Scaling GIS analysis tasks from the desktop to the cloud utilizing contemporary distributed computing and data management approaches: A case study of project-based learning and cyberinfrastructure concepts.

T.L. Swetnam[1], J.D. Pelletier,
C. Rasmussen, N.R. Callahan,
N. Merchant, E. Lyons
University of Arizona
1064 E. Lowell St.
Tucson AZ 85721
1-520-621-1052
{tswetnam, jdpellet, crasmuss, nrcallahan, nirav, ericlyons}@email.arizona.edu

M. Rynge
University of Southern California
4576 Admiralty Way #1001
Marina Del Rey, CA 90292
1 (310) 448-8414
rynge@isi.edu

Y. Liu
University of Illinois
Urbana-Champaign Urbana, IL
yanliu@illinois.edu
V. Nandigam
San Diego Supercomputer Center
viswanat@sdsc.edu
C. Crosby
UNAVCO
crosby@unavco.org

ABSTRACT
In this paper we present the experience of scaling in parallel a geographic information system modeling framework to hundreds of processors. The project began in an active learning cyberinfrastructure course which was followed by an XSEDE ECSS effort in collaboration across multiple-institutions.

CCS Concepts
• **Computing methodologies → Massively parallel and high-performance simulations.**

Keywords
CyberGIS; Makeflow; Work Queue; GRASS; GDAL

1. INTRODUCTION
Earth's Critical Zone, the near-surface environment between bedrock and the atmospheric boundary layer, is driven by inputs of mass and energy. Recent improvements in how to quantify the various sources of mass [i.e. biomass and rainfall] and energy [i.e. solar irradiation and the heat content of rainfall] within a common currency have led to major improvements in how we quantify rates of Critical Zone processes including soil formation and landscape erosion [1,2]. A crucial step was the development, by a team of investigators at the NSF-funded Santa Catalina Mountains-Jemez River Basin Critical Zone Observatory (SCM-JRB CZO), of an Effective Energy and Mass Transfer (EEMT) equation that calculates the available free energy for physical and chemical work [units in megajoules per meter square per unit time: MJ m^{-2} s^{-1}] based on available observations of mean monthly air temperature, precipitation, and solar irradiation [3-7]. A key limitation to quantifying EEMT for large areas and across a range of time scales is the heavy computational load associated with computing incident solar irradiation in complex [e.g. mountainous] terrain and the nonlinear relationships among the controlling variables. Small changes in the quantity and timing of solar energy, and lateral redistribution of precipitation as overland and subsurface flow in complex terrain can result in micro-climates with very different rates of Critical Zone evolution.

While the Earth sciences community has made rapid advances through effective utilization of cyberinfrastructure capabilities including large-scale data management, service-oriented architectures and access to high performance computing resources [8], i.e. OpenTopography, CyVerse, and XSEDE, scaling of parallelized analysis tasks onto national-scale HPC/HTC solutions from code initially developed for desktops and clusters poses unique challenges and barriers to adoption. The solar irradiation and EEMT calculations we use here rely on a suite of geospatial calculations from software which has not yet been optimized for HPC. The adaptation of such geospatial tools is a small part of a larger national-scale CyberGIS effort [9].

In Fall 2014 the Applied Cyberinfrastructure Concepts (ACIC) project-based learning course at the University of Arizona developed a parallelized workflow to calculate EEMT on HPC for the SCM-JRB CZO. Students initially formed small teams to separately tackle software installation and data management; later teams were realigned to best fit self-identified skillsets to complete the computational aspect and finalize the workflow. Their method uses the Cooperative Computing Lab's (CCL) Makeflow [10] and Work Queue [11] to launch multiple workers running similar GIS tasks [i.e. daily solar irradiation models] in parallel. The daily solar irradiation outputs are eventually summed into monthly data for use with monthly climate data from DAYMET [12] to derive EEMT (Figure 1).

XSEDE16, July17-21, 2016.

ISBN 978-1-4503-4755-6/16/07…$15.00
DOI: http://dx.doi.org/10.1145/2949550.2949573

[1] Corresponding Author.

In the second phase of the project beginning in spring 2015, the authors worked with XSEDE Extended Collaborative Support Services (ECSS) to port the students' workflow onto Open Science Grid and SDSC Comet where code was further optimized for performance on XSEDE systems. The authors also worked with the OpenTopography team to include this code in the OpenTopography processing workflow via their science gateway through which a user can select topographic data from OpenTopography or upload their own digital elevation and terrain models to produce monthly values of solar irradiation and EEMT.

options. The students identified 34 separate dependencies required to compile and run the open-source community developed GIS software on the UA campus HPC cluster. Their workflow involves shell (.sh) and python (.py) scripts working with Makeflow and Work Queue to variously import the climate variables and a digital elevation model (DEM) or digital surface model (DSM), set the region and projection of the GIS, and compute raster-based calculations without explicitly starting the typical desktop graphic user interface (GUI) (Figure 1). Grid calculations are performed tens of thousands of times across the hundreds of daily and monthly

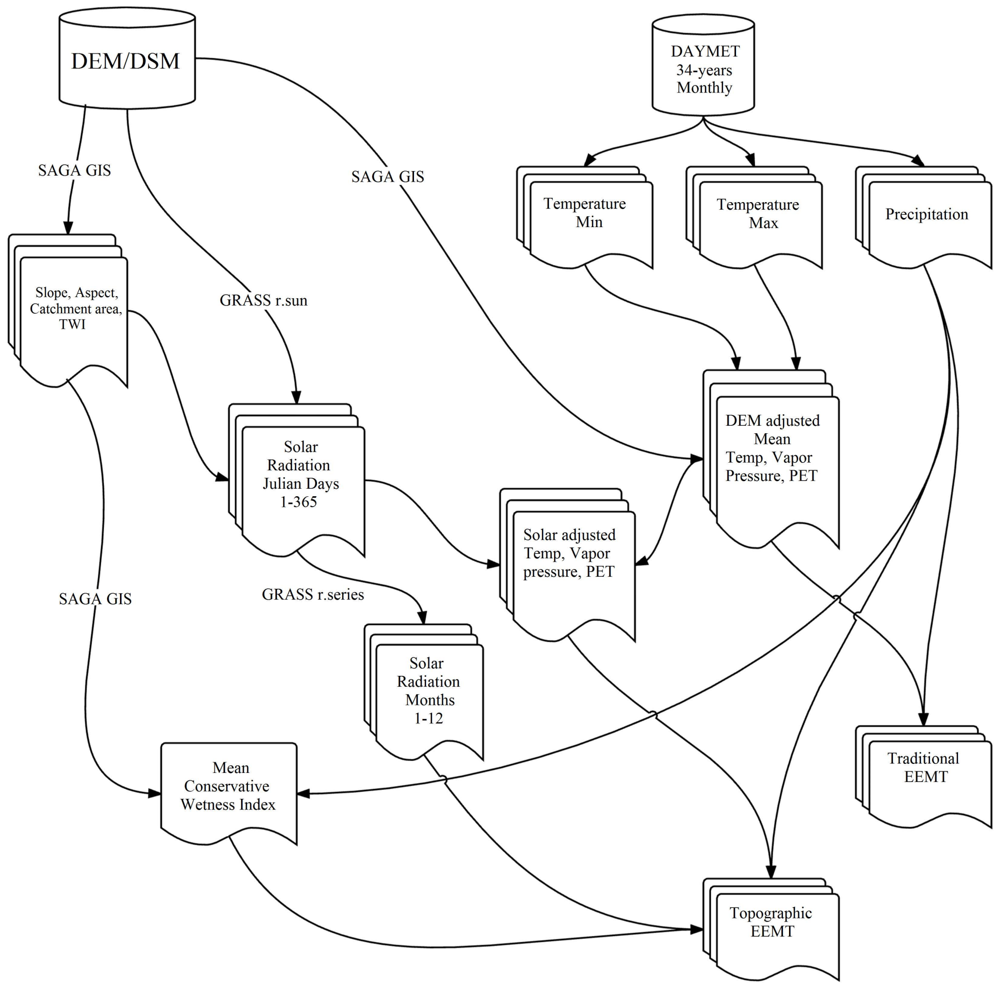

Figure 1. The workflow for generating monthly-scale solar irradiation models and EEMT.

2. COMPUTATIONAL DETAIL

The original workflow was developed by a CZO post-doctoral researcher (Swetnam) and the students of the ACIC course. The students designed appliances, containing all the necessary dependencies, by creating Docker [13] containers. These were deployed on Future Grid [14] using medium-size instances as part of their mid-term projects. Students compared performance and ease of use as part of their evaluation. The students also learned how to manage OpenStack [15] instances as part of the course. While the ease of use in cloud and Docker was valuable, the students were asked to choose a pragmatic long term solution to allow the researcher a sustained use of the students' workflow, this limited the solution to the UA campus HPC, and a start-up allocation on XSEDE resources, neither of which include cloud raster surfaces, the final output layers are packaged as a single output file or directory. By parallelizing the workflow for the daily rasters the overall calculation times can be reduced by over two orders of magnitude.

2.1 Parallelizing GIS modules developed for desktop computing

The Open Source Geospatial Foundation (OSGeo) fosters community driven global adoption of GIS technology. Within the OSGeo software stable are the Geospatial Data Abstraction Library (GDAL) [16], SAGA-GIS [17], and GRASS GIS [18,19] that are used for data management and analysis, image processing, and spatial modelling.

Optimization and scalability of the workflow depends upon the tasks that can run in parallel process, e.g. the solar irradiation model which can use up to 365 workers in the case of Sol and up to 730 workers for EEMT using GRASS *r.sun* [20] (Figure 2). Later processes include GRASS and SAGA-GIS raster calculations which generate monthly solar irradiation maps, topographic wetness indices, and calculate derived input parameters for EEMT at monthly intervals with DAYMET data. These processes require up to 816 workers for a 34-year run. Larger raster surfaces increase processing times due to their slower I/O readability. The speed at which the models are generated is also dependent upon the solar time-step interval, which can be varied depending upon the question. For example, modelling solar irradiation at high spatial resolution, e.g. a 10 cm^2 lidar DSM which includes vegetation heights in the elevation profile, executed across several hundred hectares at a 3 minute time-step will precisely integrate the minute variations in the forest-canopy modified light regime, whereas modelling regional landscapes, e.g. a 90 m^2 NASA Space Shuttle SRTM [21,22] DEM that is several hundred km wide, can be accomplished using a 15-minute interval.

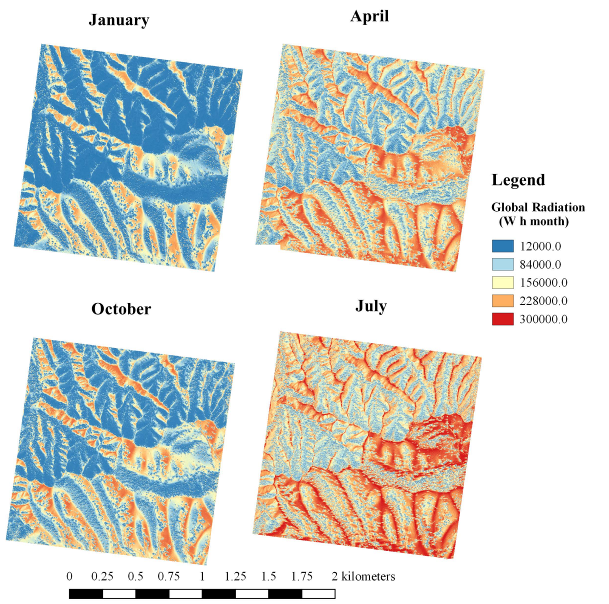

Figure 2: Example of monthly solar irradiation (W h $month^{-1}$) computed at 30 cm resolution over 1.6 km^2 on Open Science Grid. Input was a 24 Mb digital surface model derived from aerial lidar which includes the vegetation as a surface feature. For an animation click link here: https://www.youtube.com/watch?v=BKCPsZBsytk

2.2 Makeflow and Work Queue

CCL Makeflow [10] is a workflow engine for executing large complex workflows on clusters, clouds, and grids. Makeflow can handle simple or complex tasks and is portable to different systems without modification; further it does not require a distributed file system. Work Queue [11] is a framework for managing large master-worker applications across clusters, clouds, and grids. Users define tasks, submit them to a queue, and wait for them to complete; the workers communicate with the master to execute the tasks and transfer data. Both Makeflow and Work Queue are robust and highly fault tolerant and can recover from a variety of failures. Makeflow and Work Queue provide not only a data management solution in this case, but also a separation between workflow development/management on the submit node and the execution on XSEDE resources.

Workflows in Makeflow are defined in a format similar to the UNIX Make format. Each rule has a section listing dependencies (input files), a section listing output files, and one or more commands to execute in order to generate the output files. For example, a rule to calculate the daily irradiation and hours of sun can look like (on a single command line):

```
global/daily/total_sun_day_1.tif
insol/daily/hours_sun_day_1.tif : rsun.sh
mcn_p333m_converted.tif /bin/bash rsun.sh
dem_10m_converted.tif -d 1 -D
```

where the process calls rsun.sh which generates the daily output rasters: total_sun_day_1.tif and hours_sun_day_1.tif.

2.3 Work Queue and Batch Systems

Currently, the workflow can be run on one or more cores up to the total number of tasks which can run simultaneously (Table 2). This involves up to 365 tasks for the Sol module and 815 tasks for EEMT when run across the entire DAYMET 34-year monthly record [12]. Each task can be run by a separate worker, or as workers complete their task they can be reassigned if there are still queued jobs. Otherwise, they are shut down, releasing the computational resource for others to use.

Provisioning of the Work Queue workers is based on demand from currently running workflows. If it is determined the set of running workflows have enough pending tasks that cannot be met by the currently running set of workers, more workers are submitted to the appropriate XSEDE resource. Currently, the workflows are locked to a particular resource based on the input size. Smaller inputs can be handled by the Open Science Grid, while larger ones require more memory and go to SDSC Comet. The provisioning script also considers a small set of attributes, such as total running workers, to make sure we are not over-provisioned.

The Work Queue workers are also configured differently on Open Science Grid and SDSC Comet. The former being primarily a high throughput system, each worker is set up to claim a single core and 2 GBs of RAM. On the latter, more like a traditional high performance cluster, each worker is set up to claim a full node, which is 24 cores and 128 GB RAM. Work Queue partitions the available cores and memory based on the Makeflow task requirements.

Table 2: Processing speed improvements for Sol / EEMT given different DEM/DSM input file sizes on Comet.

DEM	**Single core**	**Single node (24 cores)**	**All jobs parallel**	**Wall time**	**Charged**
Mb	hours				SU
0.44	127.5	0.44	0.31	0.45	8
1.7	145	6	0.35	0.45	94
4.5	173	7.2	0.42	1.8	98
36.5	520	21.6	1.25	7.21	1186
160.5	6400	400	32	45.2	6400

2.4 OpenTopography and the Opal Toolkit

The NSF funded OpenTopography project based at the San Diego Supercomputer Center (SDSC) at University of California, San Diego, was initiated in 2009 to democratize access to Earth science oriented topography data and processing tools. It utilizes cyberinfrastructure, including large-scale data management, high-performance computing, and service-oriented architectures (SOA) to provide efficient Web based access to large, high-resolution topographic datasets [23]. The OpenTopography SOA supports a distributed computational environment and enables service interoperability. Users can both access and process data using various tools in a highly customizable workflow to generate derived products for their applications.

All data access functions and processing algorithms are packaged as web services across various cluster resources using the open source Opal Toolkit developed at SDSC [24]. The Opal Toolkit provides mechanisms to manage and track job submissions as well as real time monitoring of job and system status. Once scientific applications running on cluster and Grid resources are wrapped as web services using Opal, they can be made accessible within the OpenTopography workflow to the wider research community as new processing resources.

Two models are encapsulated as Opal2 Web Services [24]: Sol and EEMT. These services can be called as part of the point cloud processing workflow via the OpenTopography portal and made available to the end users. These two Web services are hosted on

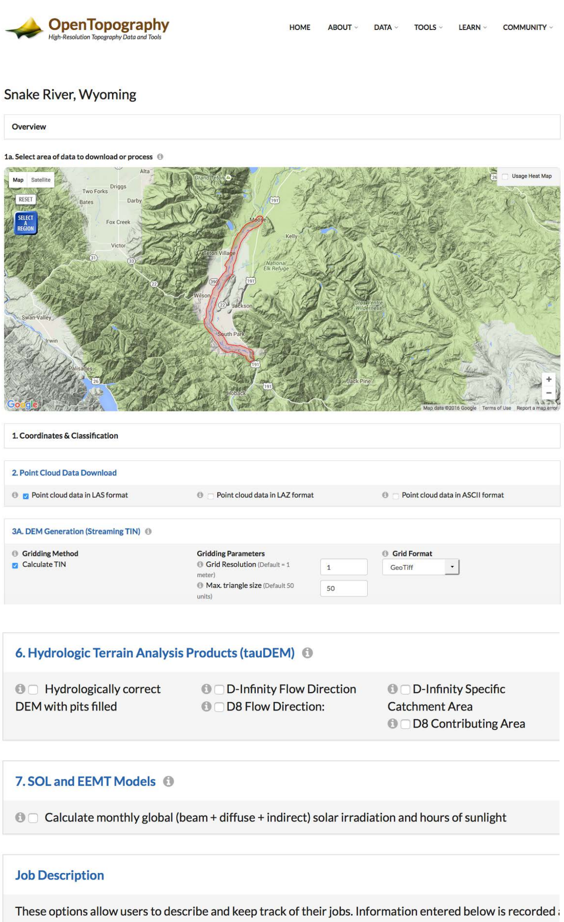

Figure 3: Example of the OpenTopography portal (Beta) with SOL and EEMT options embedded.

the XSEDE gateway host pool. Figure 4 shows the Opal2 service definition which defines both a SOAP Web service and user interface components. Since the computation occurs at Open Science Grid or Comet, the service itself acts as a single-process job proxy.

```
<appConfig xmlns="http://nbcr.sdsc.edu/opal/types"
          xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <metadata  appName="solotcomet">
   <usage><![CDATA[Compute Sol model using online DEM
e.g. from OpenTopo
                 for comet.sdsc.edu job submission
]]></usage>
   <info xsd:type="xsd:string">
      <![CDATA[ /home/eemt/realgit/sol-ot-comet/opal2-
wrapper-urldem [DEM_url] ]]>
   </info>
   <types>
       <untaggedParams>
           <param>
               <id>DEMUrl</id>
               <paramType>STRING</paramType>
               <ioType>INPUT</ioType>
               <required>true</required>
               <textDesc><![CDATA[Input DEM URL to Sol
model.]]></textDesc>
           </param>
           <param>
               <id>outputzip</id>
               <paramType>FILE</paramType>
               <ioType>OUTPUT</ioType>
               <required>true</required>
               <default>output.zip</default>
               <textDesc><![CDATA[Output zip file
name]]></textDesc>
           </param>
       </untaggedParams>
   </types>
 </metadata>
 <binaryLocation>/home/eemt/realgit/sol-ot-comet/opal2-
wrapper-urldem</binaryLocation>
 <parallel>false</parallel>
</appConfig>
```

Figure 4. Opal2 service definition for Sol model.

3. DISCUSSION

The initial collaboration between a CZO scientist, who had little substantive experience working in a shell environment or writing code, and the ACIC students [computer science, domain science, and engineering majors], who had little or no experience in virtualization and HPC, and no experience working with GIS, revealed the vital importance of clear communication and establishing realistic expectations. Most of the students' time was spent tracking down dependencies and getting the GIS software to compile properly on the HPC, rather than benchmarking analyses and looking for optimizations. Remarkably the students had almost no problem understanding the computational aspects of the GIS workflow. Following brief class lectures, student teams were given time to present their group's weekly project related problems to the rest of the class which enabled resolutions quickly when another group suggested a solution. When a team was found to be lacking in technical expertise they were able to exchange team members or to condense two teams into one to finish their work.

The collaboration of ECSS with the CZO members was critical for ensuring that the workflow could be moved across platforms. Because we are using Makeflow and Work Queue, the workflow can easily be moved and executed on any available HTC or HPC resource. The limiting factor is installing the software stack. On Open Science Grid, the software stack was installed on the FUSE/HTTP based, read-only distributed CVMFS (CernVM File System) [25], which allows access to the software from the majority of Open Science Grid compute resources. We were pleasantly surprised when SDSC decided to mount the same file system across

Comet, providing us direct access to the already installed software stack. On clusters which do not have CVMFS installed, the software stack would have to be recompiled.

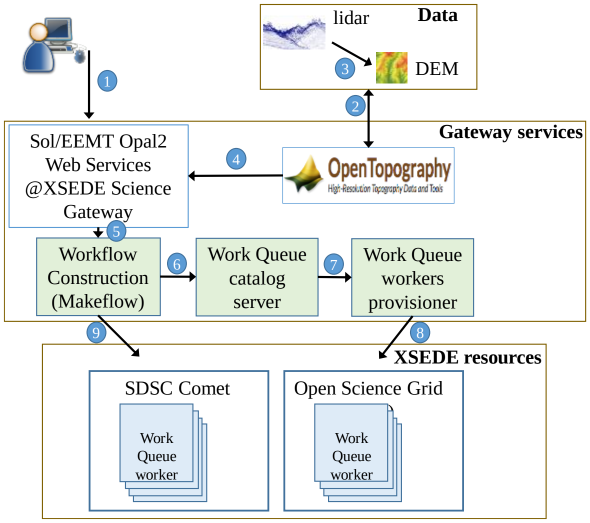

Figure 5: Architecture overview. 1) A user accesses Sol/EEMT Opal2 web interface or the OpenTopo gateway using a web browser; 2) and 3) OpenTopo generates input DEM data from lidar or direct DEM sources; 4) OpenTopo invokes Sol/EEMT Opal2 services with input DEM URL; 5) Sol/EEMT Opal2 services create a new Makeflow workflow; 6) The Makeflow is registered into the Work Queue catalog server; 7) Periodically, a provisioner checks the catalog server; 8) if there is demand, the provisioner submits new Work Queue workers to SDSC Comet or Open Science Grid; and 9) The Work Queue workers check the catalog server for Makeflow instances to attach themselves to. Once attached, Makeflow tasks starts executing.

Our workflow was optimized to run on XSEDE Open Science Grid and SDSC Comet (Figure 5). A science gateway was developed which interfaces with OpenTopography so users can produce their own solar and EEMT models on demand. Modeling frameworks like EEMT and Sol which utilize open-source GIS and local to national scale data have broad applicability: enabling production of high resolution surface models of solar irradiation for climate or ecosystem modeling or energy development; hydrological modelling and estimating landscape carbon reservoir balances is also readily possible. Future plans with the existing software stack include compiling GRASS 7 with OpenCL to utilize Comet's GPU cluster. GPU processing has the potential to greatly reduce raster I/O bottleneck times [26]; which for the *r.sun* calculations could drop processing times by over an order of magnitude and enable the calculation of much larger raster surfaces.

The current GIS software stack compiled on CVMFS makes available any of the other geospatial analysis tools in GDAL, GRASS, and SAGA-GIS to the earth science and XSEDE community. When paired with Makeflow, users can distribute hundreds to thousands of jobs across the XSEDE ecosystem; currently these include only the two configured systems, but in the future any system could be mounted or have the software stack installed.

It is important to note the success of the ACIC project-based class. Through close interactions between students, instructors, scientists, and cyberinfrastructure projects, the class has delivered a real-world solution to a computational bottleneck that will soon be available for general use through OpenTopography and US national cyberinfrastructure resources.

4. ACKNOWLEDGEMENTS

J.D. Pelletier, T. Swetnam, and N.R. Callahan were supported by NSF # EAR-1331408. N. Merchant and E. Lyons were supported by NSF # DBI-1265383. ECSS-XSEDE supported by NSF # ACI-1053575. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1053575.

5. REFERENCES

[1] Chorover, J. P., et al., 2011. How water, carbon, and energy drive critical zone evolution: The Jemez--Santa Catalina Critical Zone Observatory. Vadose Zone Journal, 10(3), 884-899. doi:10.2136/vzj2010.0132

[2] Pelletier, J. D., et al., 2013. Coevolution of nonlinear trends in vegetation, soils, and topography with elevation and slope aspect: A case study in the sky islands of southern Arizona. Journal of Geophysical Research: Earth Surface, 118(2), 741-758.

[3] Rasmussen, C., Southard, R. J., & Horwath, W. R. 2005. Modeling energy inputs to predict pedogenic environments using regional environmental databases. Soil Science Society of America Journal, 69(4), 1266-1274.

[4] Rasmussen, C., and N.J. Tabor, 2007. Applying a quantitative pedogenic energy model across a range of environmental gradients. Soil Science Society of America Journal, 71(6), 1719-1729. doi:10.2136/sssaj2007.0051

[5] Rasmussen, C., et al. 2011. An open system framework for integrating critical zone structure and function. Biogeochemistry, 102(1-3), 15-29. doi: 10.1007/s10533-010-9476-8

[6] Rasmussen, C., & Gallo, E.L.. Technical Note: A comparison of model and empirical measures of catchment-scale effective energy and mass transfer. Hydrology and Earth System Sciences 17, no. 9 (2013): 3389-3395.

[7] Rasmussen, C., et al., .2015. Quantifying Topographic and Vegetation Effects on the Transfer of Energy and Mass to the Critical Zone. Vadose Zone doi:10.2136/vzj2014.07.0102

[8] Youn, C., Nandigam, V., Phan, M., Tarboton, D., Wilkins-Diehr, N., Baru, C., Crosby, C., Padmanabhan, A., Wang, S. (2014) Leveraging XSEDE HPC resources to address computational challenges with high-resolution topography data, In Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment (XSEDE '14), ACM, New York, NY, USA, Article 59, 2014

[9] Wang, S., et al. (2013) CyberGIS software: a synthetic review and integration roadmap. International Journal of Geographical Information Science 27, no. 11 (2013): 2122-2145.

[10] Albrecht, M., P. Donnelly, P. Bui, and D. Thain. 2012. Makeflow: A Portable Abstraction for Data Intensive Computing on Clusters, Clouds, and Grids. In Workshop on Scalable Workflow Enactment Engines and Technologies (SWEET) at ACM SIGMOD.

[11] Bui P., D. Rajan, B. Abdul-Wahid, J. Izaguirre, and D. Thain. 2011. Work Queue + Python: A Framework For Scalable Scientific Ensemble Applications. In Workshop on Python for High Performance and Scientific Computing (PyHPC) at the ACM/IEEE International Conference for High Performance Computing, Networking, Storage, and Analysis (Supercomputing).

[12] Thornton, P.E., M.M. Thornton, B.W. Mayer, N. Wilhelmi, Y. Wei, R. Devarakonda, and R.B. Cook. 2014. Daymet: Daily Surface Weather Data on a 1-km Grid for North America, Version 2. Data set. Available on-line [http://daac.ornl.gov] from Oak Ridge National Laboratory Distributed Active Archive Center, Oak Ridge, Tennessee, USA.

[13] Merkel, D. 2014. Docker: lightweight linux containers for consistent development and deployment. Linux Journal, 2014(239), 2.

[14] Fox, G., von Laszewski, G., Diaz, J., Keahey, K., Fortes, J., Figueiredo, R., ... & Grimshaw, A. 2013. FutureGrid-a reconfigurable testbed for Cloud, HPC, and Grid Computing. Contemporary High Performance Computing: From Petascale toward Exascale, Computational Science. Chapman and Hall/CRC.

[15] Sefraoui, O., Aissaoui, M., & Eleuldj, M. 2012. OpenStack: toward an open-source solution for cloud computing. International Journal of Computer Applications, 55(3).

[16] GDAL. 2015. GDAL - Geospatial Data Abstraction Library: Version 1.4.3, Open Source Geospatial Foundation, http://gdal.osgeo.org

[17] Conrad, O., Bechtel, B., Bock, M., Dietrich, H., Fischer, E., Gerlitz, L., Wehberg, J., Wichmann, V., and Böhner, J. (2015): System for Automated Geoscientific Analyses (SAGA) v. 2.1.4, Geosci. Model Dev., 8, 1991-2007, doi:10.5194/gmd-8-1991-2015.

[18] Neteler, M., Bowman, M.H., Landa, M., Metz, M., 2012. GRASS GIS: A multi-purpose open source GIS. Environ Model Soft 31, 124–130.

[19] GRASS Development Team, 2015. Geographic Resources Analysis Support System (GRASS) Software, Version 6.4. Open Source Geospatial Foundation.http://grass.osgeo.org

[20] Hofierka, J., T. Huld, T. Cebecauer, and M. Šúri, 2007. Open source solar radiation tools for environmental and renewable energy applications. Environmental Software Systems, 448. (doi: 10.13140/2.1.3773.6001

[21] Jarvis, A., Reuter, H. I., Nelson, A., & Guevara, E. (2008). Hole-filled SRTM for the globe Version 4. available from the CGIAR-CSI SRTM 90m Database (http://srtm. csi. cgiar. org).

[22] Robinson, N., Regetz, J., and Guralnick, R. P. 2014. EarthEnv-DEM90: A nearly-global, void-free, multi-scale smoothed, 90m digital elevation model from fused ASTER and SRTM data. ISPRS Journal of Photogrammetry and Remote Sensing, 87:2014, 57-67.

[23] Krishnan, S., Crosby, C.J., Nandigam, V., Phan, M., Cowart, C., Baru, C., Arrowsmith, R., (2014) OpenTopography: a services oriented architecture for community access to LIDAR topography, In Proceedings of the 2nd International Conference on Computing for Geospatial Research & Applications (COM.Geo '11), ACM, New York, NY, USA, 2014, Article 7, 2014

[24] Krishnan, S. 2009. Design and Evaluation of Opal2: A Toolkit for Scientific Software as a Service. In Services-I, 2009 World Conference on, pp. 709-716. IEEE, 2009.

[25] Buncic, P., Sanchez, C. A., Blomer, J., Franco, L., Harutyunian, A., Mato, P., & Yao, Y. (2010). CernVM–a virtual software appliance for LHC applications. In Journal of Physics: Conference Series (Vol. 219, No. 4, p. 042003). IOP Publishing.

[26] Steinbach, M., Hemmerling, R., 2011. Accelerating batch processing of spatial raster analysis using GPU. Computers & Geosciences.