Reply to Referee #1.

We would like to thank Moritz Hanke for his careful review and thoughtful comments. We will reply to the comments below (in green text)

General Comments

This paper introduces the new version of the coupling software OASIS and its latest revision OASIS3-MCT_3.0. It describes in detail the most important improvements and new features of this version. In addition, it provides performance data relevant for users of the software.

It has a clear structure and is written well. It gives users of older versions of the software a good understanding of the changes and helps to decide whether to switch to the latest version or not. For developers of other coupling solutions this paper gives an interesting insight on how the current version of OASIS works.

After some modifications and clarifications regarding the presented performance results, I would recommend this paper for publication.

Specific Comments

If you are not familiar with coupling software in general or with OASIS is particular, some parts of the paper may be difficult to understand, due to usage of domain-specific terms and concepts without further explanation for example "hub coupler" in abstract, “top-level driver” in introduction, the terms “source” and “destination”, “MCT router” in 2.1 General Architecture, or “CONSERV transform” in 2.4 Conservation. Depending on the target audience this might be an issue.

We have added the following clarifications:

- " A separate top-level driver to control system sequencing is not required "
- " all coupling fields passed through a separate central hub coupler component"
- source and destination are implicitly defined in the introduction
- "Each parallel field in the source model was gathered 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",
- MCT router is an MCT datatype. We have updated the text and the only place where "router" appears in text is in the following sentence where it is defined, " Data communication and mapping rearrangement is handled internally in OASIS3-MCT via MCT routers.".
- "CONSERV" is clearly defined in section 2.4, "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."

In the paper you talk about OASIS3-MCT and its improvements compared to older OASIS versions and about its latest revision OASIS3-MCT_3.0 in particular. However this is not reflected in the title of the paper. It implies that the paper is mainly about OASIS3-MCT_3.0.

To be honest, the original title of the paper was " Development and performance of a new version of the OASIS coupler, OASIS3-MCT ", but the editor encouraged us to be more specific with regard to the version in the title prior to formal submission. The paper is written at a time when OASIS3-MCT_3.0 is the current release, and so we feel it is reasonable to include that information in the title. It is true that this paper takes a slightly broader approach by summarizing changes since OASIS3 including features added before OASIS3-MCT_3.0 (see details in Appendix A). It even includes some information about what is coming in the version 4.0 release of OASIS3-MCT. We made a few changes in the text to further clarify the scope of the paper but feel the current title is reasonable. In particular, we have added " This paper describes the development of OASIS3-MCT from OASIS3 to the current version 3.0 release and will also introduce some new features expected in the version 4.0 release." to the introduction.

You use lower and upper case when referencing figures or tables. This should be consistent.

We have updated the text so all references to figures and tables in the text are lower case unless they occur at the start of the sentence.

"2.5 Concurrency, Process Layout, and Sequencing"
I do not see why there is a need to differentiate between different executables. Since each MPI process only has a single component, shouldn't it be enough to start the differentiation at the component level? This might reduce the complexity of this paragraph. Or would there be any difference if comp2, comp3, and comp4 were run on three individual executables?

The reviewer's comments are correct. It doesn't fundamentally matter whether multiple components are run as a single executable or as multiple executables in Oasis3-MCT. I think the main point of including that statement is to make it clear that both modes are supported. We have added a sentence at the end of the second paragraph in section 2.5 to emphasize that point and address the reviewer's concerns.

The main conclusion of section "3.3 Interpolation" is that the default option of performing the mapping on the processes of the source component might not always be the best choice and that explicitly setting OASIS to do it on the processes of the component with the most resources can deliver better results. However, to draw this conclusion the presented test cases and diagrams seem to be overly complicated. Since the mapping is done based on a "simple one-dimensional" decomposition, the performance should be independent of the grid types being used. Therefore you could draw the same conclusion from a table similar to the following one (only showing the results for a one directions data exchange), which I think is much easier to understand:

# src cores	# dst cores	Mapping on src		Mapping on dst	
		transfer	mapping	transfer	Mapping
24	336	*s	*s	*s	*s
180	180	*s	*s	*s	*s
336	24	*s	*s	*s	*s

In the discussion of section "3.3 Interpolation", I would add that depending on where the mapping is executed, the amount of data that is exchanged between both components varies. This might be important in case both grids have a significantly different number of cells.

This is a reasonable point. However, Figure 5 is useful in that it shows the scaling of mapping across a broader range of pe counts which some readers might find useful. The other problem is that while it's relatively easy to time the mapping separately with appropriate barriers, it's much harder to time the transfer in these cases as there is significant load imbalance, puts are non-blocking, and some of the performance is associated with overlapping transfer and mapping work. We believe the information in table 1 is consistent with the reviewers request, and figure 5 provides additional insight into the mapping performance that goes beyond what could be done with a table.   A final point is that it's not correct to suggest the map timing is independent of grid type. In fact, the grid decomposition, number of weights, whether mapping is done on the src or dst side, number of pes in play, and distribution of the weights have a large impact on the map timing. We have updated figure 5, so the symbols and symbol key are clearer. We do agree about the comment that amount of data exchanged is important and we have added the statement, " Another point is that if there is a large disparity in the number of grid cells in the two mapped 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 the text it is nowhere mention what the abbreviation OASIS stands for.

We have added a sentence in the first paragraph of the introduction to define the OASIS project.

P1L13-15 "It includes [...] full parallelisation of the [...] grid interpolation"
This may be interpolated as OASIS being able to generate interpolation weights on-the-fly in parallel.

We have changed the sentence to read, "parallelization of the coupling communication and run time grid interpolation " to emphasize parallelization of the interpolation at run time, which is unrelated to the process of weights .

P2L21 “source neighbour weights” I do not know this term.

We have rewritten this sentence as "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."

P3L7-8 “the hub coupler [...] is no longer required”
This could be interpreted as: not required but still usable. Is that intended?

This is a good point. We have changed this sentence to "Third, the OASIS hub coupler was deprecated and is no longer needed or implemented."

P4L15-18 “Compared to OASIS3 which required two data rearranges to couple fields in order to pass through the hub, OASIS3-MCT requires just one parallel rearrange to move data between two components.”
You are comparing the coupling of fields in OASIS3 with the moving of data between components in OASIS3-MCT, which seems unfair, because in the paragraph above it is said that OASIS3-MCT also requires two data rearranges for the full coupling. Or is there a misunderstanding?

This is a very good point. We have clarified this sentence as follows, " 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. " We have also changed some of the wording in the document to provide more consistency, clarifying the terms redistribution, communication, coupling, and mapping.

P5L9 “Mapping weight files can either be read directly”
For big weight files it may be important to know whether this is done in serial or in parallel. Only in section “4. Conclusions” it is mentioned that I/O in general is done in serial.

We have added a new sentence to further define the implementation, "In OASIS3-MCT, the weight files are read serially on the root process and distributed to other processes in reasonable chunks. That chunk size is currently set to 100,000 weights at a time to limit memory use on the root process."

P5L18 “Users also have an additional option to specify the type of mapping to be carried out.” The term “type of mapping” is a little bit ambiguous. It could also refer to interpolation types (e.g. linear, nearest neighbour, or conservative interpolation).

This is a good comment. We have changed this sentence to "Users also have an additional option to set the implementation of the underlying mapping algorithm."

P6L1-11 Maybe you should mention that there is the possibility to turn off the CONSERV transform. Which is important since this operation does not make sense for all field types.

We have added the word "optional" in the first sentence of section 2.4 to reiterate the fact that CONSERV is an optional transform. We have also updated this section to reflect some new features.

P6L1-11 There is a bfb option for CONSERV transform and for mapping type. This can be confusing. Maybe clarify this

We recognize that the common keywords are not ideal and are working to differentiate them in future releases. We have added a sentence in section 2.4 to clarify, "Note that both the CONSERV operation and the underlying mapping algorithm setting share a common flag, *bfb*, but that these two settings are completely independent."

P7L8-20 whole paragraph + Figure 2
This paragraph and the associated figure seem to be out of place. I would expect them to be part of a user manual.

We have removed this section and Figure 2 from the paper. This information is in the user guide and we agree that this does not need to be duplicated in the paper.

P7L27 "a field put routine must be called before the matching get"
In case there are two components comp1 and comp2, if there is only a one directional data flow from comp1 to comp2, do all puts in comp1 actually have to be called before (in time) the respective gets in order to avoid a deadlock? Or do the gets wait until the respective put is called?

This is a good question and something we've been trying to clarify in the implementation and user guide. To answer the question, each put is non-blocking but waits for the completion of the put of the same coupling field at the previous coupling timestep before it executes. Therefore, you cannot queue up a bunch of puts before executing a get on overlapping or non-overlapping pes. We have tried to clarify this paragraph in section 2.5 by adding, "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 timestep 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."

P8L11 “16,000 cores”
Maybe you should talk about MPI processes or specify that you are using one MPI process per core.

We are constantly struggling whether to use MPI tasks, processes, cores, or pes as a way to describe parallelism. We have tried to be consistent in the paper. We have changed the text from "16,000 cores" to "16,000 MPI tasks".

P8L28-29 “There is however clearly some concern that as core counts continue to increase, the initialization time will continue to grow.”
Did you analyse the cause for the increase? Can you add some discussion on this?

To address this comment, the end of the last paragraph in section 3.1 has been updated as follows, "The initialization uses MPI heavily to initialize the coupling interactions, read in the mapping files, and setup 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 core counts continue to increase, the initialization time will continue to grow. OASIS developers continue to monitor and analyze both the runtime and initialization costs and make improvements. "

P9L8-10 With two-nearest-neighbour interpolation you should have two weights per point on the destination grid.
T799->ORCA025: 2 * 1442 * 1021 = 2,944,564 weights << 4.5 mio weights
ORCA025->T799: 2 * 843,490 = 1,686,980 weights << 3 mio weights
Did I misunderstand something or how do you explain the difference in the number of weights?

We had an error in the description, the weights are based on five-nearest-neighbor interpolation and the ORCA025 grid has masked points. 4.5 million weights for T799->ORCA025 is the equivalent of 61% unmasked points on the ocean grid, 3.0 million weights for ORCA025->T799 is 71% of the maximum number of weights if the grids were unmasked. We have corrected that section and it now reads, "Each coupling of data between a pair of components consists of a mapping operation that interpolates the masked data via a five-nearest-neighbor algorithm that includes both floating point operations and rearrangement, and then a communication operation that transfers the data between concurrent sets of MPI tasks in the different 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."

P9L13-14 “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.”
Why do you not see this behaviour in the IS-ENES2 coupling technology benchmark? Is this due to the different grids used in both test cases?

This paper does not mention nor include an analysis or comparison to the IS-ENES2 benchmark. Having said that, the comment is interesting, and we are currently looking at the benchmark results in the context of these timing tests to better understand the timing differences. The curves in the IS-ENES benchmark show roughly the same behavior although the absolute timing is quite different. These differences are likely related to the different resolutions, different mapping files, and different machines used in the two cases.

P10L25 “while for the src+bfb case, the single operation performs slightly worse”
Are you sure that the measurements (10.56 vs 11.89), this statement is referring to, are correct? (5.95 + 6.02) > 10.56 (taken from Table 2 and Table 3)

This is a good point that we should clarify. The pipo time is done without any barriers while the mapping timing is done as a separate test run with barriers around the mapping. In general, those barriers will slow the model down because any overlap in mapping and data transfer due to load imbalance will be lost with the barriers. Timing parallel kernels in a consistent way is always tricky. We have updated section 3.4, combined tables 2 and 3, and added some new timing information for the pipo time when barriered. We hope this significantly clarifies the timing information.

P11L13-14 “It's likely that the MPI memory footprint is accounting for most of this behavior (Balaji et al, 2008, Gropp, 2009).”
With a modern MPI implementation this should not happen. I have not seen this behaviour in similar measurements for the ICON model. You could verify this using for example the valgrind tool Massif.

We have updated this sentence to "It is possible that the MPI memory footprint is accounting for most of this behavior (Balaji et al, 2008, Gropp, 2009), but further investigation will need to be carried out in the future to confirm." We hope that is a reasonable response.

P20 Figure 4
In this case, I would not use a trendline or and any line between the measurement points. The number of cores has a significant impact on the decomposition, which might lead to interesting result between the provided measurements. Therefore, a line between the points implies a continuity that might not reflect reality for this test case.

We have removed the line between the measurement points in Figure 4.

P21 Figure 5
Are these single measurements or averages?

We have added 1 sentence in section 3.3 to answer this question, "Two trials were carried out and the results shown are for the best times with variability generally much less than 5% between runs."

P22 Table 2 and 3
(0.69 + 0.60) == 1.29 => Did the data exchange between the components only take a negligible amount of time?

We have rerun the tests with additional timing information, combined tables 2 and 3, updated the table with some additional results, and updated section 3.4 to clarify these results. The barriered pipo time is now shown to compare with the sum of the map time for an apples and apples comparison. Compared to the unbarriered pipo time, this also better demonstrates the amount of load imbalance and overlapping work between the mapping and communication in the unbarriered case and the text has been revised to reflect that.

P22 Table 2 and 3
(5.95 + 6.02) > 10.56 => Are the measurements correct?

See comment above.

P22 Table 2 and 3
(11.89 – (4.70 + 4.60)) > (12.15 – (4.86 + 4.97)) => time for mapping ↑ time for transfer ↓ => How do you explain this?

Again, this comes down to the barrier around map timing which we now describe in the text. See the comment above with regard to P22, Table 2 and 3. We have added some text in section 3.4 to explain the timing numbers better. The old timing information did not provide insight into the load imbalance. In fact, the mapping time does go up but you cannot immediately assume the communication time is decreased. This is hopefully clarified in the text.

P22 Table 4
(2.11 – 1.29) >> (2.17 – 1.61)
I would assume that the cost for CONSERV is independent of the src/dst option. How do you explain the difference?

It's not clear that you can make simple conclusions like this from the timing information. The timing of the pipo is complicated by load imbalance, dependencies in the communication between tasks, and other issues. In addition, the order of operations for src+bfb and dst+bfb are quite different and depending where in the sequencing the global sums are carried out, this can have an impact on the load imbalance and overall pipo time. We have updated Table 4 to reflect some new results and we have added some additional information in the discussion in Section 3.5.

Technical Comments

P1L14 "separate hub coupler **process**"

We have implemented this change to the text.

P1L23 "OASIS is **a** coupling software"

We have not made this change, we feel the current wording is ok.

P1L32-33 "OASIS-MCT supports coupling of fields on relatively arbitrary grids [...]." Is "OASIS-MCT supports coupling of fields defined on most grid types, commonly used in climate science, [...]." better?

We have updated this sentence consistent with the review.

P1L33 "via a put/get approach. This approach means components make subroutine calls [...]" "via a put/get approach, which is based on components making subroutine calls [...]"?

We have updated this sentence as suggested by the reviewer.

P2L20-21 "calculation of the **source neighbour** weights and addresses needed for the mapping"

We have updated the spelling of neighbor

P2L23-26 check use of Oxford comma

We have added a comma as suggested

P2L25 Why did you use the long form for AWI while using the abbreviation for ECMWF, KNMI, and MPI-M?

We put abbreviations everywhere.

P2L26-28 Maybe add a reference?

We added Hollingsworth et al., 2008

P2L29-30 “OASIS3-MCT extended the widely used and distributed OASIS3 version of the model.” “It extends the widely used and distributed OASIS3.”?

We have updated the sentence as suggested by the reviewer

P3L8 “Transform**ation**s are carried out” P3L21 “section 4 provides a summary” Section 4 is called “Conclusion”

We have updated this sentence as follows, "... and section 4 provides conclusions and a summary."

P6L2-3 “In OASIS3-MCT, this operation **can is** now **be** performed in parallel on the source or destination processes”
If the bfb option is used, it will still be done in serial, or not?

It will always be done in parallel. Even if the bfb option is used to compute the global sums, the corrections are applied in parallel on the decomposed fields after broadcasting those global sums to all tasks. We have added a word, "decomposed" to "...applies corrections to the decomposed mapped fields..." to make it clearer the correction is happening in parallel.

P7L1-2 “are indicated by the different lettered arrows **in Figure 1**.”

I assume the reviewer was asking us to check the capitalization of "F"? We have corrected this throughout the paper and changed all references to figures and tables to small letters unless figure or table are the first word of a sentence.

P9L10 “**there are**”

We have removed "there are" in that compound sentence.

P10L5 “(1.91**s** vs 4.70**s**)”

Units have been added

P10L32 “CONSERV unset”
In Table 4 this is called off.

We have modified table 4 and used the word unset consistently.

P10L31-33 “Table 4 shows [...]. Table 4 shows [...]” Identical start of two consecutive sentences.

We have updated the second sentence starting with "Table 4 shows" to improve readiblity.

P11L4-5 “such as area overlap conservative” Maybe place a reference to: http://dx.doi.org/10.1175/1520-0493(1999)127%3C2204:FASOCR%3E2.0.CO;2

We have added an equivalent reference as requested by the reviewer

P12L19 “**10stens** of thousands”

We have changed the wording from 10s to tens as suggested

P12L27-28 “**fastestbest** performance”

We have changed the wording of fastest to best as suggested

P16L5-7 “Valcke [...] 2012a”

We have removed this reference and changed 2012b to 2012.

P16L12-14 “Valcke [...] 2015”
Could not find references of these papers in the text.

We have added this reference in Section 1 near the end of the section.

P19 Figure 3
P20 Figure 4
P21 Figure 5
x-axis: maybe use logarithmic base 2 instead of 10 y-axis label: “**secondsTime in s**”

We have renamed the y-axis label, but left the x-axis scale as is.

P19 Figure 3
y-axis: use logarithmic scale to better show behaviour for 1 to 1000 cores per component

We believe the key to this figure is not the time at the lower core counts, but the time at the higher core counts. Switching the y-axis to log will make that information less clear. We have not changed the y-axis scale.

P21 Figure 5
The data set “T799->025,dst” seems to have two data points at 24 core per component while all others only have one.

Thanks for catching that, we have corrected that problem by eliminating a redundant point.

P21 Figure 5
For higher number of cores (> 40), the choice of the symbols for the individual data sets makes it hard to read.

This is a good point. We have changed the symbols and updated the symbol table to make the data more readable. None of the symbols are filled anymore.

P22 Table 4 “**pescores**” P22L23 “**taskscores**”

We have changed both the pes and tasks wording to cores as suggested.

P23 Figure 6
x-axis: maybe use logarithmic base 2 instead of 10

We have not changed the x or y axis scales.

P23 Figure 6
MB or MiB? per core?

MB is typically used when discussing memory use. I don't think it adds to the paper to differentiate between MB and MiB. They differ in definition by less than 5% and that difference has no impact on the plot or discussions. In fact, the scaling of the memory use is more important than the absolute memory use numbers in the plot.

Questions not necessarily relevant for the paper
P1L19-20 “10,000 two dimensional coupling fields”
In case of 3d fields, would the different levels be counted as separate fields?

In the underlying implementation of the new "bundle" feature, the 3d fields are treated under the covers are multiple 2d fields. We count multi-level 3d fields as multiple 2d fields. The requirement for using 2d bundled field is the same as the requirement for coupling multiple fields in a single namcouple statement, i.e. those fields have to share the same grids, masks and will use the same mapping file.

P7L24 “OASIS3-MCT provides some new capabilities to detect potential deadlocks before they occur” Very interesting! Can you be more specific?

Several checks were added like making sure time didn't go backwards, making sure a coupling period wasn't skipped, and others. Some of the new checks had to be removed or deprecated to support sequential coupling on overlapping pes. In general, the new capabilities are not adequate to prevent deadlocks.

P7L28-29 “In OASIS3-MCT, puts are always non-blocking while gets are blocking.” Are there plans for non-blocking gets?

There are no plans for non-blocking gets. In general, we presume that users execute a get when the data is needed. A non-blocking get would require users add a wait in their code before they could use the data which we think adds complexity with little gain. There is lack of symmetry with respect to put and get in systems such as this. If the community requests non-blocking gets, they could probably be implemented but with some additional burden on users and the user implementation.

P12L11-12 “the cost associated with generating the mapping files can be moved to a preprocessing step” which not necessarily has to be faster, if weight computation is done in parallel.

This is true. But right now, Oasis3-MCT does not provide an on-line parallel weights computation capability. Several offline tools do provide that capability. In addition, those offline tools have experts in grid and weights generation that cannot (and maybe should not) be duplicated within Oasis. The complexity associated with generating weights on (for instance) complex unstructured grids, and for many different types of gridding options (bilinear, conservative, higher order, gradient preserving, nearest neighbor, and so forth) are probably best dealt with by specialized tools outside Oasis, and these tools do already exist and exceed any capability that Oasis could build. Having said that, if future requirements, such as time evolving grids impose new requirements on Oasis for fast, parallel weights generation, Oasis will consider incorporating additional external tools into the infrastructure. This section of text in the paper was updated to reflect these ideas.