Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-294
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 24 January 2018

Compact Modeling Framework v3.0 for high-resolution global ocean-ice-atmosphere models

Vladimir V. Kalmykov[1,3], Rashit A. Ibrayev[1,2,3,4], Maxim N. Kaurkin[1,3], and Konstantin V. Ushakov[1,3]

[1]Hydrometcentre of Russia., B. Predtechensky per., 11-13, Moscow, 123242, Russia
[2]Institute of Numerical Mathematics, Russian Academy of Sciences, ul. Gubkina, 8, Moscow, 119333, Russia
[3]Shirshov Institute of Oceanology, Russian Academy of Sciences, Nahimovskiy prospekt, 36, Moscow, 117997, Russia
[4]Moscow Institute of Physics and Technology (State University), Institutskiy per. 9, Dolgoprudny, Moscow oblast, 141700, Russia

Correspondence: Maxim N. Kaurkin (maksim.kaurkin@phystech.edu)

Abstract. We present new version of the Compact Modeling Framework (CMF3.0) developed for providing the software environment for stand-alone and coupled models of the Global geophysical fluids. The CMF3.0 designed for implementation high and ultra-high resolution models at massive-parallel supercomputers.

The key features of the previous CMF version (2.0) are mentioned for reflecting progress in our researches. In the CMF3.0 pure MPI approach with high-level abstract driver, optimized coupler interpolation and I/O algorithms is replaced with PGAS paradigm communications scheme, while central hub architecture evolves to the set of simultaneously working services. Performance tests for both versions are carried out. As addition a parallel realisation of the EnOI (Ensemble Optimal Interpolation) data assimilation method as program service of CMF3.0 is presented.

1 Introduction

As was pointed at the World Modeling Summit for Climate Prediction (Shukla, 2008) there is a general agreement that a much higher resolution of the major model components (atmosphere, ocean, ice, land) is a fundamental prerequisite for a more realistic representation of the climate system and more relevant predictions (e.g., extremes, convection, tropical variability, etc.).

Along with the development of models of individual components of the Earth system, the role of the instruments organizing their coordinated work (couplers and coupling frameworks) becomes more and more important. The coupler architecture depends on the complexity of the models, on the characteristics of interconnections between models and on computer environment. Historically the development of couplers follows the development of coupled atmosphere-ocean models. On some level of complexity the development of such software became an external problem relative to development of individual components of coupled model.

First coupled models used simple algorithms for coordination of components through the file system. There was no separate coupler component and communication between models was realized as a set of model procedures for I/O and interpolation between global model grids (today this method is used, for example, in INMCM4.0 climate model (Volodin et al., 2010)).

At the next stage, coupling of components was done through the separated central sequential hub using multiply executable approach (OASIS3 (Valcke et al., 2012), Community Climate System Model cpl3 (Craig et al., 2005)).

Coupling through shared file or sequential component is acceptable only for relatively low resolution models. Increasing the size of arrays and number of model components in system will inevitably become a "bottleneck" because of the memory and performance limitations of a single core and also problems related to global network communications. Therefore it was quite natural, that the next generation of couplers introduced parallelism in internal algorithms (Community Earth System Model cpl6 (Craig et al., 2005), OASIS4 (Redler et al., 2010)). Parallel coupler architecture solves computational problems for fine grids, but increases complexity of algorithms.

New architecture of coupler was introduced for CESM1.0 model in 2012 (Craig et al., 2012). In this system, coupled model has the form of a single executable and contains high-level driver that calls few component interfaces (init, run, finalize, etc.) This approach requires some reorganization of the components code and their representation in the interfaced style understandable by the driver, but simplifies model synchronization.

Coupled system can be launched as single executable without a standalone coupler whose functions are performed in parallel on a core subset of each model. Such solution has been proposed in OASIS3-MCT (Valcke, 2013).

Another important feature of the coupled system is the scheme of working with file system. In earlier versions it was carried out independently by each model in sequential way. Obviously, such master-process scheme (used in CESM cpl6, OASIS3, OASIS3-MCT) was limited by the RAM of a node. Increasing amounts of data lead to active development of parallel I/O algorithms. Since version 1.0 the CESM system utilizes PIO library (Dennis et al., 2012a) to establish parallel data writing to NetCDF format by every component through writing delegates. GFDL FMS (Balaji, 2012) system additionally suggests fully parallel data storage with file post processing at the end of the run.

Thus, we can point out the main characteristics of coupling frameworks:

1. coupling architecture (serial, parallel, with high-level driver or as a set of procedures); design of the system defines complexity of development and maintenance of the coupled model and implicitly establishes performance limitations;

2. I/O-module architecture (serial or parallel, synchronous or asynchronous); it should be considered as balance between simplicity of algorithms and necessary rate of I/O;

3. ease of use; the level of system abstraction defines convenience of user's work and the transparency of the overall coupled model;

4. performance; the choice of underlying algorithms defines computational rate of the coupled model;

2 Background

Our work began with the development of parallel version of the ocean dynamics model. The aim at that time was to work out a high resolution WOM. We had to solve several problems, namely halo update, mapping (interpolation) of external atmospheric data to the model grid, saving solution to a file, gathering diagnostics. It was obvious, that separation of numerical mathematics for solving ocean dynamics equations from low-level service procedures is necessary to write transparent code, which will allow us to develop independently the physical model as well as service procedures.

This approach has shown its advantages in solving the problem of coupling Global atmosphere and WOM for the medium and long-term weather forecasts at the Hydrometeorological center of Russia. The purpose was to create software capable to provide effective interaction of the high-resolution (of the order of 0.1 degrees) atmosphere and ocean models with the possibility to extend the coupled model for incorporating the ice and soil components. The components of the coupled model were the WOM (Ibrayev et al., 2012) based on INMIO ocean dynamics model (Ibrayev, 2001) and the SL-AV Global atmosphere model (Tolstykh et al., 2017). It turned out that for coupling of several models one should solve similar problems as for standalone model (mapping, I/O), but also has to provide synchronization and data-consistency during interpolation for simultaneously running components.

At the beginning of our study in 2012 there were several solutions for the creation of coupled models. It should be noted that the state-of-art couplers, such as of CESM (with coupler based on MCT (Larson et al., 2005) or ESMF (Collins et al., 2005) packages) and OASIS are fairly complex programs. CESM cpl 7 (Craig et al., 2012) is written for a predefined set of components and introducing a new model requires non-trivial changes and work with internal structures. Adding a new grid still requires self-constructing interpolation weights for it (CESM). Recent tests have shown that the computational costs of CESM coupler are quite significant 20% (Craig et al., 2012), nevertheless, good results in 2.6 SYPD (Simulated Years Per wall-clock Day) rate were achieved for ultra-high resolution Earth model (Dennis et al., 2012b).

The most popular version of OASIS, the OASIS3 is widely used by many research groups around the world. As was pointed, it contains a serial coupler, which is an obvious performance bottleneck in the system both in terms of constraints on memory and from the point of view of global communications. New version OASIS3-MCT (Valcke, 2013) solves the problem of sequential interpolation using MCT procedures, executed on subset of model cores. Unquestionable advantage of non-coupler design is the absence of interference in the user code. System contains master-process I/O, which obviously limits its use for large grids. Even with parallel I/O, the solution with a subset of service processes provides double load on model processes, which manage physical calculations, coupling actions and I/O-routines. Nevertheless, such behavior could be fully acceptable for non-intensive mapping and I/O runs.

According to proposals of Earth System Modeling conference (Valcke et al., 2012), today there are several trends in coupling software development, specifically: single executable modular architecture, parallel algorithms both for calculations and I/O, use of de-facto standard libraries like SCRIP (A Spherical Coordinate Remapping and Interpolation Package) (Jones, 1999) and NetCDF.

In the CMF2.0, a framework for the ocean-ice-atmosphere-land coupled modeling on massively-parallel architectures (Kalmykov and Ibrayev, 2013), we realized basic ideas.

In this paper we present two versions of Compact Modeling Framework (CMF), version 2.0 and 3.0. As the CMF2.0 was published only in Russian (Kalmykov and Ibrayev, 2013) here we outline the basics of that version. In the CMF2.0 we combine common proposals of Earth system modeling community and experimentation with low level algorithms. We concentrate on

Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-294
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 24 January 2018

single executable hub approach with high-level abstract driver, optimized interpolation algorithms, asynchronous I/O routines and tools for pre- and post- processing stages.

In CMF3.0 pure MPI approach is replaced with PGAS (Partitioned Global Address Space) paradigm communications scheme, while central hub architecture has evolved to SOA-like architecture with a set of simultaneously working services and a common task queue.

3 CMF2.0 overview

3.1 Architecture of the coupled system

The coupled model under the control of CMF2.0 runs as a single executable. At the beginning, MPI-communicator is divided on appropriate groups according to process decomposition and then all groups work simultaneously. The coupler performs some initialization routines and enters time cycle of requests. All physical components do the same logical steps, but call predefined abstract interfaces of models, for example, *ini_grid, ini_data, main_step, finalize*. Realizations of abstract interfaces represent specific behavior of the model: initializations and registration in system of all data that will be involved in the exchanges between models; main step of physics equations; finalizing procedures, etc. This behavior could be easily extended with interceptor methods (programming pattern *Template method* (Gamma et al., 1995)).

That is, to work in a coupled system, a user only has to define derived class of physical model adapter that inherits the base Component class and to realize abstract interfaces, filling it with calls to his internal model subroutines. This approach allows one to generate different executables for different coupled model combinations and restricts the user from any changes in the code outside of his derived class. Also the addition or modification of components does not affect the main program code, because it is written for abstract Component.

3.2 Coupler-model interactions

Each coupler core interacts only with a specific subset of the component cores, which means locality of data and communications during the interpolation process or I/O actions. The example of the coupled model for 3 coupler cores and 3 parallel components is shown in Figure 1.

All actions in the system are divided into few classes: save diagnostics, save control point, read file data, send/receive mapping, etc. All these events have their own periods and define different actions with data fields.

Since all events in the system could be predefined before start, the coupler during initialization gathers information about the time of all the events. This information is used to switch between requests of components without synchronization, while components asynchronously send data. Also persistent MPI operations (combination of *MPI_SEND_INIT* and *MPI_STARTALL*) are used for all events to save time on repeated communications. Pointers to arrays are stored at the registration stage, thus sending and receiving operations will be carried out without explicit user calls but based on defined periods. Combination of

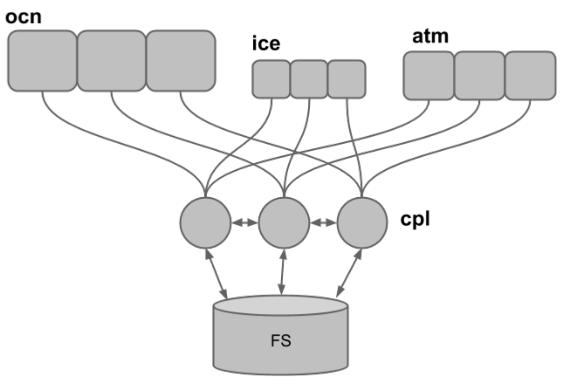

Figure 1. Architecture of coupled model in CMF2.0. For this example there are three components (ocean, atmosphere, ice) connected by 3-core coupler.

predefined time chain, persistent communications and pointer based asynchronous data sending provides maximal efficiency of data gathering and distribution.

3.3 Coupler: mapping

The interpolation algorithm uses weight files built at the off-line stage with the SCRIP package. At the run phase, components send data asynchronously, and a subset of component's cores works only with individual master core in the coupler.

The regridding process is performed on the coupler communicator and implemented as sparse matrix multiplication for the two cases, of the source and destination type (Craig et al., 2005) and currently supports logically-rectangular grids.

Process is performed in SCRIP format, where links connect destination and source cells (indices) with appropriate weights. Since single coupler core works only with a subdomain of the global model grid, it has only part of source data in memory, while other data should be gathered from neighbors by MPI-routines on every interpolation step. All necessary links are initialized at the beginning of run and are used at the calculation stage as persistent (Jacob et al., 2005).

At calculation stage every coupler core first prepares and sends source cells required by its neighbors, then it processes its local area and at last receives the missing data, completing the interpolation process. It is worth noting, that links are not sent directly, but as sorted unique cells vectors which allow one to avoid sending duplicated data. As a result, there is an overlap of computations and communication, which, in conjunction with persistent MPI transactions, determines a high efficiency of the algorithm.

Several ping-pong tests were carried out for interpolation system using coupled ocean-atmosphere model. The Test I condition, as in (Valcke et al., 2012), (Craig et al., 2012) is an exchange between two components with disabled physics routines. In our test, the ocean model sent 3 2D-fields every 2 hours to the atmosphere model and received 9 2D-fields every 1 hour.

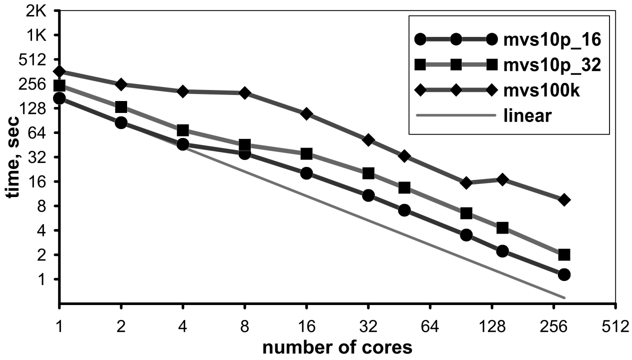

Figure 2. Time in seconds of Test I vs. number of coupler cores on MVS. Test for CMF2.0.

The ocean model has the 3600 × 1728 tripole grid and the atmosphere – 1600 × 864 latitude-longitude grid (grids were taken from current versions of ocean (Ibrayev et al., 2012) and atmosphere (Tolstykh et al., 2017) models). The mapping process consists of gathering data from the source component, regridding process inside of the coupler communicator and distributing the result to the destination component. The test has run for 10 model days, which corresponds to 240 × 9 atmosphere-ocean mappings and 120 × 3 ocean-atmosphere mappings. Sizes of communicators for ocean and atmosphere models were fixed by 1152 and 288 cores respectively. While not performing any work, they allow to simulate real communication load of overall system, reflecting packing, MPI sending, and unpacking costs. Thus charts present strong scalability of coupler interpolation algorithm. Performance is based on a standard Intel Fortran compiler.

Results were obtained on four supercomputers: MVS-100k, MVS-10p, BlueGene/P, BlueGene/Q (characteristics in Appendix). Test results for MVS supercomputers are presented in Figure 2. Two configurations - with 16 real and 32 virtual cores per node are shown for MVS-10p. The difference in the speed of their work is expected and is a result of increased communication load for a large number of cores per node. The graph shows good scalability with increasing number of coupler size. The best result of 1 second is achieved at 288 coupler cores.

It is clear that 20-40 coupler cores provide satisfactory speed for such problems, because ~10 seconds costs for 10 model days is a rather insignificant value for high-resolution ocean-atmosphere coupled modeling. The figure also shows failure of the sequential algorithm: even on the fast MVS-10p processors service activity takes about 200 seconds (work of the sequential algorithm is only possible with restriction that memory is allocated only for interpolation block, which is impossible in practice). Good coupler performance for one component-component connection is necessary for overall performance with growing number of components and their grid resolution. Test results for BlueGene supercomputers are presented in Figire 3. Timing of the algorithm is worse than on MVS-10p because of lower individual processor rate.

Test II was conducted for estimation of the increasing communication load associated with the growth of components' communicator sizes. Model grids were decomposed on much higher number of subdomains, increasing the cost of gather/distribute phase of test (mapping process inside coupler communicator remains the same). The results are shown in Figure 4 (curves

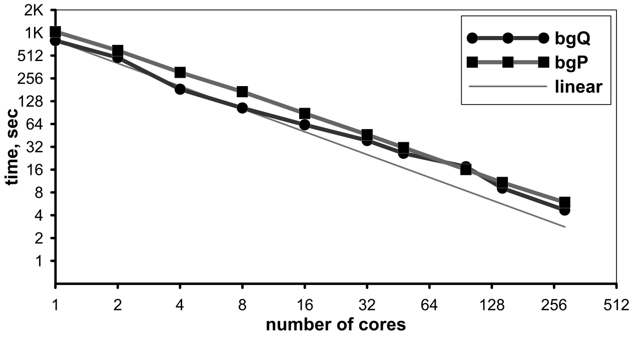

Figure 3. Time in seconds of Test I vs. number of coupler cores on BlueGene. Test for CMF2.0.

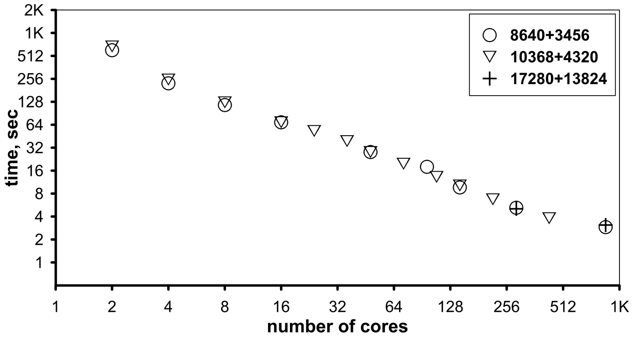

Figure 4. Time in seconds of Test II vs. number of coupler cores on BlueGeneQ system for different configurations of ocean and atmosphere models. Test for CMF2.0.

replaced by point symbols to improve the readability of the graph). Numbers of cores used for ocean and atmosphere models were equal to 8640 and 3456, 10368 and 4320, 17280 and 13824 respectively.

Graph shows two interesting facts. Firstly, single core coupler configuration does not work for Test II because of memory limitations. Secondly, increasing of communicational load affects performance only on small number of cores and at these points evaluation time is worse than in Test I. For example, test times for 2 coupler cores for model communicator sizes (8640, 3456) and (10368, 4320) are correspondingly 26% and 42% higher than for Test I communicator sizes (1152, 288). For 8 coupler cores this difference becomes 13% and 22% correspondingly. Since every coupler core communicates only with few component cores (that is, performs only local communication), increasing of the coupler communicator size leads to decreasing of communication overhead. As a result, even few tens of coupler cores are suitable to provide good performance of high-resolution mapping with huge sizes of model communicators.

3.4 Effectiveness of different I/O schemes and CMF I/O block

Since the speed of I/O-operations on supercomputers is usually slow, writing large amounts of data (such as control points which include several 3D-arrays) can take unacceptably long time. In case of frequent data dumps (e.g. forecast model with 1 hour period of saving data), time of calculations could be even comparable to time of I/O, thus it is very important to optimize file system interactions.

There are four known strategies for working with the file system: by master, direct parallel, by delegates and by external delegates.

Time of the experiment with solution subsequently recorded by master process scheme T_{total} consists of time for solving the model equations T_{run}, time for global data collection from *n* cores on a single master process $T_{gather\_glb}(n)$ and time for global array recording time T_{write_glb} :

$$T_{total} = T_{run} + T_{gather\_glb}(n) + T_{write\_glb} \tag{1}$$

Time of the experiment in the case of direct parallel scheme consists of computation time, time for recording by *n* cores to one file $T_{write\_lcl(n,1)}$ (or to different *f* files $T_{write\_lcl(n,f)}$ and then combining them $T_{u\_files(f)}$):

$$T_{total} = T_{run} + \begin{cases} T_{write\_lcl}(n,f), & \text{f=1} \\ T_{write\_lcl}(n,f) + T_{u\_files}(f), & \text{f>1} \end{cases} \tag{2}$$

In this case:

$$T_{write\_lcl(n,1)} > T_{write\_lcl(n,f)} \; if \; (f > 1), \tag{3}$$

since parallel writing to a single file is slower than to separate files. Time of the experiment in case of the delegate scheme consists of T_{run}, local data collection on *n* delegates $T_{gather\_dlg(n)}$ and time of parallel recording local arrays $T_{write\_dlg(n)}$:

$$T_{total} = T_{run} + T_{gather\_dlg}(n) + T_{write\_dlg}(n) \tag{4}$$

Recording time of parallel *n*-delegates scheme is not always better than that of sequential writing. Increase in the number of writing processes does not always increase the recording speed, but often reduces it. Such behavior is defined by actually installed supercomputer hardware. For example, presence of single I/O channel for whole cluster could serialize parallel I/O and, in opposite, special I/O-nodes allow one to achieve even some acceleration. At last, time of the experiment in the case of external *n*-delegates scheme can be equal to the time of calculation:

$$T_{total} = T_{run}, if \; (T_{write\_dlg}(n) < T_{run}). \tag{5}$$

Additionally, asynchronous component data sending makes time of the data collection phase insignificant, since calculating component is completely separated from writing delegates and can continue calculations without blocking: $T_{gather\_dlg(n)} \approx 0$. Limitation in expression appears due to the fact that the scheme allows a model to accelerate until writing time is less then time of performing the chunk of calculations. This limitation is controlled by required bandwidth:

$$B = D/T_{run}, \tag{6}$$

where D is the amount of data to be saved and depends on actually installed hardware.

Asynchronous scheme was incorporated in the latest version our framework. Since I/O algorithm is parallel, one can work with any grid sizes just increasing the number of I/O-cores. We have tested asynchronous I/O scheme with INMIO World ocean model for grid sizes $3600 \times 1800 \times 50$ (basic resolution), $5400 \times 2700 \times 50$ and $7200 \times 3600 \times 50$. Saving of control point, which includes four 3D and five 2D arrays, was successfully carried out by the coupler. In real applications rare saves could be fully overlapped by calculations on account of asynchronous messaging by model components.

3.5 Additional features

Apart from coupler, the framework also includes two helpful blocks. At preprocessor stage, CMF2.0 has got the off-line block for constructing SCRIP interpolation weights and preparation of the initial condition files. It also exploits *Template method* pattern and reduces all preparation actions (like grid definition) to realization of a few abstract interfaces in user derived class.

At the run stage, user can call different utility modules, like HaloUpdater, which is extensively used in WOM. It uses 4-neighbour scheme of any length/dimension/type update still handling diagonal cells. Impact of HaloUpdater on performance of WOM is described later.

Also CMF2.0 provides helpful tools for automatic building of various model combinations, makefile and skeleton class generation, preprocessing scripts, and other infrastructure actions.

4 CMF3.0

4.1 PGAS-communicator

CMF2.0 has shown itself as suitable framework for high-resolution coupled modeling, allowing us to perform long-term experiments which would be impossible without it. But CMF-2.0 still has several points for improvements. First of all, although pure MPI-based messaging is quite fast, it needs explicit work with sending and receiving buffers. Additionally, development of nested regional sea submodels becomes quite difficult using only MPI-routines. CMF2.0 test results showed that we can easily sacrifice some performance and choose better (but perhaps less computationally efficient) abstraction to simplify messaging routines.

We have chosen Global Arrays library (GA) (Nieplocha et al., 2006), which realizes PGAS paradigm of parallel communication. Development of this idea resulted in class Communicator, which encapsulates logic of working with GA and provides API (Application programming interface) for put/get operations of array patches from different components. Moreover, this API could be used not only for connections between nested models, but also as a communication mechanism between models and the coupler, because it allows one to hide all decomposition-to-decomposition problems rising in distributed applications. In CMF3.0 every array, which participates in intermodel communications, has its "mirror" in virtual global array. When model needs to perform some action, it puts/gets data to/from global array (this operation is local since global array internal distribution perfectly matches model decomposition) and continues calculations. Service components get array from *other side*, but

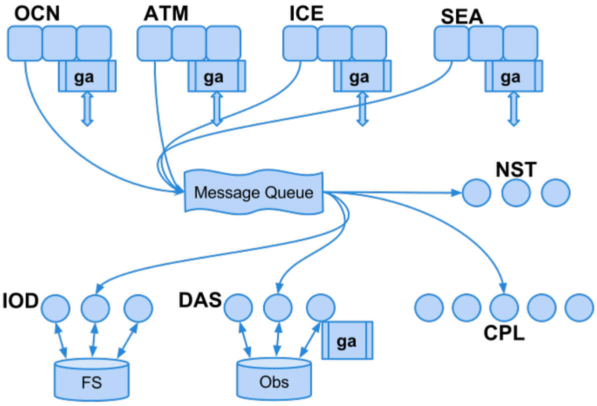

Figure 5. The architecture of the compact framework CMF3.0. There are four components in this example: ocean model (OCN), ice model (ICE), atmosphere model (ATM) and sea model (SEA). They send requests to the common message queue, where they are retrieved by coupler (CPL), data assimilation (DAS), input and output data (IOD), nesting (NST) services. The data itself is transferred through the mechanism of global arrays, which are also used for interprocessor communication in the components and services

this time on their own decomposition. For example, ocean component could store global array on 5000 cores with some 2D decomposition, while I/O procedure for saving this array could utilize only 4 cores and 1D decomposition.

4.2 SOA-architecture

As the complexity of coupled models is growing we need more easy and convenient way of connecting physical models together. SOA (Service Oriented Architecture) originally introduced for web applications, gives good pattern for component interactions. In CMF3.0 all models send their requests to common queue. Service components receive only messages they could process, get data from global arrays and perform required actions. Such architecture allows us to minimize dependencies between physical and service components and make development much easier. Moreover, since all services in CMF3.0 inherit base class Service it also allows one to easily add new. Now, we have four completely independent services: CPL (for field mapping), IOF (fast I/O device), IOS (slow I/O device), DAS (Data Assimilation Service).

CPL service represents the coupler from CMF2.0 and serves all mapping requests. It receives data using Communicator, performs interpolation and pushes data to destination global array (without request from receiving side). Although central coupler architecture of CMF2.0 allows one to collect all service operations on one external component and perform each of them in parallel, simultaneous requests sometimes can lead to inefficient usage of process time. For example, coupler in CMF2.0 can not perform parallel mapping and parallel I/O operations together. This is a disadvantage of all I/O-schemes which combine two or more actions on one process.

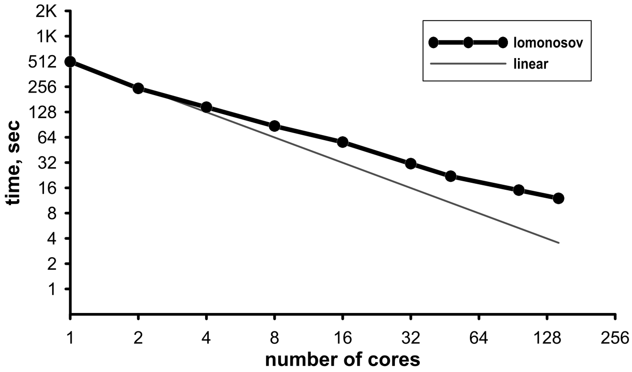

Figure 6. Time in seconds of Test I vs. number of coupler cores on Lomonosov supercomputer. Test for CMF3.0.

In CMF3.0 we decided to pick out a separate I/O-service, responsible only for working with file system. It should be noted, that one external I/O-service still solves only part of the problem, because in case of combining slow control points and fast diagnostics requests, model still would be blocked by the former. Therefore we break service into two parts: IOF and IOS – fast and slow I/O-devices (because of abstract structure of Service this separation is done via few lines of code). This mechanism provides flexible and fully asynchronous data storage, limited only by the bandwidth of file system as described earlier.

The further development of CMF has included data assimilation algorithms. For the ocean model we have added new DAS-service which implements logic of parallel data assimilation (Kaurkin et al., 2016a), (Kaurkin et al., 2016b).

4.3 Interpolation

Since logic of interpolation subroutines remains the same, we greatly simplify it by using of GA abstractions. Now all source data needed by destination cell is collected directly by Communicator routines. Optimizations regarded to ignore repeated cell requests are preserved. Disadvantage of using GA is decreasing in performance, since it can not provide persistent operations, overlapping of computations and communications and obviously has its own overheads. We take the same parameters and input files of Test I to compare CMF3.0 performance with CMF2.0 (Figure 6). Again, we measure overall timing including costs of sending event request, sending data, interpolation process and pushing data into final destination arrays. Therefore, timing reflects overall system overhead additional to timing of physical models.

Tests were conducted on Lomonosov supercomputer (characteristics in Appendix). Graph shows that results are worse than for CMF2.0 (Fig. 2) as expected, but linear scalability trend is preserved. Moreover, rate of 2-3 seconds per modeling day (on 20-50 CPL cores) is quite satisfactory for our practical purposes in high-resolution experiments.

4.4 Additional features

In CMF3.0 services are responding to messages during all run time, therefore model can send requests at every step of its time cycle. Nevertheless, sometimes we know schedule of actions (e.g. sending mapping every 2 hours, diagnostics every day and control point every month). CMF3.0 provides simple mechanism for generation of such scheduled actions. Now we have two types of event: NormalEvent, which represents uniform actions (like diagnostic saving, etc.) and SyncVarEvent, which allows one to synchronize with time variables in NetCDF-files (it is useful for prescribed forcing experiments like (Griffies et al., 2009)). Generators realize abstract class EventGenerator, so new specific generator subclasses could be easily added.

For asynchronous events (like exceptions in physics or changes in external data) user can directly call raise event for emergency data dump before termination or even change behavior of other models using special messages.

It is not difficult to migrate from CMF2.0 to CMF3.0. Only one file-adapter (about 200 lines of code) should be rewritten. It contains several procedures (*ini_main, make_step, finalize*, etc), besides global events and arrays (IO, remapping, etc) registered in it.

5 CMF examples of usage

There are several examples of using CMF for various numerical geophysical models:

1. High-resolution ocean dynamics modeling using WOM INMIO governed by CMF2.0 (Ibrayev et al., 2012),(Ushakov and Ibrayev, 2017).

2. Data assimilation using DAS of satellite observations and ARGO floats measurements for forecast and reanalysis with INMIO WOM governed by CMF3.0 (Kaurkin et al., 2016a).

3. There is a set of works with coupled atmosphere-ocean models for climate change modelling and numerical weather prediction at different spatial-time scales. The atmosphere model SL-AV (Tolstykh et al., 2017) and the WOM INMIO (Ibrayev et al., 2012) are coupled using CMF2.0 and CMF3.0 (Fadeev et al., 2016). The results of numerical experiments with the coupled model demonstrate agreement with observational data and show a possibility to use this model for probabilistic weather forecasts at time scales from weeks to year.

4. Nesting technology (as a CMF3.0 software NST-service) has been tested for the local model of Barents Sea (INMIO-based) with a resolution of 0.1°and the INMIO WOM with a resolution of 0.5°with different geophysical parametrizations (Koromyslov et al., 2017).

5. The first results of the seasonal variability simulation for Arctic and North Atlantic ocean waters and ice by the coupled model based on INMIO WOM and a sea-ice model CICE5.1 (Turner and Hunke, 2015) were obtained under CMF2.0. The numerical experiments have been performed in conditions of the CORE-II protocol (Ushakov et al., 2016).

5.1 INMIO World Ocean Model

As it was mentioned, one of the goals of the CMF is to provide tools for effective parallel calculations of stand-alone models. Historically, it was developed to provide efficient support for INMIO WOM. The INMIO WOM (Ibrayev et al., 2012) utilizes 2D-decompostition of the tripolar grid. Increasing number of cores decreases the number of performed operations for each

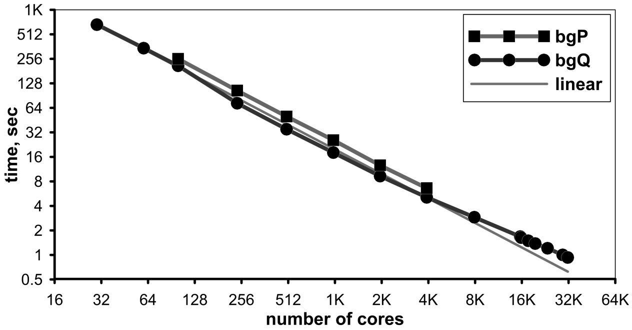

Figure 7. Time in seconds of INMIO WOM (governed by CMF2.0) 10 time steps vs. cores number of BlueGeneP supercomputer (Moscow State University. University) and BlueGeneQ supercomputer (IBM Research Center Thomas J. Watson).

process, because model uses explicit time schemes for horizontal operators which require only local halo updates. Therefore, limitations in scalability can only be associated with halo update routines and external blocks (e.g. in I/O).

Latest version of INMIO WOM model was fully integrated to CMF. At present, INMIO code consists of the hydrodynamical solver, while service work – intramodel communications (halo exchanges on tripolar and latitude-longitude grids) and work with the file system are delegated to the CMF utilities module. For experiments with CORE (Griffies et al., 2009) forcing two data models (reading CORE files) are also registered as separate atmosphere and land file components and the CMF coupler provides interpolation of their fields onto ocean high-resolution grid.

Scalability of INMIO WOM driven by CMF2.0 is shown in Figure 7. Maximum number of BlueGene/Q cores is equal to 32400. Parallel efficiency of the model for the amount of resources up to 8100 cores is 78 %. Obviously, smaller numbers of cores provide better values, but we are more interested in scalability of the program on perspective sizes of computational resources. Assuming that the time step of the model is 5 min. the result of the experiment lead to 5 simulated years per wall-clock day (SYPD) rate achieved on 20000 cores of BlueGene/Q supercomputer.

5.2 Coupled Global atmosphere - ocean model

The second application of the framework was the numerical experiment with coupled INMIO WOM (Ibrayev et al., 2012) and SL-AV Global atmosphere model (Tolstykh et al., 2017). The SL-AV atmosphere model with horizontal resolution $0.9^\circ \times 0.72^\circ$ and 28 vertical levels and INMIO WOM with resolution 0.5° and 49 vertical levels were coupled into the single program using the CMF2.0 system. Short-wave and long-wave radiation in the SL-AV model are computed with the time-step of 1 hour. Time evolution of the sea-ice surface temperature is described in the same way as in prescribed ocean experiments. The restriction of spatio-temporal resolution was implied by available computer resources and not by restrictions of CMF.

Prognostic model calculations were carried out with a time step of 6 min. for the ocean model and 3.6 min. for the atmosphere. The initial state of the ocean was a control point obtained by spin-up of standalone ocean model. Atmosphere started with objective analysis of the Hydrometeorological Center of Russia. In coupled regime every 72 min. 9 2-D arrays were transferred from the atmosphere to the ocean (components of wind stress, shortwave and long wave radiation, fluxes of sensible and

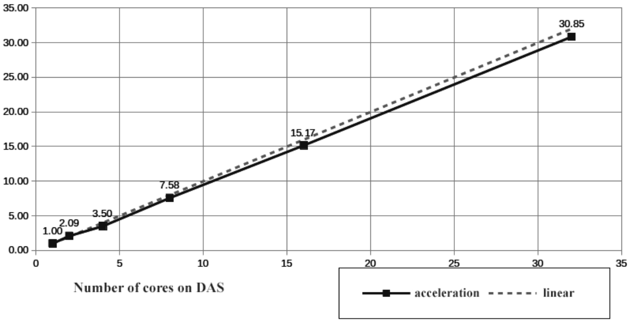

Figure 8. Scalability of EnOI method in the context of the CMF3.0 at the assimilation of 10^4 points on the Lomonosov supercomputer (Moscow State University).

latent heat, precipitation, evaporation, air temperature at 2 m), each 144 min. 3 2-D arrays from the ocean to the atmosphere (surface temperature and concentration of ice and the temperature of the upper ocean gridbox). Ice model was built into the ocean model , land model – into the atmosphere model. Coupled model works stably and along with seasonal distribution characteristics of monthly data fields reproduces enough thin elements of atmospheric and oceanic circulation.

SYPD of the coupled model on the MVS-10p supercomputer is equal to 0.75 for configuration ocean (1152 cores) - atmosphere (288 cores) - coupler (16 cores). At the moment, the maximum computational resources available for atmosphere model is limited due to the one-dimensional latitudinal model grid decomposition.

5.3 Data assimilation using DAS

As well as any service of the CMF3.0, data assimilation is performed on separate computing cores. This allows to structurize the Earth modeling system better, in order to make each software component solve its own problem. At the same time the model of the ocean does not take part in the data assimilation. Only results of the ocean modeling in the form of vector elements of the ensemble are used. On their basis the covariance matrices are approximated. Data from the ocean model is sent to the service (usually once a modeling day) without using a file system (through cluster interconnect). More over all matrix-vector operations are calculated parallel (shared-memory) using BLAS and LAPACK functions through the Global Arrays (GA) toolkit (Kaurkin et al., 2016a).

Due to the effective implementation of the EnOI method as a parallel software service DAS, the solution of the data assimilation problem is scaled almost linearly (Fig. 8). So, the assimilation of 10^4 observation points of satellite data on the 16 processor cores takes about 20 seconds instead of 5 minutes on a single core, which would be comparable to the time spent on daily ocean model forecast for 200 cores.

6 Conclusions and future work

We have presented an original modeling framework CMF3.0 developed as our initial step to high resolution modeling. The key part of it, coupler, has a sufficiently small code size for such programs (about 5000 lines of code with unit tests) and is able to manage the main parallel problems of the coupled modeling - synchronization, regridding and I/O. The coupled model follows a single executable design with main program independent of components code, and coupler dealing with all service operations. New version of CMF3.0 utilizes SOA-design which allows one to divide coupler responsibilities into small separate services and easy plug and unplug them. PGAS messaging greatly simplifies all low level interprocess communications.

Tests for parallel mapping efficiency were carried out on four modern supercomputer architectures. Tests show a near linear strong scalability of the overall communication system and regridding procedure. Satisfactory speed results could be achieved already on 20-40 coupler cores even dealing with grids of high resolution (0.1°and 0.225°). I/O tests proved the ability of the coupler delegate scheme to handle with huge amounts data. As expected, new CPL 3.0 version has lower absolute performance, but greatly simplifies code and preserves linear trend of scalability and suitable timing (2-3 seconds per modeling day on 20-50 coupler cores) for high-resolution modeling.

Originally designed for WOM support, CMF was used for overall ocean physics development and for long-term modeling of 0.1°INMIO WOM. Also first middle term forecasts of coupled 0.25°ocean - 0.225°atmosphere model became possible due to developed framework.

We think that conducted experiments cover introduction phase of our high-resolution modeling plans and CMF3.0 is ready to further evolution and establishing closer collaboration with community projects. Our future work will cover development of DAS services for operational model forecast and integrating some community instruments (like EMSF or MCT) for support of unstructured grids in perspective models (Volodin et al., 2010).

Code availability. The code of the CMF3.0 and of CMF2.0 (distributed under GPLv2 licence) is available on http://model.ocean.ru (after registration).

Appendix A: Supercomputer configurations used

MVS-100k and MVS-10p are parts of the Joint Supercomputer Center of the Russian Academy of Sciences (jscc.ru). MVS-100k consists of 1460 modules (11680 processor cores). Basic computing module is an HP Proliant server, containing two quad-core Intel Xeon, running at 3 GHz on 8 GB RAM memory. Computational modules are interconnected with Infiniband DDR. The computer MVS-10p includes 207 nodes. Each node incorporates 2 processors Intel Xeon E5-2690 (16 cores on 2.90 GHz), 64 GB of RAM, two Intel Xeon Phi coprocessor 7110H. Compute nodes are combined into FDR Infiniband network.

Supercomputer BlueGene/P is located on the faculty of Computational Mathematics and Cybernetics, Moscow State University and consists of 2048 compute nodes. Each node is a 4 core PowerPC 450 (2 Gb RAM, 850 MHz). Nodes are networked with 3D-torus topology (5.1 GB/s, DMA).

Computer BlueGene/Q is located in the IBM Thomas J. Watson Research Center and consists of several racks. Every 2 racks have 2048 computational nodes with 16 cores. The core is a PowerPC (16 GB RAM, 1.6 GHz). Nodes are networked with 5D-torus topology (40 GB/s, DMA).

Supercomputer Lomonosov is located in Lomonosov Moscow State University and consists of more than 50000 cores. We have used partition with 8 core nodes (2 x Intel Xeon 5570 Nehalem, 12 GB, 2.9 Ghz). Computational modules are interconnected with Infiniband QDR.

Acknowledgements. This work was supported by the Russian Science Foundation (project no. 14-37-00053) and performed at the Hydrometeorological Research Centre of the Russian Federation.

References

Balaji, V.: The Flexible Modeling System, pp. 33–41, Springer Berlin Heidelberg, Berlin, Heidelberg, https://doi.org/10.1007/978-3-642-23360-9_5, 2012.

CESM: CESM1.2 User guide., NCAR, http://www.cesm.ucar.edu/models/cesm1.2/cesm/doc/usersguide/ug.pdf, 2013.

Collins, N., Theurich, G., DeLuca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C., and da Silva, A.: Design and Implementation of Components in the Earth System Modeling Framework, The International Journal of High Performance Computing Applications, 19, 341–350, https://doi.org/10.1177/1094342005056120, 2005.

Craig, A. P., Jacob, R., Kauffman, B., Bettge, T., Larson, J., Ong, E., Ding, C., and He, Y.: CPL6: The New Extensible, High Performance Parallel Coupler for the Community Climate System Model, The International Journal of High Performance Computing Applications, 19, 309–327, https://doi.org/10.1177/1094342005056117, 2005.

Craig, A. P., Vertenstein, M., and Jacob, R.: A new flexible coupler for earth system modeling developed for CCSM4 and CESM1, The International Journal of High Performance Computing Applications, 26, 31–42, https://doi.org/10.1177/1094342011428141, 2012.

Dennis, J. M., Edwards, J., Loy, R., Jacob, R., Mirin, A. A., Craig, A. P., and Vertenstein, M.: An application-level parallel I/O library for Earth system models, The International Journal of High Performance Computing Applications, 26, 43–53, https://doi.org/10.1177/1094342011428143, 2012a.

Dennis, J. M., Vertenstein, M., Worley, P. H., Mirin, A. A., Craig, A. P., Jacob, R., and Mickelson, S.: Computational performance of ultra-high-resolution capability in the Community Earth System Model, The International Journal of High Performance Computing Applications, 26, 5–16, https://doi.org/10.1177/1094342012436965, 2012b.

Fadeev, R. Y., Ushakov, K. V., Tolstykh, M. A., Ibrayev, R. A., and Kalmykov, V. V.: Coupled atmosphere–ocean model SLAV–INMIO: implementation and first results, Russian Journal of Numerical Analysis and Mathematical Modelling, 31, 329–337, https://doi.org/10.1515/rnam-2016-0031, 2016.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J.: Design Patterns: Elements of Reusable Object-oriented Software, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

Griffies, S., Biastoch, A., Böning, C., Bryan, F., Danabasoglu, G., Chassignet, E., England, M., Gerdes, R., Haak, H., Hallberg, R., Hazeleger, W., Jungclaus, J., Large, W., Madec, G., Pirani, A., Samuels, B., Scheinert, M., Gupta, A., Severijns, C., Simmons, H., Treguier, A., Winton, M., Yeager, S., and Yin, J.: Coordinated Ocean-ice Reference Experiments (COREs), Ocean Modelling, 26, 1–46, https://doi.org/10.1016/j.ocemod.2008.08.007, 2009.

Ibrayev, R. A.: Model of enclosed and semi-enclosed sea hydrodynamics, Russian Journal of Numerical Analysis and Mathematical Modelling, 16, 291–304, https://doi.org/10.1515/rnam-2001-0404, 2001.

Ibrayev, R. A., Khabeev, R. N., and Ushakov, K. V.: Eddy-resolving 1/10° model of the World Ocean, Izvestiya, Atmospheric and Oceanic Physics, 48, 37–46, https://doi.org/10.1134/S0001433812010045, 2012.

Jacob, R., Larson, J., and Ong, E.: M × N Communication and Parallel Interpolation in Community Climate System Model Version 3 Using the Model Coupling Toolkit, The International Journal of High Performance Computing Applications, 19, 293–307, https://doi.org/10.1177/1094342005056116, 2005.

Jones, P. W.: First- and Second-Order Conservative Remapping Schemes for Grids in Spherical Coordinates, Monthly Weather Review, 127, 2204–2210, https://doi.org/10.1175/1520-0493(1999)127<2204:FASOCR>2.0.CO;2, 1999.

Kalmykov, V. V. and Ibrayev, R. A.: A framework for the ocean-ice-atmosphere-land coupled modeling on massively-parallel architectures, Vychisl. Metody Programm., 14, 88–95, http://mi.mathnet.ru/vmp156, (in Russian), 2013.

Kaurkin, M., Ibrayev, R., and Koromyslov, A.: EnOI-Based Data Assimilation Technology for Satellite Observations and ARGO Float Measurements in a High Resolution Global Ocean Model Using the CMF Platform, vol. 687 of *Communications in Computer and Information Science*, pp. 57–66, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-319-55669-7_5, 2016a.

Kaurkin, M. N., Ibrayev, R. A., and Belyaev, K. P.: ARGO data assimilation into the ocean dynamics model with high spatial resolution using Ensemble Optimal Interpolation (EnOI), Oceanology, 56, 774–781, https://doi.org/10.1134/S0001437016060059, 2016b.

Koromyslov, A., Ibrayev, R., and Kaurkin, M.: The technology of nesting a regional ocean model into a Global one using a computational platform for massively parallel computers CMF, vol. 793 of *Communications in Computer and Information Science*, pp. 241–250, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-319-71255-0_19, 2017.

Larson, J., Jacob, R., and Ong, E.: The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models, The International Journal of High Performance Computing Applications, 19, 277–292, https://doi.org/10.1177/1094342005056115, 2005.

Nieplocha, J., Palmer, B., Tipparaju, V., Krishnan, M., Trease, H., and Aprà, E.: Advances, Applications and Performance of the Global Arrays Shared Memory Programming Toolkit, The International Journal of High Performance Computing Applications, 20, 203–231, https://doi.org/10.1177/1094342006064503, 2006.

Redler, R., Valcke, S., and Ritzdorf, H.: OASIS4 – a coupling software for next generation earth system modelling, Geoscientific Model Development, 3, 87–104, https://doi.org/10.5194/gmd-3-87-2010, 2010.

Shukla, J.: World Modelling Summit for Climate Prediction. WCRP-131, WMO/TD-No.1468., Uk, European Centre for Medium-Range Weather Forecasts, 2008.

Tolstykh, M., Shashkin, V., Fadeev, R., and Goyman, G.: Vorticity-divergence semi-Lagrangian global atmospheric model SL-AV20: dynamical core, Geoscientific Model Development, 10, 1961–1983, https://doi.org/10.5194/gmd-10-1961-2017, https://www.geosci-model-dev.net/10/1961/2017/, 2017.

Turner, A. K. and Hunke, E. C.: Impacts of a mushy-layer thermodynamic approach in global sea-ice simulations using the CICE sea-ice model, Journal of Geophysical Research: Oceans, 120, 1253–1275, https://doi.org/10.1002/2014JC010358, http://dx.doi.org/10.1002/2014JC010358, 2015.

Ushakov, K. V. and Ibrayev, R. A.: Simulation of the global ocean thermohaline circulation with an eddy-resolving INMIO model configuration, IOP Conference Series: Earth and Environmental Science, 96, 012 007, https://doi.org/10.1088/1755-1315/96/1/012007, 2017.

Ushakov, K. V., Grankina, T. B., Ibrayev, R. A., and Gromov, I. V.: Simulation of Arctic and North Atlantic ocean water and ice seasonal characteristics by the INMIO-CICE coupled model, IOP Conference Series: Earth and Environmental Science, 48, 012 013, https://doi.org/10.1088/1755-1315/48/1/012013, 2016.

Valcke, S.: The OASIS3 coupler: a European climate modelling community software, Geoscientific Model Development, 6, 373–388, https://doi.org/10.5194/gmd-6-373-2013, 2013.

Valcke, S., Balaji, V., Craig, A., DeLuca, C., Dunlap, R., Ford, R. W., Jacob, R., Larson, J., O'Kuinghttons, R., Riley, G. D., and Vertenstein, M.: Coupling technologies for Earth System Modelling, Geoscientific Model Development, 5, 1589–1596, https://doi.org/10.5194/gmd-5-1589-2012, 2012.

Volodin, E. M., Dianskii, N. A., and Gusev, A. V.: Simulating present-day climate with the INMCM4.0 coupled model of the atmospheric and oceanic general circulations, Izvestiya, Atmospheric and Oceanic Physics, 46, 414–431, https://doi.org/10.1134/S000143381004002X, 2010.