Integrating hydrologic modeling web services with online data sharing to prepare, store, and execute hydrologic models

Tian Gan[a,b], David G. Tarboton[a], Pabitra Dash[a], Tseganeh Z. Gichamo[a], Jeffery S. Horsburgh[a]

[a] Department of Civil and Environmental Engineering and Utah Water Research Laboratory, Utah State University, 8200 Old Main Hill, Logan, UT 84322-8200, USA

[b] Corresponding author: Institute of Arctic and Alpine Research, University of Colorado, Campus Box 450, Boulder, CO 80309-0450 USA. Phone (+1) 435-754-9720. Email: gantian127@gmail.com

Abstract

Web based applications, web services, and online data and model sharing technology are becoming increasingly available to support hydrologic research. This promises benefits in terms of collaboration, computer platform independence, and reproducibility of modeling workflows and results. In this research, we designed an approach that integrates hydrologic modeling web services with an online data sharing system to support web-based simulation for hydrologic models. We used this approach to integrate example systems as a case study to support reproducible snowmelt modeling for a test watershed in the Colorado River Basin, USA. We demonstrated that this approach enabled users to work within an online environment to create, describe, share, discover, repeat, modify, and analyze the modeling work. This approach encourages collaboration and improves research reproducibility. It can also be adopted or adapted to integrate other hydrologic modeling web services with data sharing systems for different hydrologic models.

Key words: hydrologic modeling, data sharing, reproducibility, web services, HydroShare

Software availability

The software created in this research is free and open source as part of the larger HydroShare software repository. The HydroShare software repository is managed through GitHub and is available at https://github.com/hydroshare/hydroshare. The HydroShare REST API Python Client repository is available at https://github.com/hydroshare/hs_restclient.The Utah Energy Balance (UEB) web app software is available in GitHub at https://github.com/gantian127/tethysapp-ueb_app. A snapshot of the code for the app at the time of this writing was also published in Zenodo (Gan et al., 2020). Code for the HydroDS modeling web services is available at https://github.com/CI-WATER/Hydro-DS.

1 Introduction

Hydrologic modeling is essential as a guide to formulating strategies for water resources management or as a tool of scientific inquiry (Dingman, 2008). However, hydrologic modeling research presents a number of challenges. Modelers need to discover and collect data from various sources (Archfield et al., 2015) and use it to prepare model inputs. Model input preparation can be time consuming and may require a substantial learning curve, especially where programming is needed (Miles, 2014). Furthermore, modelers may need to access high performance computing (HPC) resources to effectively handle large scale or complicated hydrologic model simulations (Kumar et al., 2008; Laloy and Vrugt, 2012). Curating and sharing modeling datasets and metadata publicly is also important to improving reproducibility (Demir and Krajewski, 2013; Archfield et al., 2015; Hutton et al., 2016; Essawy et al., 2018; Chuah et al., 2020). Collaboration among people from various disciplines and areas is one of the key factors in catalyzing new research findings (Silliman et al., 2008). Computer systems as infrastructure (cyberinfrastructure) that enable collaboration have the potential to significantly advance environmental modeling research.

With the development of web technologies and standards, one promising direction is to provide web services or web applications to help people overcome these hydrologic modeling challenges and improve the efficiency of hydrologic modeling work. There are a number of systems that help acquire or preprocess datasets as model input files for hydrologic models (Leonard and Duffy, 2013; Billah et al., 2016; Gichamo et al., 2020). For instance, Billah et al. (2016)

developed web services that help to automate the grid data pre-processing workflow for preparation of model inputs for the Variable Infiltration Capacity (VIC) model (Liang et al., 1996). The workflow includes the information that allows others to independently reproduce the model results and acts as a means for documenting the steps used to create model input files. Some systems focus on simulation using a specific hydrologic model while others couple different hydrologic models to simulate integrated hydrologic processes. For example, SWATShare (Rajib et al., 2016) established a collaborative environment to publish, share, discover, and download Soil and Water Assessment Tool (SWAT) models. This cyberinfrastructure also supports SWAT model calibration running on HPC resources and visualization of model outputs. Souffront Alcantara et al. (2019) developed a large-scale streamflow prediction system and made the results available using a hydrologic modeling as a service approach (HMaaS). This approach improves accessibility to modeling results to support decision making for developing countries that may have limited hydrologic modeling capabilities. The Community Surface Dynamics Modeling System (CSDMS) (Peckham et al., 2013) created an environment that promotes the sharing, reuse, and integration of open-source modeling software. Many models in CSDMS are installed and maintained on its high-performance cluster. CSDMS members can access these resources and integrate them for complex model simulation. In addition, some systems support both model input preparation and simulation to facilitate modeling work. The AWARE framework, which is described as "A tool for monitoring and forecasting Available WAter REsource in mountain environments," was developed to offer online geospatial processing services and other tools to help users monitor and forecast water resources in Alpine regions (Granell et al., 2010). Sun (2013) migrated an environmental decision support system from the traditional server-client model to Google cloud-computing services with Google Drive holding some of the data to enable collaborative participatory modeling. Later, recognizing the computational demands of physically based hydrologic models in a web-based environment, Sun et al. (2015) explored the use of meta models to support water quality management and decision making. A similar approach was also applied to metamodeling of geological carbon sequestration (Sun et al., 2018). These prior approaches highlight the importance of easy to use server or web-based methods for collaborative and reproducible hydrologic modeling similar to those that are addressed in this paper.

Although these web services or web applications improve the efficiency of hydrologic modeling work, they do have limitations. One limitation is that they may require programming to use the web services and thus be difficult to use for those without the required programming skills or knowledge. Another limitation is related to the reproducibility of the modeling work, an essential principle in scientific research (Hutton et al., 2016). The model input/output files and the programming code for data processing and analysis are often not well curated and shared with the public (Stagge et al., 2019). This hinders the ability for the modeling community to reproduce and verify the modeling work and reuse the results.

In this research, our goal was to integrate hydrologic modeling web services with a data sharing system to provide web-based simulation that improves the reproducibility of the modeling work and the usability of these web services. We define web-based simulation as the use of web technologies to develop, execute, and analyze simulation models with the web browser playing an active role in the modeling process, either as a graphical user interface or as a container for the simulation engine (Byrne et al., 2010; Walker and Chapra, 2014). We sought to provide an online environment within which users can prepare model input, execute the model, share and analyze the results, and repeat or modify the modeling work for collaboration.

To achieve this goal, we designed an approach for system integration. The general idea was to add a browser-based graphical user interface (GUI) for the modeling web services to make them easy to use without programing knowledge and to take advantage of a data sharing system that provides advanced data curation and management capability beyond existing modeling web services. As a case study, we used this approach to integrate two example systems, HydroDS and HydroShare, to support web-based simulation for a snowmelt model. The functionality implemented was evaluated using snowmelt modeling use cases in the Animas watershed within the Colorado River Basin, USA. HydroDS (Gichamo et al., 2020) is a set of web-based, hydrological data services that provides access to input datasets and server side data processing tools for distributed hydrologic models such as the Utah Energy Balance (UEB) snow model (Tarboton and Luce, 1996). HydroDS includes a Python client library that makes it easy to use the hydrological data services in a Python programing environment to automate data processing workflows. Model input and output files can be temporarily saved in the HydroDS system and are then downloadable for further analysis. HydroShare is a hydrologic information system and repository for sharing hydrologic data, models, and analysis tools (Tarboton et al., 2014). In HydroShare, the hydrologic datasets or models can be shared as resources that can be published, collaborated around, annotated, discovered, and accessed (Horsburgh et al., 2015). Aside from the data sharing functions, HydroShare also provides a representational state transfer (REST) application programming interface (API) and corresponding Python client library that enables other systems including web applications (or apps), to interact with HydroShare.

The primary contribution of this work is that it demonstrates how the bar for collaborative and reproducible hydrologic modeling can be lowered through facilitating and better enabling the use of web-based hydrologic modeling. This is achieved through GUI and Python Notebook based web apps that serve as interfaces to web services and are underpinned by a data repository that enables users to collaborate and share their results in a reproducible way. We demonstrate how the capability of data and modeling services can be extended by providing a web browser based GUI that reduces the programming required for input data preparation and model simulation. This can make the modeling web services available to a broader user community for those who have limited programming skills. We also demonstrate how integration of modeling web services with a data sharing system can improve the accessibility of modeling work by enabling the research community to more easily discover and access modeling workflows for reuse and collaboration. With these new capabilities, this approach can facilitate research validation and experimentation in an online environment without using modelers' local computing or data storage resources. Additionally, this approach can be adopted or adapted to integrate other hydrologic modeling web services with data sharing systems for various hydrologic models to support reproducible modeling research.

In Section 2, we introduce the general architecture design and the case study that uses this approach to integrate the two example systems (HydroDS and HydroShare). In Section 3, we present the case study results, which describes the integration of the functionality implemented and tested for snow modeling use cases. Section 4 presents discussion and Section 5 summary and conclusions.

2 Methods

2.1 General approach

The purpose of the system integration presented here is to support web-based simulation that: 1) provides easy access through a web browser to the modeling web services, 2) provides online data curation and sharing to support management and reuse of the modeling work, and 3) avoids the complexity of changing existing systems to achieve system integration.

Based on these criteria, we designed a three-layer web service based architecture to integrate hydrologic modeling web services with a data sharing system. This architecture includes a user interface layer, a data service layer, and a data storage layer (Figure 1). The user interface layer can be a web app that provides a web browser based user interface for modelers to use the hydrologic modeling web services without programming. This user interface layer web app can be hosted on web servers separate from the data service or the data storage layers and interact with them through REST APIs. This design decouples the user interface web app from the other two layers and avoids significant changes in the existing systems. The data service layer is a system that hosts hydrologic data and modeling web services. This layer can receive web requests from the user interface layer to prepare model input datasets or execute hydrologic models. The hydrologic data is the general use large data, and, in our implementation, contiguous US wide data used for model input preparation (e.g., climate, land cover, and terrain input data). The data is staged in this layer to enable high availability and performant data access in responding to web service requests. The data storage layer is a data sharing system for storing and sharing the data specific to users' modeling work. This design uses the emerging functionality of data sharing systems to avoid additional software development work and provide the storage and data curation needs for systems that host hydrologic modeling web services.

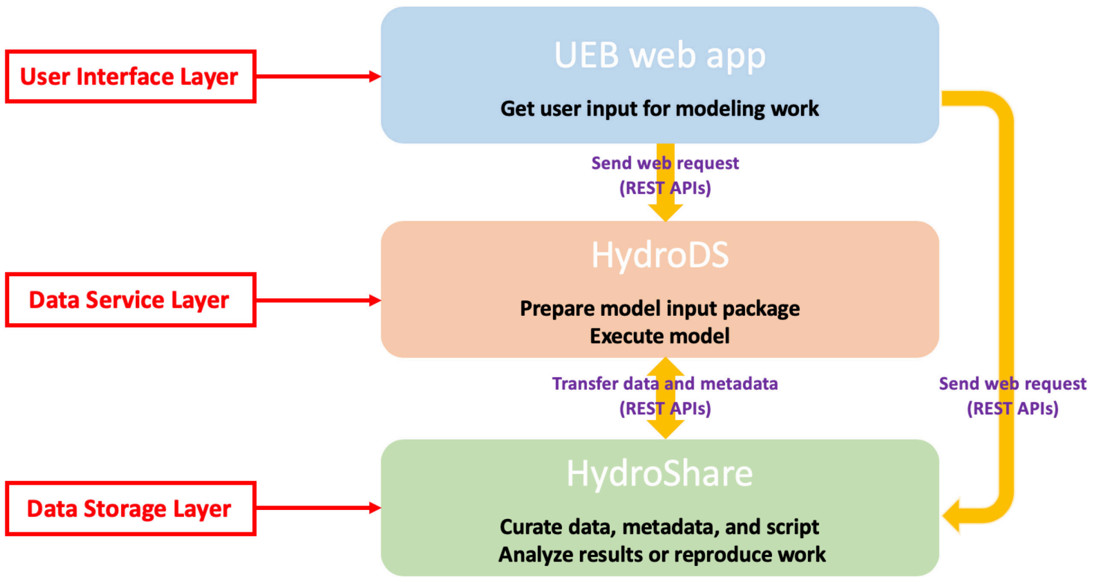


Figure 1 A three-layer web service based architecture to integrate hydrologic data and modeling web services (e.g., HydroDS) with a data sharing system (e.g., HydroShare).

2.2 Case study design

Our case study was designed to use this general approach and integrate example systems to test if the system integration can support web-based simulation to improve research reproducibility and reduce the need for coding to use the modeling web services. We used the three-layer architecture to integrate HydroShare and HydroDS, and designed use cases to evaluate the application of implemented functionality for snowmelt modeling in a test watershed. We chose these systems because: 1) they represent the general functionality of hydrologic data and modeling web services (HydroDS) and data sharing systems (HydroShare); and 2) the authors have access to both systems and are thus able to work on them for integration. In the following, we first provide background on these systems and then present the case study design.

HydroDS is a system that provides web based data services to simplify model input preparation for distributed hydrologic models (Gichamo et al., 2020). Modelers can use these web services to create model input files and save the time and energy often spent collecting datasets from multiple sources and developing code to preprocess the data into required file formats. For example, Table 1 shows the UEB model input variables and the major HydroDS Python client functions used to call the respective web services to prepare them. The UEB model requires climate, terrain, and canopy datasets as model input and uses Network Common Data Form (NetCDF; http://www.unidata.ucar.edu/software/netcdf/) as its input/output file format. Modelers can use HydroDS functions to write data processing code for input preparation. HydroDS datasets are processed and stored in GeoTiff, shapefile, and NetCDF formats based on the functions that generate the datasets. Additionally, HydroDS data conversion functions help process UEB inputs in NetCDF format.

Table 1 UEB model input variables and HydroDS Python client functions for input preparation.

Input type	**Specific variables**	**Major Python client functions for preparation**

Model domain	Watershed grid	subset_raster()
		delineate_watershed()
		raster_to_netcdf()

Terrain	Slope
	Aspect
		create_raster_aspect()
		create_raster_slope()
		raster_to_netcdf()

Canopy	Canopy cover
	Canopy height
	Leaf area index
		project_clip_raster()
		get_canopy_variable()

Climate	Incoming shortwave radiation
	Minimum air temperature
	Maximum air temperature
	Air vapor pressure
	Precipitation
		subset_netcdf()
		concatenate_netcdf()
		subset_netcdf_by_time()
		project_subset_resample_netcdf()

The HydroDS system was built using Django, an open-source Python web framework for web development (https://www.djangoproject.com/) (Figure 2). Several open-source libraries and software programs for processing NetCDF, shapefile, and raster datasets were installed in HydroDS, such as NetCDF4 Python module, NCO (Zender, 2008), GDAL (http://www.gdal.org/), and TauDEM (Tarboton, 1997). They were used to provide the required data management and processing capabilities. Additionally, datasets from multiple sources for input preparation were also stored in this system, including the National Elevation Dataset (NED) (https://www.usgs.gov/), National Land Cover Datasets (Homer et al., 2015), and Daymet climate data (Thornton et al., 2016).

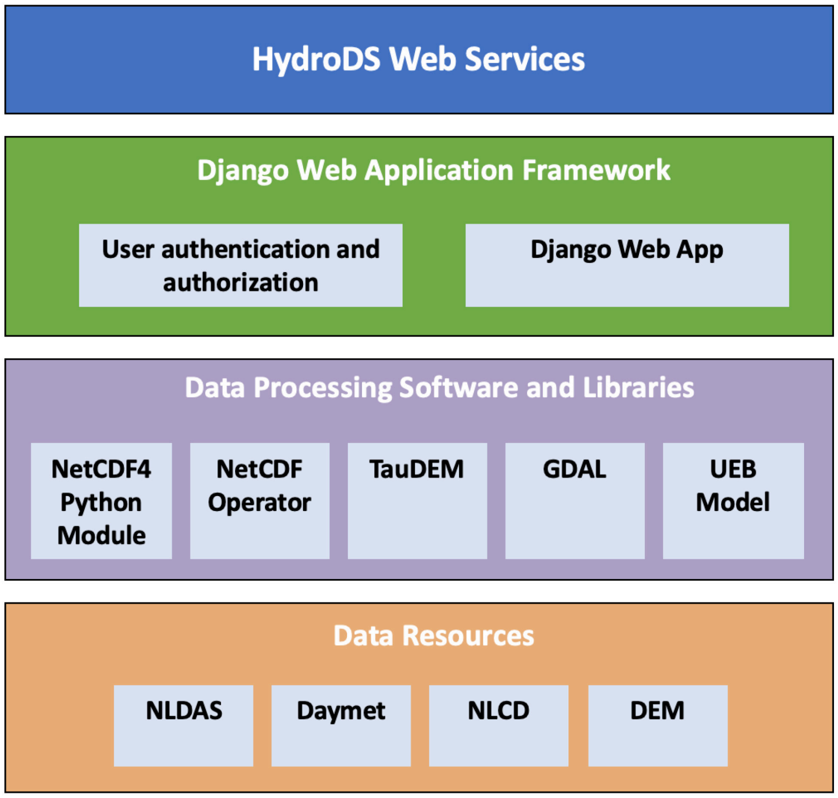


Figure 2 The HydroDS system architecture.

HydroShare's system architecture (Figure 3) is centered on several open source components (Heard et al., 2014). The major components include Django and iRODS (http://iRODS.org/). Django provides the functionality that was used to build the web user interface to help users manage their shared datasets or models. iRODS is open source data management software that is used for data storage and access control. Aside from data sharing functionality, web apps hosted on other web servers can also connect to HydroShare. For example, the Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) JupyterHub web app (http://jupyter.cuahsi.org) was developed by others (Castronova, 2016) and connected to HydroShare. This web app was built with the JupyterHub software stack (https://jupyter.org/hub) and configured with many scientific Python libraries and tools. It provides an online programming environment where researchers can load data from HydroShare and develop Python code for data analysis and visualization. Another example platform for web apps is the HydroShare Tethys Apps portal (https://apps.hydroshare.org/apps/), a system established by the HydroShare team to host multiple web apps and interact with HydroShare resources (Fig. 3). This web portal was built using the Tethys platform (Swain et al., 2016) that includes software and development kits to simplify and reduce the programming skills needed to develop web apps for environmental data visualization, analysis, and modeling applications. In order to enable information exchange between HydroShare and the HydroShare Tethys Apps portal, Oauth (https://oauth.net/) is used to support user authentication and authorization, and the HydroShare REST API Python client "hs_restclient" (https://github.com/hydroshare/hs_restclient) is used to transfer the datasets between the two systems.

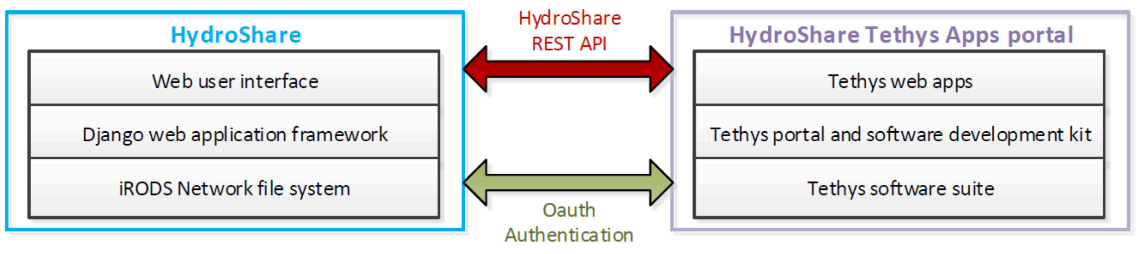

HydroShare
Web user interface
Django web application framework
iRODS Network file system
HydroShare REST API
Oauth Authentication
HydroShare Tethys Apps portal
Tethys web apps
Tethys portal and software development kit
Tethys software suite

Figure 3 System architecture of HydroShare and HydroShare Tethys Apps portal

In our case study design, we applied the three-layer architecture based on the features of HydroDS and HydroShare to support UEB modeling work (Figure 1). A Tethys web app (the UEB web app) was developed and hosted in the HydroShare Tethys Apps portal and serves as the user interface layer to provide easy access to the HydroDS web services. HydroDS is the data service layer used to prepare the model input files and execute the model. HydroShare acts as the data storage layer to store and share the results created from HydroDS. The main activity between the UEB web app and HydroDS is the transfer of user input information to HydroDS for model input preparation or model simulation. Between HydroDS and HydroShare, the activity is mainly the transfer of model input/output files and associated metadata for modeling work. The UEB web app also interacts with HydroShare to retrieve the metadata of shared model input files to facilitate model simulation. We also chose Python for our case study implementation because: 1) there is significant momentum and a growing community of Python development within the scientific computing community; 2) both HydroDS and HydroShare have available Python client libraries that facilitated more rapid development; and 3) the availability of open-source Python libraries and development tools facilitated our work.

We evaluated the system integration for two snowmelt modeling use cases. These use cases were designed to use the web-based simulation functionality to test the sensitivity of the UEB model outputs to different grid cell resolutions of the model input files. The results can help modelers evaluate the tradeoffs between model performance and computational as well as data storage requirements. In the first use case, a user prepares model input, executes the model, and curates the results in HydroShare. In the second use case, another user discovers the shared modeling work in HydroShare and modifies the work to derive new results with different grid cell resolution and compares the snowmelt model outputs from the two use cases.

3 Results

3.1 System integration

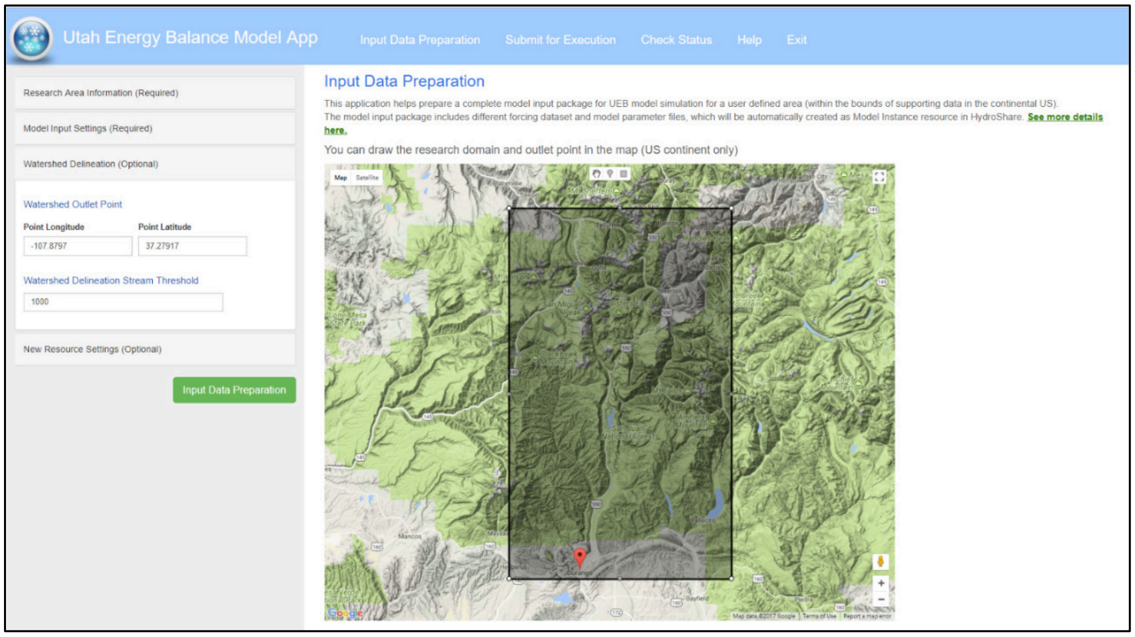
3.1.1 User interface layer

The UEB web app was developed as a Tethys web app and hosted in the HydroShare Tethys Apps portal to provide a graphical user interface for the HydroDS web services. The HydroShare Tethys Apps portal hosts various web applications to support data visualization, analysis, and model simulation. This platform was designed to lower the barrier for the development of environmental web apps and is targeted at scientists and engineers who have some scientific programming experience, but not necessarily web development experience (Swain et al., 2016). Swain et al. showed that, compared to creating a website project from scratch, using the Tethys platform can reduce the need to learn multiple languages for web app development and the total number of lines of code for each web app.

We chose HydroShare Tethys Apps portal to host the UEB web app for several reasons. First, and in general, using a web app portal decouples the user interface application from the systems that host data and hydrologic modeling web services. Loosely coupled systems allow changes in one system component without big changes in the other system components making them easier to maintain. Second, Tethys platform provides software development kits to simplify and reduce the coding and learning of web programming languages required for web app development.

The UEB web app was designed to provide three functions: model input preparation, model execution, and job status checking. Users can interact with this web app to perform modeling work without writing program code to simplify access to HydroDS. Figure 4 (a) shows the user interface for model input preparation. This has two main sections: the user input form section on the left and the map view section in the center. The user input form section allows the user to enter settings to create a complete model input package for model simulation. The map view section helps the user draw a bounding box and optionally an outlet point to specify the modeling domain. If just a bounding box is provided, the entire bounding box is used as the model domain. If an outlet point is provided, the watershed draining to the outlet is computed within the bounding box and used as the domain. The user needs to ensure that the bounding box is sufficient to contain the entire watershed draining to the outlet point.

After the user fills out the form and clicks on the "Input Data Preparation" button, the web request is sent to HydroDS and a corresponding job ID is returned so that the UEB web app can monitor the status of the submitted job. Figure 4 (b) shows the user interface for model execution. It also has two main sections: the model input information section on the left and the map view section. The model input information section allows the user to select a model input package stored in HydroShare. When the user selects a model input package, its corresponding metadata is retrieved from HydroShare and shown in this section. Furthermore, if the metadata includes the bounding box and outlet point information for the modeled domain, it will be automatically shown on the map to orient the user geographically. After the user clicks on the "Submit Model Execution" button, the web request is sent to HydroDS, and the corresponding job ID is returned so that the UEB web app can monitor the job status. Figure 5 shows the job status checking user interface where the status of submitted model input preparation or model simulation jobs is shown. When the job is completed successfully, the user is provided with a link to the resource in HydroShare that stores the model input package (in the green frame) or model output files (in the red frame). If the job fails, the user will be provided with detailed error information (in the yellow frame).


Utah Energy Balance Model App
Input Data Preparation
Submit for Execution
Check Status
Help
Exit
Research Area Information (Required)
Model Input Settings (Required)
Watershed Delineation (Optional)
New Resource Settings (Optional)
Input Data Preparation
Map
Satellite

Input Data Preparation

This application helps prepare a complete model input package for UEB model simulation for a user defined area (within the bounds of supporting data in the continental US).
The model input package includes different forcing dataset and model parameter files, which will be automatically created as Model Instance resource in HydroShare. **See more details here.**

You can draw the research domain and outlet point in the map (US continent only)

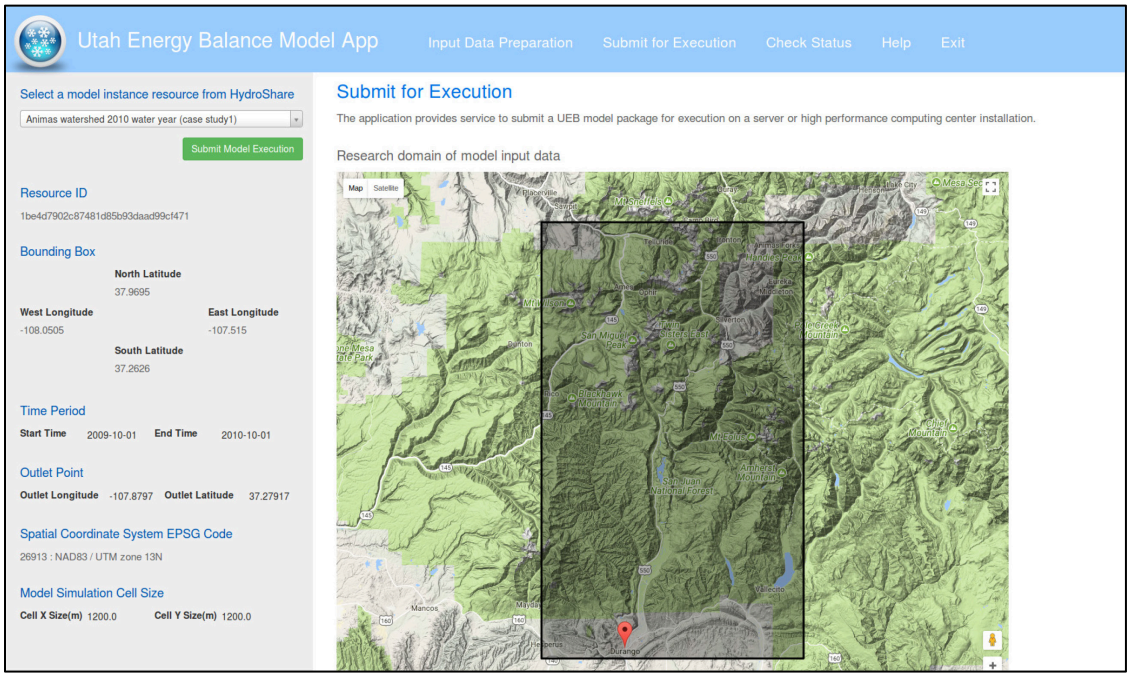
Watershed Outlet Point

Point Longitude	Point Latitude
-107.8797	37.27917

Watershed Delineation Stream Threshold

1000

(a)



Select a model instance resource from HydroShare

Animas watershed 2010 water year (case study1)

Submit for Execution

The application provides service to submit a UEB model package for execution on a server or high performance computing center installation.

Research domain of model input data

Resource ID

1be4d7902c87481d85b93daad99cf471

Bounding Box

North Latitude
37.9695

West Longitude
-108.0505

East Longitude
-107.515

South Latitude
37.2626

Time Period

Start Time 2009-10-01 End Time 2010-10-01

Outlet Point

Outlet Longitude -107.8797 Outlet Latitude 37.27917

Spatial Coordinate System EPSG Code

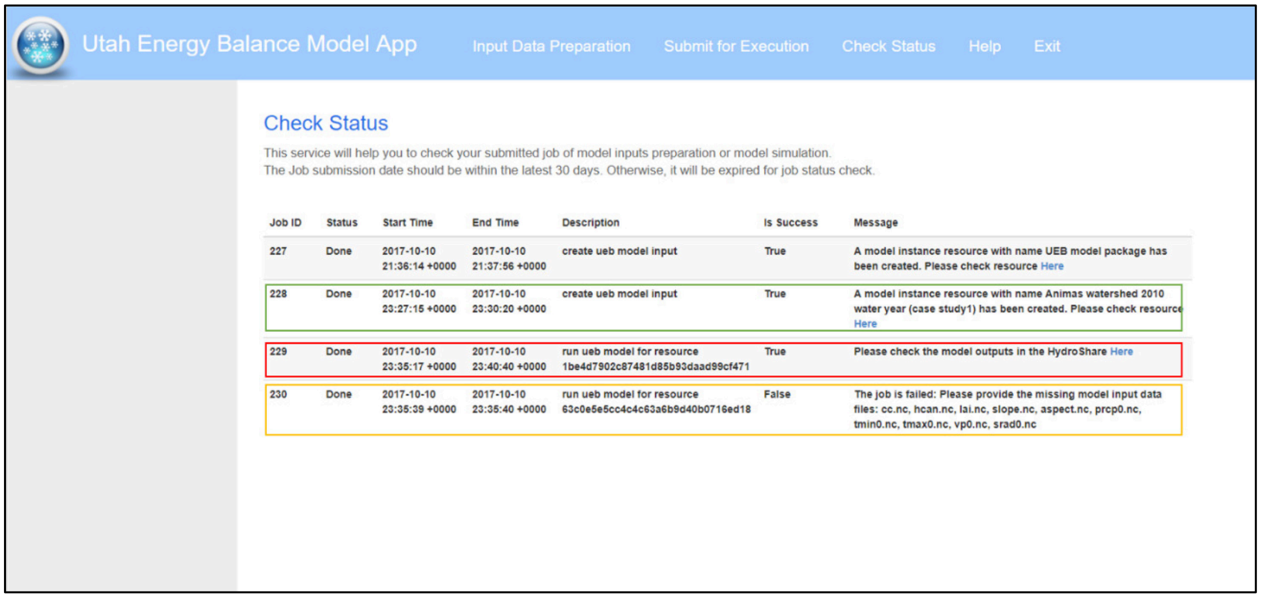
26913 : NAD83 / UTM zone 13N

Model Simulation Cell Size

Cell X Size(m) 1200.0 Cell Y Size(m) 1200.0

(b)

Figure 4 User interface of the UEB web app for input preparation (a) and model execution (b).



Check Status

This service will help you to check your submitted job of model inputs preparation or model simulation.
The Job submission date should be within the latest 30 days. Otherwise, it will be expired for job status check.

Job ID	Status	Start Time	End Time	Description	Is Success	Message
227	Done	2017-10-10 21:36:14 +0000	2017-10-10 21:37:56 +0000	create ueb model input	True	A model instance resource with name UEB model package has been created. Please check resource Here
228	Done	2017-10-10 23:27:15 +0000	2017-10-10 23:30:20 +0000	create ueb model input	True	A model instance resource with name Animas watershed 2010 water year (case study1) has been created. Please check resource Here
229	Done	2017-10-10 23:35:17 +0000	2017-10-10 23:40:40 +0000	run ueb model for resource 1be4d7902c87481d85b93daad99cf471	True	Please check the model outputs in the HydroShare Here
230	Done	2017-10-10 23:35:39 +0000	2017-10-10 23:35:40 +0000	run ueb model for resource 63c0e5e5cc4c4c63a6b9d40b0716ed18	False	The job is failed: Please provide the missing model input data files: cc.nc, hcan.nc, lai.nc, slope.nc, aspect.nc, prcp0.nc, tmin0.nc, tmax0.nc, vp0.nc, srad0.nc

Figure 5 User interface of the UEB web app for job status checking.

The UEB web app was built based on Tethys, which by default includes a narrow left panel and a wide right panel in the main app section. We designed the app to display a map in the main app section and parameter entry form with control buttons on the left. Menu bars at the top were used to switch between steps in the designed use of the app, which can provide the user with guidance on the functionality of each page. Implementing this design required customizing the default Hypertext Markup Language (HTML) and cascading style sheets (CSS) script provided by Tethys. The user input forms in the left panel were implemented using Bootstrap, an open-source front-end web framework (http://getbootstrap.com/) and the Template Gizmos API (http://docs.tethysplatform.org/en/latest/tethys_sdk/gizmos.html) from the Tethys software development kit. The map view in the right panel was implemented using the Google Maps JavaScript API (https://developers.google.com/maps/). Additionally, the HydroShare REST API Python client was used to manage all the interactions between the user interface layer and the data storage layer. For example, the metadata for existing model input packages from HydroShare can be retrieved using the Python client and displayed on the model execution interface. We also created a resource for the UEB web app in HydroShare (Gan et al., 2020). This resource stores the metadata information of the UEB web app and helps users to discover and launch the web app through HydroShare for hydrologic modeling research.

3.1.2 Data service layer

To support the work described in this paper, we implemented new web services and job submission capability in the HydroDS system, which were used by the UEB web app for model input preparation, model simulation, and job status checking. This was an extension of the original design for the HydroDS web services (Gichamo et al., 2020), which required users to make multiple web requests to process various datasets for input preparation (Table 1). It is inefficient for the UEB web app to send multiple web requests to HydroDS and periodically check for completion. Thus, we used the existing data processing functionality in HydroDS and implemented a new web service for model input preparation, which enables the user to click on the "Input Data Preparation" button in the UEB web app to submit a single web request to HydroDS to accomplish the work. Figure 6 (a) shows the detailed tasks done by this new web service. It first creates a complete UEB model input package that includes both the input data files and the model parameter files. Then, it generates a Python file to document the details of how the model input package can be created using the HydroDS Python client. Finally, it transfers all of the files and associated metadata to HydroShare. In this web service, the Python script created was designed to provide input preparation details instead of hiding the processing work behind the scenes as a black box to users. This design ensures that novices can view and learn from the syntax of the Python script, using it as an example to learn how to use HydroDS web services and create input preparation workflows for other hydrologic models. It also focuses on another major target user group for this system – i.e., modelers who want better tools to make their work easier but who still want to know the coding details of the research. For both types of users, this Python script can be reused to reproduce or derive new model input for the UEB model.

We also implemented a new web service that is called when the user clicks on the "Submit Model Execution" button in the UEB web app to make a single web request to HydroDS for model simulation. Figure 6 (b) presents the specific tasks accomplished by this web service. It first downloads the model input package from HydroShare into HydroDS. Then, it validates the model input package to check if there are missing files required for executing the model. If the validation is successful, HydroDS executes the UEB model and then transfers the model output files and stores them with the model input package in HydroShare. To support data transfer between the data service and data storage layers, the HydroShare REST API Python client "hs_restclient" was used for reading and writing files and metadata to and from HydroShare.

In order to improve the user experience by supporting job status checking and display in the UEB web app, we also added job submission capability for the two new web services. When users make web requests to HydroDS via the UEB web app, the web service responds with a job ID, and the model input preparation or model execution process can be accomplished asynchronously so that users are able to check the job status any time after the job submission (Figure 7). In HydroDS, a database was created to store information for the submitted jobs. When a job is initiated, the job ID and associated metadata are stored in the database (e.g., job creation date, job creator, and job status). After the job is launched and completed, the job status is updated. Web services for querying the job status from HydroDS were also implemented, and were used by the UEB web app to get the job details and present them on the user interface.

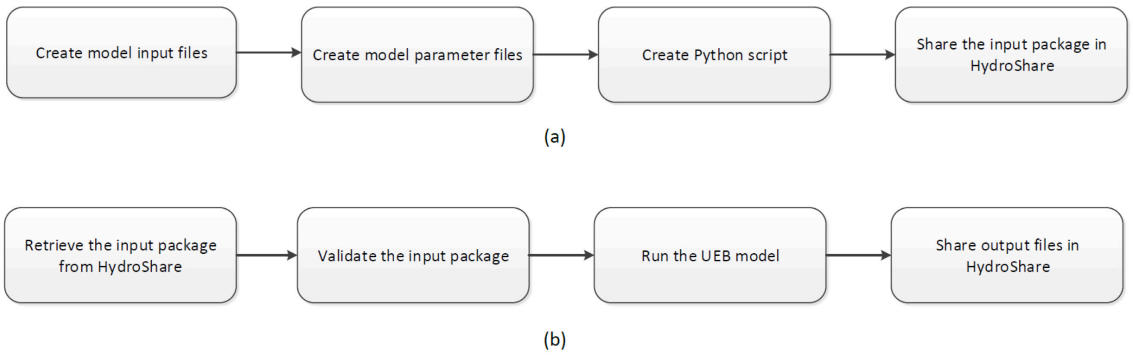

Create model input files
Create model parameter files
Create Python script
Share the input package in HydroShare
(a)
Retrieve the input package from HydroShare
Validate the input package
Run the UEB model
Share output files in HydroShare
(b)

Figure 6 The functionality of the added web services in HydroDS. Panel (a) for model input package preparation; Panel (b) for model simulation.

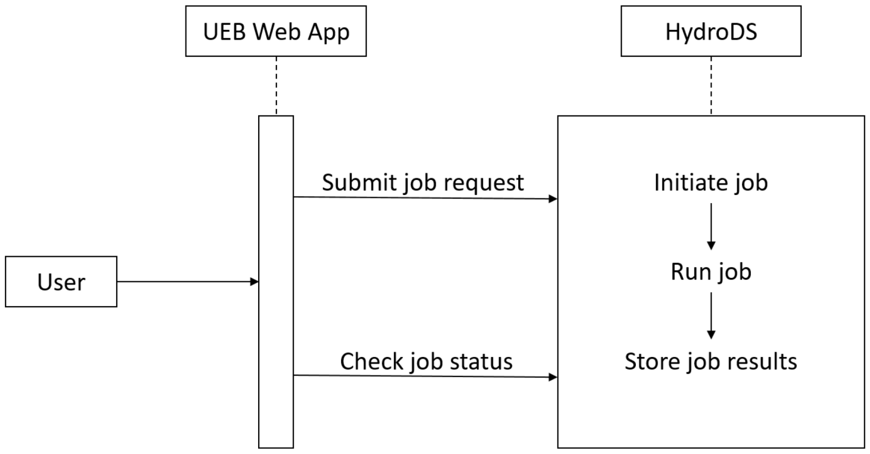
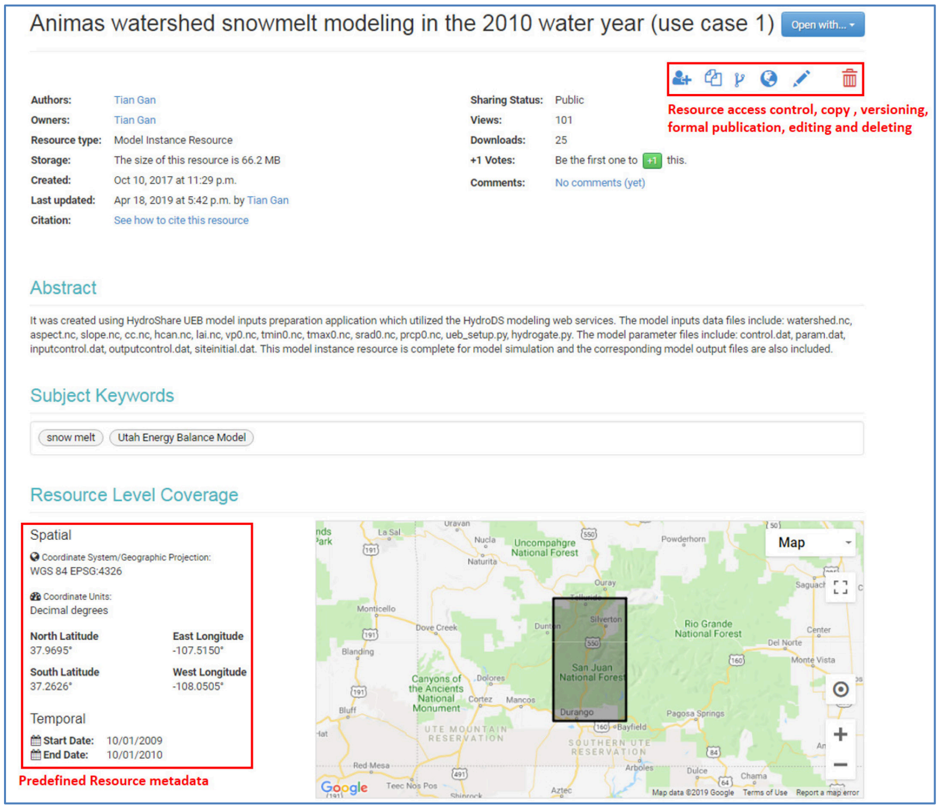


Figure 7 Job management workflow.

3.1.3 Data storage layer

In HydroShare, we chose the "model instance" resource type (Morsy et al., 2017) to support curation and sharing of the data files and metadata generated by HydroDS. This resource type was specifically designed to support the collaborative sharing of model input/output files and their associated metadata, which best suits our requirement to improve reproducibility of hydrologic modeling research (Figure 8). For example, users can store model input/output files in a HydroShare model instance resource and describe them with predefined resource-level metadata as well as user-defined key-value pair metadata. This can help others discover and access the model instance with enough information for reuse. Users can also manage the resource access control, so that it can be kept as private and accessed only by trusted users to prepare and edit the contents, or it can be shared to the public so that anyone can discover and reuse it for validation or deriving new results. In addition, users can formally publish their modeling work in HydroShare to get a digital object identifier (DOI) and formal citation information. This encourages proper citation of the shared work.


Open with...
Map
Resource access control, copy , versioning, formal publication, editing and deleting
Predefined Resource metadata

Animas watershed snowmelt modeling in the 2010 water year (use case 1)

Authors:	Tian Gan
Owners:	Tian Gan
Resource type:	Model Instance Resource
Storage:	The size of this resource is 66.2 MB
Created:	Oct 10, 2017 at 11:29 p.m.
Last updated:	Apr 18, 2019 at 5:42 p.m. by Tian Gan
Citation:	See how to cite this resource
Sharing Status:	Public
Views:	101
Downloads:	25
+1 Votes:	Be the first one to +1 this.
Comments:	No comments (yet)

Abstract

It was created using HydroShare UEB model inputs preparation application which utilized the HydroDS modeling web services. The model inputs data files include: watershed.nc, aspect.nc, slope.nc, cc.nc, hcan.nc, lai.nc, vp0.nc, tmin0.nc, tmax0.nc, srad0.nc, prcp0.nc, ueb_setup.py, hydrogate.py. The model parameter files include: control.dat, param.dat, inputcontrol.dat, outputcontrol.dat, siteinitial.dat. This model instance resource is complete for model simulation and the corresponding model output files are also included.

Subject Keywords

snow melt | Utah Energy Balance Model

Resource Level Coverage

Spatial

Coordinate System/Geographic Projection:
WGS 84 EPSG:4326

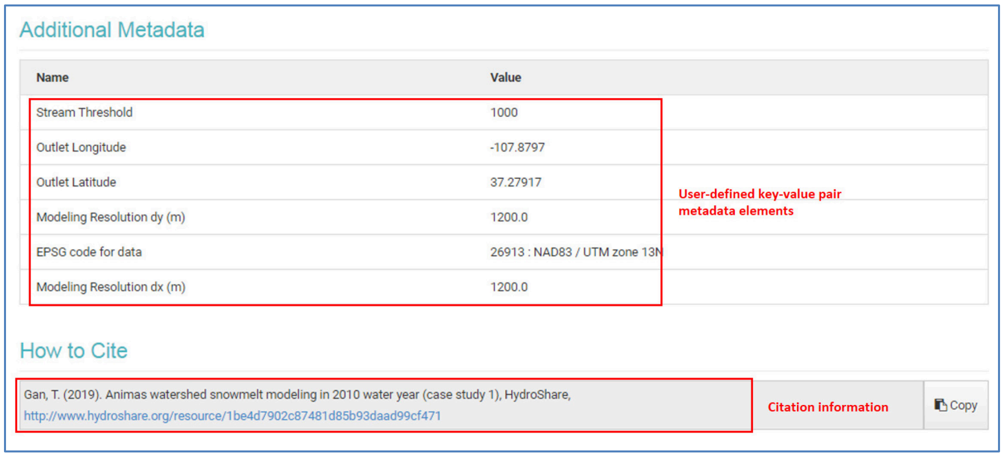
Coordinate Units:
Decimal degrees

North Latitude 37.9695°	East Longitude -107.5150°
South Latitude 37.2626°	West Longitude -108.0505°

Temporal

Start Date: 10/01/2009
End Date: 10/01/2010

(a)



Additional Metadata

Name	Value
Stream Threshold	1000
Outlet Longitude	-107.8797
Outlet Latitude	37.27917
Modeling Resolution dy (m)	1200.0
EPSG code for data	26913 : NAD83 / UTM zone 13N
Modeling Resolution dx (m)	1200.0

How to Cite

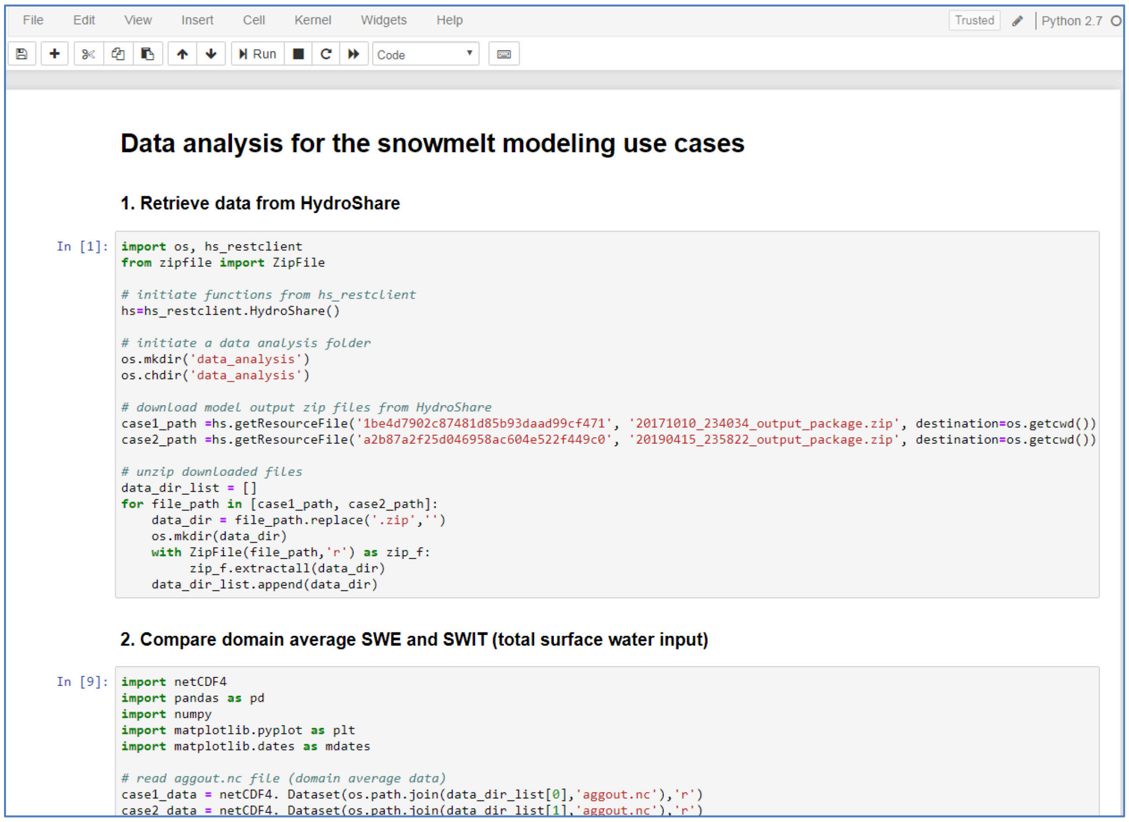
Gan, T. (2019). Animas watershed snowmelt modeling in 2010 water year (case study 1), HydroShare, http://www.hydroshare.org/resource/1be4d7902c87481d85b93daad99cf471

(b)

Figure 8 Example model instance resource in HydroShare. Panel (a) shows different resource functions and predefined metadata; Panel (b) shows the user-defined metadata and suggested citation information.

When the UEB web app is used for model input preparation, a new model instance resource is created in HydroShare to store the model input package. The information entered in the user input form of the UEB web app is stored as user-defined resource metadata in HydroShare, which saves users from manual metadata editing work to provide detailed information about the input package. When the UEB web app is used for model simulation, the model instance resource is downloaded from HydroShare into HydroDS for execution, and the resulting model output files are sent back to the corresponding model instance resource in HydroShare. In the case where a user submits a model simulation job but deletes the model instance resource before the job completes, a new model instance resource is created that includes model input package and output files after the model simulation. The user can run the simulation to generate model output multiple times with all the results stored in the same resource. Additionally, other users can use the resource copy function in HydroShare to duplicate the model instance resource as their own new resource to repeat or build on the modeling work.

In addition to using the model instance resource for data curation and sharing, we also used the CUAHSI JupyterHub web app for post-modeling analysis and to demonstrate reuse of a shared model instance. This web app provides an online programming environment that supports the development and execution of program code from a Jupyter Notebook file. The benefit of using this web app is that users do not need to download the modeling work and install software on their local computers. Instead, the model instance resource can be directly retrieved from HydroShare into this web app for reuse. Working in CUAHSI JupyterHub web app does require use of the Python programming language for post-modeling analysis. However, Python is widely used in scientific research and is, in our experience, relatively easy for modelers to understand, especially in a Jupyter Notebook context where code snippets are short, can be explained by accompanying text information, and can serve as a gentle programming and scripting entry point for users who have background with other programming languages or who are new to these concepts. Users can develop and execute Python code in a Jupyter Notebook file to visualize or analyze the model input/output datasets (Figure 9). Other users can also use this web app and the Python script from the model instance resource to repeat or modify the model input preparation workflow to validate the existing model input package or generate a new model input package (Figure 10). This provides another option for model input preparation, which is more scripted, but less graphical user interface friendly than the UEB web app.



Data analysis for the snowmelt modeling use cases

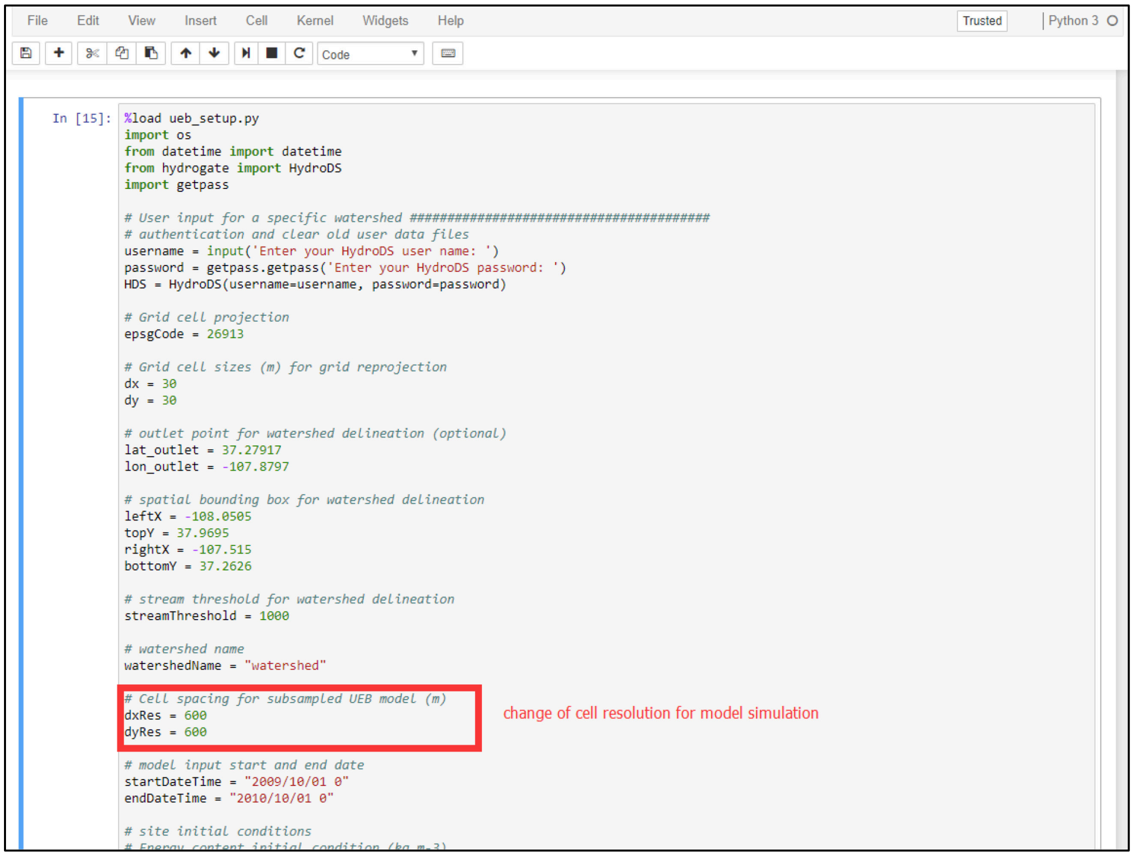
1. Retrieve data from HydroShare

In [1]:
```python
import os, hs_restclient
from zipfile import ZipFile

# initiate functions from hs_restclient
hs=hs_restclient.HydroShare()

# initiate a data analysis folder
os.mkdir('data_analysis')
os.chdir('data_analysis')

# download model output zip files from HydroShare
case1_path =hs.getResourceFile('1be4d7902c87481d85b93daad99cf471', '20171010_234034_output_package.zip', destination=os.getcwd())
case2_path =hs.getResourceFile('a2b87a2f25d046958ac604e522f449c0', '20190415_235822_output_package.zip', destination=os.getcwd())

# unzip downloaded files
data_dir_list = []
for file_path in [case1_path, case2_path]:
    data_dir = file_path.replace('.zip','')
    os.mkdir(data_dir)
    with ZipFile(file_path,'r') as zip_f:
         zip_f.extractall(data_dir)
    data_dir_list.append(data_dir)
```

2. Compare domain average SWE and SWIT (total surface water input)

In [9]:
```python
import netCDF4
import pandas as pd
import numpy
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

# read aggout.nc file (domain average data)
case1_data = netCDF4. Dataset(os.path.join(data_dir_list[0],'aggout.nc'),'r')
case2_data = netCDF4. Dataset(os.path.join(data_dir_list[1],'aggout.nc'),'r')
```

Figure 9 Python code for post-modeling analysis in the CUAHSI JupyterHub web app.



```
%load ueb_setup.py
import os
from datetime import datetime
from hydrogate import HydroDS
import getpass

# User input for a specific watershed ########################################
# authentication and clear old user data files
username = input('Enter your HydroDS user name: ')
password = getpass.getpass('Enter your HydroDS password: ')
HDS = HydroDS(username=username, password=password)

# Grid cell projection
epsgCode = 26913

# Grid cell sizes (m) for grid reprojection
dx = 30
dy = 30

# outlet point for watershed delineation (optional)
lat_outlet = 37.27917
lon_outlet = -107.8797

# spatial bounding box for watershed delineation
leftX = -108.0505
topY = 37.9695
rightX = -107.515
bottomY = 37.2626

# stream threshold for watershed delineation
streamThreshold = 1000

# watershed name
watershedName = "watershed"

# Cell spacing for subsampled UEB model (m)
dxRes = 600
dyRes = 600

# model input start and end date
startDateTime = "2009/10/01 0"
endDateTime = "2010/10/01 0"

# site initial conditions
```

Figure 10 Python script for model input preparation loaded into a Jupyter Notebook file in the CUAHSI JupyterHub web app.

3.2 Snowmelt modeling

We used the Animas watershed in the Colorado River Basin (Figure 11) as the study area to implement our two use cases for model input preparation, then simulation of snowmelt for water year 2010. This served to validate the implemented functionality and test if the system integration can provide web-based simulation to support hydrologic modeling.

In the first use case, the UEB web app was used to prepare the model input package, execute the model, and then have all the results automatically copied into a HydroShare resource. Figure 4 and Table 2 show the interfaces and detailed settings information that were used in the UEB web app for model input preparation and model simulation for the Animas watershed. Figure 5 shows the job status of the corresponding results. The green frame is the status for model input preparation, and the red frame for model simulation. Figure 8 is the resource landing page for the model instance resource (Gan, 2019a), which was created to store the model input/output files, the associated metadata, and the Python script of the input preparation workflow for the first use case.

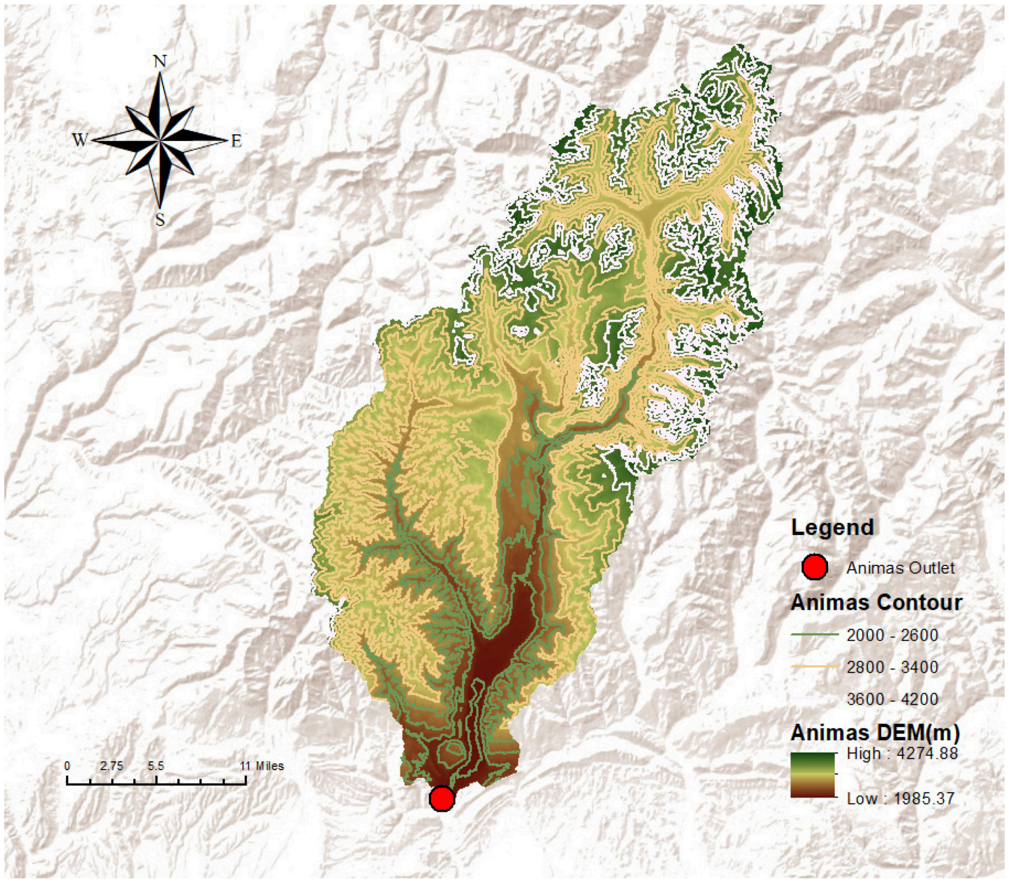


Figure 11 The Animas watershed in the Colorado River Basin.

The second use case demonstrated collaboration and showed how the modeling work created in the first use case could be discovered, modified, and reused to derive new findings. Assume that the user who prepared the model in the first use case was user 1, and the user who collaborated and reused the model was user 2. The first author of this paper actually acted as both users with separate HydroShare accounts to prepare this illustration. The second use case included the following steps. First, user 2 discovered and got access to the model instance resource created by user 1. Second, user 2 retrieved the resource into the CUAHSI JupyterHub web app, which was used by user 2 to modify the Python script from the model input package of the first use case to create a new model input package and store it in a new model instance resource in HydroShare. Third, the UEB web app was used by user 2 to execute the model with the new model instance resource. Finally, the CUAHSI JupyterHub web app was used by user 2 to develop Python code in a Jupyter Notebook to compare the model outputs from the two use cases.

Table 2 Inputs set for model input preparation in the first use case.

Item	Value	Required? (Yes/No)
Bounding box [north, south, west, east]	[37.9695, 37.2626, -108.0505, -107.5150] in degrees	Yes
Energy content initial condition	0	Yes
Snow water equivalent initial condition	0	Yes
Snow surface dimensionless age initial condition	0	Yes
Snow water equivalent of canopy condition	0	Yes
Snow surface temperature one day prior to the model starting time	0	Yes
Spatial coordinate system	NAD83/UTM zone 13N	Yes
Time period [start date, end date]	[2009/10/01, 2010/10/01]	Yes
Cell size for model simulation [dx, dy]	[1200, 1200] in meter	Yes
Watershed outlet [longitude, latitude]	[-107.8797, 37.27917] in degree	No
HydroShare resource title	Animas watershed snowmelt modeling in the 2010 water year (case study1)	No
HydroShare resource keywords	snow melt, UEB Utah Energy Balance Model	No

Figure 12 shows the discovery page in HydroShare where the model instance resource created in the first use case can be discovered. In HydroShare, users can search for resources with text or geolocation information and filter the listed results with different facets (e.g., authors or keywords) to find the needed content.

The Python script loaded into a cell in a Jupyter Notebook within the CUAHSI JupyterHub web app is shown in Figure 10. This Python script is from the model instance resource of the first use case created by user 1 and documents the workflow of model input preparation for creating the climate forcing datasets and parameter files. Figure 10 highlights where user 2 modified the

Python script and changed the model resolution from 1200 meters to 600 meters, a model configuration change being tested by user 2 in the second use case (reuse of a model previously established). This modification was designed to test the sensitivity of the model to grid cell resolution and determine whether different resolutions lead to different snow outputs. After the modification, the Jupyter Notebook file was used by user 2 to execute the script and to create a new model instance resource in HydroShare to store the results, which includes the modified Python script and the new model input package (Gan, 2019b). After the new model instance resource was created, the UEB web app was used by user 2 to execute the model to create the model output files, which were automatically stored in the same resource.

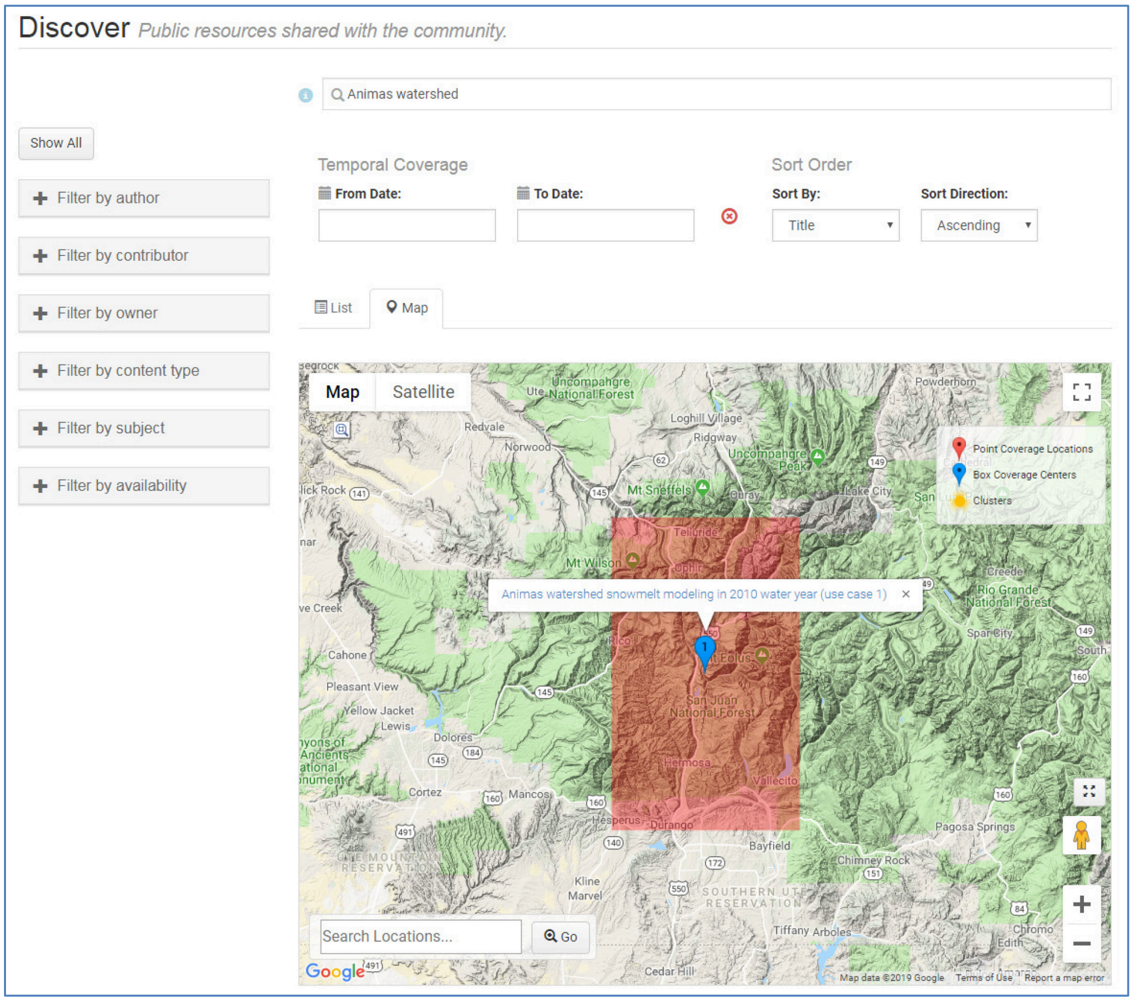


Figure 12 The HydroShare discovery page used to search for the model instance resource created in the first use case.

Finally, the CUAHSI JupyterHub web app was used by user 2 to retrieve the two resources from

HydroShare and to develop data visualization code (Figure 9) to compare the snow output from the two use cases. It was found that in the Animas watershed, the comparison of 600 meters versus 1200 meters grid cell resolutions resulted in only very small differences in the model output for snow water equivalent and total surface water input (Figure 13 and Figure 14). This is mainly because the spatial variability of the terrain and canopy input for the UEB model at the two grid cell resolutions only has small differences, which leads to similar performance for the snowmelt results. Any user can also test with higher grid cell resolutions (e.g., 100m or 300m) and compare the model outputs.

This sensitivity test is useful because UEB modelers may choose a coarser cell resolution for model simulation to decrease the simulation time and the size of input and output datasets if there is no significant difference for the snowmelt output. In addition, users may also reuse the first use case to conduct model experiments for parameter sensitivity analysis and find out the relationship between different parameter settings and model performance. The modeling and analysis process can be conducted using the web-based simulation without using the local computing and storage resources. The corresponding results for model experiments can be directly curated and shared with others for validation or reuse.

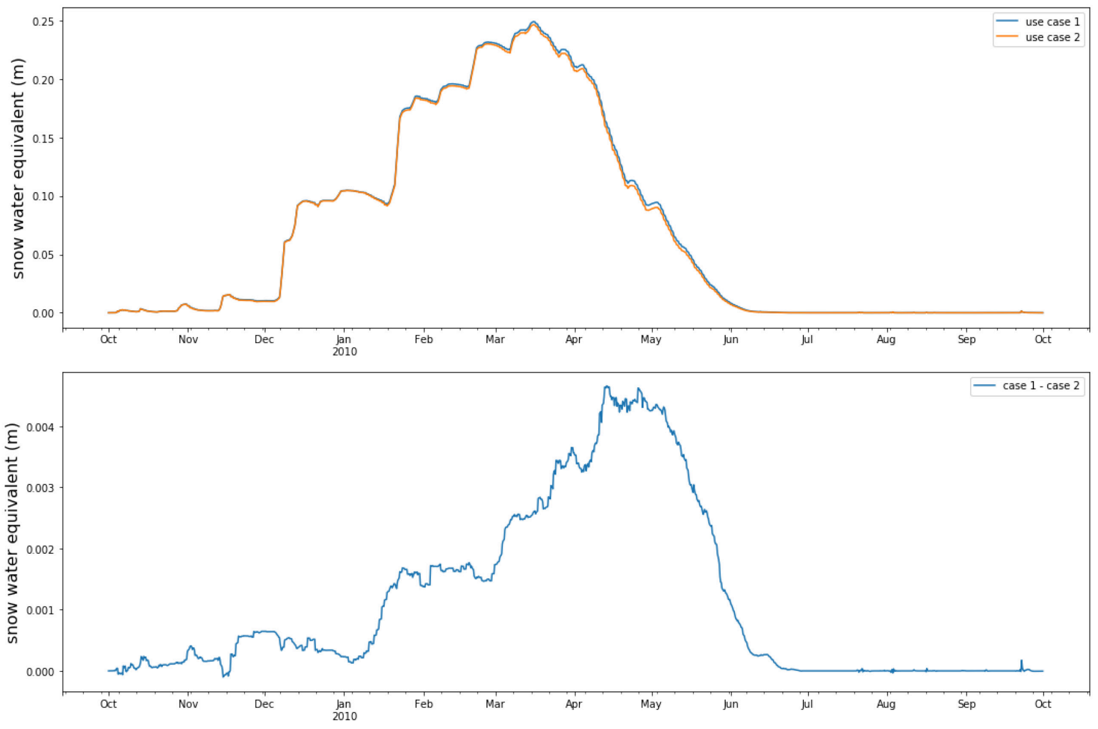


Figure 13 Comparison of snow water equivalent created by the two uses cases.

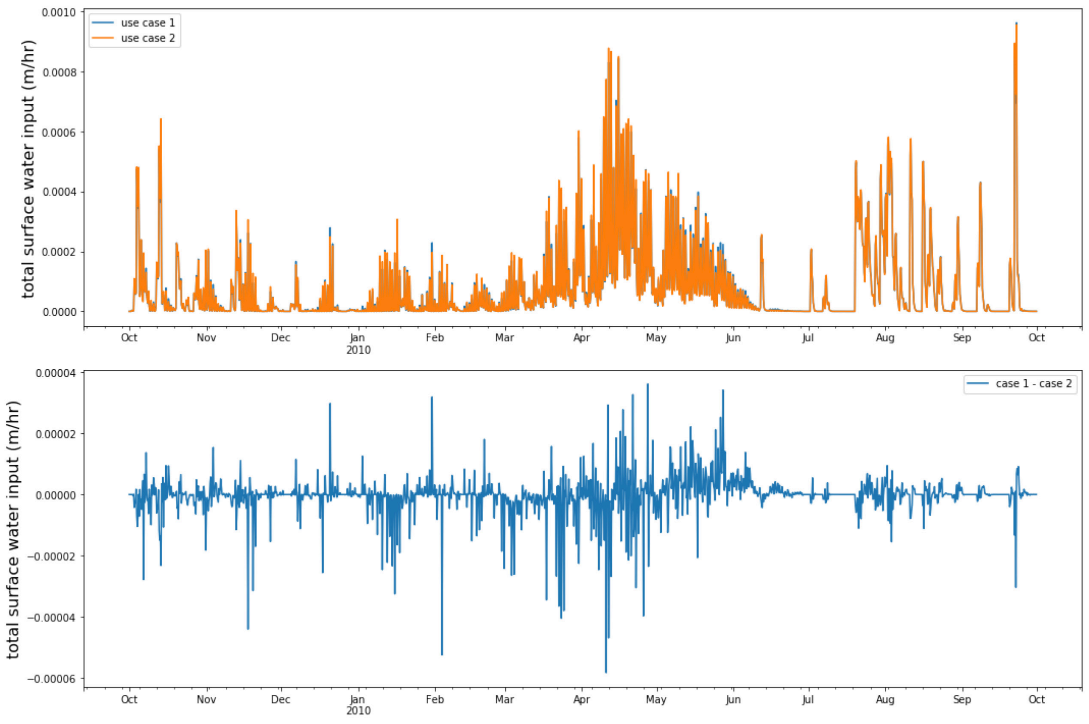


Figure 14 Comparison of total surface water input created by the two uses cases.

4 Discussion

This case study demonstrated that after using the three-layer web service based architecture to integrate example systems, users were able to develop, share, and reuse modeling work in an online environment by interacting with HydroShare and HydroShare Apps (Figure 15). The UEB web app helped to prepare the model input and execute the model through a graphical web user interface. The model instance resource in HydroShare was used to curate and share the modeling results as well as the associated metadata, which enabled others to discover and access them. The CUAHSI JupyterHub web app also provided a web-based tool with which users can modify the work and analyze the results without using data storage or computing resources on their own local computers.

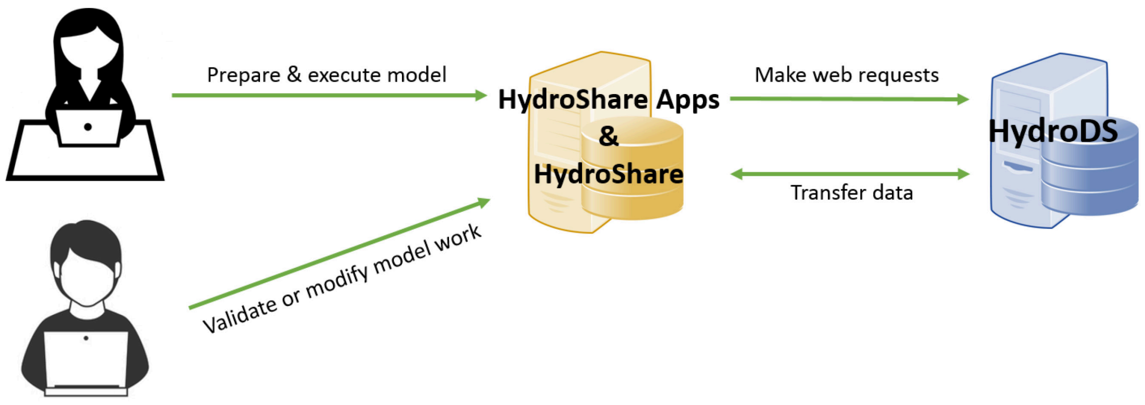


Figure 15 System integration that enables users to interact with HydroShare and HydroShare Apps for multiple modeling tasks.

We also compared three ways to accomplish the same tasks involved in the snow modeling use cases: 1) conducting research without HydroDS web services, 2) conducting research with HydroDS before system integration, and 3) conducting research with HydroDS after system integration (Table 3). The first option represents how modelers are doing modeling research now. The second option represents the use of modeling web services to simplify the work involved in the first option, which might still be difficult in a real application because of the requirement for learning and writing program code. The third option represents a new way of using the modeling web services, which provides a graphical user interface to lower the requirement of programming and the functionality to support data curation and sharing.

Table 3 Comparison of three ways to accomplish tasks for the snowmelt modeling use cases.

Modeling task	**Option1: Traditional method**	**Option2: Use HydroDS before integration**	**Option3: Use HydroDS after integration**
Prepare input and execute model	**Local PC:** • Collect data from multiple sources • Learn and write code • Install software to run script • Install and configure model	**Local PC:** • Learn about HydroDS client library and write Python script • Install Python interpreter to run script	**Data sharing system:** • Enter required information in the UEB web app
Curate and share results	**Local PC:** • Manually upload data and script to a data sharing system • Manually add metadata	**Local PC:** • Download model input/output from HydroDS • Manually upload data and script to a data sharing system • Manually add metadata	**Data sharing system:** • Data, script, and metadata directly and automatically stored in HydroShare
Repeat or modify	**Data sharing system:** • Download script and data	**Data sharing system:** • Download script and data	**Data sharing system:** • Enter required information in the UEB web app

modeling work	**Local PC:** • Learn and modify script • Install software to run script • Install and configure model	**Local PC:** • Learn and modify script • Install Python interpreter to run script	Or • Use CUAHSI JupyterHub web app to modify and run script (if familiar with HydroDS)

This comparison allowed us to evaluate whether the system integration could accomplish the modeling work with less need for coding, and fewer manual operations or data transitions among different environments. We found that the system integration provided benefits in several aspects. First, the system integration lowered the requirement for writing Python script to interact with HydroDS web services. The UEB web app only requires knowledge of the UEB model, which allows users to overcome the programming barrier, saving the time required to write Python code. Additionally, the Python script created by HydroDS to document the input preparation workflow also helps users learn and use the web services from example code.

Second, the system integration simplifies data curation and management efforts. The data files, metadata, and Python script are automatically curated in the data sharing system (HydroShare) without manually moving the files among different environments (HydroDS, local computer, and HydroShare), a process that can be error prone with potential for information loss. This automatic data transfer capability can encourage the preparation and sharing of modeling work rather that retaining it only on local computers. This also supports collaboration and makes it easier to comply with open data mandates and document reproducibility.

Third, the system integration can simplify the way for others to validate reproducibility of the modeling work, and reuse or extend it for their own work. Users can use the UEB web app and the CUAHSI JupyterHub web app to repeat or modify the modeling work without downloading the files to their local computers or configuring their local environments for model execution or data analysis.

While this work has shown that the framework of a user interface layer, data service layer, and data storage layer can facilitate web based collaborative and reproducible hydrologic modeling, there are opportunities for further work to address limitations and improve the current functionality. For example, the post-modeling analysis still requires coding for data visualization and analysis. Thus, new capabilities could be added in the UEB web app to support visualization and analysis of the model input/output datasets through a GUI (e.g., visualization of the watershed delineation result). Additionally, new capabilities for scenario generation and management could be implemented in the UEB web app to, for example, better support scenario analysis for hydrologic modeling research such as has been implemented for other models (Sun, 2013). As for new app capability, it is important to consider the balance between what is coded in a specific GUI application, such as the UEB web app, and provides specific functionality for users that the app developers anticipate are needed, versus general purpose capability in an app, such as the Jupyter Notebook platform, that can empower users more, but requires programming. User driven design and active monitoring of how systems are used can provide information to help with this balance

5 Conclusions

In hydrologic modeling research, we are starting to see the availability of more and more hydrologic modeling web services that enable users to write code and make their work more efficient. However, limitations still exist in real application of such services in terms of their usability and the reproducibility of the modeling work. Users need to learn and write code to utilize these web services, which may be a barrier for those with limited programming skills. In addition, a good mechanism is needed for curation and sharing of not only the data and metadata, but also the script of the modeling work, which can improve the research reproducibility and encourage collaborations around them.

In this paper, we presented an approach that uses a three-layer RESTful web service based architecture to integrate open source software to enable web-based simulation to support hydrologic modeling research. As an example, we integrated the HydroDS hydrologic data and modeling web services with a data sharing system, HydroShare, and tested the implemented functionality with use cases of snowmelt modeling for the Animas watershed in the Colorado River Basin. The results demonstrated that the system integration enabled users to work within an online environment to create, describe, share, discover, modify, and analyze the modeling work, which encourages collaboration around the hydrologic modeling research and significantly reduces the need for coding and manual operation for data transfer and model configuration. This approach has the advantage of reusing open source software to support hydrologic modeling research in terms of collaboration, computer platform independence, and reproducibility of modeling workflows and results.

In addition, the general design of the three-layer web service based architecture can be adopted or adapted to other open source data sharing and modeling software. Furthermore, other modeling web services can be integrated with a data sharing system such as HydroShare using the methods we described to support automated data curation and post-modeling analysis without repeating development of similar functionality. While we used HydroShare for our work, other data sharing systems could also be used. We found that the following data sharing system features were needed to ease integration with other cyberinfrastructure and add value to them. First, the system should have well-developed data sharing functionality and a corresponding web service API for interoperating with other systems over the Internet. For example, HydroShare has a REST API interface and a Python client for that API, which helped us to develop new REST API based web services in HydroDS that enable automatic data transfer between the two systems to support data curation and sharing. Secondly, the data sharing system needs to be a platform where new functionality for interacting with the shared datasets can be added as loosely coupled components (e.g., as web apps) without requiring significant changes to the existing system. For instance, the HydroShare Tethys Apps portal established by the HydroShare team was used to host the UEB web app, which provided a user interface layer to interact with HydroDS and HydroShare with minimal changes in both systems.

In the future, possible development could include a new web app that provides a graphical user interface for multiple data processing web services from HydroDS. This would benefit researchers by making it easier for them to reuse and combine different web services based on their needs and to prepare inputs for other hydrologic models without writing code, while having the results directly curated in HydroShare. Given that this work is Python-based, future work could also involve integration with wider and domain agnostic open source scientific Python environments – e.g., the PANGEO software ecosystem (https://pangeo.io/). Finally, while the work reported in this paper extended the existing HydroDS services, future work should examine how these types of services can be more standardized such that they become more generally usable in modeling workflows (e.g., Castronova et al., 2013; Qiao et al., 2019).

Funding:

This work was supported by the National Science Foundation [OCI-1148453, ICER-1639655].

Acknowledgements

This work was supported by the National Science Foundation under collaborative grants OCI-1148453 and ICER-1639655. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Archfield, S.A., Clark, M., Arheimer, B., Hay, L.E., McMillan, H., Kiang, J.E., Seibert, J., Hakala, K., Bock, A., Wagener, T., Farmer, W.H., Andréassian, V., Attinger, S., Viglione, A., Knight, R., Markstrom, S., Over, T., 2015. Accelerating advances in continental domain hydrologic modeling. Water Resour. Res. 51, 10078–10091. https://doi.org/10.1002/2015WR017498

Billah, M.M., Goodall, J.L., Narayan, U., Essawy, B.T., Lakshmi, V., Rajasekar, A., Moore, R.W., 2016. Using a data grid to automate data preparation pipelines required for regional-scale hydrologic modeling. Environ. Model. Softw. 78, 31–39. https://doi.org/10.1016/j.envsoft.2015.12.010

Byrne, J., Heavey, C., Byrne, P.J., 2010. A review of Web-based simulation and supporting tools. Simul. Model. Pract. Theory 18, 253–276. https://doi.org/10.1016/j.simpat.2009.09.013

Castronova, A.M., 2016. HydroShare Jupyterhub Notebook Server. Github repository. https://github.com/hydroshare/hydroshare-jupyterhub

Castronova, A.M., Goodall, J.L., Elag, M.M., 2013. Models as web services using the Open Geospatial Consortium (OGC) Web Processing Service (WPS) standard. Environ. Model. Softw. 41, 72–83. https://doi.org/10.1016/j.envsoft.2012.11.010

Chuah, J., Deeds, M., Malik, T., Choi, Y., Goodall, J.L., 2020. Documenting computing environments for reproducible experiments, in: Foster, I., Joubert, G.R., Kučera, L., Nagel, W.E., Peters, F. (Eds.), Parallel Computing: Technology Trends. Advances in Parallel Computing Series, Volume36, IOS Press, Amsterdam, The Netherlands, pp. 756–765. https://doi.org/10.3233/APC200106

Demir, I., Krajewski, W.F., 2013. Towards an integrated Flood Information System: Centralized data access, analysis, and visualization. Environ. Model. Softw. 50, 77–84. https://doi.org/10.1016/j.envsoft.2013.08.009

Dingman, S.L., 2008. Physcial Hydrology (second edition). Waveland Press, Inc, Long Grove, IL.

Essawy, B.T., Goodall, J.L., Zell, W., Voce, D., Morsy, M.M., Sadler, J., Yuan, Z., Malik, T., 2018. Integrating scientific cyberinfrastructures to improve reproducibility in computational hydrology: Example for HydroShare and GeoTrust. Environ. Model. Softw. 105, 217–229. https://doi.org/10.1016/j.envsoft.2018.03.025

Gan, T., 2019a. Animas watershed snowmelt modeling in the 2010 water year (use case 1). HydroShare. https://doi.org/10.4211/hs.1be4d7902c87481d85b93daad99cf471

Gan, T., 2019b. Animas watershed snowmelt modeling in the 2010 water year (use case 2). HydroShare. https://doi.org/10.4211/hs.a2b87a2f25d046958ac604e522f449c0

Gan, T., Dash, P., Tarboton D.G., 2020. Tethys web app for the Utah Energy Balance Model (v1.0.0). Zenodo. http://doi.org/10.5281/zenodo.3735456

Gichamo, T.Z., Sazib, N.S., Tarboton, D.G., Dash, P., 2020. HydroDS: Data services in support of physically based, distributed hydrological models. Environ. Model. Softw. 125, 104623. https://doi.org/https://doi.org/10.1016/j.envsoft.2020.104623

Granell, C., Dıaz, L., Gould, M., Díaz, L., Gould, M., 2010. Service-oriented applications for environmental models: Reusable geospatial services. Environ. Model. Softw. 25, 182–198. https://doi.org/10.1016/j.envsoft.2009.08.005

Heard, J., Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Bedig, A., Castronova, A.M., Couch, A., Frisby, C., Gan, T., Goodall, J., Jackson, S., Livingston, S., Maidment, D., Martin, N., Miles, B.C., Mills, S., Sadler, J., Valentine, D., Zhao, L., 2014. An architectural overview of HydroShare, a next-generation hydrologic information system. Proceedings of the 11th International Conference on Hydroinformatics, HIC 2014, New York City, USA.

Homer, C., Dewitz, J., Yang, L., Jin, S., Danielson, P., Xian, G., Coulston, J., Herold, N., Wickham, J., Megown, K., 2015. Completion of the 2011 national land cover database for the conterminous United States – Representing a decade of land cover change information. Photogramm. Eng. Remote Sens, 81(0):345-354.

Horsburgh, J.S., Morsy, M.M., Castronova, A.M., Goodall, J.L., Gan, T., Yi, H., Stealey, M.J., Tarboton, D.G., 2015. HydroShare: Sharing diverse hydrologic data types and models as social objects within a Hydrologic Information System,. J. Am. Water Resour. Assoc. 27517. https://doi.org/10.1111/1752-1688.12363

Hutton, C., Wagener, T., Freer, J., Han, D., Duffy, C., Arheimer, B., 2016. Most computational hydrology is not reproducible, so is it really science? Water Resour. Res. 52, 7548–7555. https://doi.org/10.1002/2016WR019285

Kumar, S.V., Peters-Lidard, C.D., Eastman, J.L., Tao, W.K., 2008. An integrated high-resolution hydrometeorological modeling testbed using LIS and WRF. Environ. Model. Softw. 23, 169–181. https://doi.org/10.1016/j.envsoft.2007.05.012

Laloy, E., Vrugt, J.A., 2012. High-dimensional posterior exploration of hydrologic models using multiple-try DREAM (ZS) and high-performance computing. Water Resour. Res. 48, 1–18. https://doi.org/10.1029/2011WR010608

Leonard, L., Duffy, C.J., 2013. Essential terrestrial variable data workflows for distributed water resources modeling. Environ. Model. Softw. 50, 85–96. https://doi.org/10.1016/j.envsoft.2013.09.003

Liang, X., Lettenmaier, D.P., Wood, E.F., 1996. One-dimensional statistical dynamic representation of subgrid spatial variability of precipitation in the two-layer variable infiltration capacity model. J. Geophys. Res. Atmos. 101, 21403–21422. https://doi.org/10.1029/96JD01448

Miles, B.C., 2014. Small-scale residential stormwater management in urbanized watersheds: A geoinformatics-driven ecohydrology modeling approach (Doctoral dissertation). The University of North Carolina, Chapel Hill, NC, https://doi.org/10.17615/1m6y-vs10

Morsy, M.M., Goodall, J.L., Castronova, A.M., Dash, P., Merwade, V., Sadler, J.M., Rajib, M.A., Horsburgh, J.S., Tarboton, D.G., 2017. Design of a metadata framework for environmental models with an example hydrologic application in HydroShare. Environ. Model. Softw. 93, 13–28. https://doi.org/10.1016/j.envsoft.2017.02.028

Peckham, S.D., Hutton, E.W.H., Norris, B., 2013. A component-based approach to integrated modeling in the geosciences: The design of CSDMS. Comput. Geosci. 53, 3–12. https://doi.org/10.1016/j.cageo.2012.04.002

Qiao, X., Li, Z., Ames, D.P., Nelson, E.J., Swain, N.R., 2019. Simplifying the deployment of OGC web processing services (WPS) for environmental modelling – Introducing Tethys WPS Server. Environ. Model. Softw. 115, 38–50. https://doi.org/https://doi.org/10.1016/j.envsoft.2019.01.021

Rajib, M.A., Merwade, V., Kim, I.L., Zhao, L., Song, C., Zhe, S., 2016. SWATShare - A web platform for collaborative research and education through online sharing, simulation and visualization of SWAT models. Environ. Model. Softw. 75, 498–512. https://doi.org/10.1016/j.envsoft.2015.10.032

Silliman, S.E., Hamlin, C., Crane, P.E., 2008. International collaborations and incorporating the social sciences in research in hydrology and hydrologic Engineering. J. Hydrol. Eng. 13, 13–19. https://doi.org/10.1061/(ASCE)1084-0699(2008)13:1(13)

Souffront Alcantara, M.A., Nelson, E.J., Shakya, K., Edwards, C., Roberts, W., Krewson, C., Ames, D.P., Jones, N.L., Gutierrez, A., 2019. Hydrologic Modeling as a Service (HMaaS): A new approach to address hydroinformatic challenges in developing countries. Front. Environ. Sci. 7, 158. https://doi.org/10.3389/fenvs.2019.00158

Stagge, J.H., Rosenberg, D.E., Abdallah, A.M., Akbar, H., Attallah, N.A., James, R., 2019. Assessing data availability and research reproducibility in hydrology and water resources. Scientific Data, 6, 190030, https://doi.org/10.1038/sdata.2019.30

Sun, A. Y., 2013. Enabling collaborative decision-making in watershed management using cloud-computing services. Environ. Model. Softw. 41, 93–97. https://doi.org/https://doi.org/10.1016/j.envsoft.2012.11.008

Sun, A.Y., Jeong, H., González-Nicolás, A., Templeton, T.C., 2018. Metamodeling-based approach for risk assessment and cost estimation: Application to geological carbon sequestration planning. Comput. Geosci. 113, 70–80. https://doi.org/https://doi.org/10.1016/j.cageo.2018.01.006

Sun, A.Y., Miranda, R.M., Xu, X., 2015. Development of multi-metamodels to support surface water quality management and decision making. Environ. Earth Sci. 73, 423–434. https://doi.org/10.1007/s12665-014-3448-6

Swain, N.R., Christensen, S.D., Snow, A.D., Dolder, H., Goharian, E., Jones, N.L., Nelson, E.J., Espinoza-d, G., Ames, D.P., Burian, S.J., 2016. A new open source platform for lowering the barrier for environmental web app development. Environ. Model. Softw. 85, 11-26. https://doi.org/10.1016/j.envsoft.2016.08.003

Tarboton, D.G., 1997. A new method for the determination of flow directions and contributing areas in grid digital elevation models. Water Resour. Res. *33*, 309-319, http://dx.doi.org/10.1029/96WR03137

Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Heard, J., Ames, D., Goodall, J.L., Band, L., Merwade, V., Couch, A., Arrigo, J., Hooper, R., Valentine, D., Maidment, D., 2014. HydroShare: Advancing collaboration through hydrologic data and model sharing, in: 7th International Congress on Environmental Modelling and Software. International Environmental Modelling and Software Society (iEMSs).

Tarboton, D.G., Luce, C.H., 1996. “Utah Energy Balance Snow Accumulation and Melt Model (UEB),” Computer model technical description and users guide. Utah Water Research Laboratory and USDA Forest Service Intermountain Research Station.

Thornton, P.E., Thornton, M.M., Mayer, B.W., Wei, Y., Devarakonda, R., Vose, R. S., Cook, R.B., 2016. Daymet: Daily surface weather data on a 1-km grid for north america. ORNL Distributed Active Archive Center, v3. https://doi.org/10.3334/ORNLDAAC/1328

Walker, J.D., Chapra, S.C., 2014. A client-side web application for interactive environmental simulation modeling. Environ. Model. Softw. 55, 49–60. https://doi.org/10.1016/j.envsoft.2014.01.023

Zender, C.S., 2008. Analysis of self-describing gridded geoscience data with netCDF Operators (NCO). Environ. Model. Softw. 23, 1338–1342. https://doi.org/10.1016/j.envsoft.2008.03.004