CyberWater: An Open Framework for Data and Model Integration in Water Science and Engineering

Ranran Chen† Feng Li Drew Bieger Fengguang Song Yao Liang*
Indiana University-Purdue University
Indianapolis, IN, USA

Daniel Luna† Ryan Young Xu Liang*
Dept. of Civil & Environmental Engineering
University of Pittsburgh
Pittsburgh, PA, USA

Sudhakar Pamidighantam
Indiana University
Bloomington, IN, USA

ABSTRACT

The CyberWater project is to build an open-data open-model framework for easy and incremental integration of heterogeneous data sources and diverse scientific models across disciplines in the broad water domain. The CyberWater framework extends the open-data open-model framework called Meta-Scientific-Modeling (MSM) that provides a system-wide data and model integration platform. On top of MSM, the CyberWater framework provides a set of toolkits, and external system integration engines, to further facilitate users' scientific modeling and collaboration across disciplines. For example, the developed generic model agent toolkit enables users to integrate their computational models into CyberWater via graphical user interface configuration without coding, which further simplifies the data and model integration and model coupling. CyberWater adopts a graphical scientific workflow system, VisTrails, ensuring data provenance and reproducible computing. CyberWater supports novel access to high-performance computing resources on demand for users' computational expensive model tasks. We demonstrate the merits of CyberWater by a use case of hydrologic modeling workflow.

CCS CONCEPTS

• **Information systems → Information retrieval**
• **Information systems → Information integration**

KEYWORDS: Open data and model framework, Model coupling without coding, Scientific workflow, HPC access on demand.

ACM Reference format:

Ranran Chen, Daniel Luna, Feng Li, Ryan Young, Drew Bieger, Fengguang Song, Sudhakar Pamidighantam, Yao Liang and Xu Liang. 2022. CyberWater: An Open Framework for Data and Model Integration in Water Science and Engineering. In *Proceedings of the 31st ACM Int'l Conference on Information and Knowledge Management (CIKM'22), Oct. 17-21, 2022, Atlanta, GA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/ 3511808.3557186

† Equal contribution.

* Corresponding authors (yaoliang@iupui.edu, xuliang@pitt.edu).

1 INTRODUCTION

To tackle large-scale scientific questions related to the health and resilience of the Earth, scientists need to be able to integrate diverse data and models across disciplines for modeling analyses and predictions. While there exist numerous modeling systems (as discussed in [1], [2]), they usually do not support direct access to diverse online data sources. It is also very difficult to integrate individual computational models with one another through current modeling systems which require writing "glue" code to couple each pair of the models. As existing domain models have usually been developed in isolation, writing such "glue" code is not only time-consuming but also error-prone. Besides, it is not scalable because of the pairwise nature of the "glue" interface coding. To overcome this challenge and make effective use of various available data and models across disciplines to improve our fundamental understanding of the complex behaviors and interactions of processes in the Earth system under climate changes, the CyberWater project develops an open-data open-model framework based on a workflow management system (WMS). A workflow management system provides a platform for users to describe the data dependencies of tasks, compose workflows, and launch such workflows for execution in designated computing environments. CyberWater utilizes the novel Meta-Scientific-Modeling (MSM) [1] to address the challenges of accessing heterogeneous data sources and integrating individual models in a systematic manner. Namely, instead of pairwise integration, the MSM (and its implementation *msm*) offers a general platform to support open and easy integration of diverse data and models universally. Once a model is integrated into the *msm* system via its model agent, it can be coupled with any other models already integrated into the system without developing a new pairwise integration code (see Figure 1(b)); similarly, once a data source is integrated into the *msm* system via its data agent, it can be directly accessible by any models already integrated into the system without any additional effort on data preparation or data preprocessing for these models. Furthermore, the *msm* framework supports model integration at the information level rather than at the coding level, which means that in Figure 1(b) each model's integration agent only deals with that model's input/output information rather than the model's code. As a result, the model's code is not required nor recompiled to be integrated with the *msm* framework. This does not only simplify the integration complexity but is also desirable when the model's source code is not available (e.g., the integration of commercial modeling software).

In this paper, we present and demonstrate the current CyberWater(http://www.hydroshare.org/resource/66b715b3805b476699c319a7c562f4f3) framework software system which significantly extends the *msm* framework. CyberWater now includes a set of facilities: generic model agent toolkit [2] and static parameter toolkit [3], system integration engines, and high-performance computing on demand [4]. The generic model agent toolkit enables users to construct their model agents for integration via parameter-based configuration without coding for most environmental and hydrological models. The static parameter agent toolkit aids users in creating and preparing various parameter files necessary for their models easily. The development of toolkits makes CyberWater more accessible to the general scientific community and for broad earth science communities. The developed system integration engines allow other popular system platforms such as GIS and MATLAB to be connected to the CyberWater framework as needed. Our mechanism of on-demand access to HPC platforms enables the user to offload large-scale computing tasks to remote HPC resources effectively and efficiently.

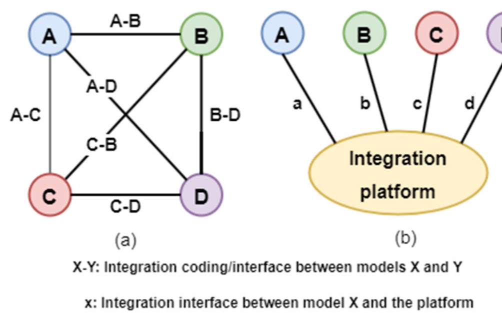
A-B
A
B
A-D
A-C
B-D
C-B
C
D
C-D
(a)
A
B
C
D
a
b
c
d
Integration platform
(b)
X-Y: Integration coding/interface between models X and Y
x: Integration interface between model X and the platform

Fig. 1: An illustration of pair-wise model integration vs. system-wise model integration. (a) Pair-wise integration, with the complexity of *O(n^2)* coupling interfaces for *n* models; (b) system-wise integration with the linear complexity of *O(n)* coupling interfaces.

2 ARCHITECTURE

The overall architecture of CyberWater framework is shown in **Figure 2**, which extends the *msm* framework with the toolkits, the integration engine interfaces for external systems, and the HPC launch agent for HPC access on demand. They will be discussed briefly as follows.

2.1 MSM

The *msm* framework interconnects with VisTrails [5], the adopted workflow engine to provide GUI-based workflows with data provenance and reproducibility. There are three major components in the *msm* system:

2.1.1 System Core. It is the core of all the components in our open-data open-model framework. In addition to interfacing with the workflow engine, the system core connects with various online data sources via data agents and users' computational models with corresponding model agents in a workflow environment.

2.1.2 Data Agents. A data agent is designed to handle the data access protocols, metadata standards, and source-specific implementations of a remote online data source. They connect to external data sources over the Internet, dynamically retrieve the data, and store them in the cache at the run time.

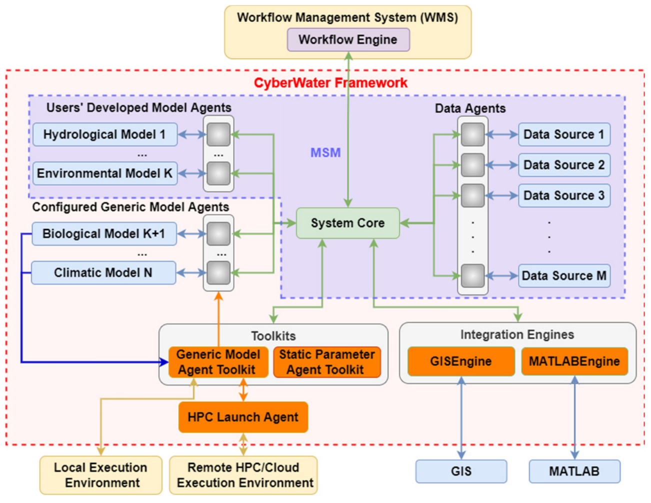

Fig. 2: The overall architecture of the CyberWater framework

2.1.3 Users' Model Agents. A model agent receives the inputs from the workflow, transforms them into the input files with the format needed by the user's model, and then invokes the external models. After execution, the model agent reads the result output files and transforms them into the workflow for visualization or the next process task in the CyberWater system.

2.2 Generic Model Agent Toolkit

This toolkit allows users to construct their model agents via workflow GUI configurations without writing code, which further significantly enhances the usability of the CyberWater framework. It has six template-based generic components for users to configure their specific models' inputs/outputs and execution environment:

MainGenerator. Set up a working directory for performing the user's model simulation.

AreaWiseParamGenerator. Organize parameter files and place the imported parameter files into the parameter folder created by the user.

ForcingDataFileGenerator. Create the forcing data brought in by the *MainGenerator* for the user's model.

InitialStateFileGenerator. Organize the data of initial states for the user's model. As the initial state files are not always required for the execution of a given model, it is optional in the workflow.

RunModuleAgent. Invoke the user's model on the local machine and retrieve the model's outputs, where the model is executed in the same way as the model runs alone without the CyberWater system environment. The user's computational model can be given as either a JAVA file/JAVA Archive (JAR) file, a Python code, or a binary executable compiled from the source codes written in C/C++ or FORTRAN. To support users' models coded in MATLAB, the RunModuleAgent module exposes an input "engine" port to be linked with our developed MATLABEngine module in the workflow. Then, the *RunModuleAgent* module writes the results back to the CyberWater system for the workflow to continue.

HPC. Access to remote High-Performance Computing (HPC) facilities on demand is enabled to provide HPC capacity for executing the user's model by seamlessly connecting to either academic supercomputers/clouds or commercial cloud platforms.

2.3 Static Parameter Agent Toolkit

This toolkit includes a suite of modules that are designed to aid users in easily creating and preparing various parameter files necessary for their models. These modules enable users to efficiently create their own parameter files on demand, significantly overcoming the hassle of conventional error-prone, time-consuming, and tedious manual processing and preparation of model parameter files. The utilization of the developed static parameter agent toolkit of CyberWater has been demonstrated in creating key parameter files for different models in geosciences, such as the VIC, DHSMV, and CASA-CNP models.

2.4 System Integration Engines

In the area of hydrology, GIS system, such as Geographic Resources Analysis Support System (GRASS), is widely applied for geospatial data management and analysis and producing graphics and maps as parameter files for models' computation. In addition, MATLAB has also been widely used in various scientific disciplines for model development. In view of this, our CyberWater framework further provides two system integration engines for integrating GIS GRASS and MATLAB with CyberWater. With this framework and integration approach, more external systems can be similarly integrated into CyberWater in the future.

2.4.1 GIS Engine. It includes two modules, *GISEngine* and *GISRunModuleAgent*. The former is aimed to create a single object of GIS GRASS for the successive-connected modules to make use of GIS functions, while the latter is to directly execute the command of GIS GRASS in the CyberWater workflow GUI.

2.4.2 MATLAB Engine. We import *matlab.engine* package into CyberWater, which enables users to start a MATLAB process, pass data, and call functions executed by MATLAB. Moreover, The MATLABEngine module is developed to allow users to simply configure the working location of the MATLAB model in the *RunModuleAgent* module and to connect the MATLABEngine module to the input "*engine*" port of the *RunModuleAgent* module. The MATLABEngine module will start a MATLAB session to execute the specified model once the entire workflow begins executing. Once the workflow finishes, the MATLAB session will close, and the MATLAB engine will stop automatically.

2.5 HPC Access on Demand

CyberWater is a standalone system installed and executed on local machines for scalability. Accessing HPC resources *on demand* is a novel aspect of CyberWater, which allows users to offload computational expensive tasks to remote HPC platforms including supercomputers and clouds. The HPC module described in Section 2.2.1 is supported by our development of *LaunchAgent* to access remote computing resources in this regard [4], so that the users can compose and configure their workflows with HPC module's GUI for their task offloading. *LaunchAgent* supports both direct Slurm-based [6] and Airavata gateway [7] connections to remote computing resources, that is, the users can access both mid-size campus-level clusters and large-size grid computing resources. User's jobs will be submitted to the HPC resources automatically when executing an HPC module in the user's workflow with CyberWater, and the results will be returned automatically to the workflow when the user's jobs on the remote HPC are completed.

3 DEMONSTRATION

To make an easy installation of CyberWater for end users, the software and all of its dependencies can be installed with a simple guided installer. As a demonstration of some of the functionality of CyberWater, a workflow aiming to study the Princeton Millstone River watershed that incorporates elements from the msm package, the generic model agent toolkit, and the static parameter agent toolkit is constructed to execute the VIC4 model (version 4.0 of the Variable Infiltration Capacity model) [8] coupled with routing. Although the special ***VicAgent*** and ***RoutingAgent*** modules are both available in msm, these model agents will be reconstructed solely by using our generic model agent toolkit to demonstrate the toolkit's functionality in integrating both models into CyberWater.

The ***TimeRange*** and ***SpaceRange*** modules are used to establish the temporal and spatial extent of the simulation. In our case study, it is run from January 1st, 2010, to January 1st, 2011, within the eastern, western, northern, and southern bounds of -74.37, -74.89, 40.55, and 40.19, respectively. To download the forcing data for VIC4, the ***NCALDASAgent*** module included within msm is used. This module is used four times to download wind speed, total precipitation, and minimum and maximum temperature. The ***PasswordDialog*** module is also used, allowing the user's NASA Earth Data credentials to be collected at runtime instead of stored within modules. To finalize the forcing data, the ***msmUnitConversion*** module is used three times to convert the temperatures to Celsius and precipitation to millimeters per day.

The parameter files needed by VIC4 can be prepared using tools from the static parameter agent toolkit. Using the ***ParameterEntry*** module repeatedly with the ***GridParameterAgent*** allows the soil parameter file to be created entirely within CyberWater. Because the VIC vegetation parameter file is nonstandard, the specialized tool ***VICVegetationParamAgent*** is included in the toolkit to allow the vegetation file to be built within the workflow. This module requires the ***GISEngine*** module to be able to hook into GRASS GIS and exercise its functionality. The remaining required parameter files are set up using the generic model agent toolkit.

To execute VIC4, the user needs to first implement the ***MainGenerator*** module from the generic model agent toolkit. This module establishes the working directory and coordinates the remaining generic agent modules. The ***ForcingDataFileGenerator*** module is used here to build the forcing data directory, and the

AreaWiseParamGenerator to build the parameter file directory. This module is also where the additional parameter files are brought in, and where the static parameter agent toolkit modules hook into the generic model agent tools. Finally, the ***RunModuleAgent*** executes the model and reads the resulting model output. Users can execute this workflow after connecting two ***msmShowChart*** modules to the output of the ***RunModuleAgent*** to view the surface runoff and baseflow predicted by VIC4. The workflow up to this point can be seen in **Figure 3**, and its resulting output in **Figure 4** with default values.

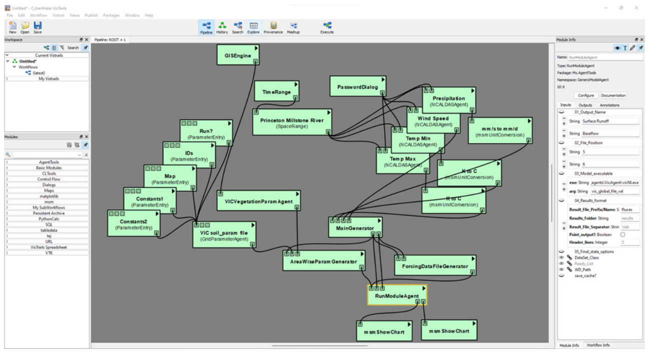

Fig. 3: Workflow where the VIC4 model is integrated and executed using toolkits.

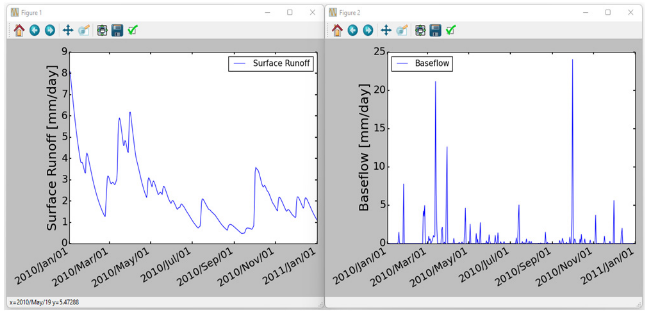

Fig. 4: Resulting surface runoff and baseflow predicted by the execution of the workflow shown in Figure 3.

Integrating the coupled routing model requires similar steps to integrating VIC4. To prepare the parameter files, we will need a digital elevation model (DEM) of the study area. The ***DEMAgent***, a data gent, included within msm automatically downloads and crops DEM data which is used here. The ***StaticDataSetToFile*** module is used to save the DEM data to an ASCII file, and a ***PythonSource*** module is used to open the file in CyberWater. The PythonSource module allows small bits of python to be run and is here used for the file to be read by ***msmComputeExactOutput***, which finds the exact coordinates of the outlet of the watershed. This exact outlet location and the DEM are then used to create the nonstandard routing parameter file using the specialized ***RoutingParameterFile*** module which is part of the static parameter agent toolkit, the only parameter file required by the model.

To execute the routing model, the generic model agent toolkit is again used. The ***MainGenerator*** constructs the working directory, the ***ForcingDataFileGenerator*** builds the forcing data directory from the output of VIC4, the ***AreaWiseParamGenerator*** creates the parameter file directory from the output of the RoutingParameterFile module, and the ***RunModuleAgent*** executes the module and reads the output.

To provide a comparison, the ***UsgsAgent*** can be used with a limited SpaceRange that only includes one USGS station to download the real streamflow at the outlet being simulated. By connecting the ***UsgsAgent*** (a data agent) and the ***RunModuleAgent*** to the same ***msmShowChart*** module, the data are plotted together. The complete workflow is shown in **Figure 5**, and the resulting streamflow with default values compared to the actual streamflow is shown in **Figure 6**.

ACKNOWLEDGMENTS

This work was supported by the U.S. NSF under [OAC-1835817] to Indiana University-Purdue University Indianapolis (IUPUI), and [OAC-1835785] to the University of Pittsburgh, respectively. It also used HPC and Cloud resources provided by the XSEDE [9].

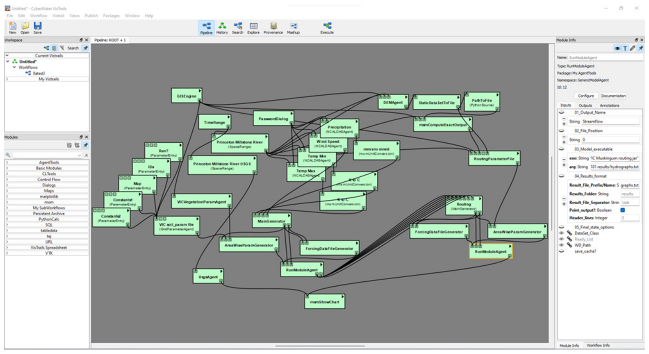

Fig. 5: Workflow where the VIC4 and routing models are coupled and executed using toolkits.

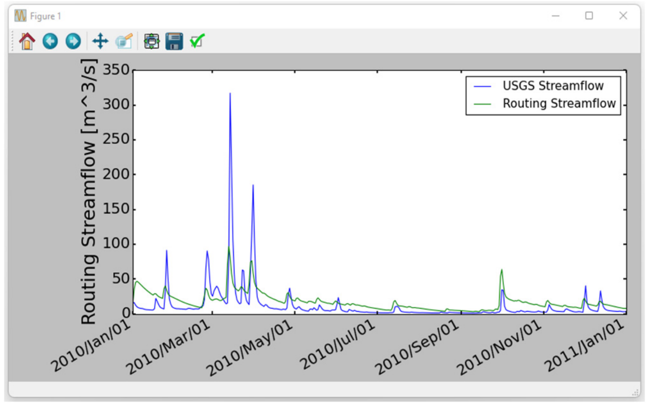

Fig. 6: Resulting streamflow predicted by the execution of the workflow shown in Figure 5, plotted against the actual data downloaded from USGS.

REFERENCES

[1] Daniel Salas, Xu Liang, Miguel Navarro, Yao Liang, and Daniel Luna, 2020. An open-data open-model framework for hydrological models' integration, evaluation and application, Environmental Modelling & Software, Vol. 126, 104622. DOI: https://doi.org/10.1016/j.envsoft.2020.104622.

[2] Ranran Chen, Daniel Luna, Yuan Cao, Yao Liang, and Xu Liang, 2022. Open data and model integration through generic model agent toolkit in CyberWater framework, Environmental Modelling & Software, Vol. 152, 105384. DOI: https://doi.org/10.1016/j.envsoft.2022.105384.

[3] Daniel Luna, Ranran Chen, Ryan Young, Yao Liang, and Xu Liang, 2021. Static Parameter Agents of CyberWater Framework for Geoscience Data and Model Integration, In American Geophysical Union Fall Meeting(hybrid), New Orleans, LA, USA.

[4] Feng Li, Ranran Chen, Yuankun Fu, Fengguang Song, Yao Liang, Isuru Ranawaka, Sudhakar Pamidighantam, Daniel Luna, and Xu Liang, 2021. Accelerating complex modeling workflows in CyberWater using on-demand HPC/Cloud resources, 2021 IEEE 17th International Conference on eScience (eScience), 196-205. DOI: https://doi.org/10.1109/eScience51609.2021.00030.

[5] Steven P Callahan, Juliana Freire, Emanuele Santos, Carlos E Scheidegger, Cl´audio T Silva, and Huy T Vo, 2006. Vistrails: visualization meets data management. In Proceedings of the 2006 ACM SIGMOD international conference on Management of data, 745–747.

[6] Andy B Yoo, Morris A Jette, and Mark Grondona, 2003. Slurm: Simple linux utility for resource management, In Workshop on job scheduling strategies for parallel processing, 44–60.

[7] Suresh Marru, Lahiru Gunathilake, and Chathura Herath, 2011. Apache airavata: a framework for distributed applications and computational workflows, In Proceedings of the 2011 ACM workshop on Gateway computing environments, 21–28.

[8] Xu Liang, Dennis P. Lettenmaier, Eric F. Wood, and Stephen J. Burges, 1994. A simple hydrologically based model of land surface water and energy fluxes for general circulation models, Journal Of Geographical Research Atmospheres, Vol. 99, 14415-14428. DOI: https://doi.org/10.1029/ 96JD01448.

[9] John Towns, Timothy Cockerill, Maytal Dahan, Ian Foster, Kelly Gaither, Andrew Grimshaw, Victor Hazlewood, Scott Lathrop, Dave Lifka, Gregory D. Peterson, Ralph Roskies, J. Ray Scott, and Nancy Wilkins-Diehr, 2014. XSEDE: Accelerating Scientific Discovery, Computing in Science & Engineering, Vol.16, 62-74. DOI: https://doi.org/10.1109/MCSE.2014.80.