The software architecture of climate models: a graphical comparison of CMIP5 and EMICAR5 configurations

By **K. Alexander and S. M. Easterbrook**

Response to Reviewers

This document is colour-coded as follows:

- Comments by reviewers are in **blue**.
- Our responses are in **black**.
- Blocks of text we have added to the manuscript are in **red**.

Anonymous Referee #1

Received and published: 9 February 2015

"The software architecture of climate models: a graphical comparison of CMIP5 and EMICAR5 configurations" investigates six CMIP5 model configurations and two EMICAR5 configurations. The paper is based on ideas presented by the authors at the AGU conference in 2011. The authors introduce a set of diagrams to visualise the dependencies between model components in the selected Earth system models. All contributing model versions are analysed following the same principles which allows for an immediate intercomparison of software architecture. The additional information about the relative size in terms of source code of the different model model components makes these 8 diagrams a felicitous representation of the general software design. The idea is worth to be published and the diagrams have the potential to become a standard ingredient for every model description paper. Some statements made by the authors
require some more proof (for details see below). Some paragraphs would benefit from
being rewritten as the reasoning and relevance for this paper is not obvious. Conclusions are too vague.

Specific comments

(1.1) P354, l 16ff: Many individual modelling groups have tried to find simple graphical representations of their own code (HadGEM2-ES see Fig 3 in Collins et al., 2011, MPI-ESMLR see Fig. 1 in Giorgetta et al., 2012, for CESM1 see Fig 2 in Hurrell et al., 2013). Are these in your terminology examples of Bretherton diagrams? It needs to be worked out more clearly what modelling groups have done so far and why the approach taken by the authors is a better one (standardised view, provides relevant information rather than dressed up pictures). Already from the three examples provided above one may guess that some figures are more self-explanatory than others in terms of architectural design.

This is a good point. As the Referee points out, there do exist examples of graphical (block-diagram) views of individual models in the literature. We have clarified this, and expanded our discussion of existing architectural diagrams in section 2. The relevant paragraph now reads:

While intercomparisons of skill scores and climatological patterns are important, these results suggest we need more insight into the nature of similarities and differences between models. The above approaches compare the outputs of the models, but tend to treat the models themselves as black boxes. There are very few representations of the high level designs of global climate models. The Bretherton diagram is perhaps the best known visualization (see Figure 3 of NASA Advisory Council (1986)), although it represents an idealized schematic of Earth system processes, rather than the specific design of any model (Cook, 2013). Other architectural diagrams have been created for individual models (see Figure 3 of Collins et al. (2011), Figure 1 of Giorgetta et al. (2013), and Figure 1 of Hurrell et al. (2013)). However, such diagrams are *ad hoc* conceptual views, and don't claim to be comprehensive, nor to accurately reflect the model codes. They therefore make it hard to compare model architectures in a systematic way. A standardized visualization of architectural structure applied to many models would be more useful for intercomparison, and may even help explain observed similarities between model outputs. For example, if the models share significant subcomponents, this would affect the diversity in a multi-model ensemble.

(1.2) P 354,l 20: some reference could be added to explain what a Bretherton diagram is: Figure 2b in Earth System Science Overview: A Program for Global Change, NASA Advisory Council. Earth System Sciences Committee, 1986?

We have included this citation in the paragraph quoted in (1.1).

(1.3) P 354, l 25: I claim that models can be architecturally similar but be still divers in terms of geoscience aspects. HadGEM2-ES, IPSL-CM5A-LR, and MPI-ESM-LR are very similar in architecture but still quite divers in terms of science as they do not share many (if any) components. See also my comment for P364, l2 ff.

While it's certainly true that models with similar architecture might still be very different scientifically, our argument goes beyond just the arrangement of components and coupling. By "architecture", we mean not just the topography of the model's structure, but also the relative sizes of components, and whether components are re-used from other models.

Note also that the architecture we show in the diagrams is only two levels deep for ease of viewing (components + selected subcomponents). In reality, there are many more levels of architecture that we don't illustrate. The diagrams we show in this paper only give a first-order visualisation of architecture, meaning that some models may seem more architecturally similar than they really are. Finding useful ways of visualizing additional layers is challenging, but may be suitable for future work.

However, this point requires deeper analysis than would make sense for the current study, so we have edited the paragraph quoted in (1.1) to avoid any claim that architecturally similar models would necessarily be scientifically similar. We have also modified our discussion of the relationship between architecture and model climatology in the discussion section to clarify that the relationship is a hypothesis – see our response to point (1.13) below.

(1.4) P 354, l 27 ff: While the authors put forward geoscientific arguments in the previous paragraphs, here they make a short detour into the development of coupler software. While there is nothing wrong about the idea of sharing and reusing model infrastructure (not only for coupling) I do not see the relevance of this statement for this paper. Using coupler A versus coupler B does not have a direct impact on the quality of science that we get out of an Earth system model. I understand Randall (2011) the way that he is addressing the coupling algorithms but not coupling software.

Our paper is not just about "the quality of science that we get out of an Earth system model". We are also interested in the qualities of software architecture that enable efficient development of the science code.

We argue that coupling software is far more important than most scientists generally assume, and that one of the advantages of our diagrams is that they make visible the infrastructure code that is otherwise invisible in scientific descriptions of the models. Because the coupling software represents a significant portion of the code base, it is also a significant part of the development effort of a climate model. Yet this is not code that can easily be handed over to software engineers to build. The specialists who build and maintain this infrastructure code at most modeling centres need expertise in the scientific domains of the software components, in the engineering choices that affect performance on HPC platforms, and, most importantly for this paper, knowledge of how the coupling architecture can facilitate or hamper the ability to address particular scientific questions with the coupled model.

The coupler also indirectly affects the scientific output of models, as we argue in a new paragraph added to section 2:

Coupling software also has an indirect effect on the scientific output of Earth system models, particularly by influencing development pathways. The design of a coupler largely determines the difficulty of using a new scientific component: whether it can be directly linked to the coupler (and if so, how easily?) or if it has to be nested within the atmosphere or ocean components. The coupler design strongly influences parallelization, particularly load balancing between components, and ultimately what simulations can be run within the given constraints on computing power. This will influence the kinds of scientific questions modellers decide to pursue, which in turn further impacts model development. In coupled model experiments, the scientific phenomena of interest (e.g. Earth system feedbacks) tend to cross the boundaries of individual model components. Hence the coupler design can have an unexpected influence on the fidelity of the physical processes in the model. In addition, the coupler design can affect the ease with which scientists can explore how well the model captures large-scale processes (such as ENSO) that cross model component boundaries.

(1.5) P 355, l 9 ff: As already stated in the introduction I appreciate the way how the authors address the general architecture of a given Earth system model, and I have no objection against the statement made in the last sentence of this para, the demand for a comparative analysis. However, I cannot follow the logics that lead to this statement. The text raises the expectation that the authors will now present a top-down analysis, but then the authors only present a top analysis and do not go down deep into the code as it would be necessary for an investigation, comparison and revision e.g. of a coupling algorithm (not coupling software).

We have deleted “top-down” in this sentence, and we have added the following text to section 2 to explain how our study represents a step towards the larger goal:

In response to this observation, we argue that a comparative analysis of the architecture of Earth system models is necessary. The analysis we present in this paper is a first step towards this goal - we focus on the design decisions represented by the top few levels of the dependency tree for each model, without extending the analysis to the low-level routines. Our method of identifying modules based on dependency structure could be applied recursively, leading to much larger and more complex diagrams. We have not pursued this further in current study due to the immense amount of code involved. Our study therefore represents a first step towards a top-down analysis of model architecture at all levels.

(1.6) P 355, l 23: Selection criteria are missing. Why do the authors select those six out of 45 model configurations, why three models from a single country (USA) but none from Asia or Australia?

We have added the following paragraph to section 3:

We could only analyse models where we had access to both the complete source code, *and* a contact at the given institution who was willing to help us pre-process the code and answer questions about the model. We generally relied on our existing contacts, which meant that the resulting models were not geographically representative of the CMIP5 and EMICAR5 participants. However, the variety of component structures and complexity levels found in these eight models suggests that we have sampled across a wide range of CMIP5 and EMICAR5 model architectures.

(1.7) P 359, l 26: If components share the same grid the nesting of those components is probably also more efficient performance wise (as MPI messages are saved) and it eliminates the problem of load balancing between those nested components or enhances the problem of load balancing as there is less freedom in distributing processes.

We have added a sentence to this paragraph acknowledging the implications of component nesting for parallelization:

When two components share the same grid (spatial discretization), nesting them in this manner is much less complicated than routing them through the coupler. It also leads to a simpler, albeit less flexible, parallelization scheme. This approach retains the historical paradigm of Atmosphere-Ocean GCMs (AOGCMs) rather than comprehensive Earth System Models (ESMs), even if the model contains all the processes found in an ESM.

(1.8) P 361, l 15ff: Line count may correlate with code complexity. But I can have numerous physical or dynamical processes implemented in my model code like different advection schemes or radiation code while only one scheme is selected via Fortran-if at run time. The software stack can become incredibly complex (even after the preprocessing step) while the really active model code can still remain remarkably simple. Can the authors comment on this?

The models in our study tend to strip out these unused options during pre-processing, so they are not present during compilation or at run-time. Most models in our study use CPP (C Pre-Processor), which encloses blocks of code in #ifdef statements rather than Fortran if statements; the contents of these #ifdef blocks are either selected or removed prior to compilation based on the options that have been selected.

Only two models in our study (CESM and MPI) may still contain unused code in the manner suggested. CESM uses a preprocessing system other than CPP, where saving the fully extracted code is not possible. The developers of this model trimmed down the code as much as possible before sending it to us. MPI-ESM requires further preprocessing for each experiment (eg CMIP5 historical runs vs RCP simulations) beyond the preprocessing required to extract the configuration MPI-ESM-LR. The developers of this model sent us the code for MPI-ESM-LR including options for all experiments, but they assured us that the amount of duplicated/unused code was minimal. Therefore, the line counts for CESM and MPI may be slightly inflated, but we do not think these issues have significantly biased our results. To acknowledge this point, we have added the following footnote to section 3:

CESM and MPI could not be fully preprocessed for our analysis. Their line counts in Figures 1, 7, and 9 may be slightly inflated; however, we do not think this has significantly biased our results.

(1.9) Is the finding of Herraiz et al. (2007) applicable to Earth system model code?

Herraiz et al demonstrated that different measures of software complexity all correlate well with line count. We have not directly tested this for Earth system model code, although it might be worth doing so as part of a broader investigation of code metrics for scientific code.

We do know that the finding is robust across a range of open source projects, and Earth system model codes share many of the important attributes of such open source projects. Most notably, two of the Earth System models for which we have obtained historical data exhibit the same steady long-term linear growth in size of the code base that characterizes open source projects, over periods as long as fifteen years of model development. Part of this analyses is included in Easterbrook and Johns 2009, as we have now discussed in section 4.3 (the rest is, as yet, unpublished):

Over the development history of a climate model, the line count tends to grow linearly. For example, Easterbrook and Johns (2009) showed that the UK Met Office model grew steadily from 100,000 lines of code in 1993 to nearly 1 million by 2008. The bulk of this code growth is due to addition of new geophysical processes to the model, and an increase in sophistication of the processes that are already incorporated.

The simplest explanation of a long-term linear growth trend in line count is that it represents a steady accumulation of new science in the model, as would be predicted by Herraiz et al's findings. The main challenge to this explanation is that we found one significant discontinuity in the growth of the code base of Had-GEM, when the old ocean component was replaced with NEMO, and it's not clear whether NEMO is a "simpler" model. This example suggests that one has to be careful in using line count as a proxy for complexity when comparing models build by different labs, rather than for analyzing the distribution of complexity within a single model.

(1.10) P362, l 1 ff: What is the knowledge we gain from this paragraph. The more source code a component has the more complex it is? What is the benefit of including poorly understood processes? Does a more complex model deliver a better climate? Is there any insight this analysis can provide about scientific quality of model components, model systems and model results?

The question of whether a more complex model delivers a better climate simulation is hotly contested across the modeling community, and we certainly don't intend to answer it here. We are also not making any claims about the relationship between complexity and scientific quality, and the Referee's point about inclusion of poorly understood processes underscores this.

The point we are making is that the distribution of complexity within a coupled model system does reveal interesting patterns about where a particular lab has invested time and effort, and hence in which parts of the earth system they are likely to have pools of expertise in, and what sort of scientific questions they address. For example, a model with a highly developed active carbon cycle model (such as IPSL-CM5A) should be well-suited to long-term analysis of paleoclimate, while a model with a detailed atmospheric chemistry component (such as HadGEM2-ES) should be better suited to short-term studies of air quality and weather systems. Such observations could be cross-checked with studies of scientific fidelity, although quantitative skill metrics for specific uses of a coupled model do not yet exist.

(1.11) P 363, l 20 ff: The logics of this paragraph is not clear to me. It starts with "organized collaborations between institutions", moves to the OASIS coupler which is built in Toulouse (at CERFACS, to my knowledge with only financial support by CNRS). Is this really a good example for a multi-institutional effort? MPI-ESM-LR is given as an example which uses OASIS. HadGEM(2-ES) is highlighted as a model which “consists almost entirely of in-house components (except for the UKCA atmospheric chemistry)”.How does this compare to the MPI-ESM-LR whose components are almost entirely developed in-house but for the coupler? The message (if any) of this paragraph is more confusing than helpful.

OASIS and NEMO both originated as single-institution components, but have transitioned to multi-institution components, through the support of EU funding and multi-lateral agreements between labs to support the development efforts. The HadGEM example is intended to illustrate that labs which have preferred to develop in-house in the past are now moving more towards the collaborative model. We have edited this paragraph to clarify:

In recent years, there have also been organized collaborations between institutions to build shared components with high levels of scientific complexity. These components are then included in several coupled modelling systems, and typically can also be run in stand-alone configurations. For example, the ocean model NEMO (Madec, 2008; Vancoppenolle et al., 2008), which was originally developed at IPSL, is now incorporated into several European models, and its ongoing development is now managed by a consortium of five European institutions. IPSL (Figure 6) and MPI (Figure 7) both use the OASIS coupler (Valcke, 2013), which was originally developed by scientists from the French institutions CERFACS and CNRS, and is now supported by a collaborative EU-funded research project. Other models are also moving in this direction. For example, the version of HadGEM (Figure 5) included in this study consists almost entirely of in-house components (the UKCA atmospheric chemistry subcomponent is the only major piece of code developed externally), but has now incorporated OASIS, NEMO, and CICE into its next release (Hewitt et al., 2011).

(1.12) P 364, l 16: The factor of 20 one probably gets when comparing EMICs with a full ESM? Is this a fair comparison? A simple box model is likely to be even smaller. But what is the message here, that adding more processes to the model system increases the number of lines of code? Hm.

We have updated the text to include this measurement restricted to the six GCMs (factor of 7), and to clarify that the factor of 20 requires comparing a GCM to an EMIC. We have chosen to retain the factor of 20 statement because we believe it is an interesting and useful illustration of how different GCMs and EMICs are in terms of total complexity, and should help a non-expert reader understand the distinction between these classes of model:

Finally, climate models vary widely in complexity, with the total line count varying by a factor of 20 between the largest GCM and the smallest EMIC we analyse (Figure 9). Even when restricting this comparison to the six GCMs, there is still a factor of 7 variation in total line count.

(1.13) P 364, l 20: The authors state that similarities in architecture lead to a similarity in the simulated climate. I cannot deduce such from Masson and Kutti (2011). On the contrary, IPSL-CM5A-LR and MPI-ESM are similar in architecture but differ in the simulated climate. Can the authors give some evidence for their believe and hypothesis?

This is a similar point to that made in (1.3) above, and our response is the same. It is certainly only a hypothesis, but one that our study offers at least weak support for. We have edited this paragraph to include quotes from Knutti’s 2013 paper that help to explain the intent of this discussion:

Our analysis also offers new insights into the question of model diversity, which is important when creating multi-model ensembles. Masson and Knutti (2011) and Knutti et al. (2013) showed that models from the same lab tend to have similar climatology, even over multiple model generations. We believe this can be explained, at least in part, in terms of their architectural structure and the distribution of complexity within the model. As Knutti et al. (2013) suggest, “We propose that one reason some models are so similar is because they share common code. Another explanation for the similarity of successive models in one institution may be that different centers care about different aspects of the climate and use different data sets and metrics to judge model ‘quality’ during development." Our analysis offers preliminary evidence to support both of these hypotheses. We hypothesize further that the relative size of each component within an Earth system model indicates the relative size of the pool of expertise available to that lab in each Earth system domain (once adjustments are made for components imported from other labs). The availability of different areas of expertise at each lab may provide a sufficient explanation for the clustering effects reported by Masson and Knutti (2011) and Knutti et al. (2013).

(1.14) P 365, l 2 ff: Even though the diagrams do help very much to get an overview of the general design (which is useful and helpful) I am not convinced that the diagrams help to understand the inner workings of climate models. Statements made by the authors are perfectly right. Then the authors speculate about potential usage of their approach. At conferences the authors have communicated their idea about how to visualise those internal structures of Earth system models for some years now. Thus, I would have been glad to read about concrete examples where their approach has already brought some light into the dark mystery of source code and how this has already helped climate scientists to understand each others source code.

We have a growing body of informal evidence that our architectural diagrams are valuable to the scientific community. Each presentation at conferences and workshops has generated a high level of interest in the audience, particularly among scientists who work directly with the models we analyse. They appreciate seeing the diagram representing their institution’s model, but are often much more interested seeing in the diagrams for other models. Seeing the top-level architectural designs of models they have never worked with allows them to see how other institutions have taken different approaches to address the same challenges. Sometimes these comparisons lead to surprises: in particular, GFDL’s strategy of routing all atmosphere-ocean fluxes through the sea ice component is so unknown to scientists outside GFDL that several have initially insisted that our diagram was mistaken.

At least three scientists have used our diagrams in their own presentations, and there may be others who we have not heard about (earlier versions of our diagrams were available freely online). We have also received several reports of eager discussion within research groups that have come across our diagrams. This work has also been useful for public communication: most notably, Scott K. Johnson included one of our diagrams in an article explaining how climate models are built, published on the popular-science website Ars Technica: http://arstechnica.com/science/2013/09/why-trust-climate-models-its-a-matter-of-simple-science/

Based on the reactions we have witnessed, we are confident in the value of our architectural diagrams as scientific communication tools. However, we are reluctant to include this evidence in the current study, as it is necessarily informal. A more rigorous approach would require formal interviews, ethics approval, and potentially the writing of a second manuscript suited to a different journal.

(1.15) P 366 Appendix: Could the list be completed with links to all models? A reference for each Earth system model would be helpful if one wants to learn more about the inner workings of the respective models.

We have added references to the description paper(s) for each model to Table 1.

(1.16) Technical remarks:
p360, l 17: . . . the the . . .

We have fixed this typo and thank the Referee for bringing it to our attention.

(1.17) additional citations:
Collins et al. (2011): Development and evaluation of an Earth-System model – HadGEM2, Geosci. Model Dev., 4, 1051-1075, doi:10.5194/gmd-4-1051-2011.
Giorgetta, M. A., et al. (2013), Climate and carbon cycle changes from 1850 to 2100 in MPI-ESM simulations for the Coupled Model Intercomparison Project phase 5, J. Adv. Model. Earth Syst., 5, 572–597, doi:10.1002/jame.20038.
Hurrell et al., 2013: The Community Earth System Model: A Framework for Collaborative Research. Bull. Amer. Meteor. Soc., 94, 1339–1360. doi:http://dx.doi.org/10.1175/BAMS-D-12-00121.1

We would like to thank Referee #1 for the time and effort providing this helpful feedback on our manuscript.

The software architecture of climate models: a graphical comparison of CMIP5 and EMICAR5 configurations

By **K. Alexander and S. M. Easterbrook**

Response to Reviewers

This document is colour-coded as follows:

- Comments by reviewers are in **blue**.
- Our responses are in **black**.
- Blocks of text we have added to the manuscript are in **red**.

Anonymous Referee #2

Received and published: 21 February 2015

Summary:
The authors analyze the source code of several climate models and determine their structure, i.e. how components are separated and interact from a software point of view. They visualize these structures in simple diagrams. They argue that these diagrams offer insights into the similarities and differences between models.

Recommendation:
This is an interesting and novel idea to look at code structures, a well written manuscript, and I recommend publication with minor changes in the discussion and implication sections. I'm still unsure whether we can learn anything about such analysis from a climate point of view (see details below), and how much this analysis affects how people think or work on models, and the authors have not demonstrated that. The discussion should be sharpened here. But the question of how such models are built, documented, tuned, how information is shared between modeling groups, and therefore how ensembles of models should be interpreted, is a very important one. Few people have looked at those broader questions, and it is great to see contributions from the software engineering community in this field. I am sure this manuscript will stimulate discussions, and in the long run help people to understand both how models are built, and how to interpret the zoo of models out there.

Specific comments:
(2.1) Abstract, page 352, line 11: "These diagrams offer insights into the similarities and differences between models, and have the potential to be useful tools for communication between scientists, scientific institutions, and the public.": I would argue the diagrams offer insight in the software structure, but not in the behaviour of the model. Two models can have similar structures but behave very differently, in fact even the same code with perturbed parameters can. In the end all of those models describe the same physical system, and whether a sea ice model is a subcomponent of the ocean or a separate component called from a coupler should not make any difference to the simulation of the climate, if things are done properly. The results from a model should be determined by the set of equations used (plus some parameterizations and numerical assumptions of course), and should be largely independent from the software framework, or the language used.

Model architecture may not have a direct impact on scientific output, but it can influence development pathways (e.g. highly modularized code is easier to edit and/or swap with another component) which then has a second-order effect on which processes get incorporated into the model, the parameterizations used, and hence ultimately the output. Architecture and past development pathways are tightly coupled. Also (as for Referee #1) our paper is not just about “the quality of science that we get out of an Earth system model”, we are also interested in the software architecture that enables efficient development of the science code.

We have modified the wording in the abstract to clarify this:
These diagrams offer insights into the similarities and differences in structure between climate models, and have the potential to be useful tools for communication between scientists, scientific institutions, and the public.

(2.2) The implication of this view is that such visual representations may be a useful communication tool for those who build and use climate models, but not necessarily for the public, and apart from the size of the components we cannot infer much (if anything) about how the model will behave, or how similar it will be to reality, or to other models. I do think the information provided here is valuable, but it is limited to software, and largely irrelevant for climate. It would be appropriate to state that clearly I think, unless the authors convince me that there is more value that I have missed so far.

We argue that these diagrams are useful to the public, who might otherwise picture climate models as black boxes that magically spit out climatology, and not stop to think about how the models are set up and the engineering challenges involved. We have anecdotal evidence that even experienced policymakers engaged in climate negotiations tend to think of climate models as fancy ways of computing linear regression over climatic trends. While our diagrams alone cannot correct such misperceptions, we believe that a visual representation of the architecture, along with some sense of the complexity of the code, can go a long way in helping explain to a lay audience what really goes into a climate model.

(2.3) In this case it would be good to see some specific examples and applications where such representations have been used, or have influenced something, or where people could infer climatic behaviour from the structure of the software.

We have a growing body of informal evidence of the value of our diagrams, but do not consider it suitable for inclusion in the current paper without a more rigorous empirical evaluation. Please see our more detailed response to point (1.14) made by Referee #1.

(2.4) Page 353, line 1: “We argue that these differences in module size offer a reasonable proxy for scientific complexity of each component”: It would be useful to define complexity early here (it comes in a parenthesis statement on page 361 but not clearly). The statement suggests that by complexity the authors mean “amount of stuff”, i.e., number of processes, or degree of detail considered. One could also interpret complexity by how rich the results are in terms of behavior. The equations of motion or a Lorentz system could be described as complex in terms of their behaviour, even though they don’t need many lines of code. Increasing the resolution of a model could massively increase the complexity in terms of the simulated patterns or structures of clouds, fronts, storms, while keeping the exact same code.

We have added wording in the introduction to clarify what we mean by complexity, and we have also added wording to section 4.3 to emphasize that we don’t include resolution of the model in this definition. The relevant sentence of the introduction now reads:

We argue that these differences in module size offer a reasonable proxy for the *scientific complexity* of each component, by which we mean the number, variety and sophistication of the set of geophysical processes included in the simulation with respect to a given part of the Earth system.

The relevant wording in section 4.3 is:

These observations allow us to treat line count as a proxy for scientific complexity, by which we mean the number and sophistication of physical processes represented in the model. Over the development history of a climate model, the line count tends to grow linearly. For example, Easterbrook & Johns (2009) showed that the UK Met Office model grew steadily from 100,000 lines of code in 1993 to nearly 1 million by 2008. The bulk of this code growth is due to addition of new geophysical processes to the model, and an increase in sophistication of the processes that are already incorporated. Note that we exclude changes in resolution of the model from our definition of scientific complexity, as changes to the grid size and timestep don't necessarily entail addition of new code.

(2.5) Page 361, line 25: the land component is not just about carbon cycle, but also about land surface processes, vegetation, and hydrology.

We have updated this sentence as follows:

In other models, particularly CESM (Figure 1) and IPSL (Figure 6), the land component is relatively substantial (although not the largest component in the model); this may reflect the growing interest by the modelling community in terrestrial carbon cycle feedbacks, land surface processes, vegetation, and hydrology.

2.6) Page 364, line 16: I'm not sure the interpretation of this factor of 20 is appropriate and useful. One might compare lines of code between different models of the same class (e.g. AOGCMs) but in a hierarchy of models the lines of code will obviously differ. An energy balance model can be coded with just handful of lines, and is also a climate model in some sense, but of course it has a very different purpose.

We have updated the text to include this measurement restricted to the six GCMs (factor of 7), and to clarify that the factor of 20 requires comparing a GCM to an EMIC. We have chosen to retain the factor of 20 statement because we believe it is an interesting and useful illustration of how different GCMs and EMICs are in terms of total complexity, and should help a non-expert reader understand the distinction between these classes of model:

Finally, climate models vary widely in complexity, with the total line count varying by a factor of 20 between the largest GCM and the smallest EMIC we analyse (Figure 9). Even when restricting this comparison to the six GCMs, there is still a factor of 7 variation in total line count.

5) Page 364, line 17ff: I like the comparison here but would add a few things. The amount of expertise may determine the amount of development of individual code, but does not imply similarity or differences to other codes a priori. In practice it may be the case, but it's a hypothesis (which is hard to test, and I'm not asking for a test).

This is true. We have updated the wording in this paragraph to indicate this is a hypothesis. Please see our response to reviewer 1, point (1.13).

Second, it would be good to emphasize that these two approaches take very different but complementary approaches. One looks at the structure of the code, how the model is built, and the other looks only at the model output, without knowing anything about the code.

We have added a sentence to the conclusions to emphasize this point:

Furthermore, the two analyses are complementary: while our analysis looks at model code without considering its outputs, Masson and Knutti (2011) and Knutti et al. (2013) analyze model outputs without looking at the code.

Finally, there is an update on the model clustering for CMIP5 by Knutti et al., GRL 2013, doi:10.1002/grl.50256.

We have added references to this publication wherever we cite the original clustering paper by Masson and Knutti, 2011.

We would like to thank Referee #2 for this thoughtful review, which led to direct improvements in our manuscript.

Manuscript prepared for Geosci. Model Dev. Discuss.
with version 2014/07/29 7.12 Copernicus papers of the LaTeX class copernicus.cls.
Date: 19 March 2015

The software architecture of climate models: a graphical comparison of CMIP5 and EMICAR5 configurations

K. Alexander[1,*] **and S. M. Easterbrook**[1]

[1]Department of Computer Science, University of Toronto, 10, King's College Rd, Toronto, Ontario, Canada
[*]now at: Climate Change Research Centre and ARC Centre of Excellence for Climate System Science, University of New South Wales, Sydney NSW 2052, Australia

Correspondence to: S. M. Easterbrook (sme@cs.toronto.edu)

Abstract

We analyse the source code of eight coupled climate models, selected from those that participated in the CMIP5 (Taylor et al., 2012) or EMICAR5 (Eby et al., 2013; Zickfeld et al., 2013) intercomparison projects. For each model, we sort the preprocessed code into components and subcomponents based on dependency structure. We then create software architecture diagrams which show the relative sizes of these components/subcomponents and the flow of data between them. The diagrams also illustrate several major classes of climate model design; the distribution of complexity between components, which depends on historical development paths as well as the conscious goals of each institution; and the sharing of components between different modelling groups. These diagrams offer insights into the similarities and differences ~~between~~ in structure between climate models, and have the potential to be useful tools for communication between scientists, scientific institutions, and the public.

1 Introduction

Global climate models are large and complex software systems, consisting of hundreds of thousands of lines of code, and a development history spanning years or even decades. Understanding what each model does and how it differs from other models is a difficult problem. Existing approaches to model comparison focus on measures of a model's skill in reproducing observed climates of the past, and on informal discussion of differences in how physical processes are resolved or parameterized within each model.

In this paper, we take a different approach. We characterize the software architecture of each model by analysing how the physical domains of the Earth system are modularized in the models, how these modules interact, and the relative sizes of these modules. The analysis reveals differences between models, both in terms of the architectural decisions regarding coupling between Earth system components, and also in terms of where the bulk of the code lies. We argue that these differences in module size offer a reasonable proxy for ~~scientific complexity~~ the *scientific complexity* of each component, by which we mean the number, variety and sophistication of the set of geophysical processes included in the simulation with respect to a given part of the Earth system. This in turn offers preliminary evidence that when modelling groups tend to specialize in different parts of the Earth system, these specializations are reflected in the architecture of their models.

2 Background

Intercomparison of models is now standard practice in Earth system modelling, as it provides insights into the strengths and weaknesses of each model, and generates standard model runs for more formal measurements of model skill. The World Climate Research Program website (2014) currently lists 45 active Model Intercomparison Projects (MIPs). Typically, these intercomparison projects proceed by defining an agreed set of model experiments which represent the different conditions models might be expected to simulate, often with (re-gridded) observational data provided as a baseline for comparison. Some of these intercomparison projects were also designed to provide a coordinated set of Earth system model runs as input to the IPCC assessment reports. In the 5th IPCC assessment report (AR5), the long term projections of future climate change were generated from CMIP5 (Taylor et al., 2012) for Global Coupled Climate Models (GCMs), and EMICAR5 (Eby et al., 2013; Zickfeld et al., 2013) for Earth System Models of Intermediate Complexity (EMICs).

Comparisons between models are normally expressed in terms of model skill relative to the given observational data, with skills scores computed by measuring mean-squared error for selected fields. For example, Reichler and Kim (2008) sum the mean squared errors at each grid point for each of 14 annually averaged variables, normalize them to account for grid variations in mass and area, and combine these to produce a single skill score for each model. Their results indicate that model error is steadily declining over successive generations of global climate models.

An alternative approach is to directly compare the climatology of the models against each other, by analyzing the spatial and temporal patterns simulated for a specific variable. For example, Masson and Knutti (2011) and Knutti et al. (2013) use this approach on monthly fields for surface temperature and precipitation to generate a cluster analysis on families of models. Their results show that models from the same lab tend to have similar climatology, even across model generations, as do models from different labs that use the same atmosphere or ocean components.

Understanding the relationships between different models is particularly important for creating model ensembles and probabilistic forecasts (Collins, 2007). Currently, model ensembles tend to be “ensembles of opportunity”, where all models of a given class are included, with no attempt to weight for either relative skill nor model similarity in the ensemble. Multi-model ensembles tend to outperform single models in overall skill, because weaknesses in any single model are compensated for by other models in the ensemble. However, these ensembles appear to have less diversity than expected (Knutti, 2008).

While intercomparisons of skill scores and climatological patterns are important, these results suggest we need more insight into the nature of similarities and differences between models. The above approaches compare the outputs of the models, but tend to treat the models themselves as black boxes. There are very few representations of the high level designs of global climate models. The Bretherton diagram is perhaps the best known visualization (see Figure 3 of NASA Advisory Council (1986)), although it represents an idealized schematic of Earth system processes, rather than the specific design of any model (Cook, 2013). ~~A comparison of the architectural structure of the models should offer useful insights that may~~ Other architectural diagrams have been created for individual models (see Figure 3 of Collins et al. (2011) , Figure 1 of Giorgetta et al. (2013) , and Figure 1 of Hurrell et al. (2013)). However, such diagrams are *ad hoc* conceptual views, and don’t claim to be comprehensive, nor to accurately reflect the model codes. They therefore make it hard to compare models in a systematic way. A standardized visualization of architectural structure applied to many models would be more useful for intercomparison, and may even help explain observed similarities between model outputs. For example, if the models ~~are~~

~~architecturally similar, or~~ share significant subcomponents, this would affect the diversity in a multi-model ensemble.

~~Interest in these architectural patterns~~ Coupling software also has an indirect effect on the scientific output of Earth system models, particularly by influencing development pathways. The design of a coupler largely determines the difficulty of using a new scientific component: whether it can be directly linked to the coupler (and if so, how easily?) or if it has to be nested within the atmosphere or ocean components. The coupler design strongly influences parallelization, particularly load balancing between components, and ultimately what simulations can be run within the given constraints on computing power. This will influence the kinds of scientific questions modellers decide to pursue, which in turn further impacts model development. In coupled model experiments, the scientific phenomena of interest (e.g. Earth system feedbacks) tend to cross the boundaries of individual model components. Hence the coupler design can have an unexpected influence on the fidelity of the physical processes in the model. In addition, the coupler design can affect the ease with which scientists can explore how well the model captures large-scale processes (such as ENSO) that cross model component boundaries.

Interest in the architectural patterns of coupled Earth system models is also driven by growing interest in the use of shared infrastructure code~~in coupled Earth system models~~ (Dickinson et al., 2002). The growing complexity of the coupling task means that couplers require more expertise to develop, and that labs can benefit by comparing their approaches, sharing lessons learnt, and re-using coupling code (Valcke et al., 2012). At the same time, there has been a move towards general reusable subcomponents (e.g. both atmosphere and ocean models using the same numerical solver), compared to earlier model generations, where the code for each component was developed entirely separately. However, code modularity remains a challenge, because nature itself isn't modular. Randall (2011) argues that insufficient attention is given to the challenge of coupling, due to a belief that the science can be contained entirely within modular components.

A discussion of these issues is hampered by a lack of detailed descriptions of the design of Earth system models. While some descriptions of software design are starting to appear (e.g. Drake, 2005), detailed analysis of the design of global climate models remains a challenge because the models have undergone continued code modification for years, and in some cases, decades. This makes a reductionist analysis of specific design decisions impossible, because each design decision is “generatively entrenched” (Lenhard and Winsberg, 2010) – that is, design features form a complex web because each has played a role in generating the others. Furthermore, each lab retains a deep but tacit knowledge base about their own models, which is readily apparent to anyone spending time in that lab (Easterbrook and Johns, 2009), but hard to share through model intercomparison projects.

In response to this observation, we argue that a ~~top-down~~ comparative analysis of the architecture of Earth system models is necessary. The analysis we present in this paper is a first step towards this goal - we focus on the design decisions represented by the top few levels of the dependency tree for each model, without extending the analysis to the low-level routines. Our method of identifying modules based on dependency structure could be applied recursively, leading to much larger and more complex diagrams. We have not pursued this further in current study due to the immense amount of code involved. Our study therefore represents a first step towards a top-down analysis of model architecture at all levels.

3 Methods

For our analysis we selected eight climate models with varying levels of complexity. These include six GCMs which participated in CMIP5 (Taylor et al., 2012), and two EMICs which participated in the EMICAR5 intercomparison project (Eby et al., 2013; Zickfeld et al., 2013). For a summary of information about each model, see Table 1. We focus on models from the CMIP5 and EMICAR5 ensembles because of the central role these projects play in the IPCC assessment reports.

We could only analyse models where we had access to both the complete source code, *and* a contact at the given institution who was willing to help us pre-process the code and answer questions about the model. We generally relied on our existing contacts, which meant that the resulting models were not geographically representative of the CMIP5 and EMICAR5 participants. However, the variety of component structures and complexity levels found in these eight models suggests that we have sampled across a wide range of CMIP5 and EMICAR5 model architectures.

The first step in analysing each model was preprocessing: stripping out unused code. Each model is a specific configuration of a larger software package (for example, CESM1-BGC is a configuration of the CESM 1.0.5 package) with many available options, including how subgrid-scale eddies are parameterized in the ocean, whether greenhouse gases are updated with emissions or with prescribed concentrations, and whether calculations of ice sheet dynamics are included or excluded. Preprocessing is the first step in the build process, and uses software such as CPP (C Preprocessor) to remove unused options from the code base.

In an earlier version of our analysis (Alexander and Easterbrook, 2011), we used the entire code base for each model, without performing any pre-processing. However, the resulting line counts tended to reflect the number of configuration choices in the code, rather than the size and complexity of the code actually used in a model run. Since we wanted to analyze the models as they were used in the Model Intercomparison Projects, we decided to use preprocessed code for our analysis, to ensure that line count would reflect the size of the actual science code used in the model runs.[1]

The preprocessed code was then analysed using the Understand software (http://www.scitools.com/) in order to extract the dependency structure: which source code files depend on which, through the use of function and subroutine calls. This structure can be interpreted as a directed graph, where any given file calls its “children”, is called by its “parents”, and has recursively defined “ancestors” and “descendants”.

[1]CESM and MPI could not be fully preprocessed for our analysis. Their line counts in Figures 1, 7, and 9 may be slightly inflated; however, we do not think this has significantly biased our results.

3.1 Classification of source code

In order to sort the code for each model into components (atmosphere, ocean, land, and sea ice), we first identified the top-level driver files for each component. For example, the ocean drivers might consist of a subroutine for initialisation, a subroutine controlling the ocean calculations at each timestep, and a subroutine controlling ocean diagnostics and output. All descendants of the ocean drivers were then classified as the ocean component. Files that were not called by any component, such as the main timestep loop and the flux routines, were classified as the coupler. These top-level files are the drivers of the entire simulation, and control data transfer between components.

Files which were called by multiple components, such as math libraries, parameter lists, and file readers, were classified as shared utilities. Other code often found in shared utilities includes numerical methods used by multiple components. For example, an implicit method commonly used to evaluate advection–diffusion equations (found in both the atmosphere and the ocean) involves solving a tridiagonal matrix. To reduce duplication of similar code, a common practice is to write one tridiagonal matrix solver which is shared by the atmosphere and the ocean.

Within each component, the classification process was repeated to identify subcomponents for atmospheric chemistry, ocean biogeochemistry (BGC), and land vegetation, if these processes were included in the model. Furthermore, sometimes one component was entirely contained in, i.e. controlled by, another: land was frequently treated as a subcomponent of the atmosphere, and sea ice as a subcomponent of the ocean. In the UVic model, sea ice is a subcomponent of the atmosphere; UVic also has a sediment component which is separate from the ocean.

Since any given file might contain several functions and subroutines, circular dependencies between files can and do exist in our analysis. It was necessary to sever some of these dependencies in order for the classification to be reasonable. For example, a low-level file reader might access a function stored in the same file as the top-level program. As a result, the main program file and all of its descendants (i.e., the entire model) would be classified as shared utilities. Only by severing (disregarding) the dependency between the file reader and the main program file could the component structure emerge. The number of dependencies severed was extremely small compared to the total number of dependencies in each model.

3.2 Software diagrams

Using David A. Wheeler's "SLOCCount" tool, we performed a line count (excluding comments and blank lines) of the source code for each component and subcomponent. Then we created diagrams for each model where each component or subcomponent is represented by an ellipse whose area is exactly proportional to the line count of the corresponding code base (See Figs. 1 to 8). The components were assigned standard colours: purple atmosphere, blue ocean, orange land, green sea ice, yellow land ice, red sediment, and grey coupler and shared utilities. Coloured arrows show fluxes between components, which we detected from the coupler code. Note that while each individual diagram is to scale, the diagrams are not to scale with each other. However, each diagram includes a legend below the title which shows the area allocated to one thousand lines of code.

4 Discussion

4.1 Architectural designs

Dividing up a complex system into modules and then arranging these modules hierarchically is an important part of making the world "theoretically intelligible to the human mind" (Simon, 1996). However, there are usually many possible choices of decomposition. While Earth system modellers strive, as Plato suggested, to "carve nature at its joints", in practice, judgment is needed to find a decomposition that is fit for purpose. Comparison of architectural patterns in software has become a standard approach for analyzing the constraints that shape such decisions (Shaw and Garlan, 1996).

The boundaries between components in an Earth system model represent both natural boundaries in the physical world (e.g. the ocean surface), and divisions between communities of expertise (e.g. ocean science vs. atmospheric physics). The model architecture must facilitate simulation of physical processes that cross these boundaries (e.g. heat transport), as well as support collaboration between knowledge communities within the work practices of model development (e.g. to study climate feedbacks). Each major model component tends to have two distinct uses: as a stand-alone component used by a specific subcommunity, and as a building block for coupled Earth system modelling. Hence, there is a tension between the need for each component to remain loosely coupled to facilitate its ongoing use as a stand-alone model, and for tighter integration to study climate interactions with the coupled system.

In our model diagrams (Figs. 1 to 8), two main architectural "shapes" are apparent. First, two of the American GCMs (CESM and GISS; Figs. 1 and 3) have a "star-shaped" architecture: each component is separate from the others, connected only through the central coupler. This design reflects a high level of encapsulation between each component of the model, which is attractive from a software engineering perspective. Once this structure is in place, further changes to any component are relatively easy to incorporate into the coupled model. It also facilitates a mix-and-match approach where, for example, an entirely different ocean model can be substituted with a minimum of effort. In fact, switching between several different ocean components is a common practice at the GISS host institution.

However, a star-shaped architecture can introduce significant challenges when building the coupler: handling fluxes between any combination of four to five components is not a trivial task. These difficulties are alleviated by the "two-sided" architecture present in all three European GCMs (HadGEM, IPSL, and MPI; Figs. 5 to 7). In these models, the only components connected to the coupler are the atmosphere and the ocean; other components are subsets of these two. In all three cases, land is contained within the atmosphere and sea ice is contained within the ocean. When two components share the same grid (spatial discretization), nesting them in this manner is much ~~simpler~~ less complicated than routing them through the coupler. ~~This approach also~~ It also leads to a simpler, albeit less flexible, parallelization scheme. This approach retains the historical paradigm of Atmosphere–Ocean GCMs (AOGCMs) rather than comprehensive Earth System Models (ESMs), even if the model contains all the processes found in an ESM.

The two EMICs (UVic and Loveclim; Figs. 4 and 8) both have intermediate architectures between star-shaped and two-sided. For both models, all components are separate except for sea ice, which is nested within a larger component (atmosphere for UVic, ocean for Loveclim). The atypical structure seen in UVic, where sea ice is treated as a subcomponent of the atmosphere rather than the ocean, was implemented because the sea ice and the atmosphere run on similar time scales. Essentially, UVic nests these components based on their temporal discretization rather than their spatial discretization (which is the same for all components in the model).

4.2 Fluxes

Mass and energy fluxes (represented as arrows, coloured based on their component of origin, in Figs. 1 to 8) are simple in two-sided models: the atmosphere and the ocean both exchange information with the other. The process is more complicated in star-shaped models, because not every component needs to receive data from all of the others. In general, the atmosphere passes fluxes to and from all components with which it shares a boundary (i.e. everything except sediment). The ocean and sea ice are also physically adjacent, so they exchange information in both directions. However, fluxes between the ~~the~~ land and ocean are one-way, since runoff (generally the only land-ocean flux which is represented) moves strictly from the land to the ocean. In GISS (Fig. 3), where a land ice component is also present, it passes runoff either directly to the ocean (including calving) or first to the land.

In GFDL (Fig. 2), quite a different dataflow structure is present. Sea ice is treated as an interface to the ocean: a layer over the entire sea surface which may or may not contain ice. All atmosphere–ocean and ~~land-ocean~~ land–ocean fluxes must first pass through the sea ice component, even if the fluxes occur at latitudes where sea ice is never actually present. This approach is convenient for interpolation, because the ocean and sea ice components share the same grid, while the atmosphere and land can differ. However, it also uniquely solves the problem of how to represent sea ice – a component immersed in the ocean but with distinct dynamical and physical processes, whose spatial domain is irregular and may change at each timestep.

4.3 Distribution of scientific complexity

Counting lines of code in a given piece of software has been used widely for decades in software engineering. It is used to estimate the amount of effort needed to build software, to measure programmer productivity, and to assess software complexity. Just as often, its validity is questioned, because it is easy to create examples where a program can be improved by making it shorter. However, in practical software development, such examples are unusual. As long as the line counting is done consistently, and comparisons are only made between programs written in the same language and for the same type of application, the number of lines of code can be remarkably useful to assess the size and complexity of a large software system, and to trace its evolution (Park, 1992). Indeed, line count strongly correlates with other measures of software complexity (Herraiz et al., 2007).

These observations allow us to treat line count as a proxy for scientific complexity~~(the number~~ , by which we mean the number and sophistication of physical processes represented in the model~~).~~ . Over the development history of a climate model, the line count tends to grow linearly. For example, Easterbrook and Johns (2009) showed that the UK Met Office model grew steadily from 100,000 lines of code in 1993 to nearly 1 million by 2008. The bulk of this code growth is due to addition of new geophysical processes to the model, and an increase in sophistication of the processes that are already incorporated. Note that we exclude changes in resolution of the model from our definition of scientific complexity, as changes to the grid size and timestep don't necessarily entail addition of new code.

We can see not only a large variation in scientific complexity between models (Fig. 9), but also variations in how this complexity is distributed within each model. For all six GCMs, the atmosphere is the largest component. This feature is particularly obvious in HadGEM (Fig. 5), which has a high level of atmospheric complexity due to MetUM's dual use as an operational weather forecasting model. However, both EMICs (UVic and Loveclim; Figs. 4 and 8) have a larger code base for the ocean than for the atmosphere. Since EMICs are built for speed, and atmospheric processes generally require the shortest timesteps in a coupled model, concessions in atmospheric complexity will give the best return on integration time. In other models, particularly CESM (Fig. 1) and IPSL (Fig. 6), the land component is relatively substantial (although not the largest component in the model); this may reflect the growing interest by the modelling community in terrestrial carbon cycle feedbacks, ~~many of which are terrestrial~~land surface processes, vegetation, and hydrology.

The distribution of complexity among subcomponents can also yield useful insights. Several models, namely CESM (Fig. 1) and HadGEM (Fig. 5), have a substantial code base for atmospheric chemistry. This subcomponent is designed to model processes such as the effects of sulfate aerosol emissions, which are likely to have a large impact on how much warming the planet experiences in the coming decades, but are nonetheless poorly understood (Rosenfeld and Wood, 2013). Other models, including IPSL (Fig. 6) and MPI (Fig. 7), put more weight on the land vegetation and ocean BGC subcomponents. These pieces of code model longer-term processes, such as carbon feedbacks, which are likely to have a large impact on the total amount of warming the planet will experience before it reaches equilibrium (Friedlingstein et al., 2006).

4.4 Shared utilities

The proportion of each model's code base stored as shared utilities also varies widely. On one end of the spectrum, IPSL (Fig. 6) contains no shared utilities at all. The atmosphere and ocean are completely separate components which call no common files. While this approach makes it easy to mix and match components in different configurations of the underlying software package, it also indicates that there is likely some duplication between the atmosphere code and the ocean code, which solve similar fluid dynamics equations.

Conversely, GFDL (Fig. 2) and UVic (Fig. 4) have particularly large proportions of their source code devoted to shared utilities. This is due to the fact that both models contain source code for a custom version of a major utility. GFDL contains a custom MPP (Message Processing Program) to enable parallelization, while UVic contains a custom version of NetCDF, a self-describing data type frequently used for climate model input and output. While most of the other models also use message passing systems and NetCDF libraries, they use unmodified versions which have been preinstalled on the given computing platform. These out-of-the-box utilities are not recompiled for every simulation, and so the source code is not stored with the model. As such, the shared utilities are correspondingly smaller.

4.5 Origin of components

While each coupled model is developed at a home institution (see Table –1), not every component was necessarily developed in-house. It is common practice for one modelling group to adopt another group's ocean component, for example, and modify it to suit the existing coupled architecture. As development continues on the adopted component, the modifications can become substantial, creating a software fork.

Institutions may decide to share components in this manner for several different reasons. Resource constraints, namely a lack of developers to build a new component in-house, particularly affect the smaller modelling groups such as that of UVic (Fig. 4). The UVic ocean component MOM2 (Weaver et al., 2001) is a modified version of a predecessor to GFDL's ocean component MOM4 (Fig. 2), developed in-house by GFDL. UVic also sourced much of its land component (including the vegetation subcomponent TRIFFID) from code written at the Hadley Centre (Meissner et al., 2003), much of which is present in HadGEM (Fig. 5). However, large modelling groups adopt components from other institutions as well. The CESM ocean POP2 (Smith et al., 2010) and sea ice CICE4 (Bailey et al., 2010) components were both built at Los Alamos National Laboratory, rather than the National Centre for Atmospheric Research (CESM's host institution), and reflect NCAR's goal of creating and supporting a community model.

In recent years, there have also been organized collaborations between institutions to build shared components with high levels of scientific complexity. These components are then included in several coupled modelling systems, and typically can also be run in stand-alone configurations. For example, ~~IPSL~~ the ocean model NEMO (Madec, 2008; Vancoppenolle et al., 2008) , which was originally developed at IPSL, is now incorporated into several European models, and its ongoing development is now managed by a consortium of five European institutions. IPSL (Fig. 6) and MPI (Fig. 7) both use the OASIS coupler (Valcke, 2013), which was originally developed by scientists from the French institutions CERFACS and CNRS~~. IPSL also uses NEMO (Madec, 2008; Vancoppenolle et al., 2008) , an ocean and sea ice model developed~~ , and is now supported by a ~~consortium of five European institutions.~~ collaborative EU-funded research project. Other models are also moving in this direction. For example, the version of HadGEM (Fig. 5) ~~, which~~ included in this study consists almost entirely of in-house components ~~in this configuration~~ (the UKCA atmospheric chemistry subcomponent is the only major piece of code developed externally), ~~is currently incorporating~~ but has now incorporated OASIS, NEMO, and CICE into its next release (Hewitt et al., 2011).

5 Conclusions

These software architecture diagrams show, in a broad sense, how climate models work: how the climate system is divided into components and how these components communicate with each other. They also illustrate the similarities and differences between the eight models we have analysed. Some models, particularly in North America, exhibit a high level of encapsulation for each component, with all communication managed by the coupler. Other models, particularly in Europe, implement a binary atmosphere–ocean architecture which simplifies the coupling process. Institutions focus their efforts on different climatic processes, which eventually causes different components and subcomponents to dominate each model's source code. However, not all models are completely independent of each other: modelling groups commonly exchange pieces of code, from individual routines up to entire components. Finally, climate models vary widely in complexity, with the total line count varying by a factor of 20 ~~among the eight models we analysed.~~ between the largest

GCM and the smallest EMIC we analyse (Figure 9). Even when restricting this comparison to the six GCMs, there is still a factor of 7 variation in total line count.

Our analysis also offers ~~some~~ new insights into the question of model diversity, which is important when creating multi-model ensembles. Masson and Knutti (2011) and Knutti et al. (2013) showed that models from the same lab tend to have similar climatology, even over multiple model generations. We believe this can be explained~~in~~ , at least in part, in terms of their architectural structure and ~~, in particular,~~ the distribution of complexity within the model. ~~We hypothesize~~ As Knutti et al. (2013) suggest, “We propose that one reason some models are so similar is because they share common code. Another explanation for the similarity of successive models in one institution may be that different centers care about different aspects of the climate and use different data sets and metrics to judge model ‘quality’ during development." Our analysis offers preliminary evidence to support both of these hypotheses. We hypothesize further that the relative size of each component within an Earth system model indicates the relative size of the pool of expertise available to that lab in each Earth system domain (once adjustments are made for components imported from other labs). The availability of different areas of expertise at each lab ~~would~~ may provide a sufficient explanation for the clustering effects reported by Masson and Knutti (2011) ~~.~~ and Knutti et al. (2013) . Furthermore, the two analyses are complementary: while our analysis looks at model code without considering its outputs, Masson and Knutti (2011) and Knutti et al. (2013) analyze model outputs without looking at the code.

Our diagrams may prove to be useful for public communication and outreach by their host institutions. The inner workings of climate models are rarely discussed in the media, even by science reporters; as such, these pieces of software are fundamentally mysterious to most members of the public. Additionally, the diagrams could be used for communication between scientists, both within and across institutions. It can be extremely useful for climate scientists, whether they are users or developers of coupled models, to understand how other modelling groups have addressed the same scientific problems. A better understanding of the Earth system models used by other institutions may open doors for international collaboration in the years to come.

Appendix A: Accessing model code

The procedure for obtaining climate model source code varies between institutions. Below are instructions for requesting access to each model.

CESM: Complete the registration form at http://www.cesm.ucar.edu/models/register/register_cesm.cgi. Instructions for accessing NCAR's Subversion repository will then be provided via email.

GFDL: Source code is open access through a GitHub repository; instructions are available at http://www.mom-ocean.org/web/downloads.

GISS: The AR5 branch of ModelE (the software package underlying GISS-E2-R-TCADI) can be downloaded as a compressed file from http://simplex.giss.nasa.gov/snapshots/.

UVic: A compressed file containing the source code can be downloaded via a password-protected link at http://climate.uvic.ca/model/. This page contains instructions for requesting a password via email.

HadGEM: Obtaining source code for climate models developed at the UK Met Office requires signing a user agreement. Contact ~~Tim Johns~~ the UM Collaboration Manager for more information~~.~~, through http://www.metoffice.gov.uk/research/collaboration/um-collaboration.

IPSL: Installation scripts can be downloaded through IPSL's Subversion repository, as described at http://forge.ipsl.jussieu.fr/igcmg/wiki/platform/en/documentation/C_installation. In order for these scripts to fully extract the model source code, a username and password must be requested via email.

MPI: Obtaining source code for climate models developed at the Max Planck Institut für Meteorologie requires signing a user agreement. Contact Reinhard Budich for more information.

Loveclim: Contact Pierre-Yves Barriat at the Université catholique de Louvain to request access to the Loveclim source code.

Acknowledgements. We are grateful to the model developers who granted us access to source code, assisted with preprocessing, answered questions about their models, and provided feedback on the diagrams: Mariana Vertenstein and Jim Edwards from the National Centre for Atmospheric Research; Venkatramani Balaji from the Geophysical Fluid Dynamics Laboratory; Gavin Schmidt from the NASA Goddard Institute for Space Studies; Michael Eby from the University of Victoria; Tim Johns from the UK Met Office; Arnaud Caubel, Marie-Alice Foujols, and Anne Cozic from the Institut Pierre Simon Laplace; Reinhard Budich and Marco Giorgetta from the Max Planck Institut für Meteorologie; and Pierre-Yves Barriat from the Université catholique de Louvain. This project was supported by the Centre for Global Change Science at the University of Toronto as well as the National Sciences and Engineering Research Council of Canada.

References

Alexander, K. and Easterbrook, S. M.: The Software Architecture of Global Climate Models, in: AGU Fall Meeting 2011, San Francisco, USA, Abstract ID 1204770, 2011.

Archer, D.: A data-driven model of the global calcite lysocline, Global Biogeochemical Cycles, 10, 511–526, doi:10.1029/96GB01521, 1996.

Bailey, D., Holland, M., Hunke, E., Lipscomb, B., Briegleb, B., Bitz, C., and Schramm, J.: Community Ice CodE (CICE) User's Guide Version 4.0, Tech. Rep., National Center for Atmospheric Research, available at: http://www.cesm.ucar.edu/models/cesm1.0/cice/ice_usrdoc.pdf (last access: 19 November 2014), 2010.

Collins, M.: Ensembles and probabilities: a new era in the prediction of climate change, Philos. T. R. Soc. A, 365, 1957–70, doi:10.1098/rsta.2007.2068, 2007.

Collins, W. J., Bellouin, N., Doutriaux-Boucher, M., Gedney, N., Halloran, P., Hinton, T., Hughes, J., Jones, C. D., Joshi, M., Liddicoat, S., Martin, G., O'Connor, F., Rae, J., Senior, C., Sitch, S., Totterdell, I., Wiltshire, A., and Woodward, S.: Development and evaluation of an Earth-System model - HadGEM2, Geoscientific Model Development, 4, 1051–1075, doi:10.5194/gmd-4-1051-2011, 2011.

Cook, K. H.: An introduction to the climate system 1, in: Climate Dynamics, Princeton University Press, 1–3, 2013.

Dickinson, R. E., Zebiak, S., Anderson, J., Blackmon, M., De Luca, C., Hogan, T., Iredell, M., Ji, M., Rood, R., Suarez, M., and Taylor, K. E.: How can we advance our weather and climate models as a community?, B. Am. Meteorol. Soc., 83, 431–434, doi:10.1175/1520-0477(2002)083<0431:HCWAOW>2.3.CO;2, 2002.

Drake, J. B.: Overview of the software design of the community climate system model, Int. J. High Perform. C., 19, 177–186, doi:10.1177/1094342005056094, 2005.

Dufresne, J., Foujols, M., Denvil, S., Caubel, A., Marti, O., Aumont, O., Balkanski, Y., Bekki, S., Bellenger, H., Benshila, R., Bony, S., Bopp, L., Braconnot, P., Brockmann, P., Cadule, P., Cheruy, F., Codron, F., Cozic, A., Cugnet, D., de Noblet, N., Duvel, J., Ethé, C., Fairhead, L., Fichefet, T., Flavoni, S., Friedlingstein, P., Grandpeix, J., Guez, L., Guilyardi, E., Hauglustaine, D., Hourdin, F., Idelkadi, A., Ghattas, J., Joussaume, S., Kageyama, M., Krinner, G., Labetoulle, S., Lahellec, A., Lefebvre, M., Lefevre, F., Levy, C., Li, Z. X., Lloyd, J., Lott, F., Madec, G., Mancip, M., Marchand, M., Masson, S., Meurdesoif, Y., Mignot, J., Musat, I., Parouty, S., Polcher, J., Rio, C., Schulz, M., Swingedouw, D., Szopa, S., Talandier, C., Terray, P., Viovy, N., and Vuichard, N.: Climate change projections using the IPSL-CM5 Earth System Model: from CMIP3 to CMIP5, Climate Dynamics, 40, 2123–2165, doi:10.1007/s00382-012-1636-1, 2013.

Dunne, J. P., John, J. G., Adcroft, A. J., Griffies, S. M., Hallberg, R. W., Shevliakova, E., Stouffer, R. J., Cooke, W., Dunne, K. A., Harrison, M. J., Krasting, J. P., Malyshev, S. L., Milly, P. C. D., Phillipps, P. J., Sentman, L. T., Samuels, B. L., Spelman, M. J., Winton, M., Wittenberg, A. T., and Zadeh, N.: GFDL's ESM2 Global Coupled Climate-Carbon Earth System Models. Part I: Physical Formulation and Baseline Simulation Characteristics, Journal of Climate, 25, 6646–6665, doi:10.1175/JCLI-D-11-00560.1, 2012.

Dunne, J. P., John, J. G., Shevliakova, E., Stouffer, R. J., Krasting, J. P., Malyshev, S. L., Milly, P. C. D., Sentman, L. T., Adcroft, A. J., Cooke, W., Dunne, K. A., Griffies, S. M., Hallberg, R. W., Harrison, M. J., Levy, H., Wittenberg, A. T., Phillips, P. J., and Zadeh, N.: GFDL's ESM2 Global Coupled Climate-Carbon Earth System Models. Part II: Carbon System Formulation and Baseline Simulation Characteristics, Journal of Climate, 26, 2247–2267, doi:10.1175/JCLI-D-12-00150.1, 2013.

Easterbrook, S. M. and Johns, T. C.: Engineering the software for understanding climate change, Comput. Sci. Eng., 11, 65–74, doi:10.1109/MCSE.2009.193, 2009.

Eby, M., Weaver, A. J., Alexander, K., Zickfeld, K., Abe-Ouchi, A., Cimatoribus, A. A., Crespin, E., Drijfhout, S. S., Edwards, N. R., Eliseev, A. V., Feulner, G., Fichefet, T., Forest, C. E., Goosse, H., Holden, P. B., Joos, F., Kawamiya, M., Kicklighter, D., Kienert, H., Matsumoto, K., Mokhov, I. I., Monier, E., Olsen, S. M., Pedersen, J. O. P., Perrette, M., Philippon-Berthier, G., Ridgwell, A., Schlosser, A., Schneider von Deimling, T., Shaffer, G., Smith, R. S., Spahni, R., Sokolov, A. P., Steinacher, M., Tachiiri, K., Tokos, K., Yoshimori, M., Zeng, N., and Zhao, F.: Historical and idealized climate model experiments: an intercomparison of Earth system models of intermediate complexity, Clim. Past, 9, 1111–1140, doi:10.5194/cp-9-1111-2013, 2013.

Friedlingstein, P., Cox, P., Betts, R., Bopp, L., von Bloh, W., Brovkin, V., Cadule, P., Doney, S., Eby, M., Fung, I., Bala, G., John, J., Jones, C., Joos, F., Kato, T., Kawamiya, M., Knorr, W., Lindsay, K., Matthews, H. D., Raddatz, T., Rayner, P., Reick, C., Roeckner, E., Schnitzler, K.-G., Schnur, R., Strassmann, K., Weaver, A. J., Yoshikawa, C., and Zeng, N.: climate–carbon cycle feedback analysis: results from the C^4MIP model intercomparison, J. Climate, 19, 3337–3353, doi:10.1175/JCLI3800.1, 2006.

Giorgetta, M. A., Jungclaus, J., Reick, C., Legutke, S., Bader, J., Böttinger, M., Brovkin, V., Crueger, T., Esch, M., Fieg, K., Glushak, K., Gayler, V., Haak, H., Hollweg, H., Ilyina, T., Kinne, S., Kornblueh, L., Matei, D., Mauritsen, T., Mikolajewicz, U., Mueller, W., Notz, D., Pithan, F., Raddatz, T., Rast, S., Redler, R., Roeckner, E., Schmidt, H., Schnur, R., Segschneider, J., Six, K. D., Stockhause, M., Timmreck, C., Wegner, J., Widmann, H., Weiners, K., Claussen, M., Marotzke, J., and Stevens, B.: Climate and carbon cycle changes from 1850 to 2100 in MPI-ESM simulations for the Coupled Model Intercomparison Project phase 5, Journal of Advances in Modeling Earth Systems, 5, 572–597, doi:10.1002/jame.20038, 2013.

Goosse, H., Brovkin, V., Fichefet, T., Haarsma, R., Huybrechts, P., Jongma, J., Mouchet, A., Selten, F., Barriat, P., Campin, J., Deleersnijder, E., Driesschaert, E., Goelzer, H., Janssens, I., Loutre, M., Morales Maqueda, M. A., Opsteegh, T., Mathieu, P., Munhoven, G., Pettersson, E. J., Renssen, H., Roche, D. M., Schaeffer, M., Tartinville, B., Timmermann, A., and Weber, S. L.: Description of the Earth system model of intermediate complexity LOVECLIM version 1.2, Geoscientific Model Development, 3, 603–633, doi:10.5194/gmd-3-603-2010, 2010.

Herraiz, I., Gonzalez-Barahona, J. M., and Robles, G.: Towards a theoretical model for software growth, in: Proceedings of the Fourth International Workshop on Mining Software Repositories, MSR '07, IEEE Computer Society, Washington, DC, USA, doi:10.1109/MSR.2007.31, 2007.

Hewitt, H. T., Copsey, D., Culverwell, I. D., Harris, C. M., Hill, R. S. R., Keen, A. B., McLaren, A. J., and Hunke, E. C.: Design and implementation of the infrastructure of HadGEM3: the next-generation Met Office climate modelling system, Geosci. Model Dev., 4, 223–253, doi:10.5194/gmd-4-223-2011, 2011.

Hurrell, J. W., Holland, M. M., Gent, P. R., Ghan, S., Kay, J. E., Kushner, P. J., Lamarque, J. F., Large, G., Lawrence, D., Lindsay, K., Lipscomb, W. H., Long, M. C., Mahowald, N., Marsh, D. R., Neale, R. B., Rasch, P., Vavrus, S., Vertenstein, M., Bader, D., Collins, W. D., Hack, J. J., Kiehl, J., and Marshall, S.: The Community Earth System Model: A Framework for Collaborative

Research, Bulletin of the American Meteorological Society, 94, 1339–1360, doi:10.1175/BAMS-D-12-00121.1, 2013.

Knutti, R.: Should we believe model predictions of future climate change?, Philos. T. R. Soc. A, 366, 4647–64, doi:10.1098/rsta.2008.0169, 2008.

Knutti, R., Masson, D., and Gettelman, A.: Climate model genealogy: Generation CMIP5 and how we got there, Geophysical Research Letters, 40, 1194–1199, doi:10.1002/grl.50256, 2013.

Lenhard, J. and Winsberg, E.: Holism, entrenchment, and the future of climate model pluralism, Stud. Hist. Philos. M. P., 41, 253–262, doi:10.1016/j.shpsb.2010.07.001, 2010.

Lindsay, K., Bonan, G. B., Doney, S. C., Hoffman, F. M., Lawrence, D. M., Long, M. C., Mahowald, N. M., Moore, J. K., Randerson, J. T., and Thornton, P. E.: Preindustrial-Control and Twentieth-Century Carbon Cycle Experiments with the Earth System Model CESM1(BGC), Journal of Climate, 27, 8981–9005, doi:10.1175/JCLI-D-12-00565.1, 2014.

Madec, G.: NEMO Ocean Engine, Tech. rep., Laboratoire d'Oceanographie et du Climat: Experimentation et Approches Numeriques, 2008.

Masson, D. and Knutti, R.: Climate model genealogy, Geophys. Res. Lett., 38, 1–4, doi:10.1029/2011GL046864, 2011.

Meissner, K. J., Weaver, A. J., Matthews, H. D., and Cox, P. M.: The role of land surface dynamics in glacial inception: a study with the UVic Earth System Model, Clim. Dynam., 21, 515–537, doi:10.1007/s00382-003-0352-2, 2003.

NASA Advisory Council: Earth System Science Overview: A Program for Global Change, National Aeronautics and Space Administration, 1986.

Park, R.: Software Size Measurement: a Framework for Counting Source Statements, Tech. Rep. CMU/SEI-92-TR-020, Software Engineering Institute, Carnegie Mellon University, available at: http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=11689 (last access: 19 November 2014), 1992.

Randall, D. A.: Should climate models be open source?, IEEE Software, 28, 62–65, doi:10.1109/MS.2011.144, 2011.

Reichler, T. and Kim, J.: How well do coupled models simulate today's climate?, B. Am. Meteorol. Soc., 89, 303–311, doi:10.1175/BAMS-89-3-303, 2008.

Rosenfeld, D. and Wood, R.: Aerosol cloud-mediated radiative forcing: highly uncertain and opposite effects from shallow and deep clouds, in: Climate Science for Serving Society, edited by: Asrar, G. R., and Hurrell, J. W., Springer Netherlands, Dordrecht, doi:10.1007/978-94-007-6692-1, 2013.

Schmidt, G. A., Kelley, M., L., N., Ruedy, R., Russell, G. L., Aleinov, I., Bauer, M., Bauer, S. E., Bhat, M. K., R., B., Canuto, V., Chen, Y., Cheng, Y., Clune, T. L., Del Genio, A., de Fainchtein, R., Faluvegi, G., Hansen, J. E., Healy, R. J., Kiang, N. Y., Koch, D., Lacis, A. A., LeGrande, A. N., Lerner, J., Lo, K. K., Matthews, E. E., Menon, S., Miller, R. L., Oinas, V., Oloso, A. O., Perlwitz, J. P., Puma, M. J., Putman, W. M., Rind, D., Romanou, A., Sato, M., Shindell, D. T., Sun, S., Syed, R. A., Tausnev, N., Tsigaridis, K., Unger, N., Voulgarakis, A., Yao, M., and Zhang, J.: Configuration and assessment of the GISS ModelE2 contributions to the CMIP5 archive, Journal of Advances in Modeling Earth Systems, 6, 141–148, doi:10.1002/2013MS000265, 2014.

Schmittner, A., Oschlies, A., Matthews, H. D., and Galbraith, E. D.: Future changes in climate, ocean circulation, ecosystems, and biogeochemical cycling simulated for a business-as-usual CO_2 emission scenario until year 4000 AD, Global Biogeochemical Cycles, 22, GB1013, doi:10.1029/2007GB002953, 2008.

Shaw, M. and Garlan, D.: Software architecture: perspectives on an emerging discipline, Prentice Hall, 1996.

Simon, H. A.: The Sciences of the Artificial, MIT Press, 1996.

Smith, R., Jones, P., Briegleb, B., Bryan, F., Danabasoglu, G., Dennis, J., Dukowicz, J., Eden, C., Fox-Kemper, B., Gent, P., Hecht, M., Jayne, S., Jochum, M., Large, W., Lindsay, K., Maltrud, M., Norton, N., Peacock, S., Vertenstein, M., and Yeager, S.: The Parallel Ocean Program (POP) Reference Manual: Ocean Component of the Community Climate System Model (CCSM) and Community Earth System Model (CESM), Tech. rep., Los Alamos National Laboratory, Report Number LAUR-10-01853, available at: http://www.cesm.ucar.edu/models/ccsm4.0/pop/doc/sci/POPRefManual.pdf (last access: 19 November 2014), 2010.

Taylor, K. E., Stouffer, R. J., and Meehl, G. A.: An overview of CMIP5 and the experiment design, B. Am. Meteorol. Soc., 93, 485–498, doi:10.1175/BAMS-D-11-00094.1, 2012.

Valcke, S.: The OASIS3 coupler: a European climate modelling community software, Geosci. Model Dev., 6, 373–388, doi: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, Geosci. Model Dev., 5, 1589–1596, doi:10.5194/gmd-5-1589-2012, 2012.

Vancoppenolle, M., Fichefet, T., Goosse, H., Bouillon, S., Beatty, C. K., and Maqueda, M. A. M.: LIM3, an advanced sea-ice model for climate simulation and operational oceanography, Mercator Quarterly Newsletter, 28, 16–21, 2008.

Weaver, A. J., Eby, M., Wiebe, E. C., Bitz, C. M., Duffy, P. B., Ewen, T. L., Fanning, A. F., Holland, M. M., MacFadyen, A., Matthews, H. D., Meissner, K. J., Saenko, O., Schmittner, A., Wang, H., and Yoshimori, M.: The UVic earth system climate model: model description, climatology, and applications to past, present and future climates, Atmos. Ocean, 39, 361–428, doi:10.1080/07055900.2001.9649686, 2001.

World Climate Research Program: WCRP Working Group on Coupled Modeling Catalogue of Model Intercomparison Projects (MIPs), available at: http://www.wcrp-climate.org/wgcm/projects.shtml (last access: 19 November 2014), 2014.

Zickfeld, K., Eby, M., Weaver, A. J., Alexander, K., Crespin, E., Edwards, N. R., Eliseev, A. V., Feulner, G., Fichefet, T., Forest, C. E., Friedlingstein, P., Goosse, H., Holden, P. B., Joos, F., Kawamiya, M., Kicklighter, D., Kienert, H., Matsumoto, K., Mokhov, I. I., Monier, E., Olsen, S. M., Pedersen, J. O. P., Perrette, M., Philippon-Berthier, G., Ridgwell, A., Schlosser, A., Schneider Von Deimling, T., Shaffer, G., Sokolov, A., Spahni, R., Steinacher, M., Tachiiri, K., Tokos, K. S., Yoshimori, M., Zeng, N., and Zhao, F.: Long-term climate change commitment and reversibility: an EMIC intercomparison, J. Climate, 26, 5782–5809, doi:10.1175/JCLI-D-12-00584.1, 2013.

Table 1. Information about each model including its full configuration name and an abbreviation for use in the text, level of complexity (GCM or EMIC), ~~and~~ the name and location of its host institution, and the relevant publication(s) describing the given configuration.

Model Name (Abbreviation)	Complexity	Institution	Country	Publication(s)
CESM1-BGC (CESM)	GCM	National Centre for Atmospheric Research	USA	Lindsay et al. (2014)
GFDL-ESM2M (GFDL)	GCM	Geophysical Fluid Dynamics Laboratory	USA	Dunne et al. (2012) , Dunne et al. (2013)
GISS-E2-R-TCADI (GISS)	GCM	NASA Goddard Institute for Space Studies	USA	Schmidt et al. (2014)
UVic ESCM 2.9 (UVic)	EMIC	University of Victoria	Canada	Weaver et al. (2001) , Meissner et al. (2003) , Schmittner et al. (2008) , Archer (1996)
HadGEM2-ES (HadGEM)	GCM	Hadley Centre for Climate Prediction and Research	UK	Collins et al. (2011)
IPSL-CM5A-LR (IPSL)	GCM	Institut Pierre Simon Laplace	France	Dufresne et al. (2013)
MPI-ESM-LR (MPI)	GCM	Max Planck Institut für Meteorologie	Germany	Giorgetta et al. (2013)
Loveclim 1.3 (Loveclim)	EMIC	Université catholique de Louvain	Belgium	Goosse et al. (2010)

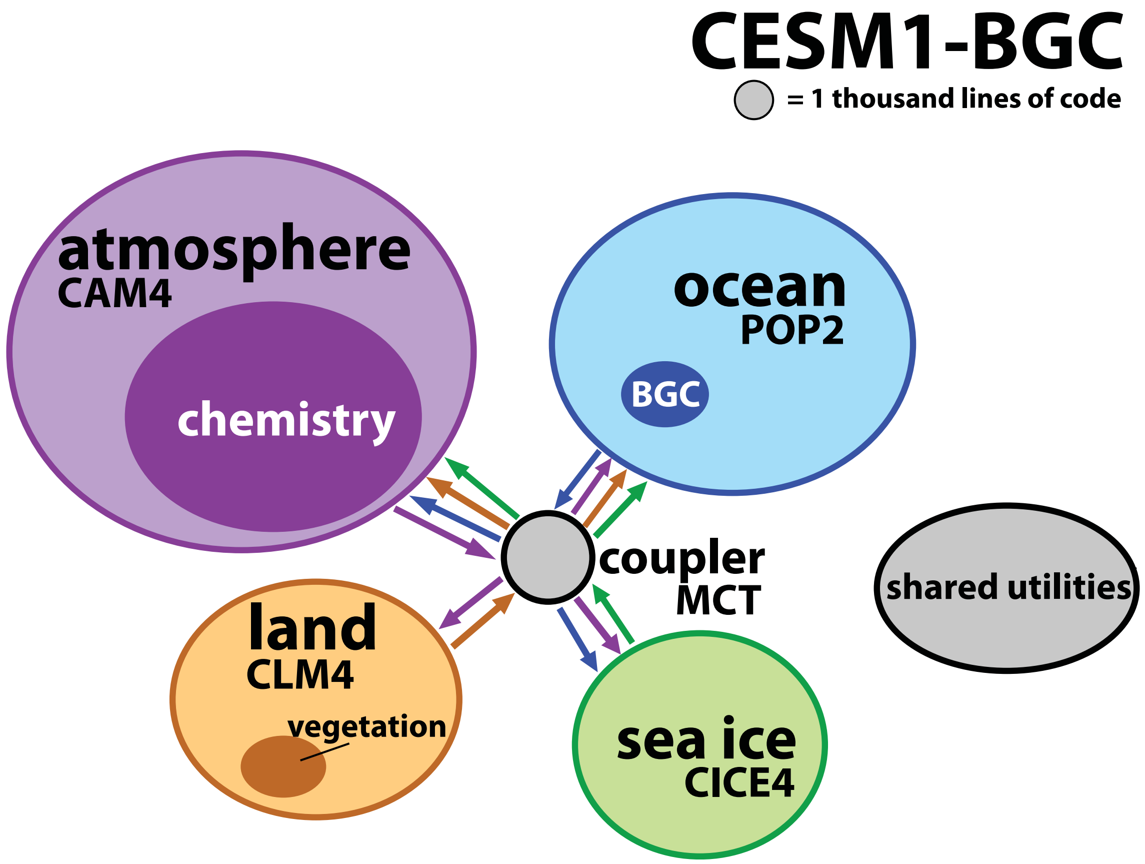

Figure 1. Architecture diagram for CESM1-BGC.

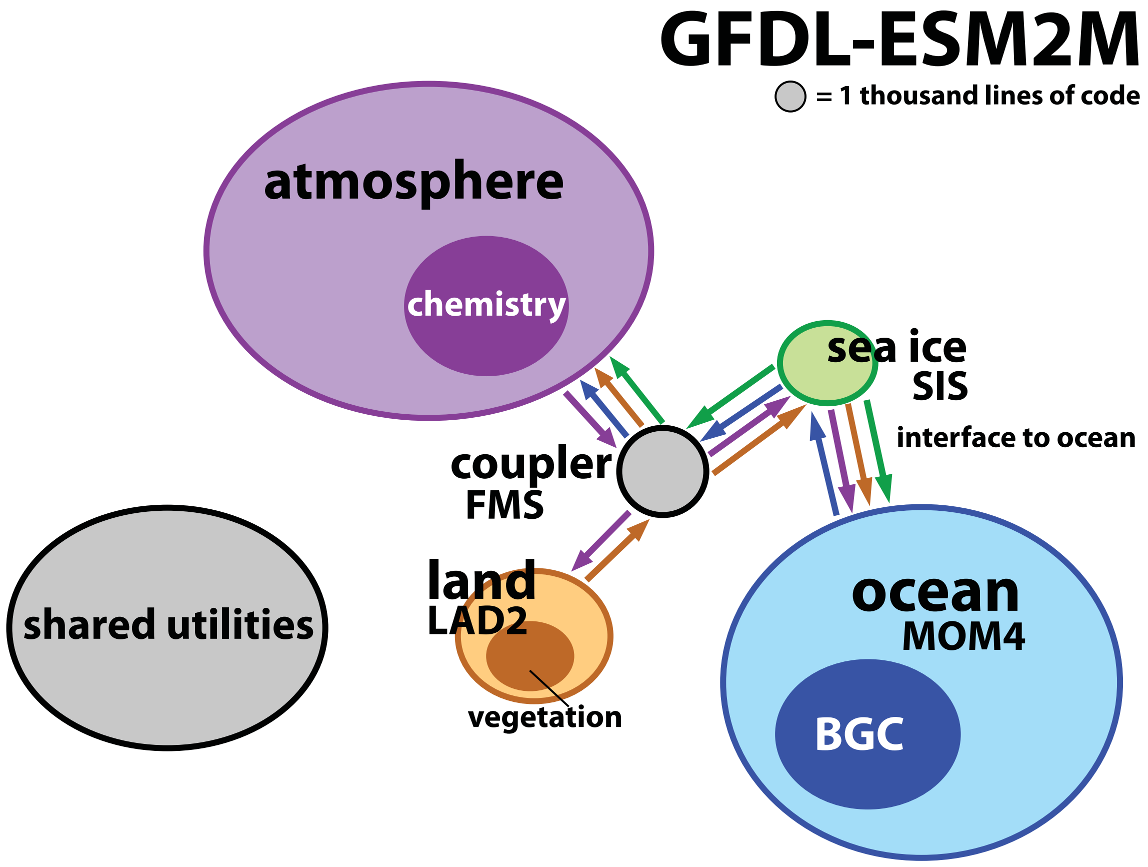

Figure 2. Architecture diagram for GFDL-ESM2M.

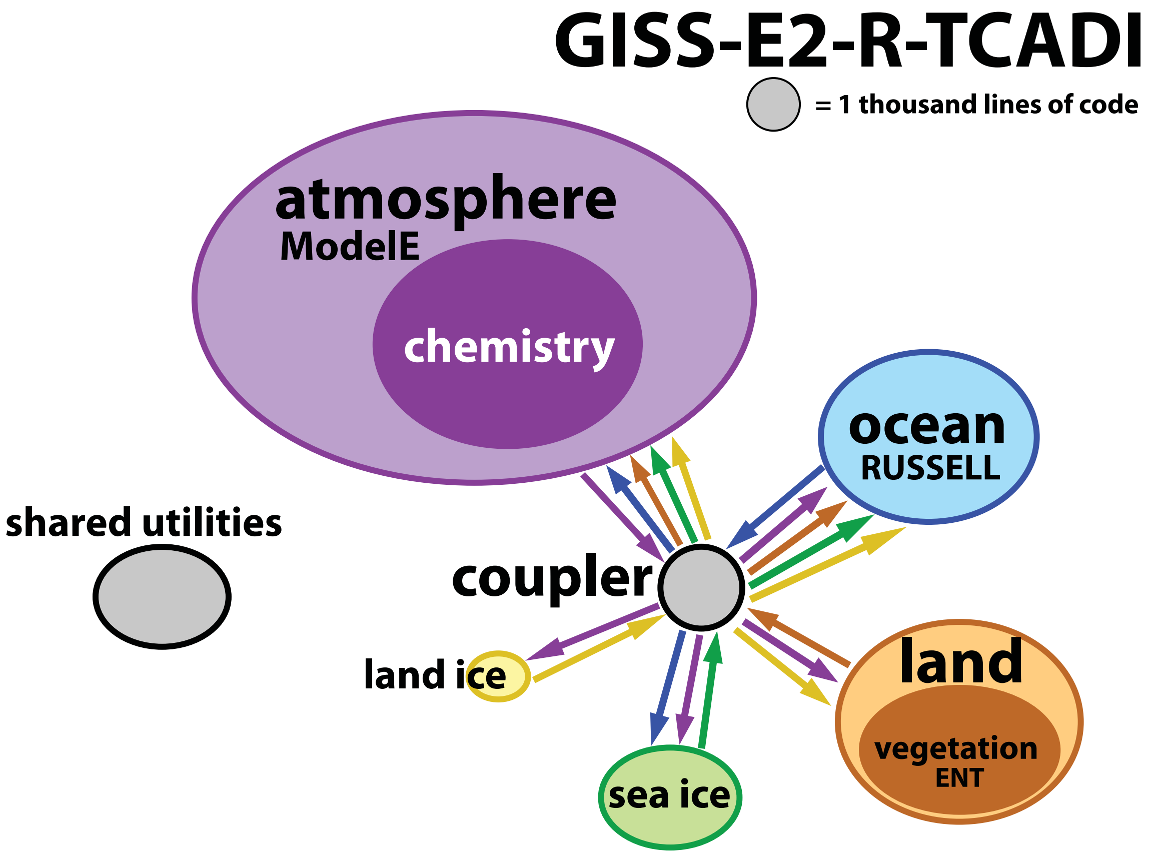

Figure 3. Architecture diagram for GISS-E2-R-TCADI.

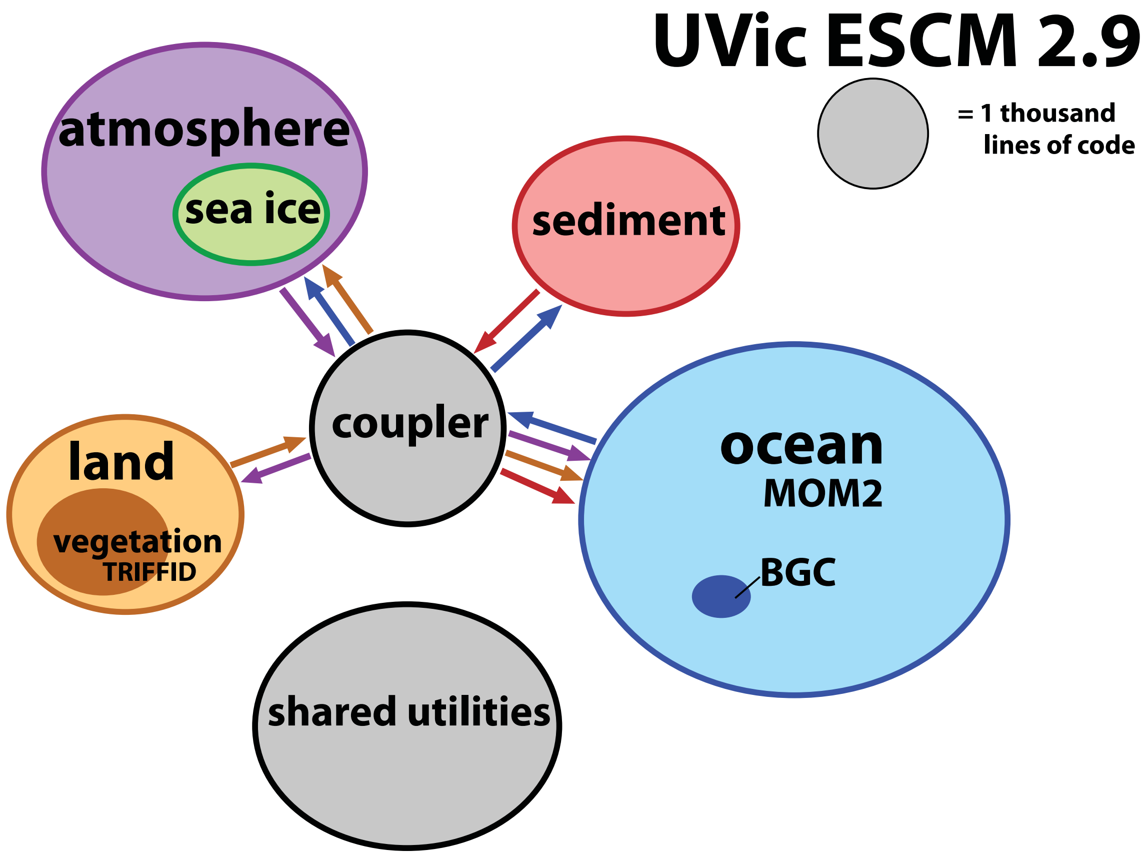

Figure 4. Architecture diagram for UVic ESCM 2.9.

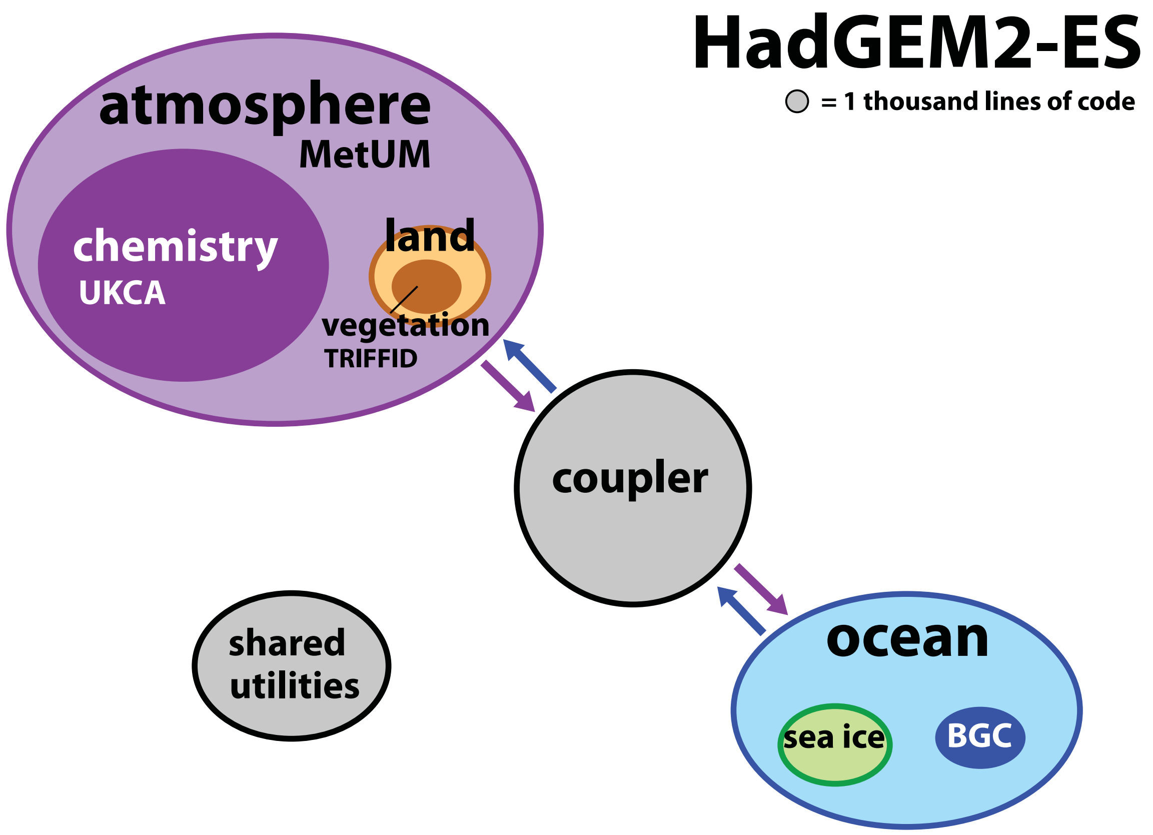

Figure 5. Architecture diagram for HadGEM2-ES.

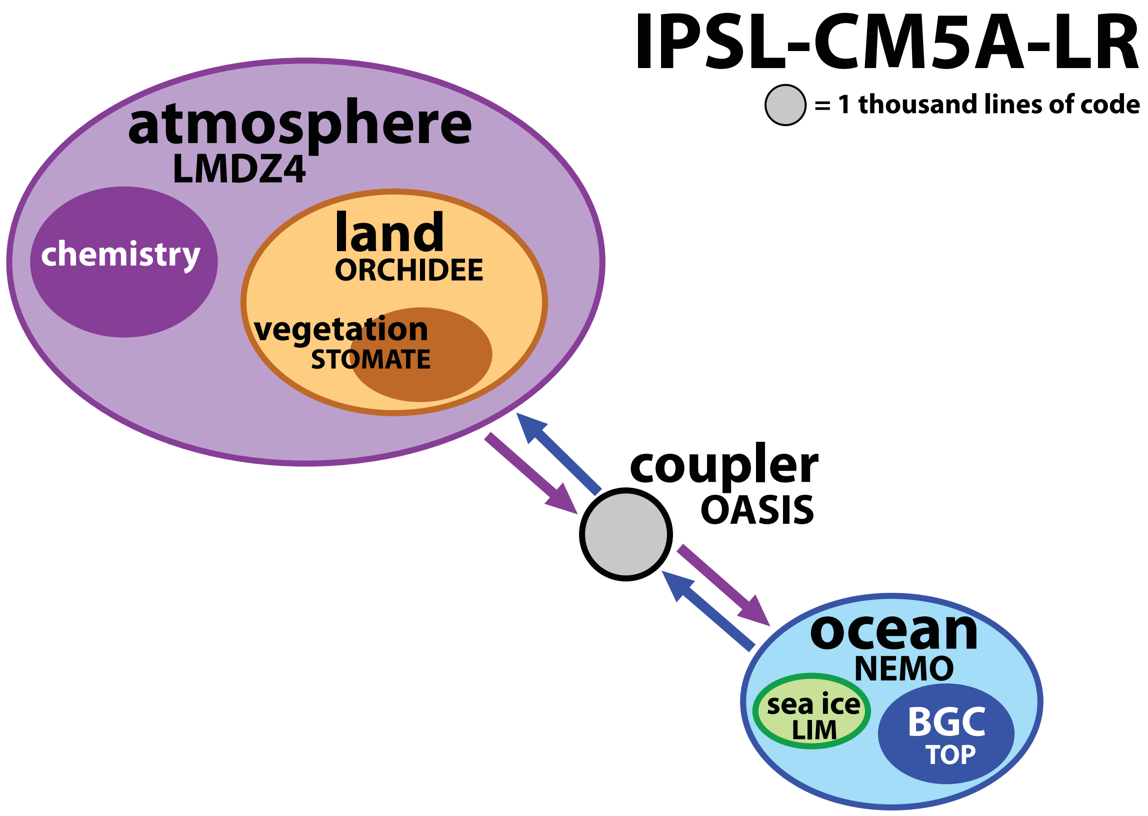

Figure 6. Architecture diagram for IPSL-CM5A-LR.

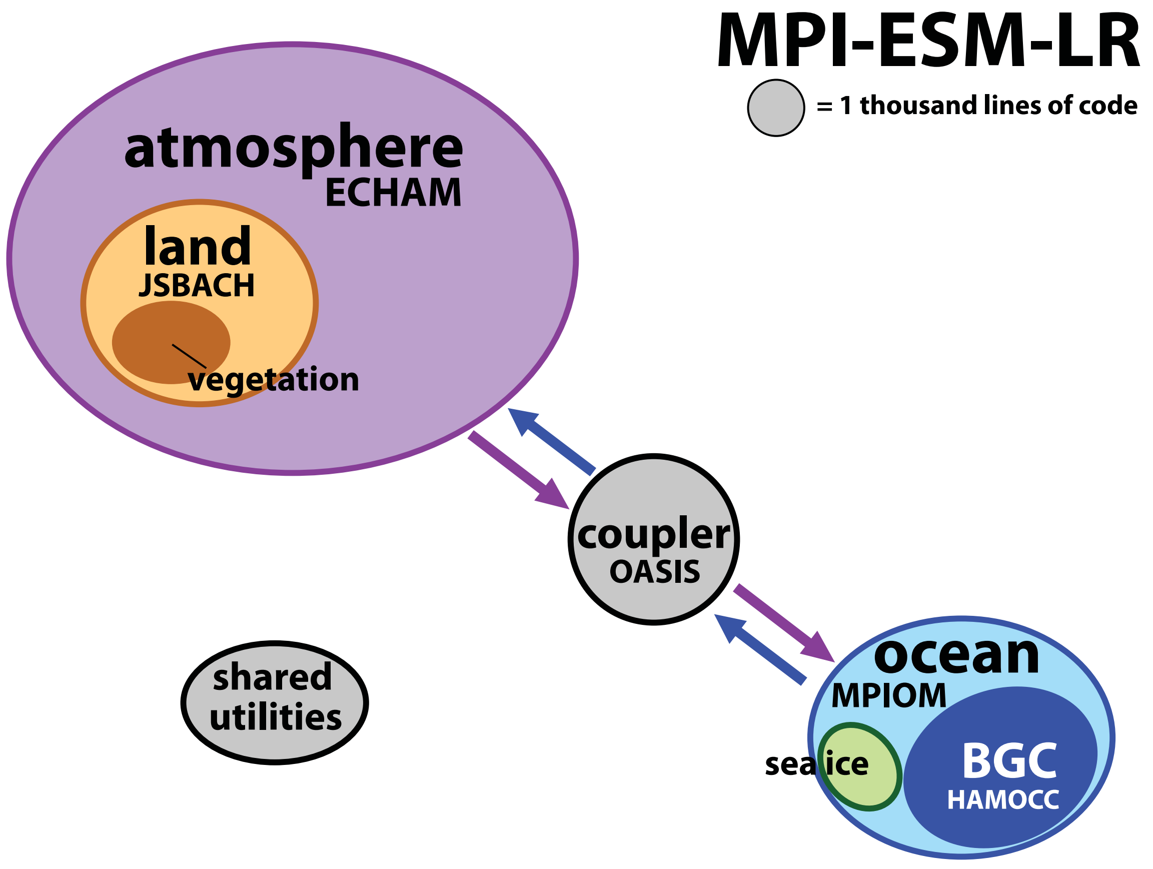

Figure 7. Architecture diagram for MPI-ESM-LR.

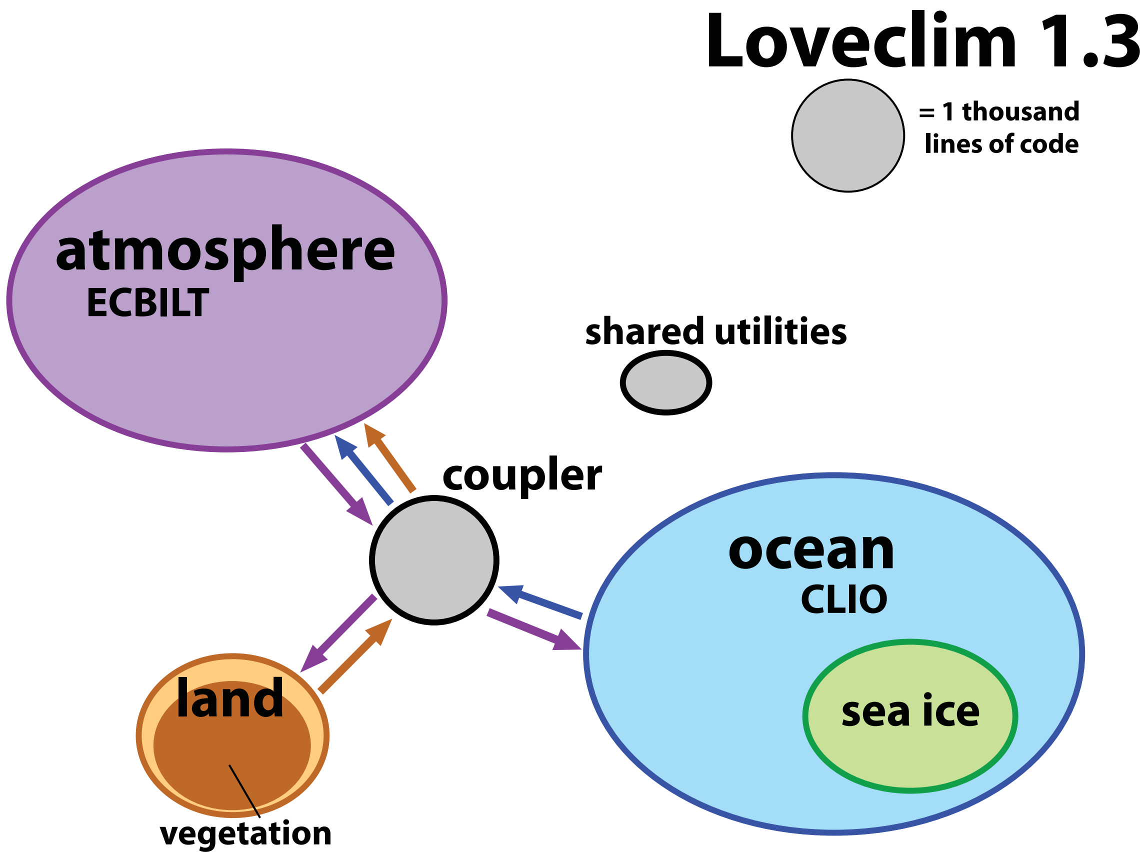

Figure 8. Architecture diagram for Loveclim 1.3.

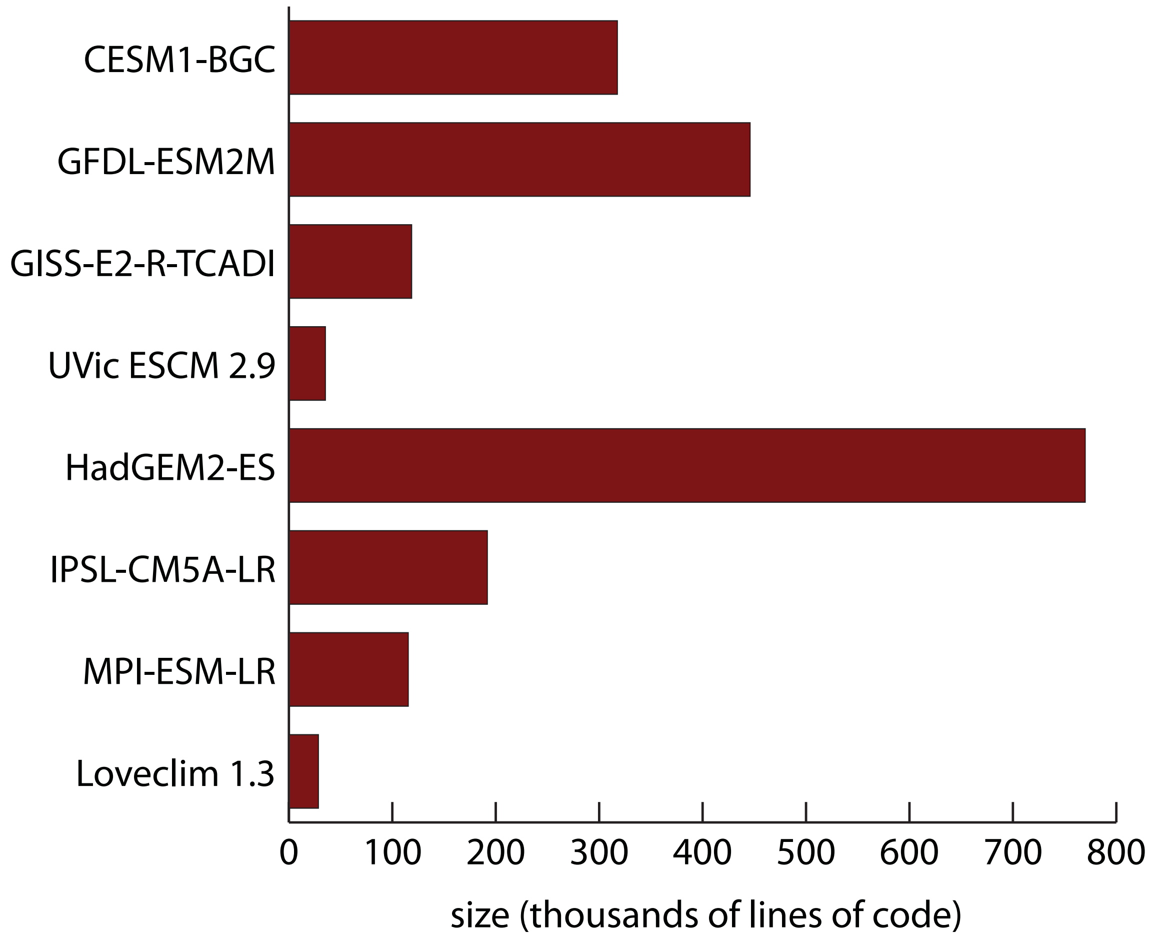

Figure 9. Line count of the source code of each model, excluding comments and blank lines. Generated using David A. Wheeler's "SLOCCount".