Geosci. Model Dev., 10, 3297–3308, 2017
https://doi.org/10.5194/gmd-10-3297-2017

Geoscientific Model Development
Open Access
EGU

Development and performance of a new version of the OASIS coupler, OASIS3-MCT_3.0

Anthony Craig, Sophie Valcke, and Laure Coquart

CECI, Université de Toulouse, CNRS, CERFACS, 42 Av. G. Coriolis, 31057 Toulouse Cedex 01, France

Correspondence to: Sophie Valcke (valcke@cerfacs.fr)

Received: 13 March 2017 – Discussion started: 7 April 2017
Revised: 2 July 2017 – Accepted: 24 July 2017 – Published: 8 September 2017

Abstract. OASIS is coupling software developed primarily for use in the climate community. It provides the ability to couple different models[1] with low implementation and performance overhead. OASIS3-MCT is the latest version of OASIS. It includes several improvements compared to OASIS3, including elimination of a separate hub coupler process, parallelization of the coupling communication and runtime grid interpolation, and the ability to easily reuse mapping weight files. OASIS3-MCT_3.0 is the latest release and includes the ability to couple between components running sequentially on the same set of tasks as well as to couple within a single component between different grids or decompositions such as physics, dynamics, and I/O. OASIS3-MCT has been tested with different configurations on up to 32 000 processes, with components running on high-resolution grids with up to 1.5 million grid cells, and with over 10 000 2-D coupling fields. Several new features will be available in OASIS3-MCT_4.0, and some of those are also described.

1 Introduction

OASIS is coupling software developed primarily for the climate community. OASIS was originally an abbreviation for "Ocean Atmosphere Sea Ice Soil", but the capabilities provided by OASIS are not restricted to just those kinds of models, so the name OASIS now represents a project to develop general coupling software. It is in relatively wide use, especially in European-based modeling efforts. It is one of a number of coupling infrastructure packages (Valcke et al., 2016) that are focused on standard and reusable methods to support coupling requirements like interpolation and communication of data between different models and different grids. OASIS is maintained and managed by the Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique (CERFACS) and the Centre National de la Recherche Scientifique (CNRS) in France. It is a portable set of Fortran 77, Fortran 90, and C routines. Low intrusiveness, portability, and flexibility are key OASIS design concepts. The current version of the software, OASIS3-MCT, is a coupling library that is compiled and linked to the component models. Its primary purpose is to interpolate and exchange the coupling fields between or within components to form a coupled system. OASIS3-MCT supports coupling of fields on grid types commonly used in climate science via a put–get approach, which means components make subroutine calls to send (put) or receive (get) data from within the component code directly. A separate top-level driver to control system sequencing is not required to use OASIS3-MCT, but a handful of subroutine calls must be added to the code to initialize the coupling, define grids, define decompositions (partitions), define coupling fields, and put and get variables between components. OASIS3-MCT leverages a text input file called the *namcouple* file to configure the interactions between components. Mapping (also known as remapping, regridding, or interpolation), time transformations, and the ability to read or write coupling data from disk are supported in OASIS3-MCT.

OASIS development began in 1991 and the first version, OASIS1, was used 2 years later in a 10-year coupled integration of the tropical Pacific (Terray et al., 1995). In the intervening decades, OASIS2 and OASIS3 were released. The history of OASIS development is well documented (Valcke, 2013). With OASIS3, the coupled models always had to run

[1] Within the text, we use "model" in the sense of a "numerical model".

concurrently as separate executables on different MPI tasks and all coupling fields passed through a separate central hub coupler component that also ran concurrently. OASIS3 allowed parallel coupling of parallel models on a per-field basis by gathering each parallel field in the source model to a single process on the hub where operations such as mapping and time averaging were executed, and the field was then scattered to the destination model. OASIS3 generated mapping weights on a single process at initialization using the SCRIP library (Jones, 1999) from the grid information specified by the component models.

A first attempt to design and develop a fully parallel coupler was started in the framework of the EU FP5 PRISM and FP7 IS-ENES1 projects (see https://is.enes.org), and that led to the development of OASIS4 (Redler et al., 2010). In particular, OASIS4 included a library that performed a parallel calculation for generation of the mapping weights and addresses needed for the interpolation of the coupling fields. This version had several other features such as the use of an xml file for specifying the configuration information. OASIS4 was used by Météo-France, ECMWF, KNMI, and MPI-M in the framework of the EU GEMS project for 3-D coupling between atmospheric dynamic and atmospheric chemistry models (Hollingsworth et al., 2008); it was also used by SMHI, AWI, and the BoM in Australia for ocean–atmosphere 2-D regional and global coupling. But OASIS4 had limited success and its development was stopped in 2011 after a performance analysis determined some fundamental weaknesses in its design, in particular with respect to the support of unstructured grids.

With OASIS3-MCT, a different approach was taken to improve the parallel performance and to address new requirements. It extends the widely used and distributed OASIS3 version of the model. This paper describes the development of OASIS3-MCT from OASIS3 to the current 3.0 release and also introduces some new features expected in the next 4.0 release. The initial requirements of OASIS3-MCT were to improve the parallel performance of the coupling, implement an ability to read in mapping weights to mitigate the cost of weight generation, support next-generation grids such as high-resolution unstructured grids running on high processor counts, and to add those features while retaining the basic OASIS3 application programming interfaces (APIs) and *namcouple* file to support backwards compatibility.

To meet these requirements, a number of changes were made. First, a portion of the underlying communication implementation was replaced with the Model Coupling Toolkit (MCT) software package (Larson et al., 2005) developed by the Argonne National Laboratory. This implementation is transparent to the user, as MCT methods and data types are only used within the OASIS3-MCT infrastructure to support parallel mapping and parallel redistribution. Second, the ability to specify pre-defined mapping files was added. Mapping files can now be generated offline using a diverse set of packages, such as SCRIP, ESMF (Theurich et al., 2016), or any locally developed methods. Third, the OASIS3 hub coupler was deprecated and is no longer needed or implemented. Transforms are carried out on the component processes, and data are transferred directly between components via MCT. These features were released in OASIS3-MCT_1.0 in 2012 (Valcke et al., 2012) and, because of backwards compatibility, OASIS3 users could upgrade easily to OASIS3-MCT.

With the release of OASIS3-MCT_3.0 in 2015 (Valcke et al., 2015), several new features were added to the coupler. OASIS3-MCT_3.0 extends the ability to couple components running concurrently and adds support for coupling within a component for grids and fields defined on overlapping or partially overlapping sets of tasks, such as between physics and dynamics modules within an atmospheric model or to and from an I/O module. OASIS3-MCT_3.0 also allows a component to define grids, partitions, and coupling fields on subsets of its tasks, and it comes with a graphical user interface (GUI) to generate the *namcouple* file.

The next section, titled Implementation, describes these features in greater detail. Section 3 provides performance and memory scaling results from OASIS3-MCT_3.0 as well as some initial results for features expected in OASIS3-MCT_4.0, and Sect. 4 provides conclusions and a summary.

2 Implementation

As discussed in the introduction, OASIS3-MCT development started with the objective to keep the OASIS3 general design. The requirements of OASIS3-MCT were focused on improved parallel performance, including parallel mapping and parallel data coupling, the ability to efficiently support unstructured grids, the ability to specify pre-defined mapping files to mitigate the serial cost of generating mapping weights on the fly, and backwards compatibility in usage of both the *namcouple* file and the OASIS3 APIs. A summary of the changes between OASIS3 and OASIS3-MCT_3.0 is provided in Appendix A as well as an initial list of features expected in OASIS3-MCT_4.0.

2.1 General architecture

To accomplish these tasks efficiently and in a timely manner, the MCT developed by the Argonne National Laboratory (Larson et al., 2005) was incorporated into OASIS3 to support parallel matrix vector multiplication and parallel distributed exchanges. Its design philosophy, based on flexibility and minimal invasiveness, is consistent with the approach taken in OASIS. MCT has proven parallel performance and is one of the underlying coupling software libraries used in the National Center for Atmospheric Research Community Earth System Model (NCAR CESM) (Jacob et al., 2005; Craig et al., 2012).

MCT handles two primary tasks in OASIS3-MCT: the parallel transfer of data from a source model to a destination model, and interpolation of fields between decomposed grids. At the present time, these two steps are independent and both are largely performance limited by MPI communication cost at moderate to high processor counts due to the data rearrangement in both. Data communication and mapping rearrangement are handled internally in OASIS3-MCT via MCT routers.

Another significant change in the OASIS3-MCT implementation compared to OASIS3 is that a separate hub coupler executable running on its own processes is no longer needed. Accumulation, temporal lagging, mapping, and other transforms are carried out in the OASIS3-MCT coupling layer on the model processes in parallel using temporary memory to store data as needed. Compared to OASIS3, which required an all-to-one communication, interpolation on the single hub process, and a one-to-all communication to couple fields, OASIS3-MCT requires just one parallel all-to-all communication between the source and destination processes and one parallel mapping which includes a rearrangement of the data on the source or destination processes. In addition, the memory needed in the infrastructure in OASIS3-MCT is much more scalable.

2.2 Coupling

OASIS3-MCT fundamentally supports coupling of 2-D logically rectangular fields, but 3-D fields and 1-D fields are also supported using a 1-D degeneration of the grid structure. If the user provides a set of pre-calculated weights, OASIS3-MCT will be able to interpolate any type of 1-D, 2-D, or 3-D field, but the capability to calculate the mapping weights by the coupler is only available for 2-D fields on the sphere.

Another new feature is the option to couple multiple fields as a single coupling operation. This is supported for fields for which the coupling options defined in the *namcouple* file are identical. This can improve performance because rather than mapping and coupling fields one at a time, the mapping and coupling can be aggregated over multiple fields. Coupling multiple fields at once is accomplished by specifying a list of colon-delimited fields in the *namcouple* file on both the source and destination sides. In this implementation, the get and put calls in the model are still individual calls on individual fields, but the coupling layer will aggregate the multiple fields specified in the *namcouple* file into a single step. On the put side, the multiple fields are not mapped or sent until all of the individual put calls are made. On the get side, the multiple fields are received and mapped on the first get call and then subsequent get calls just copy in fields that were received earlier. A user can quickly switch between coupling single and multiple fields just by changing the *namcouple* input file.

One additional feature available in the current development version and that will be released with the next official version, OASIS3-MCT_4.0, is the ability to couple a bundle of 2-D fields via extensions to the OASIS calling interfaces. An extra dimension is supported in the variable definition and in the get and put field arrays. In this case, a user can treat a bundled 2-D field as a single field in the system, while the underlying implementation treats it just like a multiple field coupling.

2.3 Interpolation

Mapping weight files can either be read directly or generated at run-time, on one processor, using the same serial method based on SCRIP as existed in OASIS3. In OASIS3-MCT, the weights are read serially by the root process and distributed to other processes in reasonable chunks, currently set to 100 000 weights at a time to limit memory use on the root process. For the interpolation, OASIS3-MCT creates a simple 1-D decomposition of the source grid on the destination processes or vice versa. Fields are then either remapped to the destination grid on the source processes and then sent to the destination processes or sent to the destination processes and then remapped to the destination grid. The user is able to specify whether the source or destination processes are used for remapping via an optional setting in the *namcouple* file. That choice will generally be made based on mapping performance and depends on the relative size of the grids, the number of weights, and the process counts of the source and destination models. In OASIS3-MCT_4.0, a new option is expected that may reduce the mapping rearrangement cost by choosing a more efficient decomposition of the source grid on the destination processes (or vice versa) compared to the current default 1-D decomposition.

Users also have an additional option to set the implementation of the underlying mapping algorithm. The *bfb* option will enforce an order of operations that will be bit-for-bit identical on different process counts. It does this by distributing the mapping weights on the destination decomposition and then redistributing the source coupling field grid point values to the destination processes before applying the mapping weights. This ensures operation order is independent of decomposition. The *sum* option does the opposite. It distributes the mapping weights on the source decomposition and then computes partial sums of the destination field on the source decomposition, before rearranging them to the destination decomposition and adding up the partial sums. This does not guarantee identical order of operations on different process counts and decompositions. In both approaches, the same number of floating operations are carried out as defined by the mapping weights. The main difference between the *bfb* and *sum* strategies is that in *bfb* mode, the source field is rearranged onto the destination distribution before the mapping weights are applied, while in *sum* mode, the mapping weights are applied on the source decomposition to form partial sums of the destination field, and then the partial sums are rearranged. From the performance point of view, it is gener-

ally better to use the method that rearranges the field on the grid that contains the fewest grid cells, to minimize the communication cost. But, of course, if bit-for-bit reproducibility on different core counts is required, then the *bfb* mode should be chosen.

2.4 Conservation

With OASIS3-MCT, the optional CONSERV transform has been refactored. In OASIS3, this operation was always performed on a single process. In OASIS3-MCT, this operation is now performed in parallel on the source or destination processes. The CONSERV operation computes global sums of the source and destination fields and applies corrections to the decomposed mapped field in order to conserve area-integrated field quantities. There are two options for computing the global sums in OASIS3-MCT_3.0. The first, *bfb*, gathers the fields onto the root process to compute the global sums in an ordered fashion that guarantees bit-for-bit identical results regardless of the number of cores or decomposition of the field. (Note that both the CONSERV operation and the underlying mapping algorithm setting share a common flag, *bfb*, but these two settings are completely independent.) The second CONSERV option, *opt*, carries out a local double precision sum of the field and then does a scalar reduction to generate the global sums. This will typically introduce a round-off difference in the results when changing process counts or decomposition, but is much faster. However, the *opt* option will be bit-for-bit reproducible if the same number of processes and decomposition are used between different runs.

In the OASIS3-MCT_4.0 release, three new options (*lsum16*, *ddpdd*, and *reprosum*) will be added to compute the global sums in CONSERV. At the same time, *opt* will be renamed *lsum8*, while *bfb* will be renamed *gather*. The rest of this paper will use the OASIS3-MCT_4.0 naming convention for CONSERV options. The first new global sum method, *lsum16*, works just like *lsum8* but uses quadruple precision to compute the local sums and to carry out the scalar reduction. The cost will be higher than *lsum8*, but there is a greater chance that results will be bit-for-bit for different decompositions than *lsum8*. The *ddpdd* is a parallel double–double algorithm using a single scalar reduction (He and Ding, 2001). It should behave between *lsum8* and *lsum16* with respect to performance and reproducibility. The third new algorithm, *reprosum*, is a fixed point method based on ordered double integer sums that requires two scalar reductions per global sum (Mirin and Worley, 2012). The cost of *reprosum* will be higher than some of the other methods, but it is expected to produce bit-for-bit results on different task counts except in extremely rare cases, and the cost should be significantly less than the *gather* method.

2.5 Concurrency, process layout, and sequencing

The ability to couple fields within one executable running on partially overlapping tasks was added in OASIS3-MCT_3.0. A number of new capabilities had to be implemented to support this feature including the ability to define grids, partitions, and coupling fields on subsets of component tasks. There also had to be a major update in the handling of MPI communicators within the infrastructure. These changes are transparent to the user. This allows, within a single model, different sets of MPI tasks to define multiple grids, multiple decompositions (partitions), and different coupling fields. These new features and updates provide the flexibility needed to couple fields between components or within a component.

Figure 1 provides a schematic of the type of coupling that can be carried out between and within components in OASIS3-MCT_3.0. Executables are defined as separate binaries that are launched independently at startup, components are defined as separate sets of tasks within an executable, and grids can be defined on all tasks or on a subset of tasks within a component. Each task will be associated with only one executable and one component in any application, but multiple grids and decompositions can exist across overlapping tasks within a component. While OASIS3-MCT supports both single and multiple executable configurations, the coarsest level of concurrency in the system is the component.

In Fig. 1, an example schematic is presented that shows how two executables, exe1 and exe2, run concurrently on separate sets of MPI tasks (0–5 for exe1 and 6–37 for exe2). Executable exe1 includes only one component, comp1, that has coupling fields defined on only one grid, grid1 (decomposed on all six tasks). Executable exe2 includes three components, comp2, comp3, and comp4, running concurrently on tasks 6–11, 12–33, and 34–37, respectively. Component comp2 participates in the coupling, with fields defined on only one grid, grid2 (decomposed on all five tasks), while comp4 does not participate in the coupling. Component comp3 exchanges coupling fields defined on three different grids, grid3 (tasks 12–21), grid4 (tasks 22–30), and grid5 (tasks 12–26, overlapping with both grid3 and grid4). Finally, comp3 has three tasks (31–33) not involved in the coupling. Different coupling capabilities are indicated by the differently lettered arrows in Fig. 1. Coupling is supported between components in separate executables, within a single executable between different components, and between overlapping, non-overlapping, or partially overlapping grids in a single component. In OASIS3, only coupling between separate executables was supported; in OASIS3-MCT_3.0, a functional and highly flexible coupled system can now be designed and implemented as either a single executable or with multiple executables.

Within OASIS, it has always been mandatory for a user to establish a set of configuration inputs that are consistent with the get and put sequencing in the components such that the coupled system will not deadlock. OASIS3-MCT provides

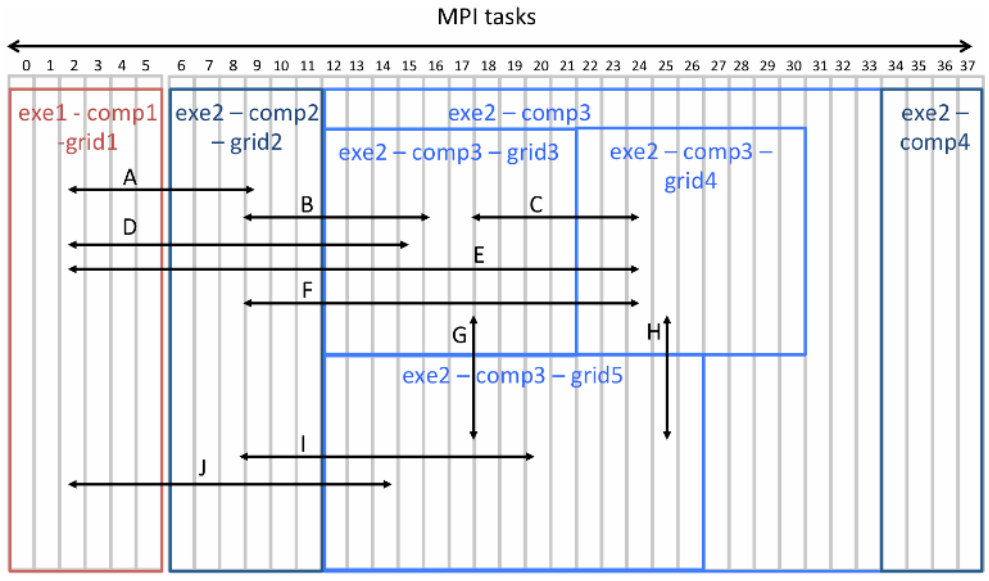

Figure 1. A schematic of the coupling capability in OASIS3-MCT_3.0. In this example, there are two executables, exe1 and exe2. exe2 has three components, comp2, comp3, and comp4, and comp3 has three grids, grid3, grid4, and grid5; comp4 is not involved in any coupling in this case. The executables, components, and grids are laid out across different tasks. Arrows indicate different coupling capabilities: A, D, E, and J between different components in different executables; B, F, and I in a single executable between different components with different grids; C between different grids in a single component on non-overlapping tasks; G between different grids in a single component on partially overlapping tasks; and H between different grids in a single component on partially overlapping and partially non-overlapping tasks.

some new capabilities to detect potential deadlocks before they occur, but it is still largely up to the user to make sure this does not happen. This is even more important for coupling components on overlapping tasks as there is almost no way to detect a deadlock ahead of time. Specifically, a field put routine must be called before the matching get (taking into account any lags specified in the configuration file) when coupling on overlapping tasks. In OASIS3-MCT, puts are generally non-blocking, while gets are blocking. More specifically, a put waits for the completion of the put of the same coupling field at the previous coupling time step before proceeding in order to prevent puts from queuing up in MPI and using excess memory. In other words, for a specific put–get pair, the last put can never be more than one coupling period ahead of the equivalent get in OASIS3-MCT. This means that the puts and gets have to be interleaved when coupling on overlapping tasks. It is not possible to queue up a series of puts over multiple coupling periods before executing the equivalent gets.

2.6 Other features

There are several additional features in OASIS3-MCT relative to OASIS3. The grid writing routines have been extended to support parallel calls from all component processes. However, even when the parallel interface is used, the grid information is still aggregated onto the root processor within the OASIS3-MCT layer and then written serially to disk.

OASIS3-MCT now also includes a GUI, which is an application of OPENTEA (Dauptain, 2014), the graphical interface developed at CERFACS. The OASIS3-MCT GUI helps users produce the *namcouple* configuration file for a specific run, without worrying about the format syntax of the file.

3 Performance

This section summarizes the performance of various aspects of OASIS3-MCT_3.0 at low and high process counts and at moderate to high resolutions. The performance and scaling of initialization, coupling, mapping, conservation, and other features will be presented. Memory usage will also be shown.

3.1 Initialization

Figure 2 shows the initialization cost for a T799-ORCA025 test case on up to 16 000 MPI tasks per component, with the two components running concurrently (32 000 tasks in total) on Curie at CEA TGCC. Curie consists of 5040 nodes with two eight-core Intel Sandy Bridge EP (E5-2680) 2.7 GHz processors per node connected with an InfiniBand QDR Full Fat Tree network. These tests were run with simple toy models that define grids and couple test data but that have practically no model initialization or run-time overhead. This configuration was chosen because it demonstrates OASIS3-MCT's ability to support high-resolution climate configurations. The T799 is a global atmospheric Gaussian reduced

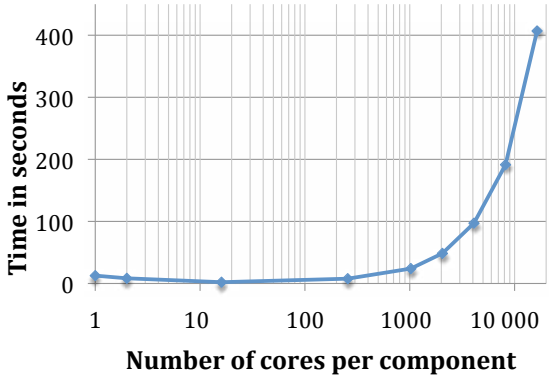

Figure 2. Initialization cost for the T799-ORCA025 toy model using OASIS3-MCT_3.0 on Curie Bullx.

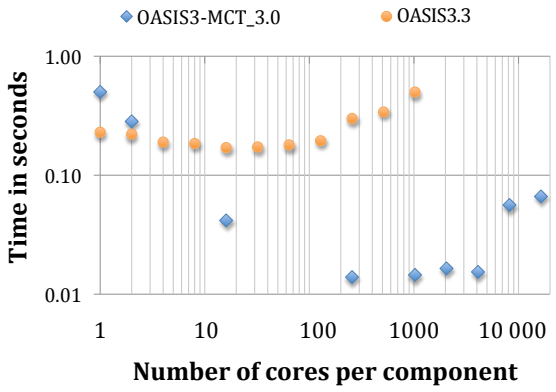

Figure 3. Comparison of the ping-pong (pipo) time for the T799-ORCA025 toy model for OASIS3.3 and OASIS3-MCT_3.0 on Curie Bullx. The time is averaged for a run where 1000 ping-pongs were carried out.

grid with a $\sim$25 km resolution and 843 490 grid points. The ORCA025 grid is a tripolar grid with 1442×1021 ($\sim$1.47 million) grid points and is one of the grid configurations used by the NEMO ocean model (http://www.nemo-ocean.eu/). The OASIS3-MCT initialization consists of several steps, including setting up the partitions, reading in and distributing the mapping weights, computing the mapping rearrangement communication patterns, and computing the coupling communication patterns. Most of these operations rely heavily on MPI to define the interactions, reconcile the coupling fields and decompositions, and set up the mapping and coupling interactions. Multiple runs were performed for each number of cores, with little variability in the timing measured. Based on the results in Fig. 2, the total initialization time for Oasis3-MCT is likely to be reasonable for most applications, even at high numbers of cores. Below 2000 MPI tasks per component, the OASIS3-MCT initialization time is less than 1 min. At 16 000 tasks per component, for this relatively high-resolution configuration, the initialization time is below 7 min. The initialization uses MPI heavily to initialize the coupling interactions, read in the mapping files, and set up the communication for the mapping rearrangement and coupling communication. In general, the initialization is not expected to scale well, but the initialization overhead is what allows the model to run efficiently during the actual run phase. There is clearly some concern that as task counts continue to increase, the initialization time will continue to grow. OASIS developers continue to monitor and analyze both the run-time and initialization costs.

3.2 Coupling

Figure 3 shows the cost of a ping-pong coupling for the same configuration as Fig. 2. The times are per single ping-pong coupling, but the test was done by running and averaging 1000 ping-pongs. In a ping-pong test, data are passed back and forth between the two components sequentially. In other words, data are sent from model 1 and received by model 2, followed by different data being sent from model 2 to model 1. Each coupling of data between a pair of components consists of a mapping operation that interpolates the non-masked data via a five-nearest-neighbor algorithm that includes both floating point operations and rearrangement, and a communication operation that transfers the data between the concurrent sets of MPI tasks of the two components. So there are four distinct MPI operations in a single ping-pong. There are 4.5 million different links (weights) between the T799 grid points and the ORCA025 grid points and 3 million weights for the mapping in the other direction. In this case, scaling is good to about 400 cores per component as the MPI cost is relatively small and the floating point operations associated with the mapping dominate the cost. Between 400 and 4000 cores per component, the ping-pong cost is relatively constant and, above 8000 cores per component, the timing is degraded relative to lower core counts. At higher core counts, the timing depends heavily on the MPI performance. At 8000 cores per component, decompositions get relatively sparse, with just 100 to 200 grid points per core. In addition, timing variability between runs (not shown) above 1000 cores and the jump in cost at 8000 cores suggest that interconnect contention is likely a problem at these core counts. Equivalent timings from OASIS3.3 are also shown in Fig. 3 (Valcke, 2013), and the ping-pong time is about an order of magnitude better in OASIS3-MCT for a large range of core counts.

3.3 Interpolation

One of the features of OASIS3-MCT is the ability to map data on either the source or destination side as described in Sect. 2.3. Figure 4 shows the timing of the mapping portion of coupling which includes both the floating point application of weights and the necessary rearrangement of the data on either the source processes (*src*) or the destination processes (*dst*) but not the communication between the source and destination processes. Two trials were carried out, and

Table 1. Comparison of the ping-pong (pipo) time for the T799-ORCA025 toy model on Lenovo on 360 cores with both the relative core count/component and the mapping location varied. The time is in seconds for 1000 ping-pongs. Columns **(a)** and **(b)** define the core count used for each component of the toy model. Columns **(c–f)** are the pipo times for four different mapping approaches: **(c)** mapping always on the source cores, **(d)** mapping always on the destination cores, **(e)** mapping on the ORCA025 cores, and **(f)** mapping on the T799 cores.

(a) ORCA 025 cores	**(b)** T799 cores	**(c)** pipo time for mapping on *src* cores (s)	**(d)** pipo time for mapping on *dst* cores (s)	**(e)** pipo time for mapping on ORCA025 cores (s)	**(f)** pipo time for mapping on T799 cores (s)
24	336	5.10	5.48	7.29	3.79
180	180	1.29	1.54	1.36	1.36
336	24	4.70	4.93	1.91	6.69

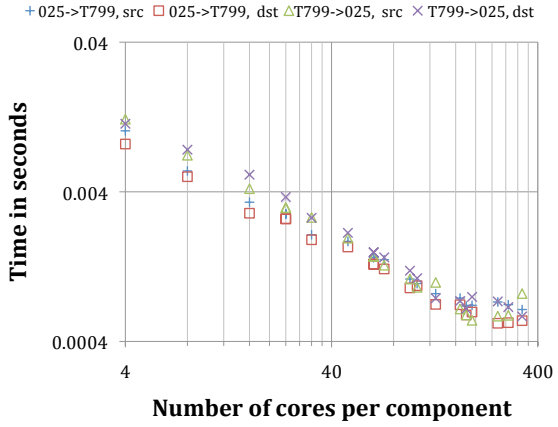

Figure 4. OASIS3-MCT_3.0 T799-ORCA025 mapping time versus core count per component on Lenovo. *src* and *dst* mapping are shown for both mapping directions using the *bfb* algorithm based on tests where 1000 ping-pongs were run.

Fig. 4 shows the best times, with variability generally much less than 5 % between runs. This test was run using the T799-ORCA025 toy model on a Lenovo Xeon based cluster at CERFACS consisting of over 6000 2.5 GHz cores connected by an Infiniband FDR. Mapping is about half the total cost of the ping-pong (not shown) in these cases. Figure 4 shows timing data for both mapping directions and for mapping done on the source (*src*) or destination (*dst*) sides. In all cases, the *bfb* algorithm is used. The mapping in this case scales well to several hundred cores. In general, the cost of the T799 to ORCA025 mapping is more expensive than the reverse, largely due to the fact that there are more mapping weights (4.5 vs. 3.0 million) to apply.

Table 1 documents the ping-pong time for 1000 trials for the same T799-ORCA025 toy model test on Lenovo. In this case, the total number of cores is held at 360, but the relative distribution of cores to each model is varied in three test configurations. The ping-pong tests were carried out with the mapping done on the source, the destination, the ORCA025, or the T799 sets of cores. In these trials, the *bfb* map algorithm was used. In this case, the best performance is when the mapping is done on the model with the highest core count because in this range of core counts, the mapping and communication are still scaling. At higher core counts or with different grids, the optimum performance may be different. For the current cases, the best time is a factor of up to 2.5 times better (1.91 s vs. 4.70 s) compared to the default setting of *src* and by an even greater factor compared to the slowest setting. Another point is that if there is a large disparity in the number of grid cells in the two grids, it should be better to exchange the coupling fields expressed on the grid with the fewest grid cells and perform the remapping on the other component tasks. In general, the number of processes per component is going to be determined by the relative cost of the scientific models, but the above analysis shows that for a given task layout, there may be ways to reduce the coupling cost by mapping on the tasks that provide the greatest performance.

3.4 Field aggregation

OASIS3-MCT provides a new feature, as described in Sect. 2.2, that allows users to aggregate coupling of multiple fields into a single coupling operation by specifying coupled fields via colon-delimited field names in the *namcouple* file. Table 2 shows unbarriered and barriered ping-pong and barriered mapping timing for the T799-ORCA025 configuration on Lenovo using single and multiple fields. For the barriered case, MPI barriers were added before the send and before the mapping in each component in both directions of the coupling to strictly enforce serialization of operations and to be able to time the mapping cost cleanly. Times are in seconds for the slowest task over the entire run. The fastest time from two test runs is shown. Variability between runs is less than 2 %. The columns in Table 2 are for a configuration with 180 cores per component using *src*+*bfb* map settings for a single field, 10 fields coupled via 10 coupling calls, 10 fields coupled via a single coupling communication, and 10 fields bundled into a single variable. The bundled fields option will be available in the OASIS3-MCT_4.0 release. The barriered pipo (ping-pong) time in Table 2 is about 50 % greater than the unbarriered time. The significant performance penalty

Table 2. Comparison of unbarriered and barriered ping-pongs (pipo) and barriered mapping time for the T799-ORCA025 toy model on Lenovo on 180 cores per component for coupling of 1 field, coupling of 10 fields one at a time, coupling of 10 fields using OASIS3-MCT multiple-coupling-field capability, and coupling of 10 fields by a single 3-D bundle. All times are for *src+bfb* mapping for 1000 ping-pongs. For barriered times, MPI barriers were introduced in both components before the send and before the mapping to force serialization of work and to time the mappings separately.

time (seconds) mapping = *src+bfb*	1 field, 1 coupling	10 fields, 10 couplings	10 fields, 1 coupling	10 fields, 1 bundle
pipo time, no barriers	1.29	10.52	11.93	12.29
pipo time, with barriers	1.87	17.63	16.56	17.48
map ORCA025 → T799, with barriers	0.67	5.48	4.61	4.68
map T799 → ORCA025, with barriers	0.56	5.28	4.76	4.81

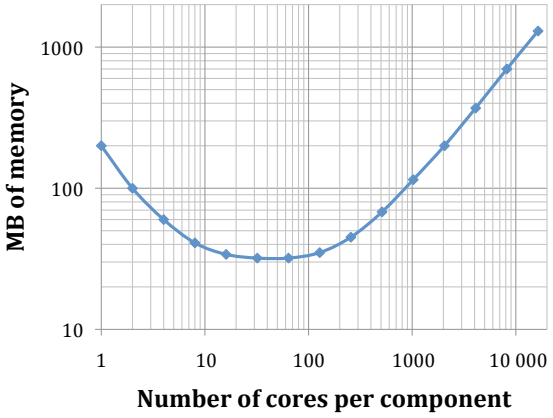

Figure 5. OASIS3-MCT_3.0 memory use on Curie Bullx for the T799-ORCA025 toy model as a function of cores per component.

with barriers suggests that there is normally some overlap of coupling communication and mapping in these timing runs when running without barriers.

The unbarriered pipo time in Table 2 shows that coupling 10 fields performs proportionally better than coupling a single field. More specifically, the case with 10 fields coupled with 10 coupling calls performs best, likely because there is a greater chance of overlapping mapping and coupling communication in this case since each field is mapped and sent independently. The barriered pipo time further suggests that the case with 10 fields coupled with 10 coupling calls has the greatest amount of overlapping work because that case has the largest performance degradation when barriers are turned on.

In contrast, the mapping time for 10 fields coupled via a single operation is faster than mapping 10 fields one at a time. This is expected as the underlying implementation aggregates the mapping rearrangement and coupling communication cost when fields are bundled. But in this case, that mapping advantage is offset by the ability to overlap less work. This simple test case carries out coupling without any real model work between calls. In a real model, the coupling performance will depend on the sequence of the coupling calls within the model, how much work can be overlapped with coupling, and the relative core counts and grid sizes of the different coupling fields.

3.5 Conservation

Table 3 shows the timings of a ping-pong test of the T799-ORCA025 case on the Lenovo cluster for four different configurations (48 and 180 cores with *src* or *dst* mapping) with CONSERV unset and CONSERV set to *lsum8* (equivalent to *opt* in OASIS3-MCT_3.0), *lsum16*, *ddpdd*, *reprosum*, and *gather* (equivalent to *bfb* in OASIS3-MCT_3.0). The CONSERV implementation and a description of the different options for the computation of the global sums are given in Sect. 2.4. Times are accumulated over 1000 ping-pongs for a single coupling field in each direction. Two trials of each case were carried out and the minimum time is shown. Differences between trials were less than 2 % except for the *gather* case where variations in time of up to 10 % were observed. The CONSERV operation increases the pipo time by at least 50 % regardless of the method used compared to CONSERV off (unset), and the *gather* option is at least an order of magnitude more expensive than other CONSERV methods. When OASIS3-MCT_4.0 is available, *lsum8* will still be the fastest CONSERV method, while *reprosum* will be the best bit-for-bit option. The cost of *reprosum* is only slightly higher than *lsum16*, but reproducibility characteristics are significantly better. When using CONSERV, it is important to test the performance of various methods and consider carefully the requirements of the science. Of course, when possible, mapping weights that are inherently conservative such as area overlap conservative (Jones, 1999) should be used to avoid use of the CONSERV operation altogether.

3.6 Memory

Figure 5 shows the memory use per core for the T799-ORCA025 test case on Curie, the same test case as in Figs. 2 and 3. Memory use was determined by calls into the gptl (http://jmrosinski.github.io/GPTL/) interface, included in the OASIS3-MCT release, which queries memory usage through

Table 3. Comparison of ping-pong (pipo) times for the T799-ORCA025 toy model on Lenovo on 48 and 180 cores per model with the CONSERV option off (unset), set to *lsum8* (*opt* in OASIS3-MCT_3.0), *lsum16*, *ddpdd*, *reprosum*, and *gather* (*bfb* in OASIS3-MCT). Times are accumulated over 1000 ping-pongs for a single coupling field in each direction.

cores, mapping	CONSERV unset	CONSERV *lsum8*	CONSERV *lsum16*	CONSERV *ddpdd*	CONSERV *reprosum*	CONSERV *gather*
48, *src+bfb*	4.00	8.27	16.78	10.65	17.34	117.72
48, *dst+bfb*	4.39	8.02	16.59	10.42	16.98	142.12
180, *src+bfb*	1.25	2.21	4.59	2.87	4.85	126.91
180, *dst+bfb*	1.56	2.26	4.62	2.92	4.90	130.01

C intrinsics. At 16 000 cores, the infrastructure uses a bit more than 1 GB per core, which while not tiny, is generally acceptable for many applications and hardware. Memory increases on a per core basis at higher core counts. It is possible that the MPI memory footprint accounts for most of this behavior (Balaji et al., 2008; Gropp, 2009), but further investigation will be carried out in the future to better understand this behavior.

4 Conclusions

OASIS3-MCT was implemented largely to address limitations in parallel performance of OASIS3 and to provide a framework for use at higher resolutions. With OASIS3-MCT, the widely used OASIS3 model interfaces (APIs) and configuration file have largely been preserved, and this explains the wide adoption of OASIS3-MCT within the OASIS user community. Since its release in May 2015, about 250 downloads of OASIS3-MCT_3.0 have been registered from most major climate modeling groups in Europe as well as from groups in North and South America, Asia, Australia, and Africa. In the last 2 years, the OASIS3-MCT coupler has been used in many state-of-the-art coupled systems, including high-resolution climate models and systems that couple 3-D atmospheric fields between global and regional models frequently among others. Other examples of coupled model applications that use OASIS3-MCT can be found on the OASIS3-MCT coupled model page[2].

The underlying software was refactored significantly in OASIS3-MCT to improve parallel performance and coupling capabilities. MCT serves as a key part of the OASIS3-MCT implementation and provides parallel capabilities for coupling operations. OASIS3-MCT_3.0 also provides new capabilities to couple fields within a single component running on concurrent, overlapping, or partially overlapping processes. This increases the flexibility of OASIS3-MCT significantly and provides a mechanism for coupling data between different decompositions or grids within a single model, among many other things. OASIS3-MCT can now be used as a coupling layer for components running sequentially, concurrently, or both; for single or multiple executable execution; to exchange coupling fields defined on a subset of the component tasks; and to support features like a separate I/O component included in the executable but not involved in the coupling. This provides significant flexibility to layout models on parallel tasks in relatively arbitrary ways to optimize overall performance and to build new features into a model beyond model coupling. OASIS3-MCT has been tested successfully at high resolution, at high processor counts, and with a large number of coupling fields.

There are other benefits in the OASIS3-MCT implementation. OASIS3-MCT still supports mapping weight generation on the fly via SCRIP using a single processor just like OASIS3. However, mapping files can also be generated offline, read in directly relatively efficiently, and more easily reused, and the cost associated with generating the mapping files can be moved to a preprocessing step using more sophisticated tools. If online weight generation needs to be upgraded in OASIS in the future to support, for instance, time-evolving grids, OASIS will consider incorporating more sophisticated external tools into the infrastructure. There are new features that support the creation of grid data using a parallel interface, that couple multiple fields in a single operation, and that generate the *namcouple* file offline via a GUI. The requirement for an OASIS3 hub coupler has been removed, all communication and mapping are done in parallel, and performance is significantly improved.

The scaling and performance results in Sect. 3 demonstrate the ability of OASIS3-MCT to support high-resolution model coupling on large core counts. However, as core counts get well into the tens of thousands and beyond, there are questions and concerns about the cost of both the initialization and coupling exchanges in OASIS3-MCT. The operations in OASIS3-MCT are ultimately constrained by MPI performance at those core counts, and developers will continue to pursue performance improvements in the underlying implementation. However, for the near-term future, say the next 5 years, OASIS3-MCT is likely to adequately meet the needs of the climate modeling community.

The flexibility and relative cost of OASIS3-MCT to map fields by various approaches was shown. A general recom-

[2] https://portal.enes.org/oasis/oasis-dedicated-user-support-1/survey-on-coupled-models-using-oasis-march-2016/coupled-models-using-oasis

mendation is to test different approaches and to choose the approach that yields the best performance. While it is always first recommended to use conservative mapping weights to avoid the use of the global CONSERV transformation, the performances of the different options of this transformation were shown for a high-resolution case. If the CONSERV transformation is needed, the more efficient *lsum8* (*opt* in OASIS3-MCT_3.0) option, implemented using partial sums, is recommended unless bit-for-bit reproducible results on different core counts are absolutely required. The partial sum option will produce bit-for-bit reproducible results for a configuration with fixed process counts and decomposition and will introduce no more than roundoff level differences when changing process counts or decomposition. CONSERV options planned for the OASIS3-MCT_4.0 release were also included in the results in Sect. 3. In OASIS3-MCT_4.0, the new *reprosum* option will significantly improve the performance of the bit-for-bit CONSERV option compared to the currently available *gather* (*bfb* in OASIS3-MCT_3.0) option.

The ability to couple multiple fields via a single coupling operation was demonstrated. While not shown in this study, OASIS3-MCT has been used to successfully couple over 10 000 fields in some coupled systems within the community. Those tests were carried out with both single field coupling and multiple field coupling with success. In that case, multiple field coupling significantly reduces the size of the *namcouple* file. Multiple field coupling was shown to reduce the mapping time compared to coupling the same number of fields individually. The performance benefit of using the multiple field feature in the overall coupling time is less clear and will depend on the sequencing and design of each coupled system.

A number of future extensions are being considered for OASIS3-MCT. In theory, it should be possible to combine the mapping and coupling steps to eliminate a field rearrangement and further reduce communication cost. As a first step, decomposition strategies that could reduce the rearrangement cost in the mapping operation are being developed for release in OASIS3-MCT_4.0. There are also many opportunities in OASIS3-MCT to improve the I/O performance. In the current version, I/O is done via a gather and/or scatter to/from a root task and data are written in serial from the root task. This is likely to eventually lead to memory and performance issues. Finally, better support within OASIS3-MCT for shared memory threading (i.e., OpenMP) and on various multi-core architectures is likely to become more important in the future.

In summary, OASIS3-MCT_3.0 is the latest released version of the OASIS coupler. OASIS3-MCT extends the well-used OASIS software with backwards compatibility with regard to usage, but has an entirely new implementation internally. It provides the functional capability to couple high-resolution structured or unstructured grids at high core counts successfully and should serve the community well for the next several years. The underlying implementation continues to be improved, and OASIS3-MCT_4.0 is expected to be ready for release in 2018.

Code availability. The OASIS3-MCT source code is available for use and testing after registration at https://portal.enes.org/oasis/download. The SVN command line to download OASIS3-MCT_3.0 is "svn checkout https://oasis3mct.cerfacs.fr/svn/branches/OASIS3-MCT_3.0_branch/oasis3-mct" (last access: September 2017). The OASIS3-MCT_3.0 source code is also available as a tar file at ftp://ftp.cerfacs.fr/pub/globc/exchanges/distrib-oasis/oasis3-mct.tar.gz (last access: September 2017).

Appendix A

The following list provides a history of changes to OASIS3-MCT since OASIS3 up to OASIS3-MCT_3.0. It also includes an initial list of some features expected in the next release, OASIS3-MCT_4.0.

OASIS3-MCT_1.0 (2012)

- requirement for separate coupler processes and hub removed
- use of MCT in the underlying coupling layer for regridding and communication
- parallel remapping
- fully parallel communication
- ability to couple a single field to multiple destinations
- extended ability to read mapping file
- improved deadlock trapping
- only MPI1 job launching supported
- ability to couple on a subset of processes
- support for 1-D coupling field arrays
- support for prism_and oasis_interface names
- restart files for LOCTRANS operations
- coupling multiple fields through a single *namcouple* entry

OASIS3-MCT_2.0 (2013)

- support for bicubic interpolation given the field gradient is specified in the interface arguments
- coupling support on a subdomain of the full grid
- update to timing and debugging capabilities
- parallel interface to grid writing

OASIS3-MCT_3.0 (2015)

- improved memory use, initialization cost, and scaling
- updated mapping file reading algorithm
- ability to implement a coupled system within a single executable
- ability to couple sequentially and on partially or completely overlapping processes

OASIS3-MCT_4.0 (2018?)

- support for bundled coupling fields
- additional CONSERV global sum methods and improved CONSERV bit-for-bit performance
- a new option for decomposing the mapped field to reduce communication cost
- an update to a newer version of MCT that may improve initialization performance

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. This research was supported by the ESiWACE H2020 European project, grant agreement no. 675191 (www.esiwace.eu), the IS-ENES2 FP7 European project, contract number 312979 (https://verc.enes.org/ISENES2), and the CONVERGENCE project funded by the French National Research Agency: ANR-13-MONU-0008.

Edited by: Claire Levy
Reviewed by: Moritz Hanke and two anonymous referees

References

Balaji, P., Buntinas, D., Goodell, D., Gropp, W. D., Kumar, S., Lusk, E. L., Thakur, R., and Traff, J. L.: MPI on a Million Processors. Recent Advances in Parallel Virtual Machine and Message Passing Interface, Lecture Notes in Computer Science, Volume 5759, 20–30, https://doi.org/10.1007/978-3-642-03770-2_9, 2009.

Craig, A. P., Vertenstein, M., and Jacob, R.: A New Flexible Coupler for Earth System Modeling developed for CCSM4 and CESM1, Int. J. High Perf. Comp. App., 26, 31–42, https://doi.org/10.1177/1094342011428141, 2012.

Dauptain, A.: OpenTEA Super-User Guide, available at: https://oasis3mct.cerfacs.fr/svn/trunk/oasis3-mct/util/oasisgui/OpenteaSUG.pdf (last access: September 2017), 2014.

Gropp, W.: MPI at Exascale: Challenges for Data Structures and Algorithms, Proceedings of the 16th European PVM/MPI Users' Group Meeting on Recent Advances in: Parallel Virtual Machine and Message Passing Interface, edited by: Ropo, M., Westerholm, J., and Dongarra, J., published by Springer-Verlag, Berlin, https://doi.org/10.1007/978-3-642-03770-2_3, 2009.

He, Y. and Ding, C. H. Q.: Using Accurate Arithmetics to Improve Numerical Reproducibility and Stability in Numerical Applications, The Journal of Supercomputing, 18, 259, https://doi.org/10.1023/A:1008153532043, 2001.

Hollingsworth, A., Engelen, R. J., Textor, C., Benedetti, A., Boucher, O., Chevallier, F., Dethof, A., Elbern, H., Eskes, H., Flemming, J., Granier, C., Kaiser, J. W., Morcrette, J.-J., Rayner, P., Peuch, V. H., Rouil, L., Schultz, M. G., Simmons, A. J., and The GEMS Consortium: Toward a Monitoring and Forecasting System For Atmospheric Composition: The GEMS Project, B. Am. Meteorol. Soc., 89, 1147–1164, 2008.

Jacob, R., Larson, J., and Ong, E.: MxN Communication and Parallel Interpolation in CCSM3 Using the Model Coupling Toolkit, Int. J. High Perf. Comp. App., 19, 293–307, https://doi.org/10.1177/1094342005056116, 2005.

Jones, P.: Conservative remapping: First-and second-order conservative remapping, Mon. Weather Rev., 127, 2204–2210, 1999.

Larson, J., Jacob, R., and Ong, E.: The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models, Int. J. High Perf. Comp. App., 19, 277–292, https://doi.org/10.1177/1094342005056116, 2005.

Mirin, A. A. and Worley, P. H.: Improving the Performance Scalability of the Community Atmosphere Model, Int. J. High Perf. Comp. App., 26, 17–30, https://doi.org/10.1177/1094342011412630, 2012.

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

Terray, L., Thual, O., Belamari, S., Déqué, M., Dandin, P., Lévy, C., and Delecluse, P.: Climatology and interannual variability simulated by the arpege-opa model, Clim. Dynam., 11, 487–505, 1995.

Theurich, G., Deluca, C., Campbell, T., Liu, F., Saint, K., Vertenstein, M., Chen, J., Oehmke, R., Doyle, J., Whitcomb, T., Wallcraft, A., Iredell, M., Black, T., Da Silva, A. M., Clune, T., Ferraro, R., Li, P., Kelley, M., Aleinov, I., Balaji, V., Zadeh, N., Jacob, R., Kirtman, B., Giraldo, F., McCarren, D., Sandgathe, S., Peckham, S., and Dunlap IV, R.: The Earth System Prediction Suite: Toward a Coordinated U.S. Modeling Capability, B. Am. Meteor. Soc., 97, 1229–1247, https://doi.org/10.1175/BAMS-D-14-00164.1, 2016.

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

Valcke, S., Craig, T., and Coquart, L.: OASIS3-MCT User Guide, OASIS3-MCT 1.0, Technical Report, TR/CMGC/12/49, CERFACS, Toulouse, France, available at: http://www.cerfacs.fr/oa4web/papers_oasis/oasis3mct_UserGuide_1.0.pdf (last access: September 2017), 2012.

Valcke, S., Craig, T., and Coquart, L.: OASIS3-MCT User Guide, OASIS3-MCT_3.0, Technical Report, TR/CMGC/15/38, CERFACS/CNRS/SUC URA No1875, Toulouse, France, available at: http://www.cerfacs.fr/oa4web/oasis3-mct_3.0/oasis3mct_UserGuide.pdf (last access: September 2017), 2015.

Valcke, S., Craig, A., Dunlap, R., and Riley, G.: Sharing Experiences and Outlook on Coupling Technologies for Earth System Models, Bu. Am. Meteor. Soc., 97, ES53–ES56, https://doi.org/10.1175/BAMS-D-15-00239.1, 2016.