules and automatically records information for reproducing simulation results.

9) Free and good service. After the public release of C-Coupler1.0, we will provide free and good service for constructing various coupled models with C-Coupler1.0.

1.4 General terms for C-Coupler

There are several general terms for C-Coupler, listed out as follows.

1) **Component model**. A component model always simulates a component or a process of the earth. Component models are classified into categories, e.g., atmosphere model, land model, ocean model, sea ice model, wave model, etc.
2) **Coupled model**. A coupled model always consists of several component models.
3) **Experiment model**. An experiment model is a version of model which is ready for simulation. Generally, it can be any kind of models, such as single-column model, stand-alone component model, regional coupling model, air-sea coupling model, nested model, climate system model, earth system model, etc. On the C-Coupler platform, we always use "compset" to stand for experiment model. Experiment models are always built from existing component models or existing coupled models, such as the models participating CMIP5. An experiment model can also be new version of coupled model with several component models which are never coupled together before.
4) **Model experiment**. Model experiment is a simulation based on certain experimental setups of an experiment model for scientific research purposes. A new model experiment can be evolved from existing model experiments.
5) **C-Coupler platform**. The C-Coupler platform is a platform (also known as runtime environment) for using the models with C-Coupler for simulation. It

provides the functions of creating, configuring, compiling, running and downloading model experiment. This user reference manual is mainly for introducing the C-Coupler platform.

1.5 How to use C-Coupler platform

The C-Coupler platform provides utilities to facilitate model experimenting, shown in Figure 1. The first step is to create an experiment from a default setup or an existing experiment. Then users can check and modify experimental setups under the working directory of the experiment. After that, user should configure the experiment and then compile the source code. Before the compilation, users can remove some objs or executables when necessary. After the compilation, users can run the experiment for simulation results. If users want further simulation after analyzing simulation results, they can improve experimental setups or create a new experiment.

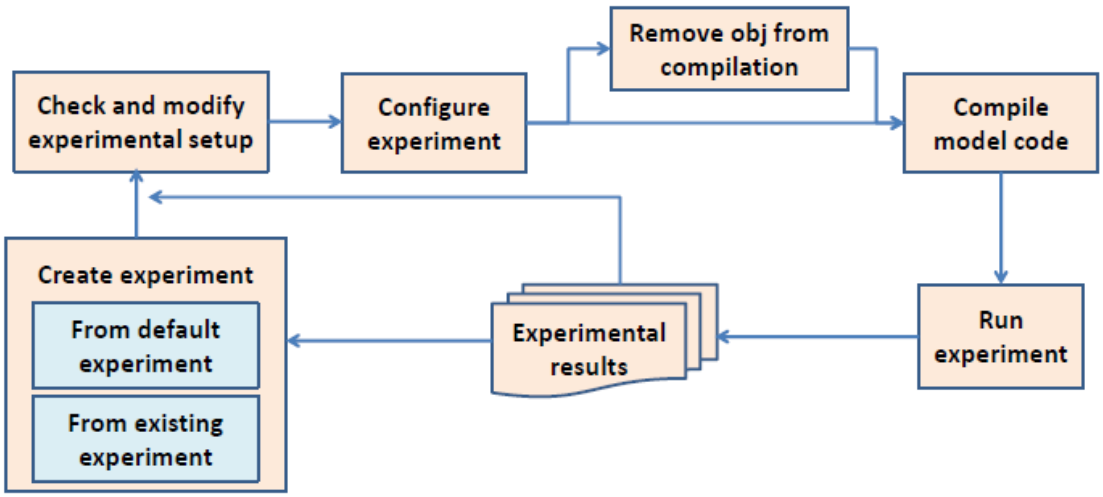

Figure 1 Flowchart for model experimenting on C-Coupler platform

2 Models on C-Coupler platform

Models on the C-Coupler platform include component models, coupled models and experiment models. Any kinds of component models and coupled models can be integrated in. Experiment models are always built from existing component models or existing coupled models, using C-Coupler for coupling and using the C-Coupler platform for experimenting. An experiment model can also be new version of coupled model with several component models which are never coupled together before.

2.1 Component models

2.1.1 Atmosphere models

2.1.1.1 GAMIL2

GAMIL is an AGCM developed mainly in the National Key Laboratory for Numerical Modeling of Atmospheric Sciences and Geophysical Fluid Dynamics (LASG), Institute of Atmospheric Physics (IAP), Chinese Academy of Sciences (CAS), which has taken part in various international model intercomparison projects and has been widely used for various science problems studies.

GAMIL is based on the Eulerian finite-difference dynamical core. Its discrete grid includes a hybrid horizontal grid with the Gaussian grid (for low resolution) or uniform grid (for high resolution) in the low and middle latitudes region and a weighted even area grid in the high latitudes and polar region, and 26-σ vertical levels (pressure normalized by surface pressure) with the model top at 2.194hPa. Its dynamical core includes a finite-difference scheme that conserves mass and effective energy (Wang et al., 2004), and a two-step shape-preserving advection scheme for the moisture equation (Yu, 1994).

GAMIL2 is the second version of GAMIL. It is the atmospheric component of the climate system model FGOALS-g2. More detailed information of GAMIL2 can be found in Li et al., (2013a).

2.1.2 Ocean models

2.1.2.1 LICOM2

LICOM is the fourth generation of LASG OGCM (Liu et al. 2004a, 2004b) since the first generation developed in 1989 (Zhang and Liang, 1989). The version 2 of LICOM (LICOM2) is used in FGOALS-g2. Comparing with LICOM1.0 and LICOM1.1, LICOM2 increases the horizontal resolutions (1°x1° horizontal resolution with 0.5° meridional resolution in the tropics) and adjusts vertical resolution (10m each layer in the upper 150m) firstly (Wu et al., 2005). Secondly, the advection scheme is introduced (Xiao et al., 2006). Thirdly, the physical processes are updated or improved (Liu et al., 2012) including the mixing schemes (Canuto et al. 2001, 2002; Liu et al. 2012), solar penetration scheme (Lin et al. 2007; Lin et al. 2011), etc (Liu et al. 2012). Several updated schemes are not adopted in the latest versions of FGOALS (Lin et al. 2012). The details can be found in Liu et al (2012) and Lin et al. (2012).

2.1.3 Land surface models

2.1.3.1 CoLM and CLM3

CoLM (Dai et al., 2008) and CLM3 (Oleson et al., 2004) are two main branches of the early Common Land Model (Dai et al., 2003, 2004) for further development of land surface models. The Common Land Model is a third-generation land surface model based on the Biosphere-Atmosphere Transfer Scheme (BATS), the Institute of Atmospheric Physics Land Surface Model (IAP94) and the Land Surface Model (LSM). Due to its outstanding simulation performance, it was selected to be the land surface model in the CCSM (Community Climate System Model) model from the National Center for Atmospheric Research (NCAR), and was further developed by NCAR. At the same time, the Common Land Model was further developed by the cooperation between Beijing Normal University and University of Texas at Austin.

2.1.4 Sea ice models

2.1.4.1 CICE4-LASG

The CICE4-LASG is an improved version of CICE4 (Los Alamos sea ice model version 4.0 (http://climate.lanl.gov/Models/CICE)). It is the sea ice component in the FGOALS-g2 model. The main difference between CICE4-LASG and CICE4 is the simulation of sea ice salinity: CICE4 uses an unchanged vertical salinity profile while CICE4-LASG formulates salinity variations and designs a simple parameterization accordingly. More information about CICE4-LASG could be found in Wang et al. (2009) and Liu (2010).

2.2 Coupled models

2.2.1 FGOALS-g2

FGOALS-g2 (Flexible Global Ocean-Atmosphere-Land System Model, Grid-point version2) (Li et al., 2013b) is a coupled climate system model developed mainly in the LASG IAP. FGOALS-g2 consists of GAMIL2, LICOM2, CICE4_LASG, and CLM3, using NCAR coupler CPL6 for model coupling. Recently, FGOALS-g2 participated in several model intercomparison projects, e.g., CMIP5 and PMIP3 (Paleoclimate Modelling Intercomparison Project, Phase 3), and showed good performance on ENSO (Bellenger, et al., 2013) and other aspects.

2.3 Experiment models

2.3.1 FGOALS-gc

FGOALS-gc is a version of FGOALS-g2, by replacing the coupler CPL6 with C-Coupler1.0. In FGOALS-gc, C-Coupler1.0 integrates the scientific algorithms in CPL6 (such as flux algorithms) and works as a component which takes physical processor cores. FGOALS-gc achieves the same (bit-identical) simulation result no matter the number of processor cores for each component. In the original version,

FGOALS-gc achieves the same (bit-identical) simulation result with FGOALS-g2. In the latest version of FGOALS-gc, GAMIL2 is updated the version with 2D parallel decomposition (Liu et al., 2014) and upgrade of TSPAS advection implementation.

2.3.2 GAMIL2-sole

GAMIL2-sole is a version of GAMIL2 for sole component run. GAMIL2-sole shares the same version of GAMIL2 with FGOALS-gc. GAMIL2-sole achieves the same (bit-identical) simulation result no matter the number of processor cores used.

2.3.3 GAMIL2-CLM3

GAMIL2-CLM3 is a coupled model with GAMIL2 and CLM3. In GAMIL2-CLM3, C-Coupler1.0 works as a library for coupling, without taking additional processor cores. GAMIL2-CLM3 achieves the same (bit-identical) simulation result no matter the number of processor cores for each component.

3 Directory structures of C-Coupler platform

3.1 Main directories of C-Coupler platform

The C-Coupler platform consists of two packages: the *C-Coupler model platform* and *input data*. In the *C-Coupler model platform*, there are three main directories: *models*, *scripts*, and *config*. *models* is used to store source code of each component model and each library. *scripts* is used to store scripts for using the C-Coupler platform, such as creating, configuring, compiling, running and downloading model experiment. *config* is used to store configuration information for component models, experiment models, machines, etc. After creating a new model experiment, the working directory of the model experiment is created by the C-Coupler platform. Please do not change the directory structure of the *C-Coupler model platform* and *input data* as possible.

3.2 Directory structure of *input data*

Similar to *models*, the *input data* includes several sub directories, e.g., *atm*, *ocn*, *lnd*, *sice*, *cpl* and *grids*, to respectively store the input data of atmosphere models, ocean models, land surface models, sea ice models, couplers, model grids, etc. Each sub directory stores the input data of several component models.

3.3 Directory structure of *models*

models includes several sub directories, e.g., *atm*, *ocn*, *lnd*, *sice*, *cpl* and *libs*, to respectively store the source code of atmosphere models, ocean models, land surface models, sea ice models, couplers, libraries, etc. Each sub directory stores the source code of several component models or libraries.

C-Coupler can work as a component or a library for coupling. Currently, its code is stored under the directories *models/cpl/c-coupler/* and *models/libs/c_coupler/*.

3.4 Directory structure of *scripts*

Under *script*, there are several scripts: *create_newcase*, *configure*, *compile*, *clean*, *runcase*, *checkout_experiment*, *register_platform.csh*, *register_platform.sh*. The former five scripts are used for creating, configuring, compiling, deleting compiled objective files and running for model experiments. *checkout_experiment* is used for checking out a model experiment from a SVN server, include configuration information, source code and input data. *register_platform.csh* and *register_platform.sh* are used for registering environment variables for the *C-Coupler model platform*. Besides these scripts, there is a sub directory *utils* which stores the common tools for these scripts.

3.5 Directory structure of *config*

config includes several sub directories, e.g., *atm*, *ocn*, *lnd*, *sice*, *cpl*, *lib*, *compset* and *common*, which stores default configuration information for atmosphere models, ocean models, land surface models, sea ice models, couplers, libraries, experiment models and common tools. Each sub directory stores the configuration information of several models or libraries.

3.6 Structure of working directory of model experiment

Each model experiment has its own working directory. Users should go to the working directory for configuration, compilation and execution of the model experiment. After configuration, under the working directory, there are four scripts, e.g., *configure*, *compile*, *clean* and *runcase*, and several sub directories: *utils*, *config*, *configure_history*, *job_logs* and *run*. These four scripts and *utils* are copied from the *scripts* directory of the *C-Coupler model platform*. *config* stores the configuration information related to the current model experiment. It is oriented from the *config* directory of the *C-Coupler model platform*. *configure_history* stores the package of experimental setups after every configuration by users, for reproducing model

simulation result. *job_logs* stores the log of every model run. *run* stores files related to model compiling and model run, for each component model and library, including namelist, input data, output data, source code for compilation, and objective and executable files produced by compilation. Given a component model with name *model_name* and type (e.g., atm, ocn, lnd, sice) *model_type*, the namelist, input data as well as output data are under the directory *run/model_type/model_name/data/*, the objective files produced by compilation are under the directory *run/model_type/model_name/obj/*, the executable file is under the directory *run/model_type/model_name/exe/*, the log files of compilation are under the directory *run/model_type/model_name/build_logs/*, and the log files of component model run are under the directory *run/model_type/model_name/run_logs/*.

4 Create, configure, compile and run model experiment

As shown in Section 1.5, users can create, configure, compile and run model experiment.

4.1 Environment variables

Table 1 lists out the environment variables for model experiments. *COMPSET*, *CASEROOT* and *MACH* are specific to each experiment. As the C-Coupler platform is a shared platform for various models and various experiments, *DATAROOT*, *CODEROOT* and *CONFIGROOT* are always common environment variables. Users can use command such as "source *register_platform.sh*" or "source *register_platform.csh*" to set *CODEROOT* and *CONFIGROOT* as common environment variables, and use command such as "source *register_inputdata.sh*" or "source *register_inputdata.csh*" to set *DATAROOT* as common environment variables. Users can also set *CODEROOT*, *CONFIGROOT* and *DATAROOT* as private environment variables of an experiment.

Table 1 Environment variables for model experiments

Environment variables	Description
COMPSET=*compset_name*	*COMPSET* is the name of an experiment model, which specifies the experiment model for model experiment
CASEROOT=*case_dir*	*CASEROOT* specifies the working directory for model experiment, where the last level specifies the name of model experiment
MACH=*machine_name*	*MACH* specifies the machine to run the model experiment
DATAROOT=*data_dir*	*DATAROOT* specifies the root directory of the input data, which is always the directory *inputdata* of the C-Coupler platform

CODEROOT=*code_dir*	*CODEROOT* specifies the root directory of model code, which is always the directory *models* of the *C-Coupler model platform*
CONFIGROOT=*config_dir*	*CODEROOT* specifies the root directory of configuration information of the *C-Coupler model platform*

4.2 Create model experiment

There are two approaches to creating a model experiment: from a default setup or from an existing experiment. We encourage users to use an existing experiment but not a default setup, to minimize modification of configuration information for a new experiment.

4.2.1 New model experiment from default setup

To create a new model experiment from a default setup, users should run command "***./create_newcase experiment_env***" under the directory *scripts* of the *C-Coupler model platform*, where *create_newcase* is the script for creating experiment, and *experiment_env* is a parameter file with environment variables of the experiment. *COMPSET*, *CASEROOT* and *MACH* must be set in *experiment_env*. If *CODEROOT*, *CONFIGROOT* and *DATAROOT* are set in *experiment_env*, they will be used as private environment variables of the experiment.

For example, with the environment variables in Table 2, users can create an experiment of model FGOALS-gc under the directory */home/liuli/model_experiments/FGOALS-gc.RCP26*.

Table 2 An example of environment variables for creating a new experiment

```
COMPSET=FGOALS-gc
CASEROOT="/home/liuli/model_experiments/FGOALS-gc.RCP26"
MACH=Tansuo100
CODEROOT="/home/liuli/C-Coupler_platform/C-Coupler_Model_Platform/models/"
CONFIGROOT="/home/liuli/C-Coupler_platform/C-Coupler_Model_Platform/models/"
DATAROOT="/home/liuli/C-Coupler_platform/inputdata/"
```

4.2.2 New experiment from an existing experiment

The C-Coupler platform provides the utility of creating a new experiment from an existing experiment. Given an existing experiment *old_exp* with experimental setup package *old_exp.setup.tar*, users can create a new experiment *new_exp* from *old_exp.setup.tar* under the following steps:

1) Create the working directory of *new_exp*.
2) Go to the working directory of *new_exp* and then untar *old_exp.setup.tar*.
3) Modify environment variable *MACH* and set *CODEROOT*, *CONFIGROOT* and *DATAROOT* as private environment variables in file *config/common/env* if necessary.

4.3 Configure experiment

To configure an experiment, users should run command "*./configure*" under the working directory of model experiment. All experimental setups of the experiment are automatically packed into a *.tar* file under sub directory *configure_history*, where the time of configuring recorded in the file name is a keyword. Before configuring, users should view and modify configuration information (experimental setups). Please refer Section 6 for details.

4.4 Compile experiment model

To compile an experiment model, users should run command "*./compile*" under the experiment working directory of the model. The log files of compilation are stored under the sub directory *run* of the experiment working directory. Users can run command "*./clean para*" to remove objective files produced by compilation, where *para* means parameter for *clean*. The parameter can be the name of a component model or a library, or *all*. For command "*./clean all*", all files produced by compilation will be removed.

4.5 Run experiment

To run an experiment, users should run command "*./runcase*" under the experiment working directory. Note that, "*./runcase*" is common to any kind of experiments on any kind of hardware platforms.

5 Configuration information of models

5.1 Configuration information of component models

The configuration information of a component model is under directory *config/model_type/model_name/* of the *C-Coupler model platform*, where *model_type* and *model_name* are type and name of component model respectively. There are six files, including *field_buf_register.cfg*, *private_field_attribute.cfg*, *config.sh*, *form_src.sh*, *compiler.cfg* and *build.sh*. File *field_buf_register.cfg* records fields which should be registered by component model. File *private_field_attribute.cfg* records information for outputting private model fields (not coupling fields) of component model. File *config.sh* is a script for generating private namelist of component model. File *form_src.sh* is a script which specifies source code directories of component model. File *compiler.cfg* is a file which specifies private compiling options of component model. File *build.sh* is a script for compiling component model, which can generate some header files.

5.2 Configuration information of experiment models

The configuration information of an experiment model is under directory *config/compset/model_name/* of the *C-Coupler model platform*, where *model_name* is the experiment model. It contains a subdirectory *coupler* and a file *compset.settings*. The subdirectory *coupler* stores the configuration files for driving C-Coupler runtime software system. There are two sections in the file *compset.settings*, *common* and *model*. Section *common* records overall information of the experiment model, including component models, libraries and a template of namelist. Section *model* records information of each component model, including *realname* (real name of the component model), *grid* (label of model grid), *type* (type of component model, including *atm*, *ocn*, *lnd*, *sice*, *wave*, *cpl*, etc.), etc.

6 Configuration Information of Model Experiment

We suggest users go to the private working directory of a model experiment to check and modify configuration information of this model experiment. Please **do not** modify the shared directory *config* of the whole C-Coupler platform. Please **do not** modify any file under subdirectory *run* of experiment working directory, such as *namelist* files, because these files are temporal and generated by the scripts *configure* and *compile* according to the configuration files under directory *config* of experiment working directory. Note that, the experiment model (*compset*) for an experiment cannot be modified. If users want use another experiment model for simulation, please create a new experiment. **For reproducibility of model simulation result, if the configuration information of an experiment is modified, users should configure the experiment (run the command "*./configure*") before running it.** Then configuration information of the experiment will be stored automatically to facilitate reproducing model simulation result.

6.1 Experiment name and working directory

The name of a model experiment is the same as the name of the working directory. When users want to modify the experiment name or migrate the model experiment to another directory, please take the following two steps:

1) Move or copy the experiment to a new working directory.
2) Configure the experiment under the new working directory with command "./configure". Then, users can run the experiment after compilation.

6.2 Initial run or restart run

There are two choices to start a simulation, initial run and restart run. For initial run, please set variable *run_type* in Section *common* of file *config/common/case.conf* to "*initial*" and then configure the experiment. For restart run from the original experiment, please take the following steps:

1) Set variable *run_type* in Section *common* of file *config/common/case.conf* to “*restart*”.

2) Set variables *original_case_name*, *original_config_time*, *run_restart_date* and *run_restart_second* in Section *common* of file *config/common/case.conf*. *Original_case_name* and *original_config_time* specify the name and configuration time of the original experiment. *run_restart_date* and *run_restart_second* specify the simulation time restarted from. The format of variable *run_restart_date* is “YYYY-MM-DD”. When these variables are different from that in the restart data files, simulation cannot be restarted.

3) Configure the current experiment.

6.3 Start time and stop time

Variables *run_start_date* and *run_start_second* in Section *common* of file *config/common/case.conf* specify the start time of model experiment, while *run_stop_date* and *run_stop_second* specify the stop time. The format of variables *run_start_date* and *run_stop_date* is “YYYY-MM-DD”. Please configure the model experiment after the modification. **Frequency of writing restart file**

Variables *rest_freq_unit* and *rest_freq_count* in Section *common* of file *config/common/case.conf* specify the frequency of writing restart file. *rest_freq_unit* is the unit of frequency which can be “*seconds*”, “*days*”, “*months*” or “*years*”. For example, given *rest_freq_unit=days* and *rest_freq_count*=5, the frequency of writing restart file is per 5 days. Please configure the model experiment after the modification.

6.5 Nonleap year and leap year

Variable *leap_year* in Section *common* of file *config/common/case.conf* specifies whether leap year is on. If it is “true”, leap year is on. Otherwise, leap year is off.

Please configure the model experiment after the modification.

6.6 Experiment description

Variable *case_desc* in Section *common* of file *config/common/case.conf* specifies the description of model experiment. This description will be written into output data file through C-Coupler I/O utility. Please configure the model experiment after its modification.

Experiment description is important for comparing, evaluating and reproducing simulation result. We propose the providers of experiment leave their names (even email) into the description.

6.7 Parallel settings

Variables *num_thread*, *num_x_proc*, *num_y_proc* and *num_total_proc* in Section *model* of file *config/common/case.conf* specify parallel settings of the corresponding component model. *num_thread* is the number of threads (e.g., OpenMP threads). *num_x_proc* and *num_y_proc* are the numbers of processes on two directions, which is useful for 2D parallel decomposition. These two variables must be set or unset at the same time. *num_total_proc* is the number of total processes. If *num_x_proc* and *num_y_proc* are set, *num_total_proc* must be the multiple of *um_x_proc* and *num_y_proc*.

Please configure the model experiment after modifying parallel setting. For the experiment models which use macro define to control parallelism but not namelist, please clean obj files and recompile the source code.

6.8 Compilation options

The compilation options of a component model or library consist of its specific compilation options and common compilation options for the whole experiment model. Given a component model *comp_name* with type *comp_type*, its specific

compilation options are specified in file *config/comp_type/comp_name/compiler.cfg*. Given a library with name *lib_name*, its specific compilation optimizations are specified in file *config/lib/lib_name/compiler.cfg*. The common compilation options depend on the machine (computer platform) that the model experiment runs on. Given a machine with name *mach_name*, the common compilation options are specified in file *config/common/machine/mach_name/common_compiler.mach_name.cfg*. Please clean related objective and executable files after modifying compilation options. The following table shows the meaning of main variables in the *common_compiler.mach_name.cfg* file.

Variable	Meaning
FC	Fortran compiler
CC	C compiler
CXX	C++ compiler
CPP	Tool for precompiling according to macro definition
AR	Tool for building library
LD	Linker for all objective files for generating executable file
CFLAGS	Compiling options for CC (C compiler)
FFLAGS	Compiling options for FC (Fortran compiler)
CXXFLAGS	Compiling options for CXX (C++ compiler)
CPPFLAGS	Compiling options for CPP (precompiling)
LDFLAGS	Compiling options for LD (linker)
ULIBS	Compiling options of linking libraries for LD (linker)
NETCDFINC	Directory of NetCDF header files
NETCDFLIB	Directory of NetCDF library files
MPIINC	Directory of MPI compiler header files
MPILIB	Directory of MPI compiler library files

6.9 Directories of source code

On C-Coupler platform, each component model or library specifies its own

directories of source code. Given a component model *comp_name* with type *comp_type*, its directories of source code are specified in file *config/comp_type/comp_name/form_src.sh*. Given a library with name *lib_name*, ts directories of source code are specified in file *config/lib/lib_name/ form_src.sh*. Please clean related objective and executable files after modifying source code directories.

6.10 Namelist of component model

Given a component model comp_name with type comp_type, its namelist is generated by file *config/comp_type/comp_name/config.sh*. Please modify *config.sh* to modify the namelist. Please do not modify the namelist file under the *run* directory of the model experiment.

7 Computer systems

7.1 Recommended system setups

We recommend users to use Linux operating system for the C-Coupler platform as well as models and experiments. We propose users to use Intel compiler (including Fortran, C and C++) with the version around 11 and Intel MPI library with the version around 3 for compiling the source code. For Netcdf library, we propose to use the version around 4.

7.2 High-performance computers

Currently, the C-Coupler platform successfully runs on several high-performance computers. Under the directory *config/common/machine* of the *C-Coupler model platform*, there are 4 machines corresponding to the environment variable *MACH*: *generic_linux*, *Tansuo100*, *IAP_avatar*, and *FIO_HPC*. *generic_linux* stands for common single-node server with multiple cores. *Tansuo100* stands for the high-performance computer Tansuo100 in Tsinghua University. *IAP_avatar* stand for the high-performance computer avatar in Institute of Atmospheric Physics (IAP), Chinese Academy of Sciences (CAS). *FIO_HPC* stands for a high-performance computer in the First Institution of Oceanography (FIO), State Oceanic Administration (SOA).

If users want to use the C-Coupler platform on new high-performance computer, we'd like to help migrate to the computer and integrate the configuration information of the computer into the C-Coupler platform.

8 Download

Users can download a model experiment as well as the *C-Coupler model platform* from a SVN server of the C-Coupler platform in a simple way. Given that *exp_setups.tar* is a package of experimental setups of an experiment, users can download this experiment with command "*checkout_experiment exp_setups.tar local_env*", where *checkout_experiment* is a script which has been introduced in Section 3.4, and *local_env* is a file with environment variables. Table 4 lists out all environment variables in *local_env*.

After downloading the experiment, users can follow Section 4 to create, configure, compile and run the experiment.

The local computer for downloading the experiment must have installed two kinds of tools: SVN and FTP (such as wget, lftp and curl).

Table 4 Environment variables for downloading experiment

Environment variables	Description
USER=*username*	Username for downloading the experiment from the SVN server
PASSWORD=*password*	Password for downloading the experiment from the SVN server
DATAROOT=*dataroot_dir*	Local directory for storing input data of the experiment downloading from the SVN server
PLATFORMROOT=platformroot_dir	Local directory for storing the *C-Coupler model platform* downloading from the SVN server

9 Frequently Asked Questions

1) Cannot find MPI and Netcdf header files when compiling

Solution: Please check the path of compiler (use commands such as “which mpiifort”), and then modify compiling option variables *MPIINC, MPILIB, NETCDFINC* and *NETCDFLIB* accordingly (please refer Section 6.8 for details).

2) Error happens when linking Netcdf library

Solution: Please check the version of Netcdf to verify the option of linking Netcdf library, such as *-lnetcdf* and *-lnetcdff*, and then modify compiling option variable *NETCDFLIB* accordingly (please refer Section 6.8 for details).

10 Copyright

C-Coupler and the C-Coupler platform are protected by copyright and patent. They are free and open-source software for non-commercial usage. We encourage users to use them for model coupling, simulation, etc. Models and experiments will keep their own copyrights on the C-Coupler platform. C-Coupler and the C-Coupler platform will not infringe these copyrights.

References

Balaji, V., J. Anderson, I. Held, M. Winton, J. Durachta, S. Malyshev, and R. J. Stouffer, 2006: The Exchange Grid: a mechanism for data exchange between Earth System components on independent grids. In: Proceedings of the 2005 International Conference on Parallel Computational Fluid Dynamics, College Park, MD, USA, Elsevier, 2006.

Bellenger, H., Guilyardi, E., Leloup, J., Lengaigne, M., Vialard, J., 2013: ENSO representation in climate models: from CMIP3 to CMIP5. Climate Dyn., DOI 10.1007/s00382-013-1783-z

Canuto, V. M., A. Howard, Y. Cheng, and M. S. Dubovikov, 2001: Ocean Turbulence. Part I: One-Point Closure Model–Momentum and Heat Vertical Diffusivities. J. Phys. Oceanogr., 31, 1413-1426.

Canuto, V. M., A. Howard, Y. Cheng, and M. S. Dubovikov, 2002: Ocean Turbulence. Part II: Vertical Diffusivities of Momentum, Heat, Salt, Mass, and Passive Scalars. J. Phys. Oceanogr., 32, 240-264.

Craig, A. P., M. Vertenstein, and R. Jacob, 2012: A New Flexible Coupler for Earth System Modeling developed for CCSM4 and CESM1. *Int. J. High Perform. C*, 26-1, 31–42, doi:10.1177/1094342011428141.

Craig, A. P., R. L. Jacob, B. Kauffman, T. Bettge, J. W. Larson, E. T. Ong, C. H. Q. Ding, Y. He, 2005: CPL6: The New Extensible, High Performance Parallel Coupler for the Community Climate System Model. *International Journal of High Performance Computing Applications*, 19(3): 309-327.

Dai, Y., et al., 2003: The Common Land Model (CLM), Bull. Am. Meteorol. Soc., 84, 1,013–1,023, doi:10.1175/BAMS-84-8-1013.

Dai, Y., R. E. Dickinson, and Y.-P. Wang, 2004: A two-big-leaf model for can- opy temperature, photosynthesis, and stomatal conductance, J. Clim., 17, 2,281–

2,299, doi:10.1175/1520-0442(2004)017<2281:ATMFCT>2.0.CO;2.

Dai, Y., DY Ji, 2008: The Common Land Model (CoLM) Technical Guide. [at http://globalchange.bnu.edu.cn/download/doc/CoLM/CoLM_Technical_Guide.pdf]

Hill, C., C. DeLuca, V. Balaji, M. Suarez, and A. da Silva, 2004: Architecture of the Earth System Modeling Framework. *Comput. Sci. Eng.*, 6, 18–28.

Jacob, R., J. Larson, and E. Ong, 2005: M x N Communication and Parallel Interpolation in Community Climate System Model Version 3 Using the Model Coupling Toolkit, Int. J. High. Perform. C, 19, 293–307.

Larson, J., R. Jacob, and E. Ong, 2005: The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models. *Int. J. High Perform*, C, 19, 277–292.

Li L., B. Wang, L. Dong, L. Liu, S. Shen, N. Hu, W. Sun, Y. Wang, W. Huang, X. Shi, Y. Pu, G. Yang, 2013a: Evaluation of Grid-point Atmospheric Model of IAP LASG version 2 (GAMIL2). Advances in Atmospheric Sciences, 30(3), 855-867.

Li, L. J., P. F. Lin, Y. Q. Yu, B. Wang, T. J. Zhou, L. Liu, J. P. Liu, Q. Bao, S. M. Xu, W. Y. Huang, K. Xia, Y. Pu, L. Dong, S. Shen, Y. M. Liu, N. Hu, M. M. Liu, W. Q. Sun, X. J. Shi, W. P. Zheng, B. Wu, M.-R. Song, H. L. Liu, X. H. Zhang, G. X. Wu, W. Xue, X. M. Huang, G. W. Yang, Z. Y. Song, and F. L. Qiao, 2013b: The Flexible Global Ocean-Atmosphere-Land System Model: Grid-point Version 2: FGOALS-g2. Adv. Atmos. Sci., 30(3), 543-560.

Lin, P. F., and Coauthors, 2013: Long-term Stability and Oceanic Mean State Simulated by the Coupled Model FGOALS-s2. Adv. Atmos. Sci., 30(1), 175-192.

Liu, H. L., Zhang, X.H., Li, W., Yu, Y.Q., Yu, R.C., 2004a: A eddy-permitting oceanic general circulation model and its preliminary evaluations, Adv. Atmos. Sci. 21, 675–690.

Liu, H. L., Y. Q. Yu, W. Li, and X. H. Zhang, 2004b: Manual for LASG/IAP Climate System Ocean Model (LICOM1.0). Science Press, Beijing, 1–128. (in Chinese).

Liu, H. L., P. F. Lin, Y. Q. Yu, and X. H. Zhang, 2012: The baseline evaluation of LASG/IAP Climate system Ocean Model (LICOM) version 2.0. Acta Meteorologica Sinica.

Liu, J., Sensitivity of sea ice and ocean simulations to sea ice salinity in a coupled global climate model, Science in China Series D: Earth Sciences, 53(6), 911-916, 2010.

Liu L., R. Li, G. Yang, B. Wang, L. Li, Y. Pu, 2014: Efficient Parallelization Strategies for the Finite-Difference AGCM on Modern High-Performance Computers. Submited to Journal of Atmospheric and Oceanic Technology, minor revision.

Oleson, K. W., Y. Dai, and Coauthors, 2004: Technical description of the Community Land Model (CLM), NTIS #PB2004-105836.

Redler, R., S. Valcke, and H. Ritzdorf, 2010: OASIS4 - a coupling software for next generation earth system modeling. *Geosci. Model Dev.*, 3, 87–104, doi:10.5194/gmd-3-87-2010, 2010

Valcke, S., V. Balaji, A. Craig, C. DeLuca, R. Dunlap, R. W. Ford, R. Jacob, J. Larson, R. O'Kuinghttons, G. D. Riley, and M. Vertenstein, 2012: Coupling technologies for Earth System Modelling. *Geosci. Model Dev.*, 5, 1589-1596, 2012

Valcke, S., 2013: The OASIS3 coupler: A European climate modelling community software. *Geosci. Model Dev.*, 6, 373–388, doi:10.5194/gmd-6–373-2013

Wang, B., H. Wan, Z. Z. Ji, X. Zhang, R. C. Yu, Y. Q. Yu, and H. L. Liu, 2004: Design of a new dynamical core for global atmospheric models based on some efficient numerical methods. Science in China (A), 47, 4--21.

Wang, X.C., J.P. Liu, Y.Q. Yu, H.L. Liu, and L.J. Li, 2009: Numerical simulation of polar climate with FGOALS− g1.1. Acta Meteorologica Sinica, 67, 961– 972. (in Chinese)

Wu, F., H. Liu, W. Li, and X. Zhang, 2005: Effect of adjusting vertical resolution on the eastern equatorial Pacific cold tongue. Acta. Meteor. Sinica, 24, 1-12.

Xiao, C., 2006: Adoption of a two-step shape-preserving advection scheme in an OGCM and its coupled experiment. M.S. thesis, Institute of Atmospheric Physics, Chinese Academy of Sciences, 89pp. (in Chinese)

Yoshimura, H. and S. Yukimoto, 2008: Development of a Simple Coupler (Scup) for Earth System Modeling. *Meteorology and Geophysics*, 59, 19-29

Yu, R., 1994: A test for numerical weather prediction of real-time for china flood season precipitation in 1993 by a regional eta-coordinate model. Scientia Atmospherica Sinica, 18, 284-292.

Zhang, X., and X. Liang, 1989: A numerical world ocean general circulation model. Adv. Atmos. Sci., 6, 43-61.