Jun 18th, 2:00 PM - 3:20 PM

Software Engineering for Scientific Application: Effort Report on the Community Land Model within the Earth System Modeling Framework

Dali Wang
The Climate Change Science Institute, Oak Ridge National Laboratory, wangd@ornl.gov

Yang Xu
The University of Tennessee, yxu30@utk.edu

International Environmental Modelling and Software Society (iEMSs)
7th Intl. Congress on Env. Modelling and Software, San Diego, CA, USA,
Daniel P. Ames, Nigel W.T. Quinn and Andrea E. Rizzoli (Eds.)
http://www.iemss.org/society/index.php/iemss-2014-proceedings

Software Engineering for Scientific Application: Effort Report on the Community Land Model within the Earth System Modeling Framework

Dali Wang[a], Yang Xu[b]

[a] The Climate Change Science Institute, Oak Ridge National Laboratory,
Oak Ridge, TN 37831, wangd@ornl.gov
[b] Department of Geography, The University of Tennessee,
Knoxville, TN 37966, yxu30@utk.edu

Abstract: One key factor in the improved understanding of earth system science is the development and improvement of high fidelity models. Along with the deeper understanding of earth system processes, the software complexity of those modelling systems becomes a barrier for further rapid model improvements and validation. In this paper, we present our experience on better understanding of the community land model (CLM) within an earth system modeling framework. After the science and software background of CLM, we represent three groups of CLM software engineering practices, which aim to 1) better understand the software system for rapid software system development on future computing platforms; 2) facilitate new model development via model-data comparison at field measurement level; 3) engage broad user communities via web services and cloud computing. Since better software engineering practices are much needed for general scientific software systems as we are adapting the integrated environmental modeling methodology, we hope those considerations can be beneficial to many other environmental modeling research programs involving multiscale system dynamics.

Keywords: Earth System Model, Community Land Model, Functional Testing, Experiment-inspired Software Design, Cyberinfrastructure.

1. INTRODUCTION

Environmental system modeling presents a variety of challenges. Through the past several decades, along with the rapid development of computing technologies, and strong interests of understanding large-scale environmental phenomena, many computer models have been developed to capture our knowledge on numerous facts of environmental systems, and to explore better options for system-wide management. The complexity of environmental models varies greatly and reflexes modelers' own view on the environmental system. Recently, researchers have emphasized integrated environmental system modeling (Estreguil et al. 2014, Laniak et al. 2013), which involves complex interactions between some or all components of an integrated environmental system, and results in model systems that link multiple components. Using those model systems, researchers can investigate the system behaviors using several modeling approaches developed by different research groups. There are non-disputable advantages to embrace this modeling approach. It is a very practical method to leverage existing modeling efforts (especially based on open source models) into a comprehensive system, which otherwise cannot be built by any single institution. Those model systems can naturally broaden community engagement. Also from the system design perspective, multicomponent system will impose restrictions over each individual component and provide a way to valid those subsystems (Li et al. 2007).

One implication of these efforts is that monolithic software development within a traditional computing framework is hindering further sustained innovation in those highly integrated model simulations using a group of existing scientific applications. The coupled earth system models are a typical example of

integrated environmental modeling systems. Herein, we report our software engineering practices on the Community Land Model (CLM) within a unified earth system modeling framework. First, we layout our specific goals related to CLM software engineering. Then we represent our efforts on the community land model in three categories that aim to 1) better understand software structure and performance; 2) facilitate experimental-inspired model-data comparison and 3) engage broad user communities. We hope those CLM software engineering efforts can inspire further activities for other integrated environmental system modeling.

2. SCIENCE AND SOFTWARE BACKGROUND OF THE COMMUNITY LAND MODEL

Over the past several decades, researchers have made significant progress in developing high fidelity earth system models to advance our understanding on the Earth systems, and to improve our capability of better projecting future scenarios. The Community Earth System Model (CESM) is one of leading models in US and is widely used for US Department of Energy's climate and environmental research. CESM system is based on a modeling framework that uses a coupler to integrate several component models (such as atmosphere, land, ocean, and sea-ice, etc.) into a complete earth system model. The whole system of CESM is reconfigurable, which provides a great flexibility to the community to design their own computational experiments. In our study, CESM has been configured into an offline land model simulation, which includes a data atmosphere model, an active land model and stub models for ocean, ice and glacier. Scientifically, this configuration uses historical climate forcing to drive active land component simulation, and provides unique capability to verify and calibrate land modeling activities with observational datasets (Wang et al. 2011 and 2013).

The active land model used in our study is the Community Land Model (CLM) that simulates surface energy, water, carbon, and nitrogen fluxes and state variables for both vegetated and non-vegetated land surfaces (Oleson et al. 2010). CLM has a long development history, which can be tracked back into the official release of LSM (http://www.cgd.ucar.edu/tss/lsm/) in 1996 on several targeted Cray machines. CLM2.0 was released in 2002 with improvements on many aspects of biogeophysics and ecosystem dynamics, and river routing etc. CLM2.0 can be executed efficiently on several parallel machines including IBM SP, SGI Origin 2000, and Compaq/alpha. CLM 3.0 was released in June 2004, with significant improvements on the terrestrial carbon cycle, dynamic vegetation, land-surface physics to reduce temperature biases, etc. CLM3.0 also contains radically changes related to data structure and loading balance methods. CLM 3.0 can be executed efficiently on both scalar parallel machines and vector parallel machines such as Earth Simulator or the Cray X1. CLM4 was officially released in April 2010. The model represents several aspects of the land surface including surface heterogeneity and consists of components or submodels related to land biogeophysics, the hydrologic cycle, biogeochemistry, human dimensions, and ecosystem dynamics (http://www.cesm.ucar.edu/models/clm/). The latest release is CLM version 4.5, which contains several submodels, including Canopy Fluxes, Ecosystem Dynamics, Hydrology, Urban, Soil Temperature, Hydrology, Fire, Dust, etc. Interested readers can look into Wang et al. 2014a for a visualization of those models.

3. SOFTWARE ENGINEERING PRACTICES FOR THE CLM

The general purposes of our software engineering practices are 1) to better understand the existing software structure to expedite rapid software developments on future computational facilities, 2) to facilitate new model development via model-data comparison at multiple scales, and 3) to engage broad user communities via cyberinfrastructure. In the following sections, we summarize our practices into three categories. The first type of activities, described in section 3.1, are designed to collect detailed information on current CLM's software structure, data structure, and computational characteristics. The second type of activities, described in section 3.2, focus on science community engagement for rapid new model developments. The third type of activities, described in section 3.3, focus on community oriented cyberinfrastructure to accelerate CLM model developments.

3.1 Computing oriented practices

As a typical scientific application, after almost two decades of active developments, the CLM software system became very complicated. The whole CLM modeling system consists of more than 1800 source files and over 350,000 lines of source code. The activities, described in this section, focus on several basic aspects of the CLM software system, which include data structure investigation, call tree generation, computational performance evaluation.

3.1.1 Data structure investigation

Data structure is the fundamental of any software system design. The CLM contains several submodels related to land biogeophysics, the hydrologic cycle, biogeochemistry, human dimensions, and ecosystem dynamics. The landscape surface is a key, hierarchical data structure for the CLM model. The majority of data arrays and derived datatypes within the CLM landscape surface data structure are declared as globally accessible variables. Therefore, understanding the relationship between the landscape surface data structure and each individual CLM function (module or subroutine) are very important. For this purpose, a CLM Fortran-syntax specific Perl-script was developed to decompose CLM code into simpler forms. This script marked and extracted information on 1) the global variables within CLM landscape surface data structures, 2) the name of each subroutine as well as 3) the procedure of subroutine invocation. The information on global variables with landscape surface data structure was further used to create interactive graphs to show the interconnections between CLM subroutines. The information on CLM subroutines is also extensively used in the following activities, such as software call-tree generation and performance evaluation. Interested reader can look into Xu et al, 2014 for more information. Besides the Perl-script, advanced software debuggers, such as the Allinea DDT and Rogue Wave TotalView, were used to collect the basic characteristics (range, mean, medium, deviation, etc.) of those global variables within landscape surface data structure.

3.1.2 Call tree generation

Call-graph analysis is a widely used tool for understanding the interactions between different parts of an application. It is straightforward to generate a static call tree graph using the subroutine information extracted from the source code. This kind static call tree graph provides very useful information on the interconnections between subroutines via shared global variables. Wang et al. 2014a described innovative methods to use advanced profiling and tracing software, the Vampir framework (http://www.vampir.eu) and the Tuning and Analysis Utilities (TAU) (http://www.cs.uoregon.edu/research/tau/) to collect necessary information for constructing CLM call graph based on run-time information. Those dynamic call tree graphs can provide extra information on subroutine invocation procedures, execution time associated with each individual subroutine, as well as the number of subroutine invocations during the whole simulation period.

3.1.3 Performance evaluation and analysis

With the collaborations with staff from National Center for Computational Sciences and Dresden University of Technology, the Vampir framework was also used in our research to collect CLM performance data on variety of computing platforms. The complete list of subroutines, which generated from the previous mentioned Perl-script, was used together with TAU instrumentation within Vampir Framework, to collect detailed performance data associated with each CLM individual subroutine. More detailed information on how to configure the Vampir framework for this purpose can be found in Domke and Wang 2012.

In order to better understand the potential impacts from the rapid hardware shift of computational facilities, CLM's global data-structure access patterns was tracked and analyzed using the Gleipnir tracing tool (Janjusic et al. 2011) within the Valgrind framework (http://www.valgrind.org). Based on those data-access patterns, cache simulators can be created to relate data access patterns with cache memory performance. That information is very valuable for CLM migration and further

performance tuning on emerging computing architectures (such as Graph Processing Units (GPUs) and Many integrated Cores (MICs)) as we are in the process of building pre-exascale supercomputer in 2017 (https://asc.llnl.gov/CORAL/).

Figure 1 summarizes the workflow for our computing oriented practices, along with software tools. It starts the source files analysis, which uses advanced debugging tools, or in-house code parser to collect the basic information on CLM subroutines, landscape surface data structure (global variables), as well as their interconnections. The basic information was then used for CLM call tree graph generation and automatic instrumentation. The performance data in our study mainly came from Vampir framework and an in-house memory pattern analysis tool, Gleipnir. At last, a variety of software tools have been used for data visualization and analysis.

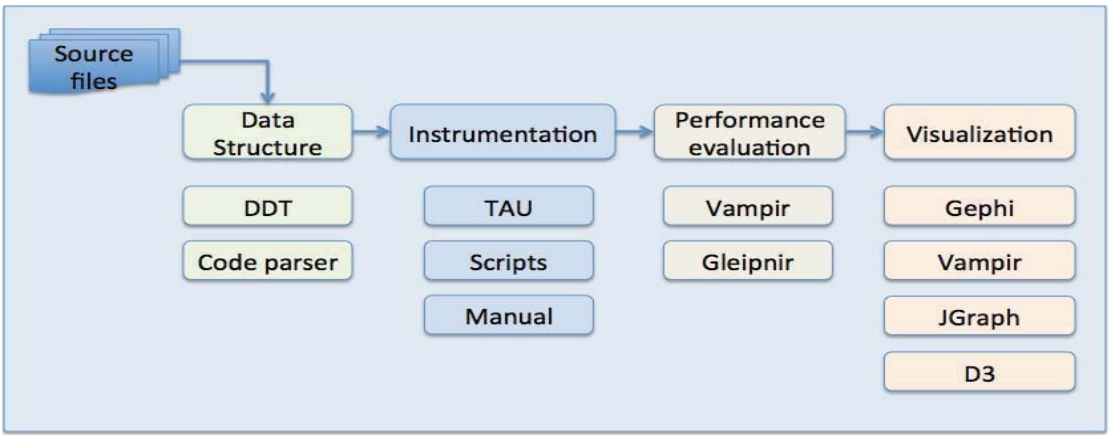

Figure 1: Workflow of computational oriented practices

3.2 Science oriented practices

3.2.1 Experiment-inspired modular design

Current CLM module design represents the historic view of ecological climatology. For example, individual modules were developed for surface radiation, canopy fluxes, hydrology, soil temperature, as well as surface albedo, etc. Since the landscape surface was represented by the combination of land-units (such as urban, lake, vegetated portion with soil column, etc.), individual modules for urban, lake, ecosystem dynamic were also developed separately within the CLM.

Biological and Environmental Research (BER) program within US Department of Energy has a distinguished history of designing, testing, and implementing leading-edge experimental approaches to study long-term effects of climate and atmospheric composition on the structure and functioning of terrestrial ecosystems. Distinguish projects include The Free-Air CO2 Enrichment (FACE) experiments, Next Generation Ecosystem Experiment (NGEE), Spruce and Peatland Responses Under Climatic and Environmental Change (SPRUCE). BER investments also provide management and support infrastructure for AmeriFlux, the interagency activity coordinating long-term CO2 (and energy) flux measurements across North America. (science.energy.gov/ber/research/cesd/ terrestrial-ecosystem-science/). Terrestrial ecosystem processes can be described by an exchange between four major compartments, such as foliage, structural material, including roots and wood, surface detritus or litter and soil organic matter (including peat) (Wang et al. 2011). Therefore, an experiment-inspired software view, based on current CLM software system, is being developed for new ecosystem functional module developments. Figure 2 shows a snap of current CLM subroutines or their child subroutines that uses root related global variables. We are working to develop a new root module for further root model development and functional testing.

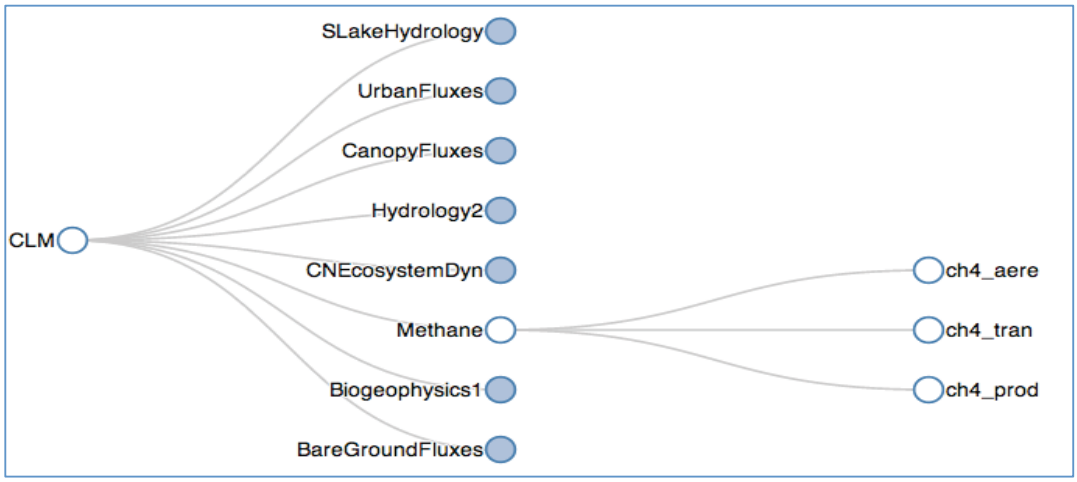

Figure 2: CLM Subroutines Accessing Root-related Global Variables. Gray Nodes are expandable.

3.2.2 Model development via functional testing

Variety of observational instruments, networks and datasets, ranging from satellite images to experiment results from site-based process studies, are available for the CLM model comparison (Wang et al, 2011). There are many reports that use remote sensing data and gap-filled synthesis data to validate and verify the regional and global CLM simulation results. In order to better represent the key biogeophysical and biogeochemical functions within CLM using the variety of datasets from a variety of field experiments, a functional test platform is developed (Wang et al. 2014b) to create direct linkages between site measurements and key ecosystem functions within CLM. The platform consists of three major parts: 1) interactive user interfaces, 2) functional test models and 3) observational datasets. It provides much needed integration interfaces for both field experimentalists and ecosystem modelers to improve the model's representation of ecosystem processes within the CESM framework without large software overhead.

Figure 3 presents two typical usages of our functional testing. The first case is designed to setup computational experiments and to test a target functional element under different environmental settings. The second case is designed to setup observation-inspired computational experiments, which in turn, to enable a direct model-data comparison based on similar environmental settings.

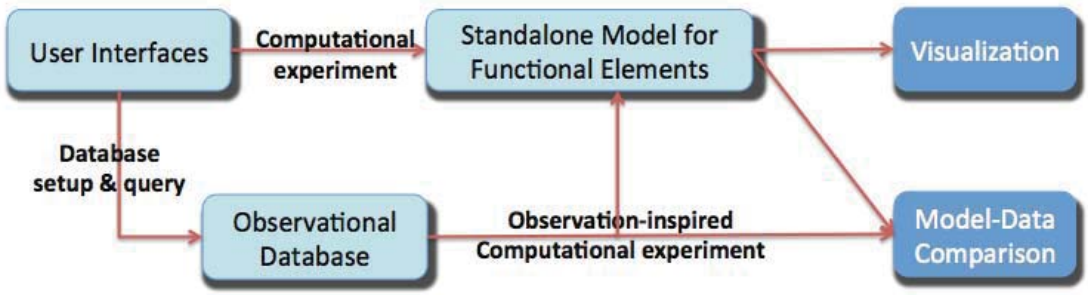

Figure 3: Two Typical Usage of Ecosystem Functional Testing

A case study of CLM functional testing of photosynthesis module was presented in Wang et al. 2014b. Currently, with the collaborations of modelers, field scientists, and observational dataset providers, several modules have being developed, including Canopy Fluxes, Microbe and the previous mentioned Root module. Observational datasets connected with our functional testing platform include a global leaf property database (Leafweb: leafweb.ornl.gov), a subset of a long-term ecosystem measurement network (AmeriFlux: ameriflux.ornl.gov), and a Microbial and Methane

dataset (Xu et al. 2014). This functional testing approach is also a part of an international effort, which aims to develop a path forward to improve the representation of fine roots in terrestrial biosphere models.

3.3 Community oriented practices

As we mentioned in the previous sections, CLM software system has been investigated and a functional testing platform was developed to bridge the gaps between modelers, field scientists, as well as the observational datasets providers. We believe that better engagement with large and diverse user communities, who have difference research focus and computing background, is the key to future CLM success.

In our previous studies, desktop-based computing technologies for both software structure analysis and functional testing. For example, within the functional testing platform, desktop-based Java applications have been developed as user interfaces for individual functional module testing. Open source Java-based libraries have been used to visualize the model-data comparison, and observational datasets have been preloaded in a local database for real-time data query. Furthermore, the procedure to generate standalone model for individual functional element requires advanced computing environment configurations related to compilers, file manipulation, software libraries installation, operating system preferences, etc. Similarly, software structure was represented using desktop-based graph visualization software, such as Gephi (www.gephi.org), and JGraph (www.jgraph.com), etc.

Deliver those applications to broad communities are still quite challenging. It is known that web services are only few common computing technologies widely adapted by field scientist, observational datasets provider and modelers. Therefore, a cyberinfrastructure is being developed to accelerate CLM development by further reducing the computational barriers via web-based services and cloud computing technologies. Web-based services development aims to provide Internet-based, interactive interfaces for computational experiment configuration and results exploration. Cloud computing technologies will be used to deliver a customized computing environment directly to the users with minimum user efforts on compiler, library configuration, as well as local database establishment.

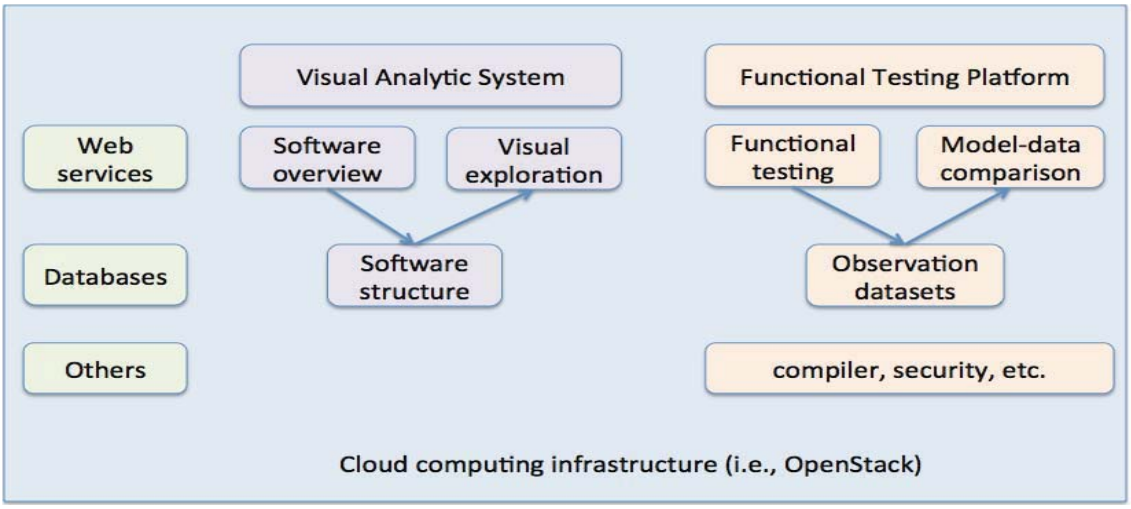

Figure 4: Cyberinfrastructure for Accelerated CLM Development

Figure 4 shows the major components of our evolving cyberinfrastructure, which currently contains two systems: 1) online visual analytic system to explore and analyze the CLM software structure information, and 2) a web-based functional testing platform to enable direct model-data comparison

for rapid model development. Three major building blocks of our system are: web services (e.g. an example of software structure representation can found at http://www.ornl.gov/~7xw/CLM_Overview.html), online databases (e.g. AmeriFlux http://ameriflux.ornl.gov/) and cloud-based computing systems within ORNL's Compute and Data Environment for Science (CADES) (http://computing.ornl.gov/cades/).

4 SUMMARY AND FUTURE WORK

In this paper, we have presented our software engineering practices related to a scientific application: the Community Land Model with an earth system modeling framework. The primary goals of those practices are 1) to facilitate new model development and 2) to engage broad user communities. Future work will focus on our system enhancements, such as CLM software structure analysis, new module design and validation, etc. We are also working hard to encourage the broad communities to share those modeling-oriented datasets around CLM. We hope our open cyberinfrastructure can further facilitate new modularized ecosystem functional module design and development. Since better software engineering practices are much needed for integrated environmental modeling efforts, we hope those considerations can be beneficial to many other environmental modeling research programs involving multiscale system dynamics.

5. ACKNOWLEDGMENTS

This research was funded by the U.S. Department of Energy (DOE), Office of Science, Biological and Environmental Research (BER). This research used resources of the Oak Ridge Leadership Computing Facility, located in the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the Department of Energy under Contract DE-AC05-00OR22725. Oak Ridge National Laboratory is managed by UT-Battelle LLC for the Department of Energy under contract DE-AC05-00OR22725.

6. REFERENCES

Domke, J., Wang, D., Runtime Tracing of the Community Earth System Model: Feasibility Study and Benefits, 2012, 12th Workshop on Tools for Program Development and Analysis in Computational Science, Omaha, Nebraska, June procedia CS Vol 9, pp1950-1958

Estreguil, C., Eigo, D., Caudullo, G., 2014, A proposal for an integrated modelling framework to characterise habitat pattern, Environmental Modeling and Software, Vol. 52, pp176-191

Janjusic T, Kavi KM, and Potter B, 2011, Gleipnir: A Memory Analysis Tool, Procedia Computer Science, Vol 4, pp 2058–2067.

Laniak, G.F, Olchin, G., Goodall, J., Voinov, V. , Hill, M., Glynn, P., Whelan, G., Geller, G., Quinn, N., Blind, M., Peckham, S., Reaney, S., Gaber, N., Kennedy, R., Hughes, A.,2013, Integrated environmental modeling: A vision and roadmap for the future, Environmental Modeling & Software, Vol39, p3-23

Li, K., Peng Zhang,John C. Crittenden,et al., 2007, Development of a Framework for Quantifying the Environmental Impacts of Urban Development and Construction Practices. Environmental Sciences and Technologies, Vol 41(14), pp5130–5136.

Oleson, K., Lawrence, D., Gordon, B., Flanner, M., Kluzek, E., Peter, J., Levis, S., Swenson, S., Thornton, P., and Feddema J., 2010, Technical description of version 4.0 of the Community Land Model (CLM). http://www.cesm.ucar.edu/models/cesm1.0/clm/CLM4_Tech_Note.pdf

Wang, D. Domke, J., Mao, J., Shi, X., Ricciuto, D. 2013. A Scalable Framework for the Global Offline Community Land Model Ensemble Simulation, International Journal of Computational Science and Engineering, (in press)

Wang, D., Ricciuto, D., Post, W., Berry, M., 2011. Terrestrial Ecosystem Carbon Modeling, Encyclopedia for Parallel Computing, Editor, David A Padua, Springer, 2011, DOI 10.1007/978-0-387-09776-4, pp 2034-2039

Wang, D., Post, W., Wilson, B., 2011. Climate Change Modeling: Computational Opportunities and Challenges, IEEE Computing in Science and Engineering, Vol 13(5), pp36-42

Wang, D., Schuchart, J., Janjusic, T., Winkler, F., and Xu, Y. 2014a. Toward Better Understanding of the Community Land Model within the Earth System Modeling Framework, International Conference on Computational Science, Cairns, Australia, 2014, (accepted)

Wang, D., Xu, Y., Thornton, P., King, A., Gu, L., Steed, C., Schuchart, J., 2014b. A Functional Testing Platform for the Community Land Model, Environmental Modeling and Software, Vol. 55, pp25-31, 10.1016/j.envsoft.2014.01.015

Xu, X., Schimel, J.P., Thornton, P., Song, X., Yuan, F., Goswami. S., 2014. Substrate and Environmental Controls on Microbial Assimilation of Soil Organic Carbon: A Framework for Earth System Models. Ecology Letters, DOI: 10.1111/ele.12254. 2014.

Xu, Y., Wang, D., Janjusic, T., Xu, X., (in review) A Web-based Visual Analytic System for Understanding the Structure of Community Land Model, International Conference on Software Engineering Research and Practice, June, 2014.