Procedia Computer Science

Volume 29, 2014, Pages 1722–1732

ICCS 2014. 14th International Conference on Computational Science

Design and Implementation of Kepler Workflows for BioEarth

Tristan Mullis[1], Mingliang Liu[2], Ananth Kalyanaraman[1], Joseph Vaughan[2], Christina Tague[3], Jennifer Adam[2]

[1]School of Electrical Engineering & Computer Science, Washington State University, Pullman
[2]Department of Civil & Environmental Engineering, WSU, Pullman
[3]Bren School of Environmental Science & Management, University of California, Santa Barbara
Corresponding author: ananth@eecs.wsu.edu

Abstract

BioEarth is an ongoing research initiative for the development of a regional-scale Earth System Model (EaSM) for the U.S. Pacific Northwest. Our project seeks to couple and integrate multiple stand-alone EaSMs developed through independent efforts for capturing natural and human processes in various realms of the biosphere: atmosphere (weather and air quality), terrestrial biota (crop, rangeland, and forest agro-ecosystems) and aquatic (river flows, water quality, and reservoirs); hydrology links all these realms. Due to the need to manage numerous complex simulations, an application of automated workflows was essential. In this paper, we present a case study of workflow design for the BioEarth project using the Kepler system to manage applications of the Regional Hydro-Ecologic Simulation System (RHESSys) model. In particular, we report on the design of Kepler workflows to support: 1) standalone executions of the RHESSys model under serial and parallel applications, and 2) a more complex case of performing calibration runs involving multiple preprocessing modules, iterative exploration of parameters and parallel RHESSys executions. We exploited various Kepler features including a user-friendly design interface and support for parallel execution on a cluster. Our experiments show a performance speedup between 7–12x, using 16 cores of a Linux cluster, and demonstrate the general effectiveness of our Kepler workflows in managing RHESSys runs. This study shows the potential of Kepler to serve as the primary integration platform for the BioEarth project, with implications for other data- and compute-intensive Earth systems modeling projects.

Keywords: Kepler, scientific workflows, environmental modeling, Earth system modeling, data-intensive application, compute-intensive application, parallel workflows.

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2014
© The Authors. Published by Elsevier B.V.

1 Introduction

The Biosphere-relevant Earth system model (BioEarth[1]; Adam *et al*., 2014) is a regional-scale Earth system modeling (EaSM) framework, being developed by a multi-disciplinary team, that incorporates biogeochemistry and dynamic vegetation in both managed and unmanaged landscapes, and couples the atmospheric, hydrospheric, and anthrospheric realms to capture important feedbacks. The anthrosphere addresses the impacts of socio-economic activities in human-managed systems such as forestry and agriculture. The models integrated within BioEarth include atmospheric models (for meteorology and atmospheric chemistry), land surface models (for hydrology, biogeochemical cycling, and dynamic vegetation), aquatic models (for reservoir operations and nutrient export via rivers), and economic models. The overarching science goal of BioEarth is to improve understanding of decadal (and shorter) time-scale interactions among the coupled carbon (C), nitrogen (N) and water (H_2O) cycles and human actions for the region, under global change (Adam *et al*., 2014; Allen *et al*., 2013; Liu *et al*., 2013). We plan to achieve these goals by exploring the coupling of multiple stand-alone models within a modular framework to generate region-specific information about the impacts of changes in climate, policy and management on cropping, rangeland, and forested agro-ecosystems, while explicitly considering environmental impacts, water resource sustainability, and hydropower generation.

BioEarth integrates multiple existing models addressing atmospheric, terrestrial, aquatic, and economic subsystems with varying levels of coupling (Fig. 1). The models are as follows: regional climate model (Weather Research and Forecasting – WRF, Skamarock *et al*., 2008), air quality model (Community Multi-scale Air Quality – CMAQ, Byun and Schere, 2006), biogenic emission model (Model of Emissions of Gases and Aerosols from Nature – MEGAN, Guenther, 2012), and land surface/terrestrial ecosystem models. The land surface models include i) an integrated VIC-CropSyst model (Rajagopalan *et al*., 2014, in preparation), which couples the macro-scale hydrological model (Variable Infiltration Capacity – VIC, Liang *et al*., 1994) and the agricultural model (CropSyst, Stockle *et al*., 2003), and ii) a catchment-scale biogeochemical and eco-hydrological model (Regional Hydro-Ecologic Simulation System - RHESSys, Tague and Band, 2004).

Adam *et al*. (2014) describes two levels of integration within BioEarth for these models: one-way ("offline") and two-way ("online" or "full") coupling. For instance, models can either be linked offline (run sequentially) or fully-coupled at finer (30- to 60-minute) or coarser (24-hour) time-steps. Many BioEarth components can be coupled at longer time-steps – *e.g.*, the reservoir model (Columbia River Simulation Model ColSim, Hamlet and Lettenmaier, 1999), the nutrient transport model (Global Nutrient Export from Water(S)sheds – Global NEWS, Seitzinger *et al*., 2005), and an economic and land-use model. As BioEarth is a regional EaSM, it uses simulated results from a Global Climate Model (Gent *et al*., 2011) for boundary conditions. Software for the above models is publicly available and the programming languages include C, C++, Fortran, Python and Excel.

Designing workflows for BioEarth is challenging for several reasons, first of which is the scale of integration required to meet project goals. There are a large number of interacting components, as illustrated in Fig. 1, and several different input options. To design complex workflows that can seamlessly integrate these components while remaining understandable, reusable, and modifiable is a significant challenge. Furthermore, the diversity in computing platforms and programming environments complicates workflow design. High performance computing (HPC) requirements dictate the need for a workflow to support distributed and/or parallel computing. Also, for a project of this scale, having a number of dynamically evolving groups actively contributing both code and data makes versioning, tracking and provenance capabilities essential. Lastly, given the diversity in expertise expected of the user group, usability, such as from GUI support, becomes critical. Consequently, in this paper we explore the Kepler workflow development tool (Altintas *et al*. 2004)

[1] http://www.cereo.wsu.edu/bioearth/

for its support for all these features, and report on our progress. Specifically, this paper makes the following *contributions*. We present the design, development and evaluation of Kepler workflows for implementing one of the core BioEarth models, *viz*. the RHESSys model. We present workflows for supporting two types of RHESSys model executions: a single stand-alone execution of RHESSys, and calibration involving multiple parameter settings and logging features. One of the key improvements we achieved, as a result of using Kepler, is support for parallelism in RHESSys executions.

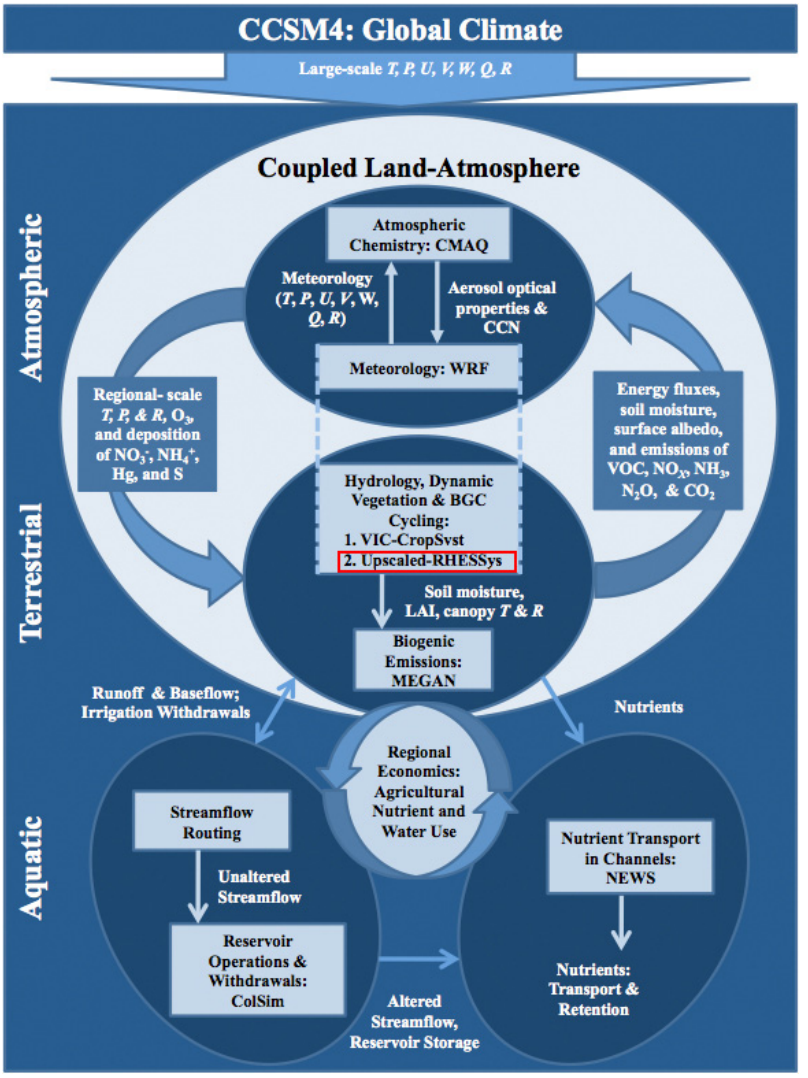

Figure 1: Linkages between the atmospheric, terrestrial, aquatic, and economic components of the BioEarth model. Abbreviations: T=temperature, P=precipitation, U,V,W=wind components, Q=water mixing ratio, R=radiation, CCN=cloud condensation nuclei, O_3=ozone, NO_3^-=nitrate, NH_4^+=ammonium, Hg=mercury, S=sulfur, VOC=volatile organic compounds, NO_x=NO + NO_2=nitric oxide + nitrogen dioxide, NH_3=ammonia, N_2O=nitrous oxide, CO_2=carbon dioxide, LAI=leaf area index (Adam et al., 2014).

We report on this parallel design as well as experimental results relating to scalability and performance.

Following this introduction: Section 2 introduces the RHESSys model; Section 3 discusses the problem of managing RHESSys runs, and indeed of all such models as jointly constitute BioEarth; Section 4 details the Kepler workflows for RHESSys; Section 5 reviews our experimental results and discusses our main findings; and Section 6 concludes our report on developments to date for Kepler workflows for BioEarth.

2 RHESSys Overview

The Regional Hydrologic-Ecologic Simulation System, RHESSys (Tague and Band, 2004), is a computational watershed model designed to simulate climate and land-use change impacts on ecosystem carbon and nutrient cycling and hydrology. RHESSys integrates a set of sub-models, for ecosystem carbon and nutrient cycling, within a spatially distributed hydrological model simulating above and below ground biogeochemical and hydrological processes for a watershed. RHESSys has been used to study the effects of climate and land-use change on hydrology and ecosystem function in a number of different environments, including the Pacific Northwest (Christensen *et al*., 2008; Tague and Grant, 2009; Tague *et al*., 2013).

In BioEarth, RHESSys will function as: 1) a land surface model for simulating terrestrial ecosystem feedbacks (including energy and greenhouse gases fluxes) to regional climate, and 2) an impact model for assessing effects of climate change and human activities on the structure and function of managed and natural ecosystems. Major inputs for RHESSys include meteorology (temperature, precipitation, wind speed, and radiation) and atmospheric nitrogen deposition. Required spatial inputs include contemporary land cover (vegetation types), soil texture and chemistry, and topology. Use of RHESSys for a specific watershed requires calibration to that watershed by finding a set of parameters so that RHESSYs results match historic stream-flow observations. Thus watershed-specific RHESSys calibration requires solving for these six parameters: Ksat, an effective (patch to hill-slope scale) saturated hydraulic conductivity; m, the decay of Ksat with depth; pa and po, pore size index and air entry pressure (for determining available water capacity); gw1, the amount of infiltrate water bypassing the soil matrix directly into deep groundwater; and gw2, a parameter controlling deep groundwater drainage.

To prepare a RHESSys simulation, a hierarchical set of spatially explicit simulation objects must be delineated. These objects organize the representation of different meteorological, ecological and hydrological processes. Watersheds are subdivided into hill-slopes that drain to individual stream reaches; within hill-slopes, zones represent areas with similar meteorological conditions. Vertical hydrological and soil and litter biogeochemical cycling are modeled within patches. Water and nutrients are also transported laterally between patches, although flow is constrained from crossing hill-slope boundaries. Patch definition includes strata supporting the modeling of vegetation layers, such as overstory and understory, and their hydrologic, carbon and nutrient cycling processes. When coupled with a regional climate system model (*e.g.,* WRF), the patch-level outputs from RHESSys will be aggregated into WRF grid cells; when coupled offline to the NEWS model, the RHESSys stream-flow and runoff will be inputs for simulating the nutrient yield and export processes in riverine systems.

3 Challenges in Management of Models

BioEarth's models are currently managed with complicated shell scripts. This leads to a variety of challenges. Plug-and-play features are *not* supported, and any extension or modification requires direct editing. The execution environment is hard-coded into the scripts, making interoperability and portability difficult. These manually managed BioEarth scripts likely underutilize resources and delay executions. Bookkeeping for metadata describing model runs (parameters, input and output files, and version control) is managed manually. The lack of standardization in protocols further complicates

management. Coupling multiple models for more complex runs is also very difficult, as it requires more shell scripting. Scripting also poses several challenges to human usability. Scripts are hard to understand and maintain, leading to difficulty with script reuse. The lack of a GUI also limits the ease of use. Due to all these shortcomings of scripting for model management (McPhillips *et al.*, 2009), we chose to pursue the use of scientific workflows using Kepler (Altintas *et al.*, 2004).

4 Design of Kepler workflows for RHESSys case-study

The RHESSys model is implemented in C and takes as input: a world-file, a tec-file, an output directory and various parameters including the simulation period. A world-file details a hierarchy of watersheds, hill-slopes, and patches. A tec-file specifies the temporal events controlling the model. The RHESSys output is available for hourly, daily, monthly, or yearly time intervals for all spatial units; all output files are tabular. The data flow for RHESSys execution is shown in Fig 2. There is a need for handling multiple input sources to supply a wide range of input data as shown in Fig. 2, and each option requires specific preprocessing to generate an input for RHESSys. Post-processing of the output depends upon the scientific objectives.

Our development of Kepler workflows for executing the RHESSys model has so far resulted in three distinct variants of workflow, for either single execution or calibration. The first variant is a standalone serial execution workflow, and is described in section 4.1.1. The second variant is a standalone parallel execution, and is described in section 4.1.2. The third variant is designed to perform calibration runs by iterating over the second variant, and is described in Section 4.2.

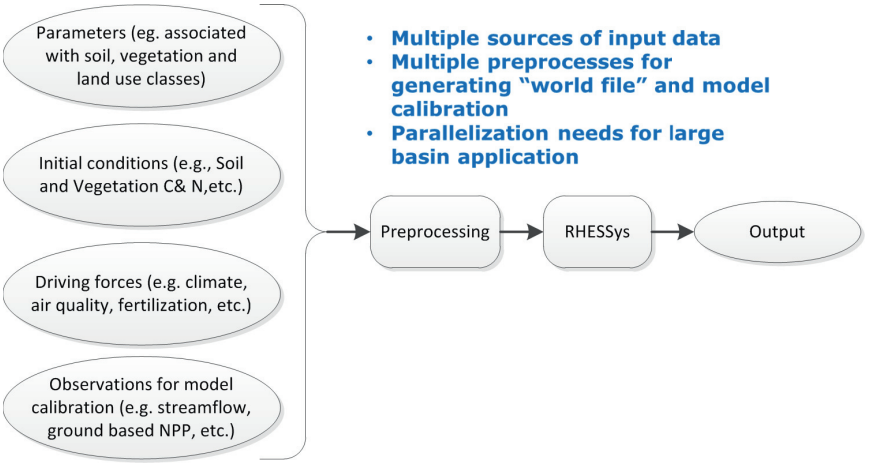

Figure 2: An overview of the main steps in RHESSys application for BioEarth. Ellipses represent data and rectangles represent processing.

4.1 RHESSys Standalone Execution

4.1.1. Workflow for Serial Execution

For the implementation of the serial execution RHESSys workflow, we present a layered design in which the basic parameter and input/output controls are provided at the highest level for easy accessibility, while the internal parameters and execution components are implemented within reusable actors that could also be optionally reconfigured by an advanced user. RHESSys parameters

fall into three categories: 1) parameters for remote system configuration include such as username, server, and working directory, used for setting the compute environment; 2) parameters specific to RHESSys; and 3) parameters for script generation and configuration of the compute cluster's queuing system, which is currently defaulted to Portable Batch System (PBS). User control for the first two parameter sets are provided at the highest level of the workflow, while the third set can be controlled from within a subsidiary actor of the workflow at a lower level. As a whole, this design is intended to provide a high degree of configurability to the end-user (application scientist), while hiding most of the lower-level implementation details.

A high level illustration of the serial execution workflow is shown in Fig. 3a. The first task is to manage the network connections to HPC resources. Next, a job script for running RHESSys is automatically generated using the user-specified input parameters. This script contains both the environmental setup including the queuing system and the command to execute the RHESSys model. After the script is generated, it is submitted to the job scheduler, which then assigns a unique job ID. The final stage of the workflow uses this job ID to wait for the job to complete. This workflow design is modular and is intended to allow reuse and subsequent incorporation into more complex workflows.

4.1.2. Workflow for Parallel Execution

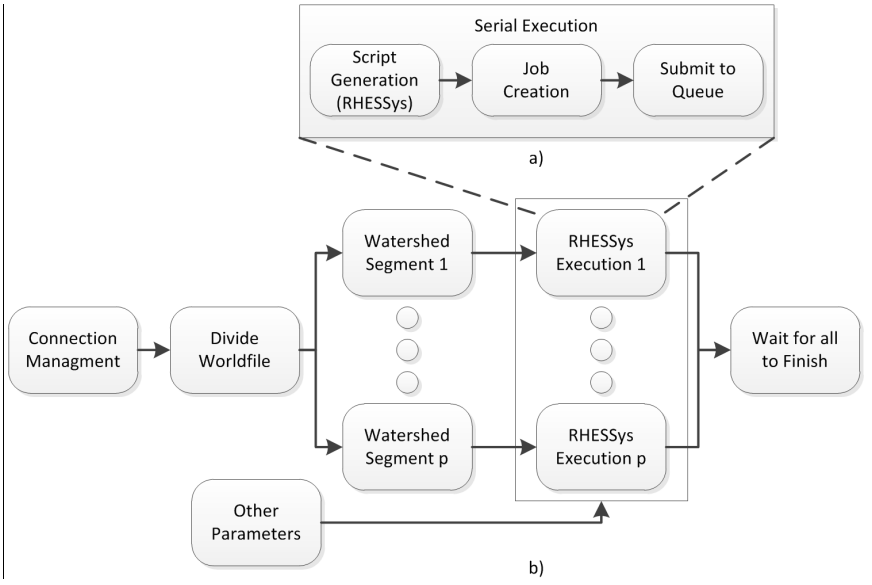

Figure 3: Part a) shows the serial execution workflow for RHESSys. Part b) shows the parallel workflow in which the world-file is divided into multiple independent serial executions.

In this section, we present our design of Kepler workflows to support a parallel execution of RHESSys. Even though the currently available distribution of RHESSys is serial, it is amenable to parallelization because input watersheds (in the world-file) can be independently processed. Consequently, we designed a Kepler workflow for parallelizing RHESSys executions as follows (see Fig. 3b):

Step 1) The world-file, which provides all input information (vegetation type, soil type, climate station ID, and watershed hierarchical structure, *etc*.) for each simulation, is analyzed (using a Python script) to determine the "size" of each watershed (measured in the number of patches per watershed)

as the runtime is dependent on this size. Let *n* denote the total number of all patches across watersheds.

Step 2) Let *p* denote the number of processing cores available in the parallel platform. The set of watersheds in the world-file is then partitioned into *p segments* such that no watershed is split across two different segments and the total number of patches assigned to each processing core is roughly *O(n/p)*. While such a distribution cannot be guaranteed for all inputs, the scheme is effective in distributing the load in a balanced manner in most of the cases. Each segment, which represents a subset of watersheds, is then output into a separate world-file.

Step 3) The Kepler workflow is then used to instantiate a parallel execution whereby each of the *p* world-files is processed by an instance of the RHESSys serial execution workflow described in Section 4.1.1.

Step 4) The workflow completes when all the *p* segments have been processed and when the output of all the *p* RHESSys runs have been merged into a single set of output files. Part of the output naming convention is specified by the user as a parameter, and the remainder is a prefix assigned from RHESSys, such that naming and reconstruction of the final output files can be completely automated, since the process is completely deterministic.

Parallelization is required to ensure scalability as a large number of RHESSys runs will be conducted over the course of the project. For example, a single RHESSys serial run over a simulation period of two years on a system with 10 watersheds containing 68503 patches takes 2 h 12 min.

Additional implementation details include: i) Reuse of workflow components under the parallel setup is achieved by placing the serial components in single actors that are run individually until completion, while a blocking wait is placed outside the actors (as part of the caller module). ii) The user has the option of assigning either the number of compute nodes or the number of cores as the value for *p*. This provides for flexibility in allocating more memory per RHESSys process, or for supporting a multithreaded implementation of RHESSys, if such an implementation were to become available in the future. iii) The parameter interface used in the parallel workflow is same as that in the single RHESSys execution Kepler workflow, with an addition specifying the number of cores (*p*). The output prefix file name is modified internally so the output of each run is unique and can later be aggregated in one set of files.

4.2 Calibration Workflow

Calibration of RHESSys parameters for watersheds being modeled is an essential step in the model's application. This step often involves exploring tens to hundreds of different parameter settings, and typically consumes significant compute resources, as each calibration run requires simulation of hundreds of years to reach the equilibrium state. Currently, calibration runs are managed manually requiring significant person-hours of effort. To manage the procedure in an automated fashion, to the extent possible, we designed a Kepler workflow, described as follows.

Calibration is an iterative process which, for each watershed, involves several steps: i) defining parameter spaces (ranges) and assembling parameter sets to execute (this is internally controlled by an R script); ii) running RHESSys under each parameter set; iii) comparing output to recorded (observed) flows to adjust the parameters; and iv) repeating steps i through iii until an optimal parameter set has been identified. All the above steps except step iii are automated in our Kepler workflow. Step iii has been partially automated (to perform the necessary comparisons) so that the user is involved only at the refinement stage. Note that each RHESSys execution explores a different parameter set independently; therefore this step exposes a second dimension for parallelization if required. For instance, in our experiments we used 25 different parameter sets, automatically defining 25 parallel runs. The unique parameter sets encode the parameters described in section 2 and are selected

randomly from uniform distributions. Additional parameters control operation of the R script. Due to the lack of space, the finer implementation details of the workflow are omitted.

5 Experimental Results and Discussion

The experimental setup is as follows: for our test platform, we used an in-house 25-node, 704-core Linux cluster with 16 compute nodes (8 Intel 2.4 GHz Xeon cores per node, 8GB per node) connected via a 10 Gbps Ethernet and 190TB ZFS storage. The desktop used to run Kepler 2.4 utilizes the Java(TM) SE Runtime Environment 1.7.0_45, and R version 3.0.1 32-bit. The input data sets used are as follows. A small dataset containing one watershed and one patch was used for validation purposes; its serial execution takes under a second. The second data set (referred to as *RHESSys-training*) contains a larger watershed with a total of 70 hill-slopes and 5789 patches. The third data set (referred to as *RHESSys-training-2*) contains 10 watersheds, 811 hill-slopes, and 68503 patches. All runs were performed over the simulation period 1990-1992.

5.1 Results

A single execution of our Kepler version of the serial execution workflow for the RHESSys-training input completed in ~8 min. Alternatively, the use of shell scripts to manage the runs resulted in a completion time of 7 min. A majority of the additional minute can be attributed to the network connection latency, entering the password by the user, and the queuing system allocating resources and launching the task through Kepler. This overhead is a small constant that is expected to become negligible as large-scale runs are performed. On the other hand, through its automated capabilities, the new workflows have now made it significantly easier to manage jobs, share workflows and enable usage by untrained users.

Over the RHESSys-training-2 input, the single serial execution workflow took 2 h 12 min to run. In contrast, the parallel execution workflow completed in ~20 min using 16 cores. This represents a 7x speedup, compared to the theoretical maximum achievable speedup of 10x based on the number of watersheds. The marginal loss in speedup was due to two reasons, the individual time of processing watersheds being limited by the maximal run of 14 minutes, and the additional four and a half mins required to combine the separate output files.

Executing RHESSys calibration manually is an involved task, requiring editing several files and scripts. The final script that is generated does the calibration runs serially, while incrementally updating relevant output files, contributing to its long run time. This script alone takes 3 h for 25 parameter sets on RHESSys-training. Execution of the RHESSys calibration workflow on the RHESSys-training input took 15 min on average to run for 25 parameter sets using 16 cores. This represents a 12x speedup over the manual method. When compared to the time of the single standalone execution workflow it takes 7 min longer to handle 24 more runs of RHESSys. The reason for this was our using fewer than 25 nodes on which to run jobs in parallel, thus two batches had to be run serially. Using more nodes we would expect to break the 10-minute mark in runtime, getting closer to the maximum achievable speedup of 25x.

5.2 Discussion

The Earth is a very complex system including atmosphere, biosphere, hydrosphere, geosphere, and anthrosphere (human). Normally, Earth system models (EaSM) simulate several components of the Earth system using specific sub-models. The interactions between these sub-models are implemented with couplers. These couplers transfer information on energy, momentum, material flows, *etc*., handling translation between disparate temporal and spatial scales. For the user (normally a scientist– not a software engineer) the convenience and transparency of adjusting major parameters and selecting options such as spatial and temporal domains, assumptions or algorithms for major

processes, are the most important considerations for using the EaSM. Repeatability of experiments and reproducibility of results are essential for good science, providing evidence of robust methods as are needed for modern research into complex systems and interdisciplinary questions. Knowing the integrity of each sub-system may not guarantee a correct simulation of the Earth system as a whole, which also requires that the interactions and feedbacks among the sub-systems are correct. As demonstrated in this study, the Kepler implementation of a critical component of one EaSM can save substantial time on repetitive computation tasks (i.e., RHESSys calibration) and achieve parallelization without the burden of recoding models, and also provide for the visualizing of workflows. These workflows can be conveniently edited for reuse by other users without detailed knowledge of the embedded procedures (preprocessing of the input data sets, model calibrations *etc*.) for running a specific model and/or coupled models.

BioEarth is an integrated modeling approach to investigate the interactions at the interfaces of various sub-systems through coupled C-N-H2O processes (Adam *et al*., 2013; Liu *et al*., 2013). The Kepler workflow implementation for RHESSys will be emulated for other models and their individual Kepler workflows should also be flexible for both off-line (or one-way, sequential) couplings and on-line (two-way) couplings (Liu *et al*., 2013). For these model couplings, for each individual model included within the workflow, the encapsulation of pre-processing for inputs and post-processing for outputs must also handle required scaling and interpolation in spatial and temporal domains and data format conversion between different models. An effective way to assess the efficiency of workflow implementations is to measure the time-to-solution and the reductions achieved in the time to use the tools by the end user. However, other measures such as memory usage, storage and stability need to be taken into consideration as well.

6 Conclusion

Our Kepler workflow case study thus far has shown promising results. All the workflows developed are more user-friendly than are scripts, and can easily be reconfigured for different data sets. The workflows performed with minimal overheads, and in the case of the parallel workflow implementation, we observed significant speedups over serial executions. Automation also saves significant hours of personnel resources spent in manual management. Our immediate goal is to incorporate the provenance feature of Kepler in order to perform bookkeeping of metadata associated with model runs, and provide a common interface to such metadata. Another improvement is to develop a heuristic that automatically determines the size of compute resources (RAM size, number of cores, *etc*.) required for RHESSys runs, with the goal of improving resource utilization. A longer term goal is to implement a Distributed Data-Parallel framework (Wang *et al*., 2012) and compare it to our current implementation of the data parallel RHESSys workflow.

Acknowledgements

We thank the Kepler development team, led by Dr. Ilkay Altintas at SDSC, for training, user support and other guidance. The Biosphere-relevant Earth system modeling project, BioEarth, is supported by the U.S. Department of Agriculture (grant no: 20116700330346).

References

Adam, J. C., Stephens, J. C., Brady, M. P., Chung, S. H., Evans, R. D., Kruger, C. E., Lamb, B. K., Liu, M., Stockle, C. O., Vaughan, J. K., Chen, Y., Guenther, A., Harrison, J. A., Kalyanaraman, A., Leung, L. R., Perleberg, A. B., Tague, C. L., Yoder, J., Hamlet, A. F., Nijssen, B., Chinnayakanahalli, K., Choate, M. J., Jiang, X., Nelson, R., Yoon, J. H., Yorgey, G. G., Zhu, J., Allen, E., Anderson, S., Malek, K., Nergui, T., Poinsatte, J., Rajagopalan, K., & Reyes, J. (2014).

BioEarth: a regional biosphere-relevant earth system model for understanding the Impacts of agricultural and natural resource management decisions, *Climatic Change,* (under revision).

Allen, E., Kruger, C., Leung, F.Y., & Stephens, J. C. (2013). Diverse Perceptions of Stakeholder Engagement within an Environmental Modeling Research Team. *Journal of Environmental Studies and Science*, 3(3), 541-551. DOI: 10.1007/s13412-013-0136-x.

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., & Mock, S. (2004). Kepler: an extensible system for design and execution of scientific workflows. In *Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on* (pp. 423-424). IEEE.

Byun, D., & Schere, K.L. (2006). Review of the governing equations, computational algorithms, and other components of the models-3 community multiscale air quality (CMAQ) modeling system. Appl. Mech. Rev. 59(1), 51–77.

Christensen, L., Tague, C., & Baron, J. (2008). Spatial patters of simulated transpiration response to climate variability in a snow dominated mountain ecosystem. *Hydrological Processes*, 22(18), 3576–3588. DOI:10.1002/hyp.6961.

Gent, P.R., Danabasoglu, G., Donner, L. J., Holland, M.M., Hunke, E.C., Jayne, S.R., Lawrence, D.M., Neale, R.B., Rasch, P.J., Vertenstein, M., Worley, P.H., Yang, Z.-L., & Zhang, M. (2011). The community climate system model version 4. *Jounal of Climate*, 24(19), 4973–4991.

Guenther, A. B., Jiang, X., Heald, C. L., Sakulyanontvittaya, T., Duhl, T., Emmons, L. K., & Wang, X. (2012). The model of emissions of gases and aerosols from nature version 2.1 (MEGAN2.1): an extended and updated framework for modeling biogenic emissions, *Geoscientific Model Development*, 5(6), 1471–1492. DOI: 10.5194/gmd-5-1471-2012.

Hamlet, A. F., & Lettenmaier, D. P. (1999). Effects of climate change on hydrology and water resources in the Columbia River basin1. *JAWRA Journal of the American Water Resources Association*, 35(6), 1597–1623.

Liang, X., Lettenmaier, D. P., Wood, E. F. & Burges, S. J. (1994). A simple hydrologically based model of land-surface water and energy fluxes for general-circulation models, *Journal of Geophysical Research-Atmospheres*, 99(D7), 14415–14428. DOI: 10.1029/94JD00483.

Liu, M., Rajagopalan, K., Chung, S. H., Jiang, X., Harrison, J., Nergui, T., Guenther, A., Miller, C., Reyes, J., Tague, C., Choate, J., Salathé, E. P., Stöckle, C. O., & Adam, J. C. (2013). What is the importance of climate model bias when projecting the impacts of climate change on land surface processes?, *Biogeosciences Discuss*., 10, 17145-17192. DOI: 10.5194/bgd-10-17145-2013

McPhillips, T., Bowers, S., Zinn, D., & Ludäscher, B. (2009). Scientific workflow design for mere mortals. *Future Generation Computer Systems* 25(5), 541-551.

Rajagopalan, K., Chinnayakanahalli, K. J., Nelson, R., Stöckle, R., Brady, M., Barber, M.E., Yorgey, G., Kruger, C., Dinesh, S. T., Malek. K., Hamlet, A., & Adam, J. C. (2014). (in prep). Integrated modeling to assess the impacts of changes in climate and socio-economics on agriculture in the Columbia River basin. *Water Resources Research*.

Seitzinger, S. P., Harrison, J. A., Dumont, E., Beusen, A. H. W. & Bouwman, A. F. (2005). Sources and delivery of carbon, nitrogen, and phosphorus to the coastal zone: An overview of Global Nutrient Export from Watersheds (NEWS) models and their application, Global Biogeochemical Cycles, 19(4), DOI: 10.1029/2005GB002606.

Skamarock, W. C., Klemp, J. B., Dudhia, J., Gill, D. O., Barker, D. M., Duda, M., Huang, X. Y., Wang, W., & Powers, J. G. (2008). A description of the advanced research WRF version 3. Mesoscale and Microscale Meteorology Division, *NCAR technical note.* NCAR/TN–475+ STR.

Stockle, C. O., Donatelli, M., & Nelson, R. (2003). CropSyst, a cropping systems simulation model, *European Journal of Agronomy*, 18(3-4), 289–307. DOI: 10.1016/S1161-0301(02)00109-0.

Tague, C. L. & Band, L. E. (2004). RHESSys: regional hydro-ecologic simulation system-an object-oriented approach to spatially distributed modeling of carbon, water, and nutrient cycling. *Earth Interactions*, 8(19), 1-42.

Tague, C. L., Choate, J. S., & Grant, G. (2013). Parameterizing sub-surface drainage with geology

to improve modeling streamflow responses to climate in data limited environments. *Earth Syst. Sci.*, 17, 341-354. DOI: 10.5194/hess-17-341-2013.

Tague, C. and Grant, G. (2009). Groundwater dynamics mediate low flow response to global warming in snowdominated alpine regions. *Water Resources Research*, 45(7), DOI: 10.1029/2008WR007179.

Wang, J., Crawl, D., & Altintas, I. (2012). A framework for distributed data-parallel execution in the Kepler scientific workflow system. *Procedia Computer Science*, *9*, 1620-1629.