Geophysical Research Abstracts, Vol. 8, 05467, 2006
SRef-ID: 1607-7962/gra/EGU06-A-05467

# PyMCT, PyCPL and PyCCSM: Toward a scripting layer in high performance climate modeling

M. Tobis, M. Steder, R. Jacob and R. Pierrehumbert
University of Chicago (mtobis@geosci.uchicago.edu)

We describe our efforts to replace the coupling control layer in a contemporary climate model with an equivalent script. Specifically, research at the Department of the Geophysical Sciences has been focused on examining the feasibility of replacing the coupler module of the CCSM with an equivalent code written in Python.

The history of computing in the private sector shows a tendency towards ever higher levels of abstraction. A significant recent advance has been the emergence of powerful very-high level languages, notably Python and Ruby. While emerging from the tradiation of system administration scripting languages, these languages have advanced considerably from their roots. Use of these languages is widely reputed to dramatically improve programmer productivity and code reuse.

Importing this strategy, and the so-called agile methodologies that have emerged along with it, into climate science promises to offer large advantages and only modest disadvantages.

The primary disadvantage of very high level languages as opposed to conventional compiled languages is performance. By leaving critical per spatial element computations in a compiled language, it is possible to replace much of a model's structure with a script without dramatically compromising performance, especially at the top levels of the call tree. In exchange for this modest performance cost, scientists seeking to modify the code are presented with a far more readable and writable specification of the model's behavior.

By presenting a control layer capable of abstracting away the details presented by compiled codes, we both demonstrate the possibility of a new approach to model construction and provide a platform for experimentation with alternative coupling

schemes within the CCSM architecture.

Our efforts are complicated by the necessity to cope with codes written in Fortran 90, a language whose interoperability with scripting languages is quite limited, and by the necessity to provide an interface with MPI, the message passing interface. In overcoming these difficulties, we have been fortunate to be able to draw upon the designers of the existing coupling layer of CCSM, called the Model Coupling Toolkit or MCT, as well as on a set of interoperability bindings that the MCT developers have provided for the toolkit using the Babel tool suite and the Scientific Interface Definition Language or SIDL.