The Software Architecture of Global Climate Models

Kaitlin Alexander[1,2], Steve Easterbrook[2]

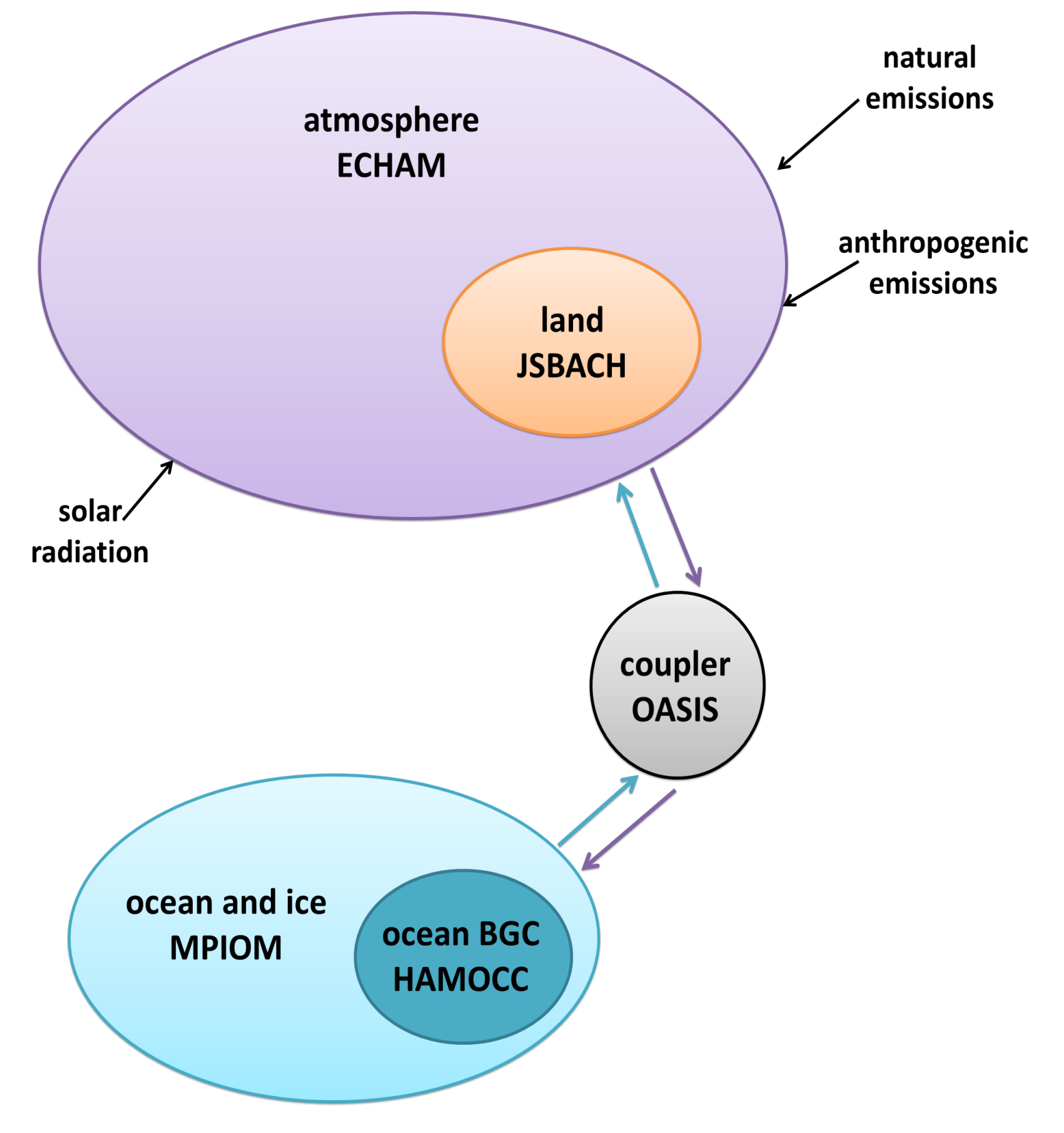
umalexak@cc.umanitoba.ca sme@cs.toronto.edu

[1]Department of Mathematics, University of Manitoba

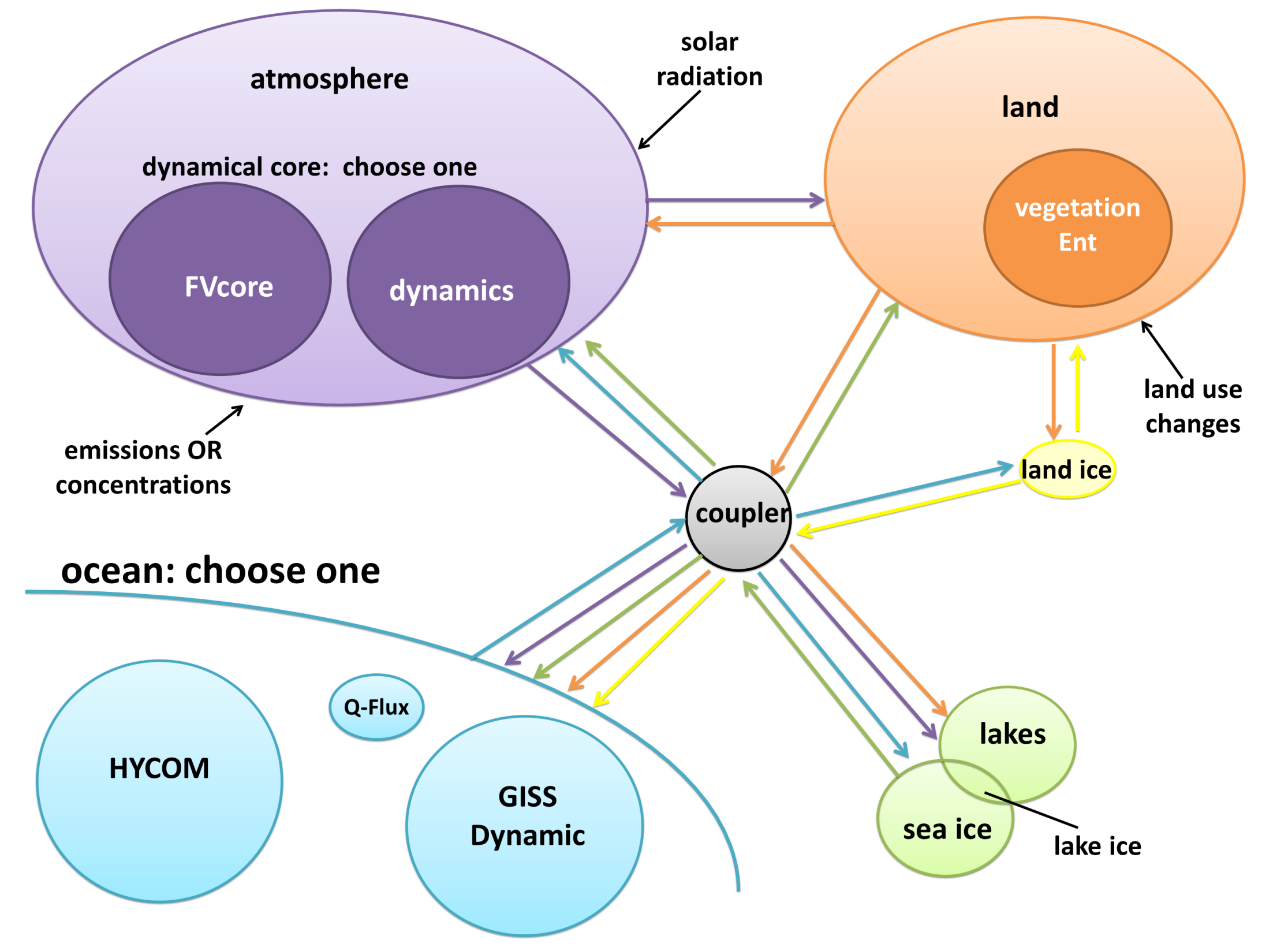
[2]Software Engineering Lab, Department of Computer Science, University of Toronto

IN41A-1394

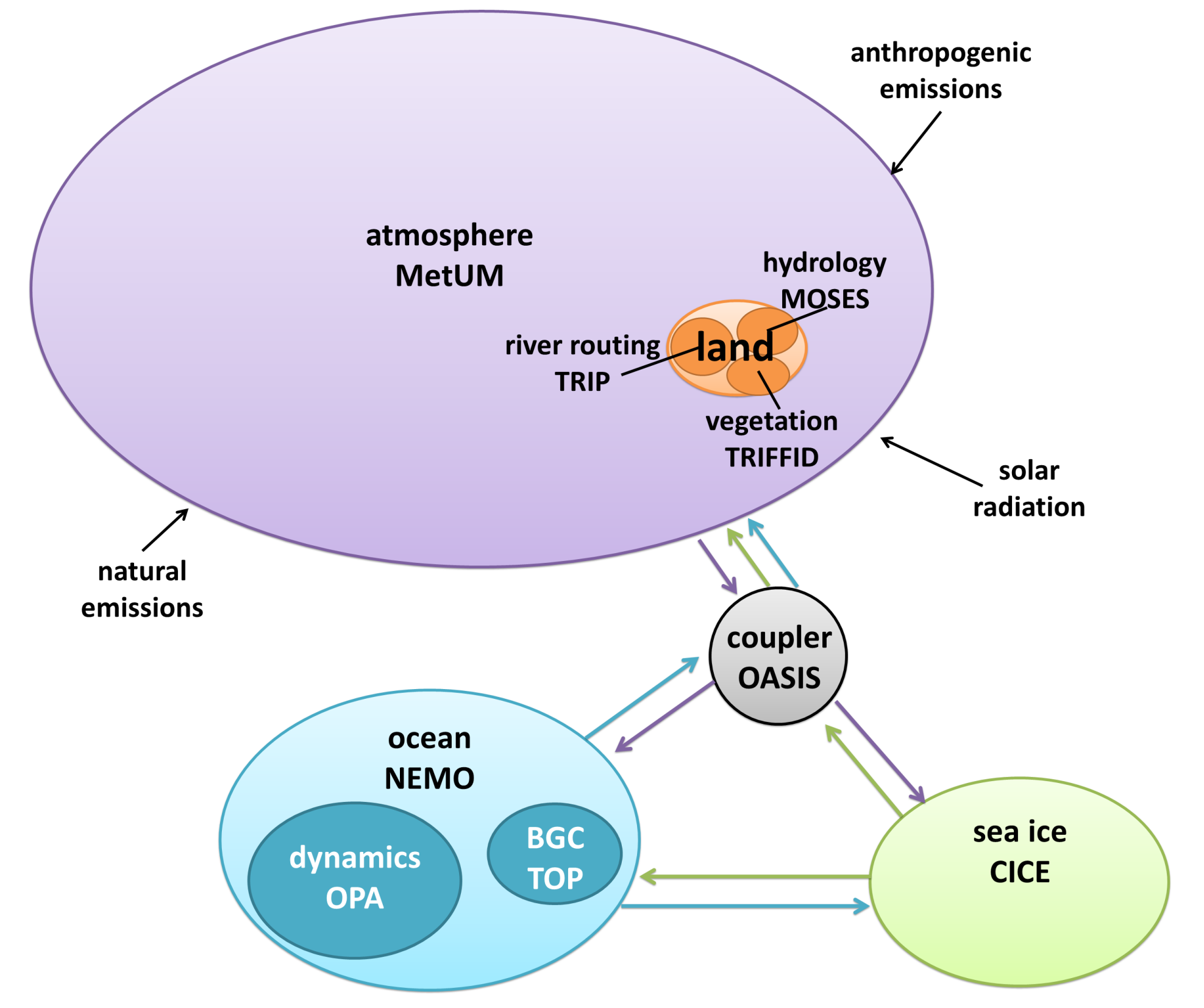
COSMOS 1.2.1
Max-Planck-Institut für Meteorologie, Germany



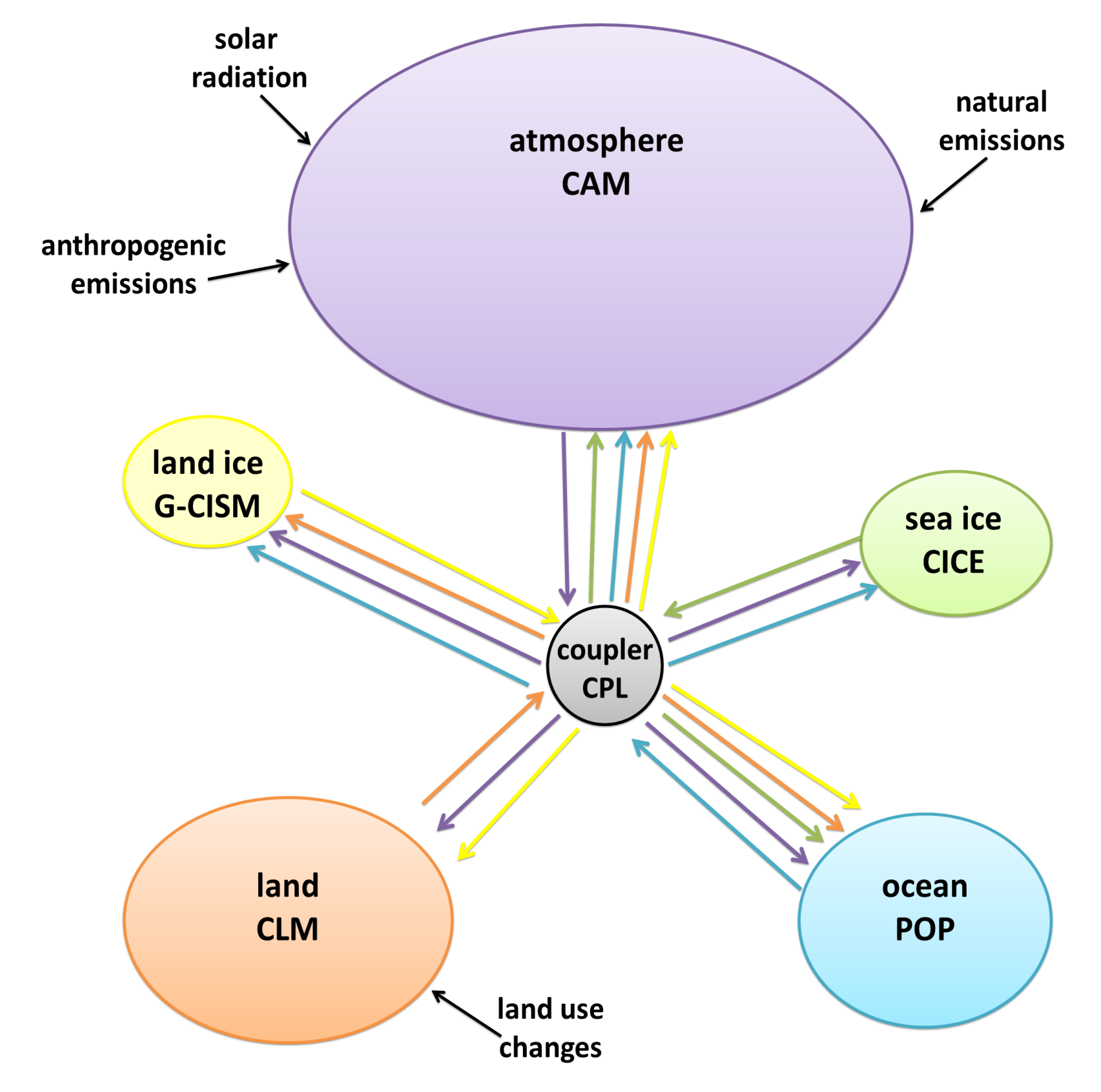
Model E October 11, 2011 revision
NASA Goddard Institute for Space Studies, USA



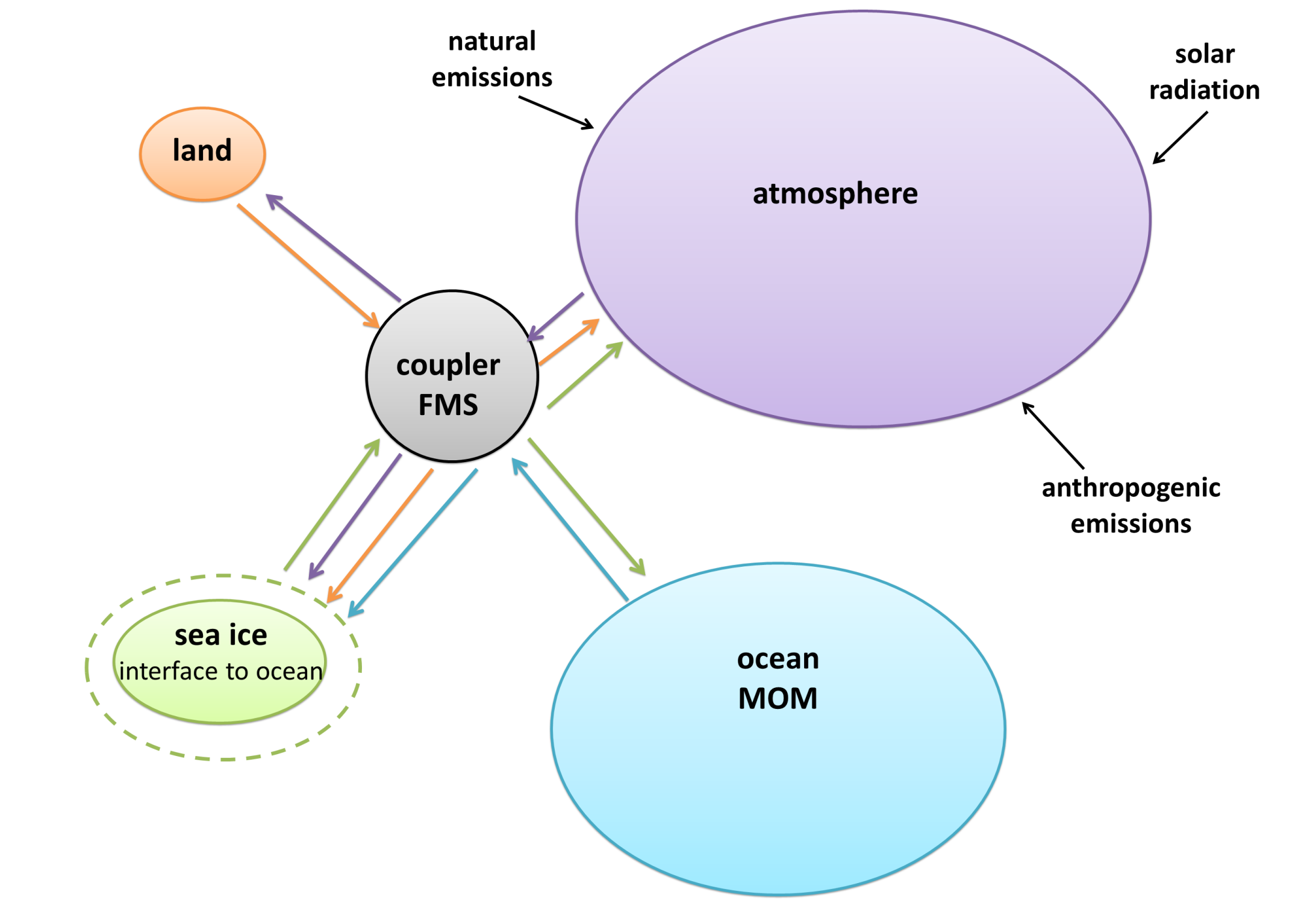
HadGEM3
Met Office, UK



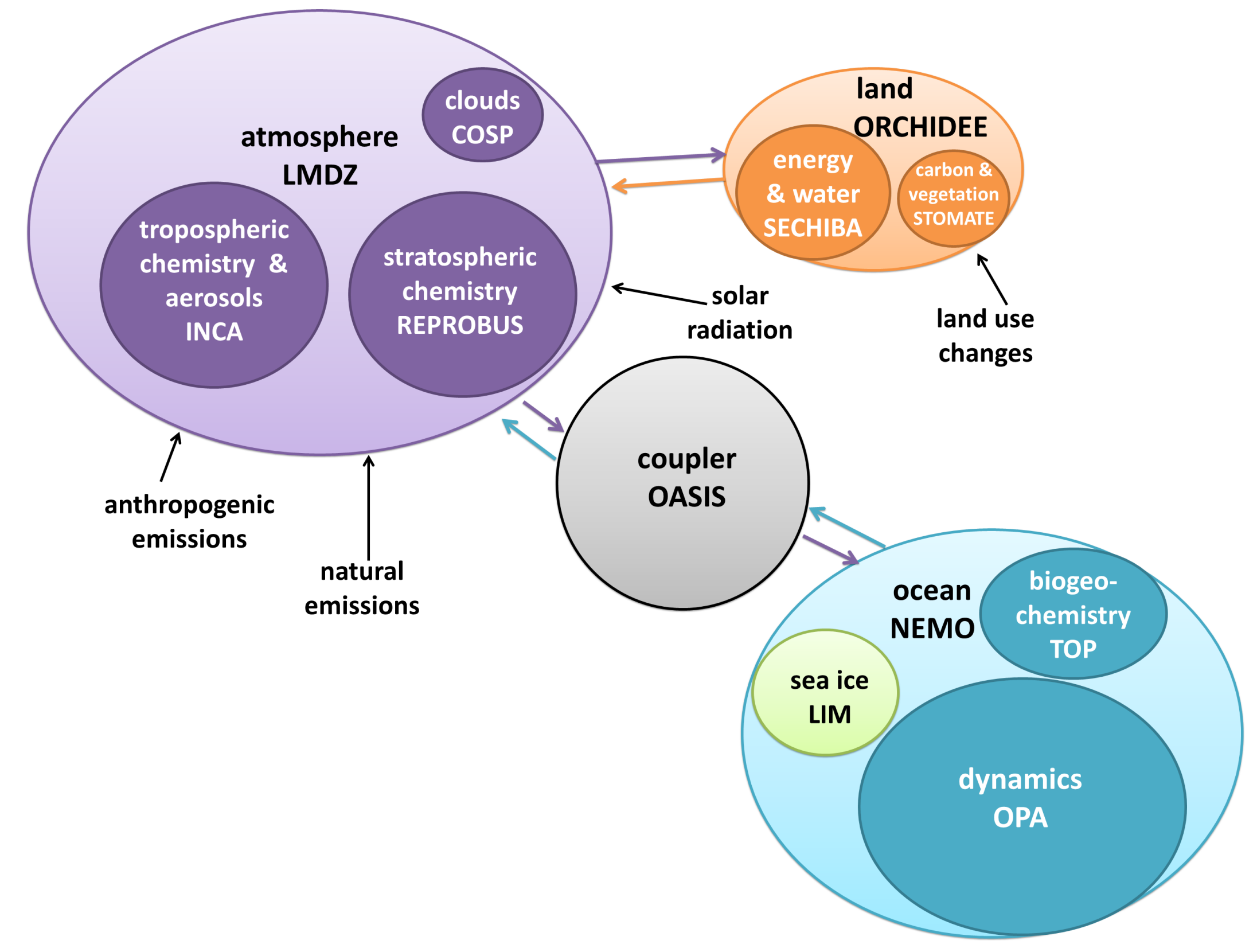
CESM 1.0.3
National Center for Atmospheric Research, USA



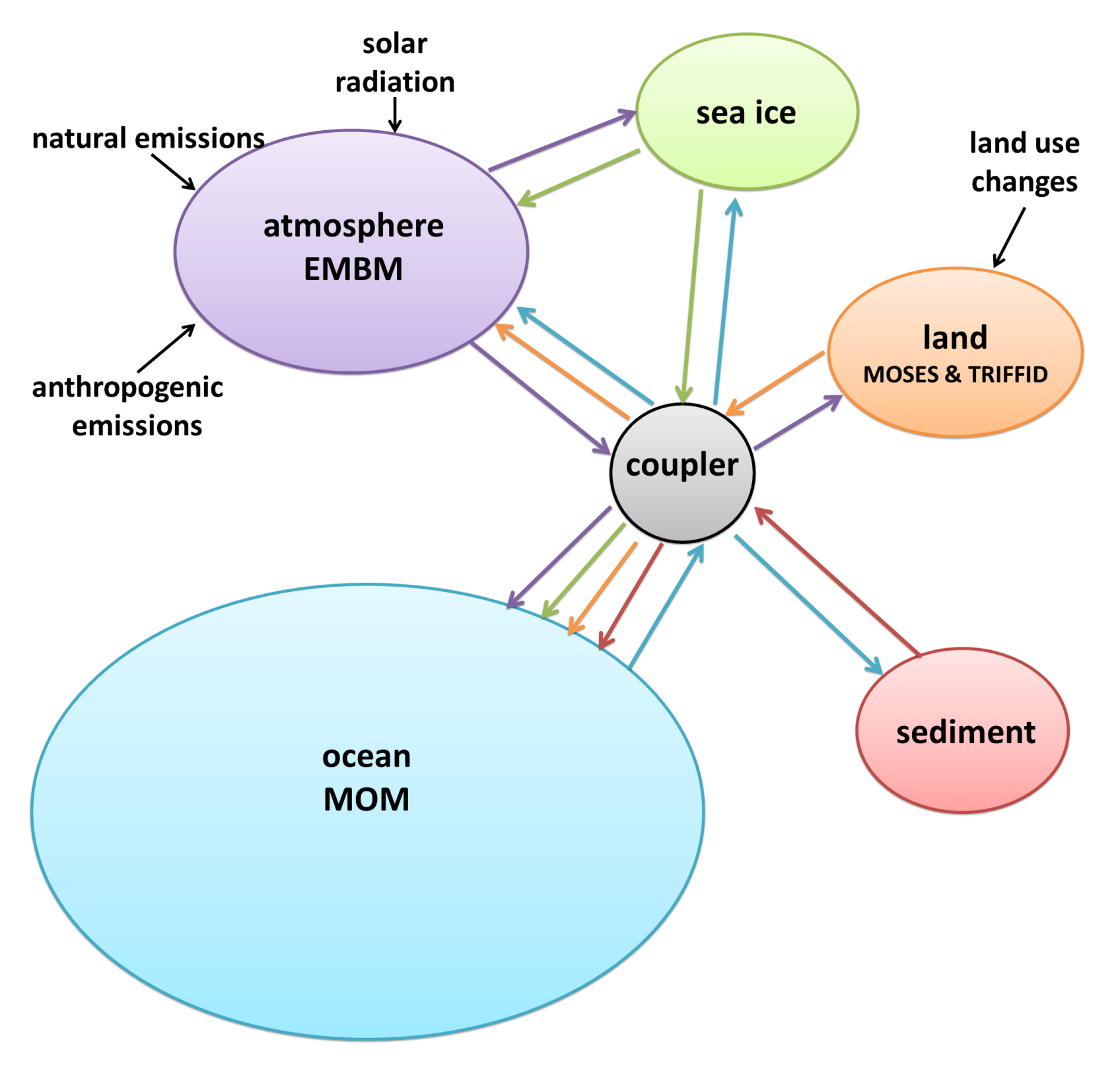
GFDL Climate Model 2.1 (coupled to MOM 4.1)
Geophysical Fluid Dynamics Laboratory, USA



IPSL Climate Model 5A
Institut Pierre Simon Laplace, France



UVic Earth System Climate Model 2.9
University of Victoria, Canada



Key to Diagrams

Each component of the climate system has been assigned a colour:
atmosphere ocean land sea ice land ice sediment

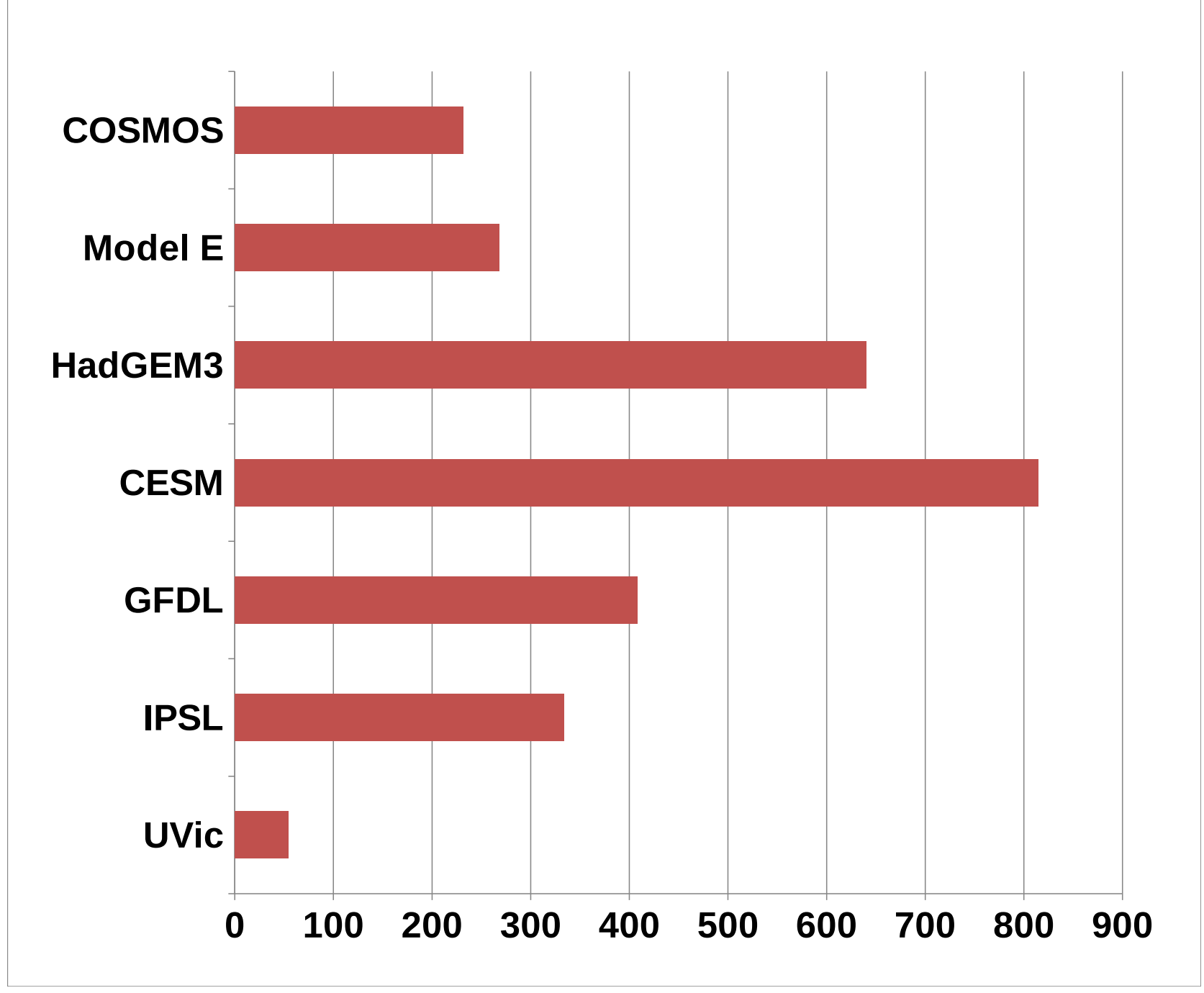
Model code for a component is represented with a bubble. ● Fluxes are represented with arrows, in a colour showing where they originated.

Couplers are grey. ● Components can pass fluxes either directly to each other or through the coupler.

The area of a bubble represents the size of its code base, relative to other components in the same model.

A smaller bubble within a larger one ◉ represents a small, highly encapsulated model of a system (eg clouds) that is used by the component.

Radiative forcings are passed to components with plain arrows.



Generated using David A. Wheeler's "SLOCCount".

Introduction

It has become common to compare and contrast the output of multiple global climate models (GCMs), such as in the Climate Model Intercomparison Project Phase 5 (CMIP5). However, intercomparisons of the software architecture of GCMs are almost nonexistent. In this qualitative study of seven GCMs from Canada, the United States and Europe, we attempted to fill this gap in research. By examining the model source code, reading documentation, and interviewing developers, we created diagrams of software structure and compared metrics such as encapsulation, coupler design, and complexity.

Component-Based Software Engineering

A global climate model is really a *collection* of models (components), each representing a major realm of the climate system, such as the atmosphere or the land surface. They are highly encapsulated, for stand-alone use as well as a mix-and-match approach that facilitates code sharing between institutions.

This strategy, known as component-based software engineering (CBSE), pools resources to create high-quality components that are used by many GCMs. For example,

- **UVic** uses a modified version of GFDL's ocean model, MOM.
- **HadGEM3** and **CESM** both use CICE, a sea ice model developed a third institution (Los Alamos).

Contrary to CBSE goals, there is no universal interface for climate models, so components need to be modified when they are passed between institutions. Furthermore, the right to edit the *master* copy of a component's source code is generally restricted to the development team at the hosting institution. As a result, many different branches of the software develop.

A drawback to CBSE is the fact that, in the real world, components of the climate system are not encapsulated. For example, how does one represent the relationship between sea ice and the ocean? Many different strategies exist:

- **CESM**: sea ice and ocean are completely separate components.
- **IPSL**: sea ice is a sub-component of the ocean.
- **GFDL**: sea ice is an interface to the ocean. All fluxes to and from the ocean must pass through the sea ice region, even if no ice is actually present.

Acknowledgements

Gavin Schmidt (NASA GISS); Tim Johns (Met Office); Gary Strand (NCAR); Arnaud Caubel, Marie-Alice Foujols, and Anne Cozic (IPSL); Reinhard Budich (MPI); and Michael Eby (University of Victoria) answered questions about their work developing GCMs and helped to verify our observations. Additionally, Michael Eby from the University of Victoria was instrumental in improving the diagram design.

This project was funded by NSERC and the Centre for Global Change Science at the University of Toronto.

The Coupling Process

Since the climate system is highly interconnected, a CBSE approach requires code to tie the components together - interpolating fluxes between grids and controlling interactions between components. These tasks are performed by the coupler. While all GCMs contain some form of coupler, the extent to which it is used varies widely:

- **CESM**: Every interaction is managed by the coupler.
- **IPSL**: Only the atmosphere and the ocean are connected to the coupler. The land component is directly called by the atmosphere.
- **HadGEM3**: all components are connected to the coupler, but ocean-ice fluxes are passed directly, since NEMO and CICE have similar grids.

A CBSE approach has even affected coupling. OASIS, a coupler used by many models (including COSMOS, HadGEM3, and IPSL) is built to handle any number and any type of components, as well as the flux fields within.

Complexity and Focus

A simple line count of GCM source code serves as a reasonable proxy for relative complexity. A model that represents many processes will generally have a larger code base than one that represents only a few. Between models, complexity varies widely. Within models, the bulk of a GCM's complexity is often concentrated in a single component, due to the origin of the model and the institution's goals:

- **HadGEM3**: atmosphere-centric. It grew out of the atmospheric model MetUM, which is also used for weather forecasting, requiring high atmospheric complexity.
- **UVic**: ocean-centric. It began as a branch of MOM, and kept the combination of a complex ocean and a simple atmosphere due to its speed and suitability to very long simulations.
- **CESM**: atmosphere-centric, but land is catching up, having even surpassed the ocean. It is embracing the "Earth System Model" frontier of terrestrial complexity, particularly feedbacks in the carbon cycle.

Conclusions

While every GCM we studied shares a common basic design, a wide range of structural diversity exists in areas such as coupler structure, relative complexity between components, and levels of component encapsulation. This diversity can complicate model development, particularly when components are passed between institutions. However, the range of design choices is arguably beneficial for model output, as it inadvertently produces the software engineering equivalent of perturbed physics (although not in a systematic manner).

Additionally, architectural differences may provide new insights into variability and spread between model results. By examining software variations, as well as scientific variations, we can better understand discrepancies in GCM output.