GIScience & Remote Sensing

ISSN: 1548-1603 (Print) 1943-7226 (Online) Journal homepage: https://www.tandfonline.com/loi/tgrs20

A loosely integrated data configuration strategy for web-based participatory modeling

Songshan Yue, Min Chen, Chaowei Yang, Chaoran Shen, Bowen Zhang, Yongning Wen & Guonian Lü

To cite this article: Songshan Yue, Min Chen, Chaowei Yang, Chaoran Shen, Bowen Zhang, Yongning Wen & Guonian Lü (2019) A loosely integrated data configuration strategy for web-based participatory modeling, GIScience & Remote Sensing, 56:5, 670-698, DOI: 10.1080/15481603.2018.1549820

To link to this article: https://doi.org/10.1080/15481603.2018.1549820

Published online: 21 Nov 2018.

GIScience & Remote Sensing, 2019
Vol. 56, No. 5, 670–698, https://doi.org/10.1080/15481603.2018.1549820

A loosely integrated data configuration strategy for web-based participatory modeling

Songshan Yue[a,b,c,d], Min Chen*[a,b,c], Chaowei Yang[d], Chaoran Shen[a,b,c], Bowen Zhang[a,b,c], Yongning Wen[a,b,c] and Guonian Lü[a,b,c]

[a]*Key Laboratory of Virtual Geographic Environment, Ministry of Education, Nanjing Normal University, Nanjing 210023, China;* [b]*State Key Laboratory Cultivation Base of Geographical Environment Evolution (Jiangsu Province), Nanjing Normal University, Nanjing 210023, China;* [c]*Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing Normal University, Nanjing 210023, China;* [d]*NSF Spatiotemporal Innovation Center and Department of Geography and Geoinformation Science, George Mason University, Fairfax, VA 22030-4444, USA*

(*Received 26 May 2018; accepted 8 November 2018*)

Participatory modeling is an important approach for solving complex geo-problems from a comprehensive and holistic viewpoint, and it brings together stakeholders from multiple disciplines to provide diverse resources, including modeling, data fields and computational assets. Data configuration work (e.g., preparing appropriate input data for model execution, connecting a model's output to the input data of another model) is important for constructing and executing a participatory modeling task. Most current data configuration methods depend on the model integration logic, which presents a challenge when adding new modeling resources into a model to dynamically create and execute new modeling tasks. To support the construction of participatory modeling tasks in a web environment, this article proposes a loosely integrated data configuration strategy for decoupling data configuration work from the execution process of a participatory modeling task. A model service controller is designed for model input/output (I/O) configuration, and a data service controller is designed for data access configuration. These two controllers can help modelers link the data I/O demands of a model-service with the appropriate data-services; thus, different modeling instances can be dynamically joined to a participatory modeling task and executed without reconstructing the original data configuration settings. A prototype participatory modeling system is proposed to demonstrate the flexibility and feasibility of the proposed method using an experimental modeling case. The results show that the proposed data configuration strategy supports the integration of different model-services based on the data dependency relationships and that the complexity and difficulty in configuring data for a participatory modeling tasks in the web environment are minimized.

Keywords: data configuration; participatory modeling; model service; data service; web environment

1. Introduction

Geo-analysis models are necessary tools for understanding various geo-processes and phenomena of the earth's environment (Ward, Murray, and Phinn 2000; Al-Sabhan, Mulligan, and

*Corresponding author. Email: chenmin0902@163.com
Songshan Yue, Min Chen and Chaowei Yang conceived and designed the experiments; Bowen Zhang, Yongning Wen and Guonian Lü contributed experimental materials and analysis tools; Songshan Yue, Min Chen and Chaoran Shen wrote the paper.

Blackburn 2003; Selvam et al. 2014; Fonseca et al. 2014; Gutiérrez, Snell, and Bugmann 2016; Shahparakia et al. 2017). With the advancement of Integrated Environmental Modeling (IEM) studies, it is widely accepted that sharing and integrating different geo-analysis models can help researchers solve complex geo-problems from a holistic and systemic viewpoint (Bastin et al. 2013; Laniak et al. 2013; Zhang et al. 2016; Moore and Hughes 2017). A range of studies about the integration of Geographic Information Science (GIS), Remote Sensing (RS), and geo-analysis models have been conducted in various disciplines, such as terrain analysis (Xie, Pearlstine, and Gawlik 2012; Deng et al. 2017), land use modeling (Rodrigues 2016; Omrani, Tayyebi, Pijanowski 2017and Zhai et al. 2018), water resource analysis (Chen et al. 2014), and GIS-based decision-making research (Nascimento et al. 2017; Cenci et al. 2018; Eisman, Gebelein, and Breslin 2017). Among the efforts in IEM and related research, participatory modeling is an important approach for addressing the dynamic complexities of geo-problems (Voinov and Gaddis 2008; Langsdale et al. 2009; Addison, Bie, and Rumpff 2015; Paolisso and Trombley 2017).

Generally, participants engaged in a participatory modeling task play a range of different roles, such as modeling scientist, decision-maker, policy-maker, and nonscientist (Sandker, Campbell, and Suwarno 2008; Voinov and Bousquet 2010; Seidl 2015; Videira, Antunes, and Santos 2017). The rapid development of Information Technology (IT) has promoted a trend in participatory modeling that involves models being constructed and conducted in a web environment to help participants from different places provide various modeling-resources (e.g., model resources, data resources, computational resources) as reusable web services. By constructing a web-based participatory modeling task, participants can collaboratively work together to solve geo-problems by integrating these modeling-resources (Mustajoki, Hämäläinen, and Marttunen 2004; Voinov et al. 2016). In this article, participants in a web-based participatory modeling task are regarded as both modeling-resource providers and geo-problem solvers.

The ability to achieve participatory modeling targets mainly relies on building a software-based computational system to organize and integrate models, data, computers, and expert elicitations (Argent 2004; Liu et al. 2008; Bergez et al. 2013; Belem and Saqalli 2017). When using such a computational system to accomplish participatory modeling tasks, the input/output (I/O) data of an involved model are the main source for participants to link the model with specific geo-problems or practical decision-making. In addition to the important role that configuring the appropriate input data plays in a single model's execution, when integrating multiple models, the configuration of data transfer among these models is also crucial, as one model's input could be another model's output. To address the data configuration issue, current efforts in participatory modeling can be categorized into two groups based on their software implementation method: centralized data configuration methods and distributed data configuration methods.

Centralized data configuration methods are designed in studies of a centralized, integrated modeling framework. Examples of integrated modeling frameworks include the Earth System Modeling Framework (ESMF) (Hill et al. 2004), the Open Modeling Interface (OpenMI) (Gregersen, Gijsbers, and Westen 2007), the European Union's Program for Integrated Earth System Modeling (PRISM) (Guilyardi, Budich, and Valcke 2003), the Object Modeling System (OMS) (David et al. 2013) and a range of other initiatives (Rao, Rubin, and Berkenpas 2004; Fischer et al. 2010; Welsh et al. 2013). Within this mode, models are wrapped as a range of different model components and are executed in a standalone computational platform (Argent 2004; Laniak et al. 2013). A model component in an integrated modeling framework is regarded as a

software module with specific calling and data I/O interfaces according to the specifics of the modeling framework. Since model components within a certain integrated modeling framework are interdependent and somewhat fixed, different model components are wrapped by the same data I/O interfaces so that data configuration is unified (David et al. 2013; Granell, Schade, and Ostländer 2013). By utilizing a centralized modeling framework, modelers can conveniently couple various model components. However, because the data I/O interfaces of each model component tightly bind with the modeling framework, heterogeneous models (with different data I/O modes) still have difficulty joining the integrated modeling process (Peckham, Hutton, and Norris 2013; Whelan et al. 2014). The centralized characteristics also make it difficult for researchers from distributed areas to work together.

Distributed data configuration methods are most often designed in service-based model sharing and integrated studies. Sharing and integrating models as distributed web services can promote the accessibility of modeling-resources (Goodall, Robinson, and Castronova 2011; Nativi, Mazzetti, and Geller 2013; Li et al. 2017). Benefiting from the openness of the World Wide Web, modelers from widely distributed areas can work together. Along with the development of Service-Oriented Architecture (SOA) and geospatial information interoperation technologies, many related studies are conducted for "converting models to reusable model-services" (Wen et al. 2013; Zhao, Foerster, and Yue 2012; Santoro, Nativi, and Mazzetti 2016; Wen et al. 2016). A prominent example is the Web Processing Service (WPS) standard, which was developed by the Open Geospatial Consortium (OGC). Within a WPS-based model-service, the I/O data for model execution are described using the Extensible Markup Language (XML) (Lanig et al. 2008; Castronova, Goodall, and Elag 2013). Other Web Service Description Language (WSDL)-based studies have been conducted for sharing models as services, such as the Community Surface Dynamic Modeling System (CSDMS) (Peckham, Hutton, and Norris 2013; Jiang et al. 2017) and the Open Geographic Modeling and Simulation project (OpenGMS) (Wang et al. 2018; Zhang et al. 2018; Chen et al. 2018). Both WPS-based and WSDL-based model-services employ XML as the medium to help users understand the models' I/O data demands. By chaining, orchestrating, or mediating different model-services, integrated modeling systems can be constructed to help modelers solve complicated geo-problems (Meng, Bian, and Xie 2009; de Jesus et al. 2012; Giuliani et al. 2012; Sun, Yue, and Di 2012; Belete, Voinov, and Laniak 2017). In such a web-based integrated modeling system, data configuration is conducted based on "uploading data (or transferring data among servers) and then obtaining results." However, the data configuration processes among different model-services remain a problem. A model involved in an integrated modeling task usually includes a series of computational steps, which requires modelers to prepare all data in different steps and package them as a single "model input" to drive model execution. This data configuration process limits the designed participatory modeling tasks that are conducted in a static context, and different participants cannot join the modeling process dynamically.

In summary, previous centralized and distributed data configuration methods treat the execution of a participatory modeling task as a fixed workflow. Moreover, the model calling (control flow) and data transmission (data flow) processes are tightly combined when integrating model-services (Granell, Díaz, and Gould 2010; Zhang, Bu, and Yue 2017; Herle and Blankenbach 2018). Thus, data configuration must be redone once the integration logic is modified, especially when new participants join the modeling task and new models and data are integrated. Targeting these problems, this article focuses on reducing the difficulties in data configuration work when executing a web-based participatory modeling

task. By separating the data flow from the control flow, an architectural strategy is presented for data configuration. The proposed data configuration strategy reduces the difficulties of using model-services and helps participants integrate multiple model-services and data-services in a more dynamic manner.

Through this study, an alternative for data configuration is introduced as a contribution to web-based participatory modeling studies. Existing data configuration approaches have mainly focused on supporting the execution of a well-constructed modeling solution; however, exploring and constructing a reasonable modeling solution were less considered. To solve a complex geo-problem, a modeling solution generally requires several rounds of improvements and modifications based on communication among participants. Previous modeling practices regard data configuration as the static settings of models; thus, these settings are difficult to integrate with the dynamic construction process of a participatory modeling task. By using the proposed model service controller and data service controller, data configuration work can be decoupled from the execution process of a participatory modeling task, which enables modelers to work collaboratively in a more flexible manner.

2. Participatory modeling and data configuration based on web services

Participatory modeling brings stakeholders together for environment simulation and geo-problem solving (Jonsson et al. 2007; Voinov and Gaddis 2008; Gray et al. 2012; Scheer et al. 2017). Along with the development of geographic modeling, there are various types of participatory modeling. Some participatory modeling tasks focus on bringing together scientists from a specific discipline to construct a complex model or build a decision-making tool (Antunes 2006; Beall and Zeoli 2008; Robinson and Fuller 2017); some focus on collecting knowledge from both scientists and nonscientists to help in building, evaluating and improving a model (Jones et al. 2009; Falconi and Palmer 2017); and some focus on collecting modeling resources to form a community-based modeling framework (Mendoza and Prabhu 2005, 2006; Jankowski 2009). Although the modeling target and the implementation method differ among these studies, the underlying idea is knowledge sharing and resource integration. This article focuses on web-based participatory modeling, which collects service-based modeling resources in the web environment and conducts modeling tasks by integrating different model-services and data-services. Participants in web-based participatory modeling are model resource providers (provide models as executable model-services), data resource providers (provide data as reusable data-services), computer resource providers (provide computer resources across the web) and geo-problem-solving participants (modelers from various disciplines, decision-makers, and the general public).

Based on the resources offered by providers, the model repository, data repository, and computer repository are formed. In the web environment, models and data are deployed on different computers as model-services and data-services, respectively. By employing these modeling-resource repositories, geo-problem-solving participants collaborate in defining modeling targets, collecting modeling-resources, constructing the integration logic and conducting other related work. Through such collaborative work, the constructed participatory modeling task is implemented by integrating different model-services and data-services based on the scientific process. In this article, a participatory modeling task is considered an integrated modeling scenario in which information about the geographic environment (ranging from local to global scales) is organized and modelers from various disciplines explore modeling targets

and results from their perspectives. In this context, participants join a participatory modeling task dynamically, and new model-services and data-services are added to a participatory modeling task according to the demands of the participants.

The proposed data configuration strategy for participatory modeling involves two basic modules to mediate the involved model-services and data-services: (1) a model service controller for linking model-services with appropriate data-services and (2) a data service controller for managing and searching appropriate data-services for each involved model-service. These two controllers are used to create, integrate and organize different model-service and data-service instances. The data transmission between a model-service and a related data-service is conducted in a request–response mode. In contrast to the commonly used "assign data for a model" mode, the proposed data configuration strategy is conducted as follows: "a model execution instance requests data along with its computation process, and appropriate data-services are searched in the modeling task to respond to such a data request." In a participatory modeling task, the request–response process is conducted along with the execution of each involved model. By using the model service controller and the data service controller, modelers can be more focused on configuring the data demands of a model-service, and the required data-service is provided by other participants or generated by other model-services. Consequently, modelers can explore their modeling targets within the participatory modeling task.

Figure 1 illustrates the basic process of data configuration when using the model service controller and the data service controller in executing a modeling task. The involved model-services and data-services form the corresponding model-service collection and data-service collection for constructing a modeling task. When driving a modeling task to execution, the model service controller provides the data I/O configuration and the data service controller provides the data access configuration.

For the input data, when an execution step of a model-service requires certain data, five steps are effected: (1) the request message is passed to the model service controller; (2) the model service controller receives the request and notifies the data service controller to prepare the corresponding data; (3) the data service controller checks whether such data exist and returns the message that the requested message is accepted; (4) the data service controller asks for the data from the data-service collection in a modeling task asynchronously without blocking the execution process for the other model execution instances; and (5) once the content of the required data-service is ready, the data service controller notifies the model service controller, and the model execution instance receives the location of the corresponding data-service.

For the output data, once the model execution generates the intermediate or final results, two steps are effected: (1) result data are posted to the data I/O controller and the data access controller is notified; and (2) data are registered into the collection as new data-services that are accessible by other model-services.

The proposed data configuration strategy relies on the data I/O configuration and the data access configuration. The data I/O and access configurations are handled on the model and data side, respectively.

3. Design of the data configuration strategy

3.1. Data I/O configuration

A geo-analysis model can be regarded as a scientific logic-based executable program that usually involves multiple execution steps. These steps are organized as data I/O states in

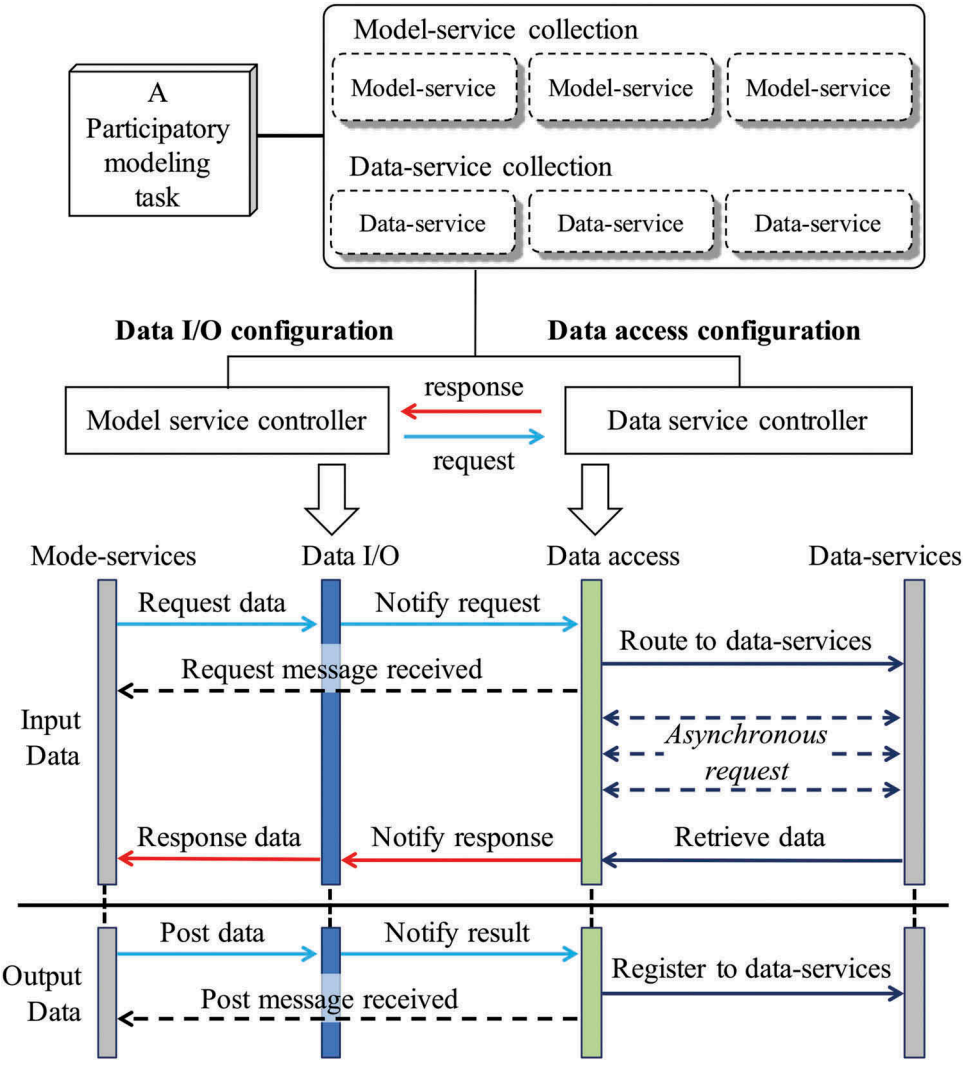


Figure 1. Basic process of data configuration for executing a modeling task.

the proposed method. The execution process consists of data I/O states (simple models may contain only one data I/O state), and computation is implemented by switching from one state to another. Each state contains corresponding data I/O events to transfer information between the model and the model service controller. A data I/O event is used to send or receive messages regarding a data request or a data response. Consequently, there are two types of data I/O events: the request data event (which requires data from the model service controller) and the response data event (which posts data to the model service controller).

In Figure 2(a), two models from the TauDEM (Terrain Analysis Using Digital Elevation Models) toolkit (http://hydrology.usu.edu/taudem/taudem5/) are employed to introduce the state-based execution process. TauDEM is a suite of Digital Elevation Model (DEM) tools for the extraction and analysis of hydrologic information from

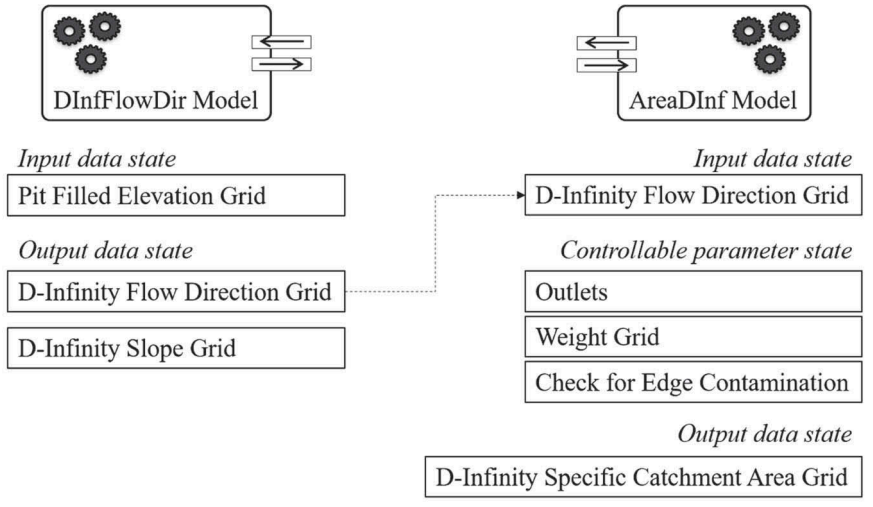
topography, as represented by a DEM, and it can be imported as a plugin to many GIS platforms (e.g., ArcGIS, QGIS, MapWindow). For the *DInfFlowDir* model, which is used to calculate flow directions based on the D-infinity flow method, two states exist: the input data state (which contains one data I/O event that requires a pit-filled elevation grid) and the output data state (which contains two data I/O events that post a D-infinity flow direction grid and a D-infinity slope grid, respectively). For the *AreaDInf* model, which is used to calculate a grid of a specific catchment area using the multiple flow direction D-infinity approach, three states exist: the input data state (which contains one data I/O event that requires a D-infinity flow direction grid), the controllable parameter state (which contains three data I/O events that require outlets, a weight grid and an edge contamination check flag, in series) and the output data state (which contains one data I/O event that posts a D-infinity specific catchment area grid).

The D-infinity flow direction grid generated by the *DInfFlowDir* model is used as the input data for the *AreaDInf* model. These two data I/O events are not directly connected. All data I/O events are processed by the model service controller, which is regarded as a web-based mediator for configuring the data I/O events of all involved models. The model service controller is constructed with (1) a web interface for the information transfer between model-services, (2) a web interface for the information transfer between the data service controller that manages all involved data-services, and (3) a "model execution instance collection" for managing the data configuration requirements of each model execution instance separately (Figure 2(b)). The model execution instance collection is constructed as a data I/O configuration document list, in which one document links to one model execution instance.

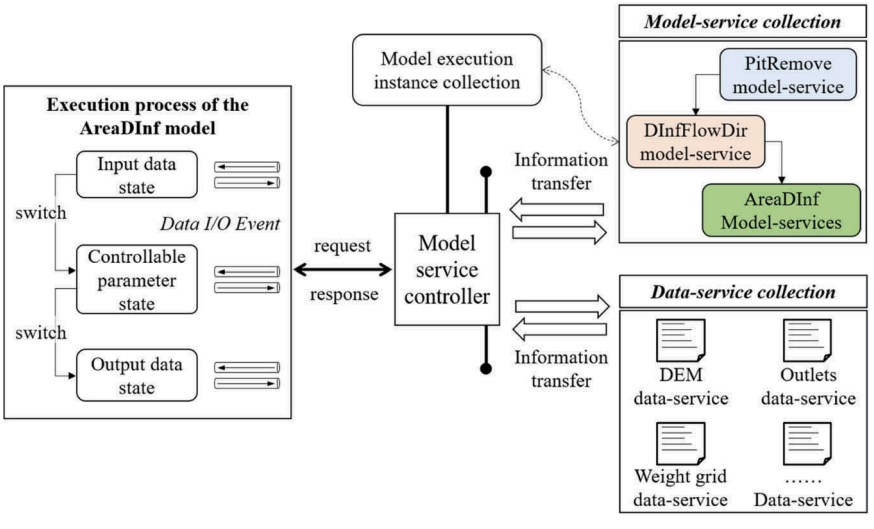
3.1.1. Data I/O configuration document for one model execution instance

In Figure 3, a sample data I/O configuration document for the AreaDInf model is illustrated. XML is employed to organize the data I/O configuration documents in a structured manner. There are four basic XML nodes in the document:

(1) The Instance node is the root node of the data I/O configuration document for one model execution instance. The Instance node contains a series of State nodes to represent the execution steps, and these execution steps are controlled by the model itself. In the Instance node, *id* indicates the unique identifier of a model execution instance, and *name* indicates the human-readable name to which this model execution instance relates.
(2) The State node indicates one execution step of the model execution instance. The State node contains a series of Event nodes to represent the data I/O demands within the corresponding execution step. In the State node, the *name* attribute is employed to indicate the human-readable name information of this execution step.
(3) The Event node is used to describe every data I/O demand in the model execution instance. An Event node owns a single Data node to link detailed data resources with the execution step. In the Event node, *name* indicates the human-readable name information of this data I/O event; *type* indicates whether the event is requesting input data or responding with output data.
(4) The Data node belongs to the parent Event node and is employed to represent the data description information and configured data resources. The Data node contains two attributes: the *description* attribute indicates how to understand the corresponding data content and the *content* attribute indicates where the reusable data resources can be accessed.



(a) State-based execution process



(b) Design of the model service controller

Figure 2. Model service controller: (a) the structure of the state-based execution process using TauDEM models as examples, (b) the basic design of a model service controller constructed with two web interfaces for model-service collection and data-service collection, and a model execution instance collection for managing integration.

The *description* attribute is a Unique Resource Link (URL) on the web. For example, the DescribeProcess operation in a WPS-based model service provides an XML document through a URL for describing its input and output data (Rautenbach, Coetzee, and Iwaniak 2013; Li et al. 2017). Other service-oriented data description approaches also describe a model's related data in the URL-based manner (Yue et al. 2015, 2016).

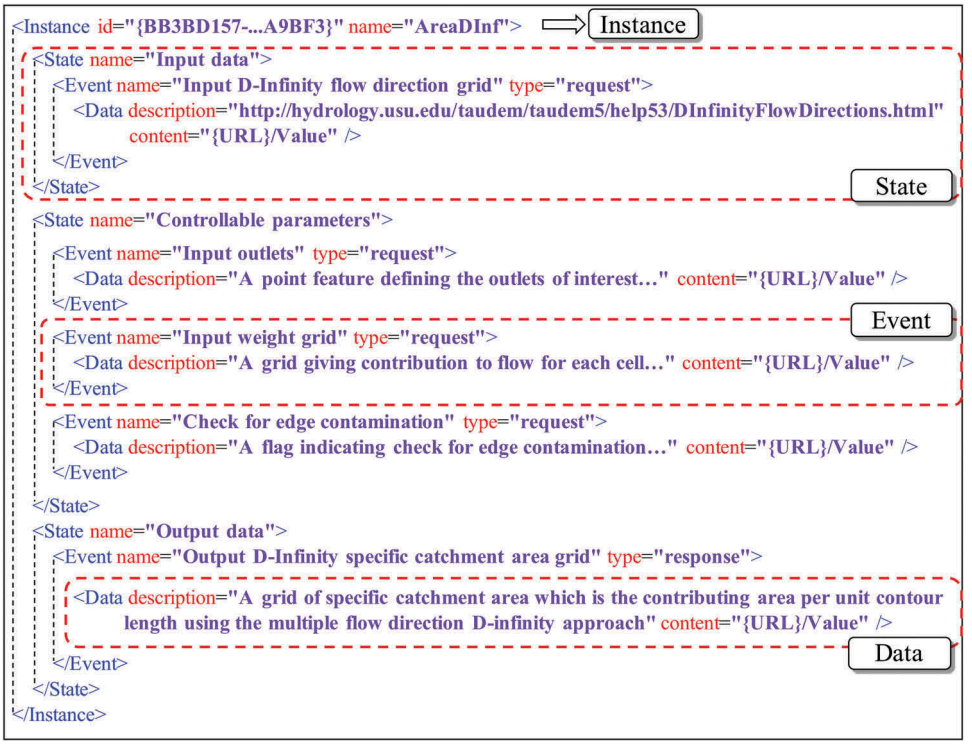


Figure 3. Structure of a typical data I/O configuration document; there are four basic XML nodes: Instance, State, Event and Data node.

The *description* attribute is also filled as one descriptive string, supporting the understanding of the data demands of the other participants in the modeling task.

For the *content* attribute, either a URL or the detailed value can be filled. If the related Event node is a request type, the content attribute is empty. The corresponding model execution step is executed only if the modeling participants provide an accessible URL or a rational value. The model service controller is responsible for checking the accessibility and rationality of the input data. If the related Event node is the response type, the model service controller generates one unique identifier for this output data; and the data are provided as a reusable data-service based on the computer on which this model execution instance is deployed.

3.1.2. Service-oriented method for wrapping data I/O events in a model

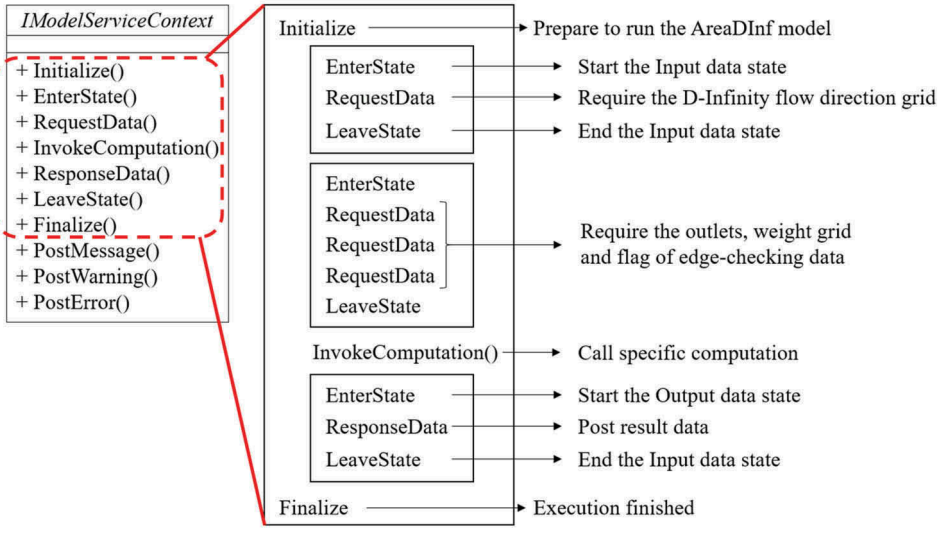
In the model service controller, data I/O configuration documents are generated for every involved model execution instance and the data I/O events within a model execution instance are handled according to the linked configuration document. To bridge the intercommunication between a model execution instance and the model service controller, the data I/O events of a model are wrapped based on the request–response structure.

The data I/O wrapping method is implemented using the IModelServiceContext interface, which is designed with seven core functions and three auxiliary functions. The core functions (i.e., Initialize, EnterState, RequestData, InvokeComputation,

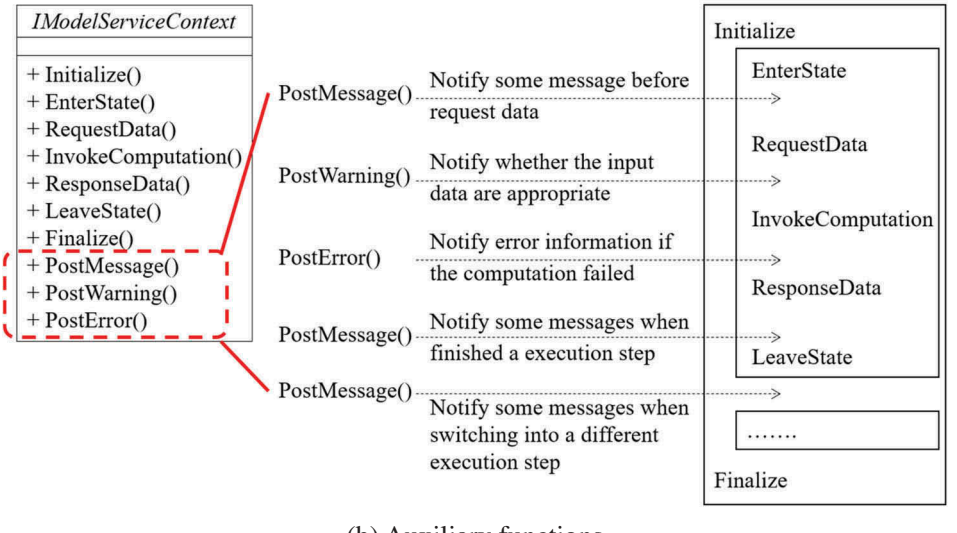
ResponseData, LeaveState, Finalize) describe the full execution process of a model, and the auxiliary functions (i.e., PostMessage, PostWarning, PostError) support the transfer of more detailed execution information between a model and the external applications.

Figure 4(a) illustrates the basic structure when using the IModelServiceContext interface to wrap the data I/O of the AreaDInf model.

(1) Initialize is used at the beginning of the execution process to prepare to run a model, such as by checking the memory capacity and loading dependent libraries. Once the



(a) Core functions



(b) Auxiliary functions

Figure 4. Functions in the IModelServiceContext interface: (a) the core functions using the AreaDInf model as an example and (b) the auxiliary functions when wrapping the AreaDInf model.

Initialize message is sent to the model service controller, the controller links the model execution instance with a unique data I/O configuration document. This data I/O configuration document is built when the execution instance is added to the modeling task, and the configuration content is modified by participating modelers based on the involved data-services.

(2) EnterState is used at the beginning of an execution step to send a message that "a specified execution step is starting" to the model service controller. The model service controller finds the corresponding Event node in the configuration document and returns the result indicating whether the input data are ready or not.
(3) RequestData is used to require input data from the model service controller. The data request message is routed to the data service controller from the model service controller. Once the required data can be accessed, the service location of the data resource is sent back to the model execution instance through the data service controller and the model service controller.
(4) InvokeComputation is used to conduct the specified model execution process. The input data prepared by the RequestData function is used for computation.
(5) ResponseData is used to post the result data to the model service controller. The location of the result data is also transferred to the data service controller allowing other models can reuse these data.
(6) LeaveState is used when finishing the current execution step. The next execution step can be executed after the LeaveState.
(7) Finalize is used at the end of the execution process. The data I/O configuration document is saved and uninstalled from the model service controller.

Through such a structured wrapping method, the execution process of a model can be organized flexibly, and the data I/O operation of a model is handled in a decoupled manner.

To gain a better understanding of modeling objects and obtain better solutions for geo-problems, the designed modeling task usually needs to be adjusted by modelers based on the preliminary execution results. Modelers normally need to know detailed running information about the modeling task (e.g., the value of the intermediate variable, reason for the errors) so that they can make adjustments accordingly. Therefore, the auxiliary functions in the IModelServiceContext interface are designed to satisfy such demands. In Figure 4(b), a sample use of the auxiliary functions in implementing the IModelServiceContext interface is illustrated.

(1) PostMessage is used to notify external applications of messages about the execution information of a model through the model service controller. For example, the PostMessage function can be used before the RequestData function (the related data resource provider can be aware of the usage) and after the ResponseData function (other participants can be aware of the generation of new data).
(2) PostWarning is used to notify external applications of warnings in the execution process through the model service controller. Between the RequestData function and the InvokeComputation function, the PostWarning function is used to tell modelers if the input data are appropriate. For example, the input data are in a vector data format (e.g., ESRI Shapefile), and there may be no projection information within the input data. The InvokeComputation regards these data with a default projection (e.g., WGS 84), and then, this warning information can notify external applications using the PostWarning function.

(3) PostError is used to notify external applications of errors that occurred in the execution process through the model service controller. After the InvokeComputation function, if the computation has not executed completely, the PostError function is used to notify modelers of error-related information.

A range of technological approaches and strategies can be employed to implement the IModelServiceContext interface. For legacy models, the data I/O redirection approach, the dynamic library loading approach, and the external process calling approach can be used. In addition, for models previously published in the WPS-based or the WSDL-based service style, the web message redirection method can be used.

3.2. Data access configuration

In a participatory modeling task, the data configuration work must be considered from both the model execution side and the data resources access side. The data service controller is designed to access data resources. In the data service controller, each involved data resource is assigned a data stub; the data stub is used to help participating modelers handle the data requirement routed from the model service controller.

In Figure 5, the data stub strategy is illustrated. A data stub represents the content information of a reusable data resource. The data service controller manages all involved data resources (represented as data-services) in a modeling task as different data stubs. A data I/O event of a model execution instance can be configured to link with a data stub in the data service controller, and the requested data content underlying the data stub can be retrieved by the corresponding data-service. By employing the XML-based structure, data stubs are organized into a "data stub collection document." In this document, the DataCollection node is the root node that contains a range of DataStub nodes to indicate all the related data resources.

For the DataStub node, the *id* attribute indicates the unique identifier of the corresponding data resource and the *name* attribute indicates the human-readable name. There are three child nodes in the DataStub node: the Description node (which represents the description or explanation information of the contents of the data resource), the Request node (which represents the web request method for the data resource), and the DataStatus node (which represents whether the data resource can be accessed).

In the Description node, the *value* attribute is used to explain the data content through an external URL or a descriptive string. This node employs the same description strategy as the data I/O configuration document (as introduced in Section 3.1) so that a data I/O event can be compared with a specified data stub.

The Request node is used to represent how to access the data resource underlying a data stub via web requests. The REST (REpresentation State Transfer) specification is employed to transfer data resources among participating computers. According to the design of the REST specification, the *baseUrl* attribute in the Request node indicates the base location of a data resource as a hyperlink string. The Request node also contains a collection of Parameter nodes, which are used to construct the full available data resource URL combined with the content in the *baseUrl* attribute. The *key* attribute in the Parameter node indicates the name of the query string in the related RESTful data-service, and the *description* attribute is used to explain the query string.

In the DataStatus node, the *type* attribute indicates to which status the corresponding data resource belongs. There are three typical types of data resource status: (1) the Initial

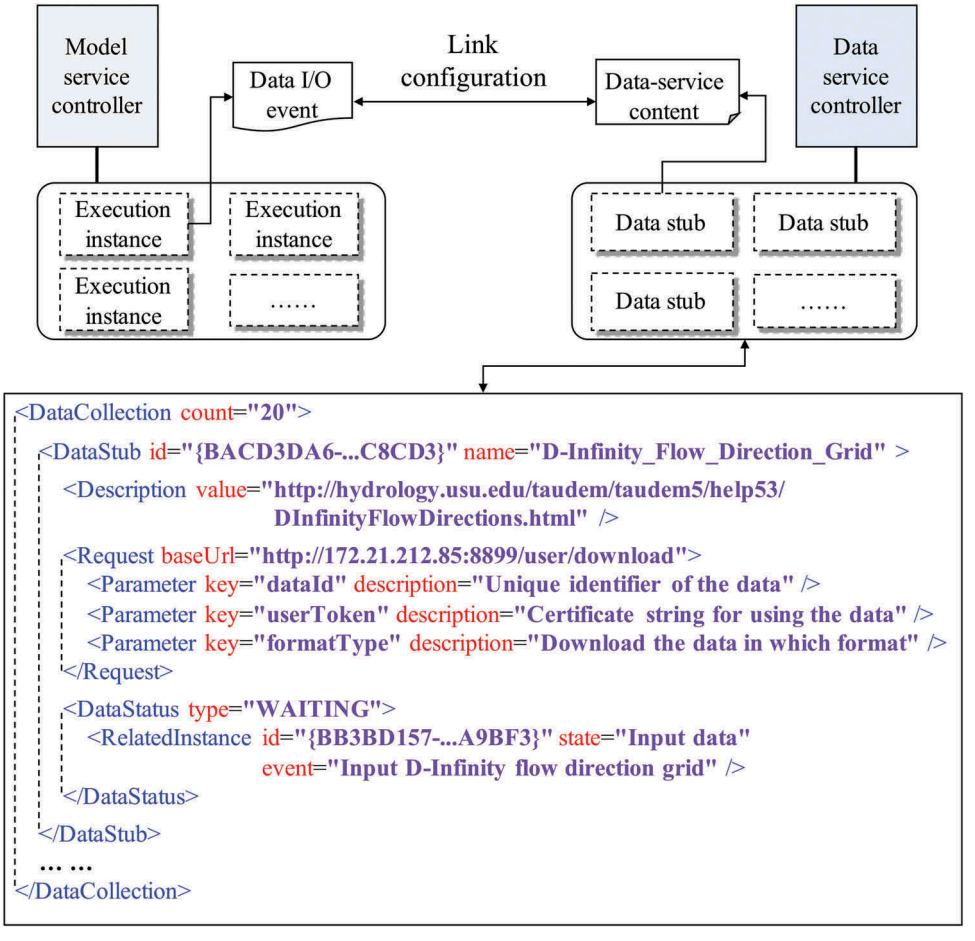


Figure 5. Data stub strategy used in the data service controller; all data stubs are organized in an XML-based document that is managed by the data service controller.

status indicates that the related data resource is not ready to be accessed (it is predefined and required by certain participant modelers); (2) the Waiting status indicates that the related data are dependent upon the computation results of a certain model execution instance, and they are not ready to be accessed; and (3) the Ready status indicates that the related data resource can be accessed and provided by certain participants (data resource providers can offer their data resource as data-services in different styles, such as the OGC WFS, WMS, and WCS specifications) or generated by a model execution instance.

3.2.1. Data input event configuration

When a model execution instance requests input data from the model service controller, the request will be routed to the data service controller. The data service controller will return the request result based on the linked data stubs. Based on the status of a data stub (Initial, Waiting or Ready), a data input event in a model execution instance can be handled for three different conditions.

If the linked data stub for a model execution instance is in the Initial status, the data service controller returns the result of "preparing data" to the model service controller and to the corresponding execution instance. The computation process of the execution instance is suspended until the required data are ready. As shown in Figure 6, a "demand queue" is constructed in the data service controller, which is used to collect the request demands (a request demand indicates the required data stub and the related execution instance). The data service controller has a separate thread to iterate through the request demands in the queue. Once the status of a data stub has been changed to Ready, the message that contains this data stub will be sent to the model service controller and then to the model execution instance. Based on the content in the received data stub, the model execution instance can continue its computation process.

If the linked data stub for a model execution instance has the Waiting status, the data service controller will return the request result "preparing data" to the model service controller and the execution instance. The Waiting status means that the data resource behind the data stub depends on the computation results of another model execution instance. Similar to the working process illustrated in Figure 6, once the dependent data are computed by the related model, the status of this data stub will be changed to the Ready status. Then, the original model execution instance can continue its computation process based on the data stub object received from the data service controller.

If the linked data stub of a model execution instance has the Ready status, the data service controller will return the corresponding data stub directly to the model service controller and the execution instance. The above conditions (i.e., Initial status and Waiting status) can be converted to the Ready status. The status of a data service stub can be changed to the Ready status either through resource sharing by data resource

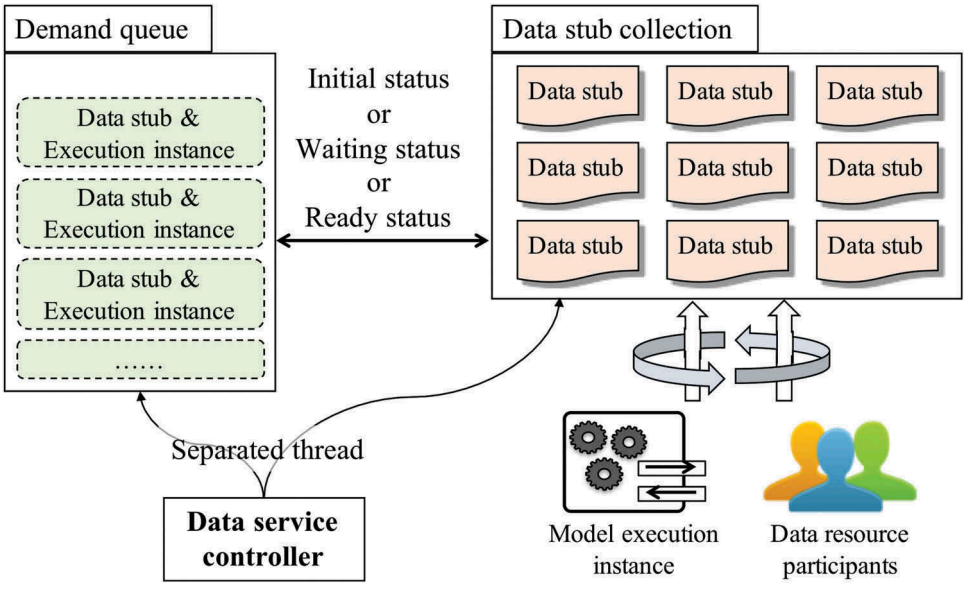


Figure 6. Basic process of the data input event configuration; demand queue indicates the requirements of models, data stub collection indicates the data stubs in the modeling task, and the data service controller links them in a separated thread.

participants or the generation of other model execution instances; the latter is related to the data output event configuration.

3.2.2. Data output event configuration

When a model execution instance responds with output data to the model service controller, the response data notifies the data service controller, and the data service controller then fills the content of the corresponding data stub. In a model execution instance, if a computation step generates output data, the related data I/O event will be fired, and the message will be sent to the data service controller. There are two approaches for making the data content in a data output event a reusable data-service: (1) publish in the data service controller and (2) publish in the computer where the model execution instance runs.

For the first approach, the computation result is sent to the data service controller as a web stream and the data service controller publishes it as a RESTful service. As shown in Figure 7(a), the generated RESTful service's URL is configured to fill the content of the related data stub, and the original Waiting status of the data stub can be changed to the Ready status.

For the second approach, as shown in Figure 7(b), the computation result is published in the model-executing computer as a RESTful service and the URL of this RESTful

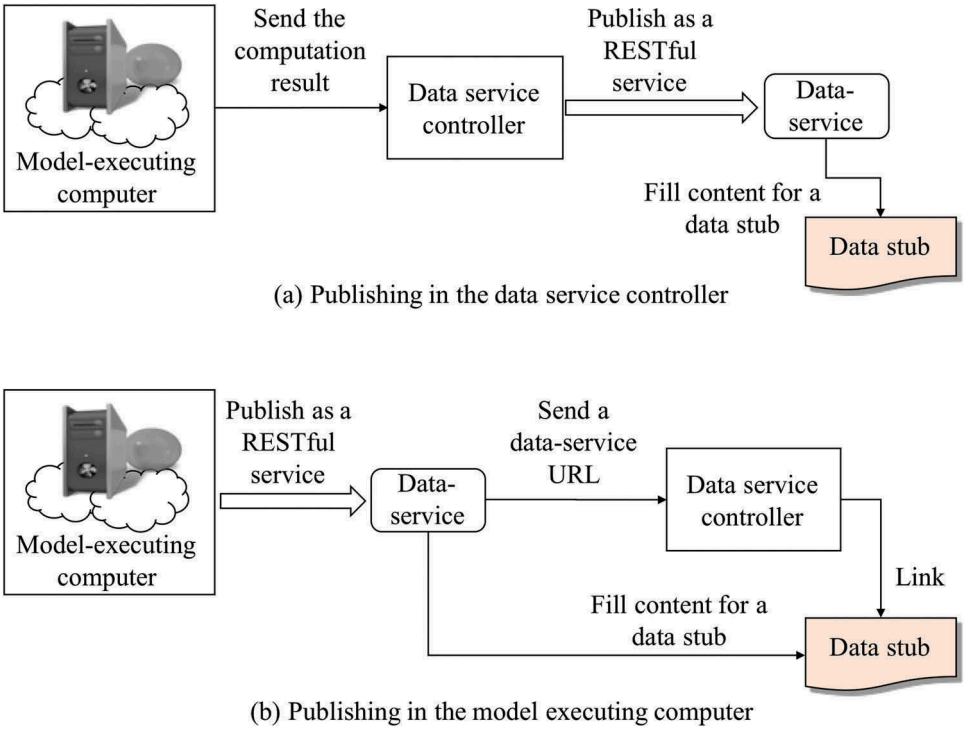


Figure 7. Two approaches for configuring a model output data to a data stub: (a) publish data as a service in the data service controller and (b) publish data as a service in the computer where the model is executed.

service is sent to the data service controller. Additionally, the content of the related data stub is filled with this RESTful URL.

Regarding the publishing method of a computation result as a RESTful service in the data service controller or the model-executing computer, a data-service host is designed to generate the URL of each data resource. In a data-service host, according to the Request node in the data service stub collection document, the *baseUrl* attribute is constructed based on the IP address (the data service controller or the dependent model-executing computer). A request Parameter node is added to the Request node that contains a unique ID for the corresponding result data. The designed data-service host is in charge of parsing the request URL and returning the data.

4. Experiments

To demonstrate the capability and feasibility of the proposed data configuration strategy for supporting the web-based participatory modeling task, a prototype system was built and a modeling case was established under the web environment.

4.1. Prototype system for configuring data in a participatory modeling task

Figure 8 introduces the implementation architecture of the participatory modeling system. This study employed Node.JS as the web service container for publishing data-services and model-services. Node.JS is a lightweight and open-source server framework that uses the JavaScript programming language to produce dynamic web applications. The cross-platform and asynchronous feature allow models and data in various operating systems (e.g., Windows, Linux, Mac OSX) to be published as services using Node.JS. Considering the service accessibility and transfer efficiency, both the Hypertext Transfer Protocol (HTTP) and the Socket connection method are used in the system designation. In the participatory modeling system, model-services and data-services are shared as RESTful services by participants. The HTTP web communication method was employed to publish a model-service and a data-service in the Node.JS environment. Additionally, the service-oriented model wrapping interface was implemented using the native C++ programming language, the Java programming language, and the C# programming language. The model repository and data repository are organized and managed by a MongoDB database system. Model providers can choose one of these technical approaches to wrap their models as model-services.

The model service controller and the data service controller were implemented in the Node.JS environment. For the model service controller, the data I/O configuration document building and managing engine was developed within itself. Three web ports (listening to messages from clients) were used to handle the web request–response message: (1) a port for communicating with model execution instances, which uses the HTTP method to receive and return messages from execution instances; (2) a port for communicating with the data service controller, which uses the Socket method to maintain a long connection between the model service controller and the data service controller; and (3) a port for communicating with the participants who constructed the modeling task, which uses the HTTP method to support the modelers conducting the data configuration work. Similarly, the developed data service controller had three corresponding ports for communicating with data-services, communicating with the model service controller and communicating with the participants of the modeling task. In addition, the data service controller contained a configuration engine for data stub collection.

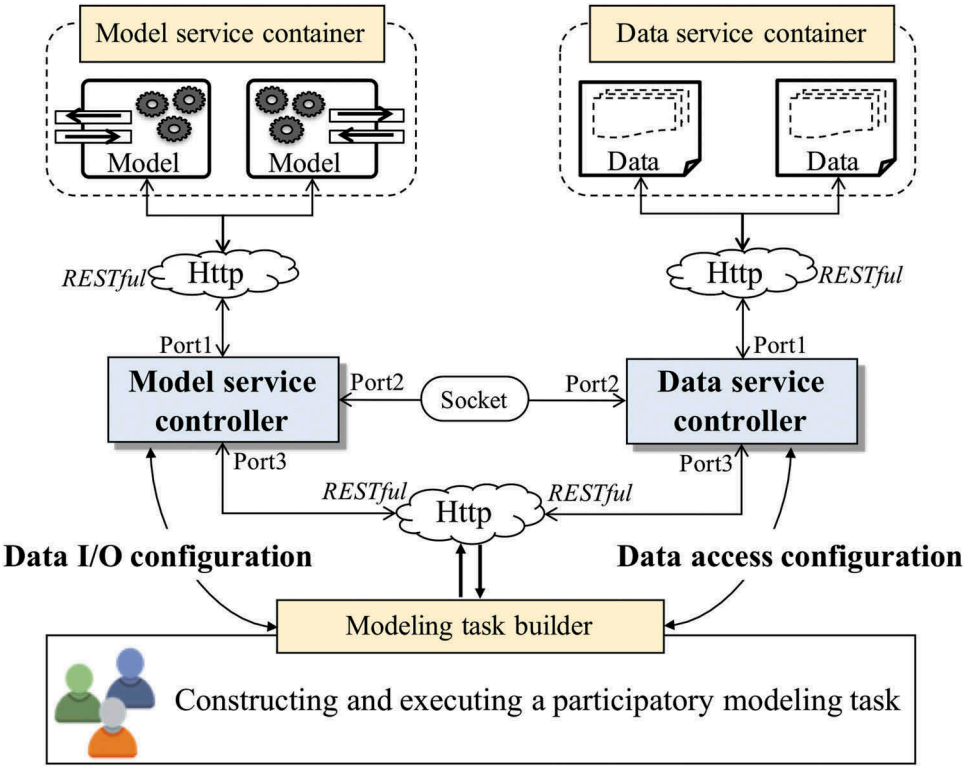


Figure 8. Basic architecture of the participatory modeling system; there are five components: model service container, data service container, modeling task builder, model service controller and data service controller.

In addition to the model service controller and the data service controller, to help modelers construct a participatory modeling task more efficiently, we developed a model service container, a data service container, and a modeling task builder. As shown in Figure 8, the model service container can be used to publish various model-services and communicate with the model service controller. The data service container can be used to publish data-services and communicate with the data service controller. The modeling task builder was designed and developed to discover available and appropriate model-services and data-services. Modeling participants can dynamically join a modeling task built using the modeling task builder. The data I/O configuration and data access configuration tasks for each model execution instance can also be conducted in the modeling task builder. Figure 9 presents snapshots of the prototype system. We developed Graphical User Interface (GUI)-based web applications for the model service container, the data-service container, and the modeling task builder. The data configuration functionalities of the model service controller and the data service controller can be used through the modeling task builder.

4.2. Modeling case study

An experimental modeling case was conducted based on the proposed methods and the implemented modeling system. The theme of the modeling case was terrain analysis and

watershed management. A set of DEM analysis models (TauDEM) and the Soil and Water Assessment Tool (SWAT) were wrapped and published as model-services in the model service container.

The DEM analysis models can be used to compute various terrain characteristics and analyze hydrological information, such as flow path extraction, flow direction computation, and watershed delineation. Table 1 presents some of the DEM analysis models involved in the designed modeling case. Detailed information about these DEM analysis models can be found on the TauDEM homepage (http://hydrology.usu.edu/taudem/taudem5/documentation.html).

As shown in Figure 10(a), in this modeling case, DEM analysis models are published as model-services on computing servers, and these servers can connect to the model service controller and the data service controller. Figure 10(b) illustrates the data-dependent relationships among these models. The required DEM data, outlet data, and other related data are also provided by participants as data-services. Modeling targets can be achieved in this modeling case. As shown in Figure 10, one modeling target is delineating the watersheds of the research region. We designed a modeling instance, i.e., Watershed_Delineation, based on the graphical web applications developed in the prototype system. The corresponding data I/O configuration document and data service stub collection document were built during the experiment. These two documents were serialized and saved as reusable solutions in the modeling task builder. The data configuration results and the modeling results are shown in Figure 11.

In addition, another modeler participant tried to join the modeling task and create a modeling instance (Hydrological_Analysis) based on the SWAT model. A new SWAT model-service was published in a specified model service container. One key input for the

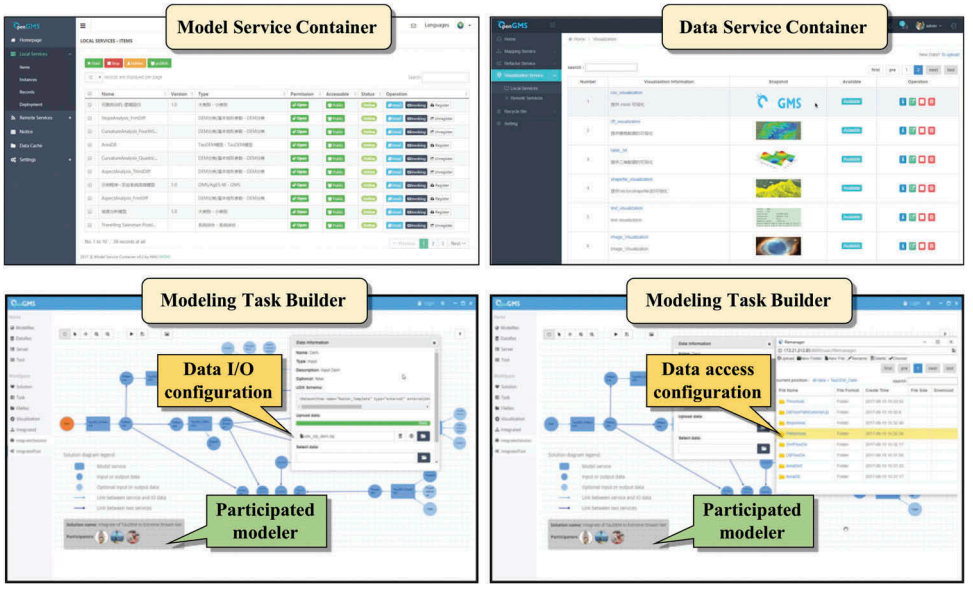


Figure 9. Prototype implementation of the participatory modeling system considering dynamical data configuration: model service container (upper left) and data service container (upper right); modeling task builder with data I/O configuration (bottom left) and modeling task builder with data access configuration (bottom right).

Table 1. Data input and output of the related DEM analysis models.

Model	Brief description and main Input & Output data
PitRemove	**Description**: Identifies all pits in the DEM and raises their elevation to the level of the lowest pour point around their edge. **Input**: Original DEM data. **Output**: Filled DEM data.
DInfFlowDir	**Description**: Assigns a flow direction based on the D-infinity flow method using the steepest slope of a triangular facet. **Input**: Filled DEM data. **Output**: Slope data (D-infinity) and Flow direction data (D-infinity).
AreaDInf	**Description**: Calculates a catchment area, which is the contributing area per unit contour length, using the multiple flow direction D-infinity approach. **Input**: Flow direction data (D-infinity), Outlets data (points), and other controllable parameters. **Output**: Catchment area data (D-Infinity).
SlopeArea	**Description**: Creates a grid of a slope area based on the slope and specific catchment area grid input. **Input**: Slope data (D-infinity), Catchment area data (D-Infinity), Slope Exponent parameter, and Area Exponent parameter. **Output**: Slope area data.
D8FlowDir	**Description**: Computes the flow direction based on the D8 method and generates the slope data. **Input**: Filled DEM data. **Output**: Flow direction data (D8), slope data (D8).
D8FlowPathExtremeUp	**Description**: Evaluates the extreme (either maximum or minimum) upslope value from an input grid based on the D8 flow model. **Input**: Flow direction data (D8), slope area data, outlets data (points), and other controllable parameters. **Output**: Extreme upslope data.
Threshold	**Description**: Operates on any grid and outputs an indicator (1,0) grid that identifies the cells with input values ≥ the threshold value. **Input**: Extreme upslope data, threshold value, and mask data. **Output**: Stream data.
AreaD8	**Description**: Calculates a grid of contributing areas using the single-direction D8 flow model. **Input**: Flow direction data (D8), outlets data (points), and other controllable parameters. **Output**: Contributing area data (D8).
StreamNet	**Description**: Produces a vector network and watershed area from the stream raster grid. **Input**: Filled DEM data, flow direction data (D8), contributing area data (D8), stream data, and other controllable parameters. **Output**: Stream order data, network connectivity tree data, network coordinates data, stream reach data, and watershed data.

SWAT model is the Hydrology Response Unit (HRU) data. To generate the HRU data, required data can be found in the previous Watershed_Delineation modeling results, such as filled DEM data, watershed data, and stream order data. Based on the proposed data configuration strategy, the existing Watershed_Delineation configuration documents can be reused. As shown in Figure 12, the data configuration of the Watershed_Delineation modeling instance was imported to the Hydrological_Analysis modeling instance as a "WatershedDelineation" module. According to the execution process introduced in

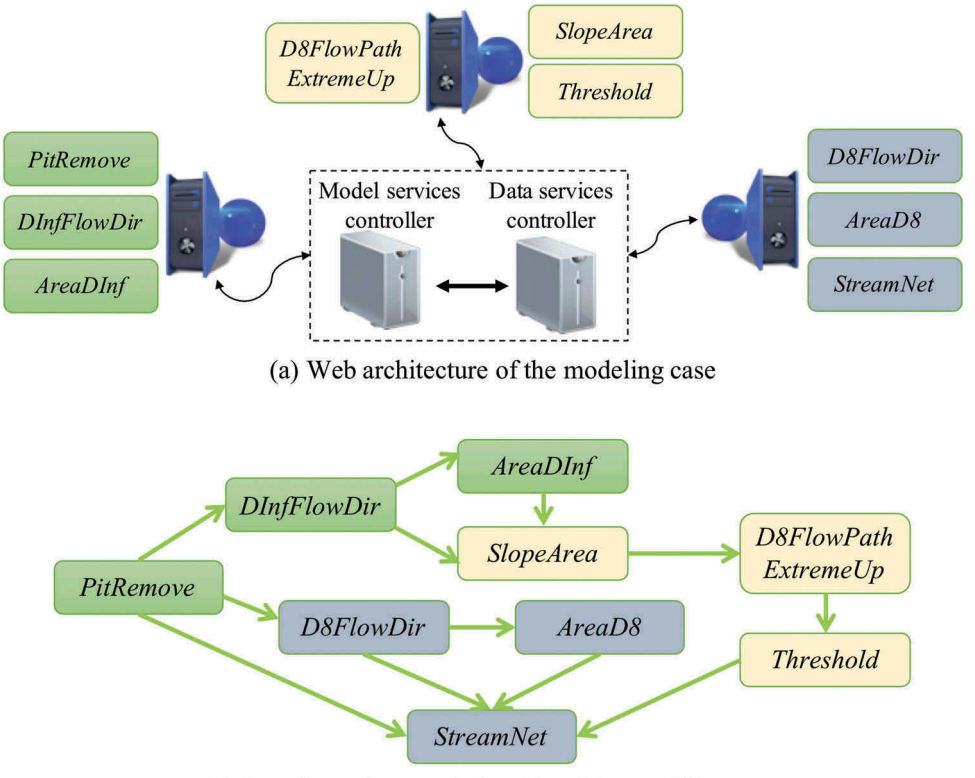


Figure 10. Web architecture and data dependence relationship of the modeling case: (a) the web architecture of the modeling case that consists of three computer nodes for executing models and two computer nodes for the model service controller and the data service controller; (b) the data dependence relationship of the modeling case.

Section 2, the previously constructed Watershed_Delineation modeling instance can maintain its running state, and the later constructed Hydrological_Analysis modeling instance can be dynamically added to the running process of the modeling task.

4.3. Validation

Based on the above analysis, in the prototype system, the open-source Cesium (https://cesiumjs.org/) and D3.js (https://d3js.org/) software are employed for visualizing the data. As shown in Figure 13(a), the watershed delineation result is presented as polygons in the Cesium virtual globe, while the result of the SWAT model is presented as a histogram by D3.js.

To validate the modeling results, the MapWindow platform – a GIS application system that can import TauDEM models and SWAT model as plugins – is employed to conduct the modeling process with the same datasets. In the MapWindow platform, all input data must be prepared on a single computer, and the involved computing steps need to be operated by users manually. However, in the proposed prototype system, models and data are published as reusable web services in distributed computer nodes; once the

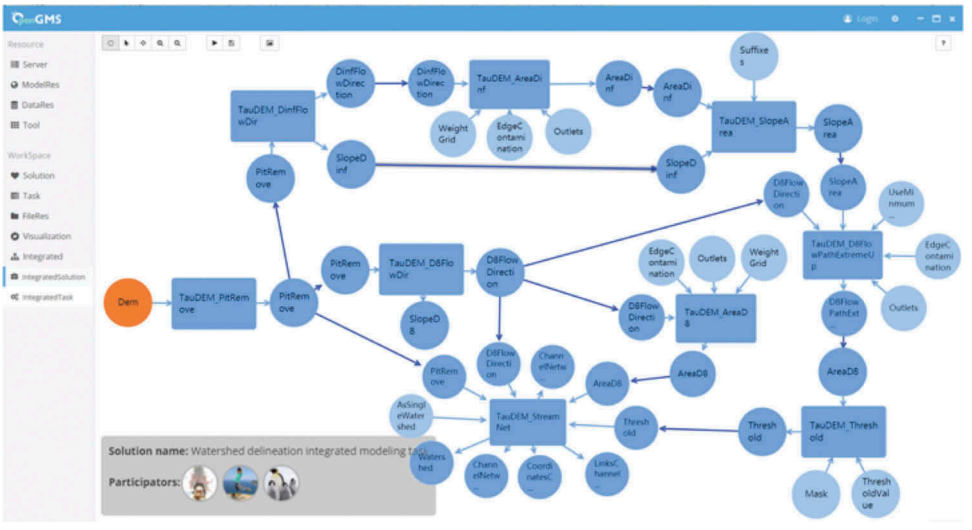


Figure 11. Watershed delineation model based on the prototype system; a rectangle indicates a model, a circle indicates data, and an arrow indicates a relationship between a model-service and a data-service.

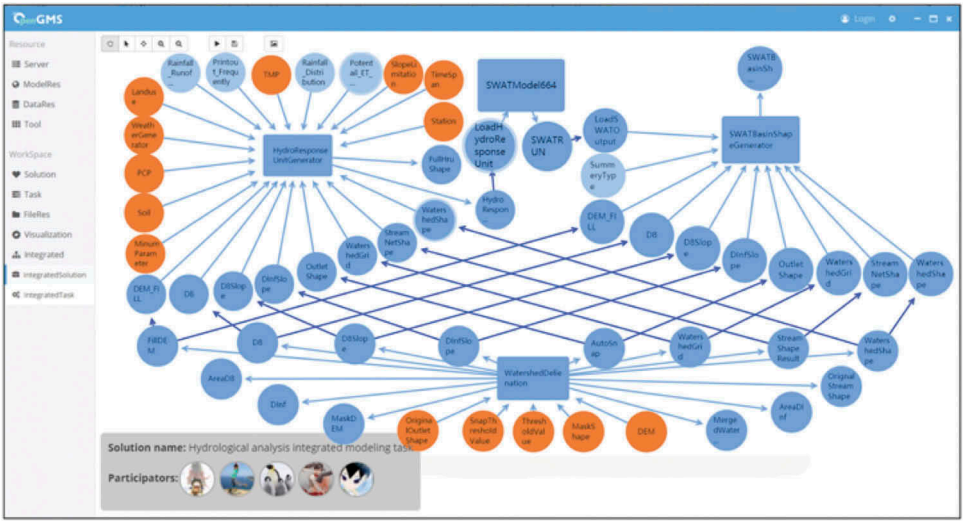
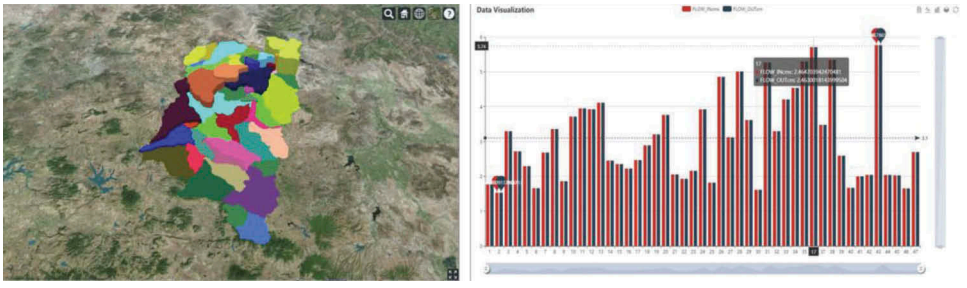


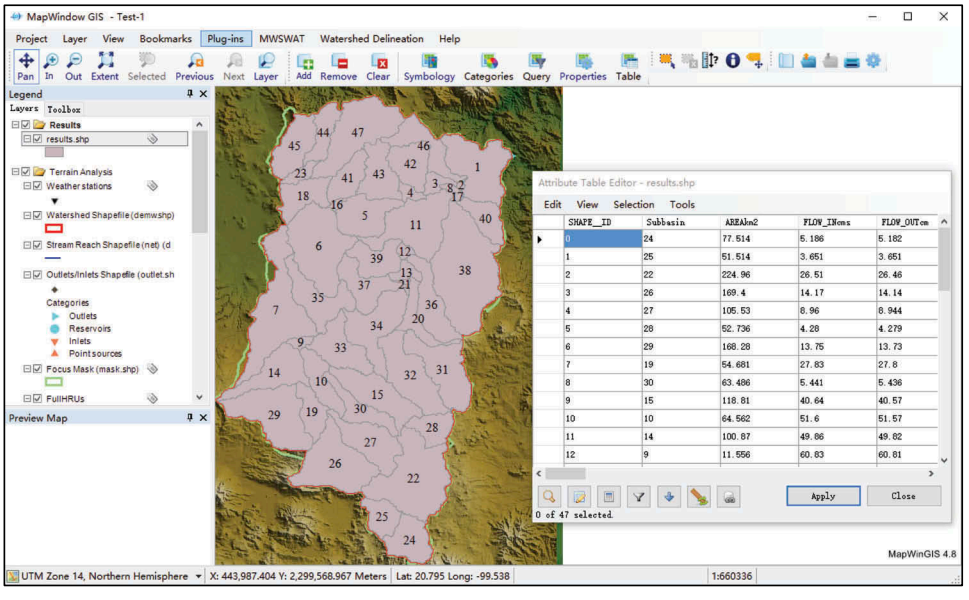
Figure 12. Hydrological analysis model based on the prototype system, the Watershed_Delineation modeling instance is presented at the bottom as a standalone component of the Hydrological_Analysis modeling instance.

relationships among model-services and data-services are configured, the computing process can be executed automatically. The modeling results can also be visualized based on the MapWindow platform, as shown in Figure 13(b).

Based on the proposed data configuration strategy, the construction of a modeling task in the prototype system can be mainly divided into two parts: the integration logic



(a) Visualization in web pages



(b) Visualization in the MapWindow platform

Figure 13. Partial computation results for the modeling case: (a) the visual effects in the prototype system and (b) the visual effects in the MapWindow platform.

design part and the execution-oriented data configuration part. The two parts are loosely decoupled and can be collaboratively undertaken by different participants. Data-services and model-services involved in a modeling task can be dynamic modified, while the model integration logic remains stable. Therefore, participants in a modeling task can work together conveniently.

5. Conclusions and future research

An approach to configuring data in a participatory modeling task is described in this article. Considering the collaborative characteristics of constructing a participatory modeling task in a web environment, the proposed data configuration strategy decouples the model execution process and the data transmission process using two controllers: the model service controller (for data I/O configuration) and the data service controller (for

data access configuration). By employing the proposed data configuration strategy, the difficulties existing in the construction of a web-based participatory modeling task can be reduced. The prototype system established in this study also presented the flexibility and feasibility of the proposed data configuration strategy when exploring dynamic modeling targets in a participatory modeling task.

The model service controller and the data service controller were designed as middleware to decouple the model calling process and the data transmission process, but this approach depends on web communication capabilities. All computer nodes that publish model-services must be able to connect to the computer node that contains the model service controller, and all computer nodes that publish data-services must be able to connect to the computer node that contains the data service controller. The two designed controllers must be connected to each other, and the modeling participants also must be able to connect to the servers containing the two controllers. Based on this web architecture, a modeling participant does not need to connect to the servers used to publish model-services and data-services. Participants can access the modeling resources in a modeling task through the two controllers. While we have designed the HTTP and Socket method for transferring data and executing messages, more comprehensive web communication strategies can be explored to improve efficiency and user convenience.

In addition, considering the synthetic characteristics of the earth environmental system and the complexity of conducting participatory modeling tasks, further work on web-based participatory modeling is needed, especially with regard to the following aspects.

(1) Regarding the integration of a model-service and its required data-services, a request–response data transfer strategy was designed in this study. This strategy makes a model-service the computation "center" and all other data resources are transferred to it. However, data transfer can be time consuming due to network transfer speed limitations. If the data-service and model-service are deployed on the same compute node, the request–response data transfer strategy can be implemented by setting the corresponding file path. Further investigation regarding when the size of the required data is too large to be transferred from other computer nodes is needed. This difficulty may be managed by transferring model-services to the computer nodes that store data resources (Schaeffer 2008; Müller, Bernard, and Brauner 2010, Müller, Bernard, and Kadner 2013).
(2) The data configuration strategy proposed in this article uses the integration (chaining) of model-services and data-services to support the modeling process. The service chaining in this study is a simple and effective method that relies on the data dependency relationships: the logical relationships among integrated models are implicitly represented by the dependencies of the input and output data. This approach is similar to the data provenance studies from the viewpoint of web service chaining. However, data quality, data extent, ontology, metadata and other information have not been considered in this study. The linkage between a model-service and a data-service is determined and customized by the modelers. Achievements made in data provenance studies, especially in geospatial data provenance studies, can be integrated into this work. The proposed data configuration strategy can be improved through more comprehensive and structural descriptions of data-services, which can be found in other data provenance studies (Sahoo, Sheth, and Henson 2008; Yue et al. 2011).

(3) The integration of models in a participatory modeling task was considered from the viewpoint of data dependency. There are also many other web-service-based model integration methods that have been studied to support the solving of geo-problems, such as the BPMN method, the uncertainty execution effect method, and the Bayesian Network method (Düspohl, Frank, and Döll 2012; Meek, Jackson, and Leibovici 2016; Wang et al. 2016). The data configuration study needs further improvement regarding its technical compatibility when employing these methods in constructing a participatory modeling task. Additionally, the data demands of a model and the content of a data-service are usually matched directly. More data preprocessing and postprocessing work is also required in a modeling task. This article focused on the model–data link configuration to support model execution, and the data processing work is considered a model-service in the designed prototype system. From the viewpoint of the participatory modeling work, data processing, data recommendation, data transmission control and other data-related demands should also be included in the data configuration study.

Acknowledgements

We appreciate the detailed suggestions and comments from the editor and the anonymous reviewers. We express heartfelt thanks to the other members of the OpenGMS team.

Computer code availability

Architecture Designer: Min Chen, Songshan Yue and Yongning Wen
Software Developer: Songshan Yue and Chaoran Shen
Address: No.1 Wenyuan Road Qixia District, Nanjing, 210,023 P.R.China
Email: yss123yss@126.com
First available: 2018
Hardware requirements: 1 GHz CPU 1024 MB RAM
Software requirements: Windows (or other platforms supporting NodeJS)
Availability: Open source (no license, source code available upon request)
Cost: Free
Program language: JavaScript, C++ and C#
Program size: 145 MB
Software Access: Sent by email

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the National Natural Science Foundation of China [41701441]; The National Natural Science Foundation of China [41622108]; Priority Academic Program Development of Jiangsu Higher Education Institutions [164320H116]; The National Natural Science Foundation of China [41471317]; The National Basic Research Program of China (973 Program) [2015CB954103]; and the Priority Academic Program Development of Jiangsu Higher Education Institutions [grant number 164320H116].

References

Addison, P. F. E., K. Bie, and L. Rumpff. 2015. "Setting Conservation Management Thresholds Using a Novel Participatory Modeling Approach." *Conservation Biology* 29 (5): 1411–1422. doi:10.1111/cobi.12544.

Al-Sabhan, W., M. Mulligan, and G. A. Blackburn. 2003. "A Real-Time Hydrological Model for Flood Prediction Using GIS and the WWW." *Computers, Environment and Urban Systems* 27 (1): 9–32. doi:10.1016/S0198-9715(01)00010-2.

Antunes, P., R. Santos, and N. Videira. 2006. "Participatory Decision Making for Sustainable Development—The Use of Mediated Modelling Techniques." *Land Use Policy* 23 (1): 44–52. doi:10.1016/j.landusepol.2004.08.014.

Argent, R. M. 2004. "An Overview of Model Integration for Environmental Applications—Components, Frameworks and Semantics." *Environmental Modelling & Software* 19 (3): 219–234. doi:10.1016/S1364-8152(03)00150-6.

Bastin, L., D. Cornford, R. Jones, G. B. Heuvelink, E. Pebesma, C. Stasch, … M. Williams. 2013. "Managing Uncertainty in Integrated Environmental Modelling: The UncertWeb Framework." *Environmental Modelling & Software* 39: 116–134. doi:10.1016/j.envsoft.2012.02.008.

Beall, A., and L. Zeoli. 2008. "Participatory Modeling of Endangered Wildlife Systems: Simulating the Sage-Grouse and Land Use in Central Washington." *Ecological Economics* 68 (1): 24–33. doi:10.1016/j.ecolecon.2008.08.019.

Belem, M., and M. Saqalli. 2017. "Development of an Integrated Generic Model for Multi-Scale Assessment of the Impacts of Agro-Ecosystems on Major Ecosystem Services in West Africa." *Journal of Environmental Management* 202: 117–125. doi:10.1016/j.jenvman.2017.07.018.

Belete, G. F., A. Voinov, and G. F. Laniak. 2017. "An Overview of the Model Integration Process: From Pre-Integration Assessment to Testing." *Environmental Modelling & Software* 87: 49–63. doi:10.1016/j.envsoft.2016.10.013.

Bergez, J. E., P. Chabrier, C. Gary, M. H. Jeuffroy, D. Makowski, G. Quesnel, … P. Debaeke. 2013. "An Open Platform to Build, Evaluate and Simulate Integrated Models of Farming and Agro-Ecosystems." *Environmental Modelling & Software* 39: 39–49. doi:10.1016/j.envsoft.2012.03.011.

Castronova, A. M., J. L. Goodall, and M. M. Elag. 2013. "Models as Web Services Using the Open Geospatial Consortium (Ogc) Web Processing Service (Wps) Standard." *Environmental Modelling & Software* 41: 72–83. doi:10.1016/j.envsoft.2012.11.010.

Cenci, L., L. Disperati, M. G. Persichillo, E. R. Oliveira, F. L. Alves, and M. Phillips. 2018. "Integrating Remote Sensing and GIS Techniques for Monitoring and Modeling Shoreline Evolution to Support Coastal Risk Management." *GIScience & Remote Sensing* 55 (3): 355–375. doi:10.1080/15481603.2017.1376370.

Chen, M., C. Yang, T. Hou, G. N. Lü, Y. Wen, and S. S. Yue. 2018. "Developing a data model for understanding geographical analysis models with consideration of their evolution and application processes." *Transactions in GIS*. doi: 10.1111/tgis.12484.

Chen, Z. Q., H. Lin, M. Chen, D. Liu, Y. Bao, and Y. L. Ding. 2014. "A Framework for Sharing and Integrating Remote Sensing and GIS Models Based on Web Service." *The Scientific World Journal*: 354919: 1–13. doi: 10.1155/2014/354919.

David, O., J. C. Ascough, W. Lloyd, T. R. Green, K. W. Rojas, G. H. Leavesley, and L. R. Ahuja. 2013. "A Software Engineering Perspective on Environmental Modeling Framework Design: The Object Modeling System." *Environmental Modelling & Software* 39: 201–213. doi:10.1016/j.envsoft.2012.03.006.

de Jesus, J., P. Walker, M. Grant, and S. Groom. 2012. "WPS Orchestration Using the Taverna Workbench: The eScience Approach." *Computers & Geosciences* 47: 75–86. doi:10.1016/j.cageo.2011.11.011.

Deng, Z., Y. Ke, H. Gong, X. Li, and Z. Li. 2017. "Land Subsidence Prediction in Beijing Based on PS-InSAR Technique and Improved Grey-Markov Model." *GIScience & Remote Sensing* 54 (6): 797–818. doi:10.1080/15481603.2017.1331511.

Düspohl, M., S. Frank, and P. Döll. 2012. "A Review of Bayesian Networks as A Participatory Modeling Approach in Support of Sustainable Environmental Management." *Journal of Sustainable Development* 5 (12): 1.

Eisman, E., J. Gebelein, and T.A. Breslin. 2017. "Developing a geographically weighted complex systems model using open-source data to highlight locations vulnerable to becoming terrorist safe-havens." *Annals of GIS* 23 (4): 251–267. doi:10.1080/19475683.2017.1368705.

Falconi, S. M., and R. N. Palmer. 2017. "An Interdisciplinary Framework for Participatory Modeling Design and evaluation—What Makes Models Effective Participatory Decision Tools?" *Water Resources Research* 53 (2): 1625–1645. doi:10.1002/2016WR019373.

Fischer, G., W. Winiwarter, T. Ermolieva, G. Y. Cao, H. Qui, Z. Klimont, … F. Wagner. 2010. "Integrated Modeling Framework for Assessment and Mitigation of Nitrogen Pollution from Agriculture: Concept and Case Study for China." *Agriculture, Ecosystems & Environment* 136 (1): 116–124. doi:10.1016/j.agee.2009.12.004.

Fonseca, A., D. P. Ames, P. Yang, C. Botelho, R. Boaventura, and V. Vilar. 2014. "Watershed Model Parameter Estimation and Uncertainty in Data-Limited Environments." *Environmental Modelling & Software* 51: 84–93. doi:10.1016/j.envsoft.2013.09.023.

Giuliani, G., S. Nativi, A. Lehmann, and N. Ray. 2012. "WPS Mediation: An Approach to Process Geospatial Data on Different Computing Backends." *Computers & Geosciences* 47: 20–33. doi:10.1016/j.cageo.2011.10.009.

Goodall, J. L., B. F. Robinson, and A. M. Castronova. 2011. "Modeling Water Resource Systems Using a Service-Oriented Computing Paradigm." *Environmental Modelling & Software* 26 (5): 573–582. doi:10.1016/j.envsoft.2010.11.013.

Granell, C., L. Díaz, and M. Gould. 2010. "Service-Oriented Applications for Environmental Models: Reusable Geospatial Services." *Environmental Modelling & Software* 25 (2): 182–198. doi:10.1016/j.envsoft.2009.08.005.

Granell, C., S. Schade, and N. Ostländer. 2013. "Seeing the Forest through the Trees: A Review of Integrated Environmental Modelling Tools." *Computers, Environment and Urban Systems* 41: 136–150. doi:10.1016/j.compenvurbsys.2013.06.001.

Gray, S., A. Chan, D. Clark, and R. Jordan. 2012. "Modeling the Integration of Stakeholder Knowledge in Social–Ecological Decision-Making: Benefits and Limitations to Knowledge Diversity." *Ecological Modelling* 229: 88–96. doi:10.1016/j.ecolmodel.2011.09.011.

Gregersen, J. B., P. J. A. Gijsbers, and S. J. P. Westen. 2007. "OpenMI: Open Modelling Interface." *Journal of Hydroinformatics* 9 (3): 175–191. doi:10.2166/hydro.2007.023.

Guilyardi, E., R. Budich, and S. Valcke. 2003. "PRISM and ENES: A European Approach to Earth System Modelling." In: *Realizing Teracomputing, Proceedings of the Tenth ECMWF Workshop on the Use of High Performance Computing in Meteorology*, eds. W. Zwieflhofer and N. Kreitz, 146–164. World Scientific. Singapore. ISBN 981-238-376-X.

Gutiérrez, A. G., R. S. Snell, and H. Bugmann. 2016. "Using a Dynamic Forest Model to Predict Tree Species Distributions." *Global Ecology and Biogeography* 25 (3): 347–358. doi:10.1111/geb.12421.

Herle, S., and J. Blankenbach. 2018. "Enhancing the OGC WPS Interface with GeoPipes Support for Real-Time Geoprocessing." *International Journal of Digital Earth* 11 (1): 48–63. doi:10.1080/17538947.2017.1319976.

Hill, C., C. DeLuca, S. M. Balaji, and A. D. Silva. 2004. "The Architecture of the Earth System Modeling Framework." *Computing in Science & Engineering* 6 (1): 18–28. doi:10.1109/MCISE.2004.1255817.

Jankowski, P. 2009. "Towards Participatory Geographic Information Systems for Community-Based Environmental Decision Making." *Journal of Environmental Management* 90 (6): 1966–1971. doi:10.1016/j.jenvman.2007.08.028.

Jiang, P., M. Elag, P. Kumar, S. D. Peckham, L. Marini, and L. Rui. 2017. "A Service-Oriented Architecture for Coupling Web Service Models Using the Basic Model Interface (BMI)." *Environmental Modelling & Software* 92: 107–118. doi:10.1016/j.envsoft.2017.01.021.

Jones, N. A., P. Perez, T. G. Measham, G. J. Kelly, P. d'Aquino, K. A. Daniell, … N. Ferrand. 2009. "Evaluating Participatory Modeling: Developing a Framework for Cross-Case Analysis." *Environmental Management* 44 (6): 1180. doi:10.1007/s00267-009-9391-8.

Jonsson, A., L. Andersson, J. Alkan-Olsson, and B. Arheimer. 2007. "How Participatory Can Participatory Modeling Be? Degrees of Influence of Stakeholder and Expert Perspectives in Six Dimensions of Participatory Modeling." *Water Science and Technology* 56 (1): 207–214.

Langsdale, S. M., A. Beall, J. Carmichael, S. J. Cohen, C. B. Forster, and T. Neale. 2009. "Exploring the Implications of Climate Change on Water Resources through Participatory Modeling: Case Study of the Okanagan Basin, British Columbia." *Journal of Water Resources Planning and Management* 135 (5): 373–381. doi:10.1061/(ASCE)0733-9496(2009)135:5(373).

Laniak, G. F., G. Olchin, J. Goodall, A. Voinov, M. Hill, P. Glynn, … S. Peckham. 2013. "Integrated Environmental Modeling: A Vision and Roadmap for the Future." *Environmental Modelling & Software* 39: 3–23. doi:10.1016/j.envsoft.2012.09.006.

Lanig, S., A. Schilling, B. Stollberg, and A. Zipf. 2008. "Towards Standards-Based Processing of Digital Elevation Models for Grid Computing through Web Processing Service (WPS)." *Computational Science and Its Applications–ICCSA* (2008: 191–203.

Li, Z., C. Yang, Q. Huang, K. Liu, M. Sun, and J. Xia. 2017. "Building Model as a Service to Support Geosciences." *Computers, Environment and Urban Systems* 61: 141–152. doi:10.1016/j.compenvurbsys.2014.06.004.

Liu, Y., H. Gupta, E. Springer, and T. Wagener. 2008. "Linking Science with Environmental Decision Making: Experiences from an Integrated Modeling Approach to Supporting Sustainable Water Resources Management." *Environmental Modelling & Software* 23 (7): 846–858. doi:10.1016/j.envsoft.2007.10.007.

Meek, S., M. Jackson, and D. G. Leibovici. 2016. "A BPMN Solution for Chaining OGC Services to Quality Assure Location-Based Crowdsourced Data." *Computers & Geosciences* 87: 76–83. doi:10.1016/j.cageo.2015.12.003.

Mendoza, G. A., and R. Prabhu. 2005. "Combining Participatory Modeling and Multi-Criteria Analysis for Community-Based Forest Management." *Forest Ecology and Management* 207 (1): 145–156. doi:10.1016/j.foreco.2004.10.024.

Mendoza, G. A., and R. Prabhu. 2006. "Participatory Modeling and Analysis for Sustainable Forest Management: Overview of Soft System Dynamics Models and Applications." *Forest Policy and Economics* 9 (2): 179–196. doi:10.1016/j.forpol.2005.06.006.

Meng, X., F. Bian, and Y. Xie. 2009, October. "Geospatial Services Chaining with Web Processing Service." Proceedings of the International Symposium on Intelligent Information Systems and Applications (IISA'09), Qingdao, China, 7–10.

Moore, R. V., and A. G. Hughes. 2017. "Integrated Environmental Modelling: Achieving the Vision." *Geological Society, London, Special Publications* 408 (1): 17–34. doi:10.1144/SP408.12.

Müller, M., L. Bernard, and D. Kadner. 2013. "Moving code–Sharing Geoprocessing Logic on the Web." *ISPRS Journal of Photogrammetry and Remote Sensing* 83: 193–203. doi:10.1016/j.isprsjprs.2013.02.011.

Müller, M., L. Bernard, and J. Brauner. 2010. "Moving Code in Spatial Data Infrastructures–Web Service Based Deployment of Geoprocessing Algorithms." *Transactions in GIS* 14 (s1): 101–118. doi:10.1111/tgis.2010.14.issue-s1.

Mustajoki, J., R. P. Hämäläinen, and M. Marttunen. 2004. "Participatory Multicriteria Decision Analysis with Web-HIPRE: A Case of Lake Regulation Policy." *Environmental Modelling & Software* 19 (6): 537–547. doi:10.1016/j.envsoft.2003.07.002.

Nascimento, V. F., N. Yesiller, K. C. Clarke, J. P. H. B. Ometto, P. R. Andrade, and A. C. Sobral. 2017. "Modeling the Environmental Susceptibility of Landfill Sites in California." *GIScience & Remote Sensing* 54 (5): 657–677. doi:10.1080/15481603.2017.1309126.

Nativi, S., P. Mazzetti, and G. N. Geller. 2013. "Environmental Model Access and Interoperability: The GEO Model Web Initiative." *Environmental Modelling & Software* 39: 214–228. doi:10.1016/j.envsoft.2012.03.007.

Omrani, H., A. Tayyebi, and B. Pijanowski. 2017. "Integrating the Multi-Label Land-Use Concept and Cellular Automata with the Artificial Neural Network-Based Land Transformation Model: An Integrated ML-CA-LTM Modeling Framework." *GIScience & Remote Sensing* 54 (3): 283–304. doi:10.1080/15481603.2016.1265706.

Paolisso, M., and J. Trombley. 2017. "Cognitive, Material and Technological Considerations in Participatory Environmental Modeling." In *Environmental Modeling with Stakeholders*, 3–23. New York, NY: Springer International Publishing.

Peckham, S. D., E. W. Hutton, and B. Norris. 2013. "A Component-Based Approach to Integrated Modeling in the Geosciences: The Design of CSDMS." *Computers & Geosciences* 53: 3–12. doi:10.1016/j.cageo.2012.04.002.

Rao, A. B., E. S. Rubin, and M. B. Berkenpas. 2004. *An Integrated Modeling Framework for Carbon Management Technologies*, 3890. Vol. 15213. Pittsburgh, PA: Department of Engineering and Public Policy.

Rautenbach, V., S. Coetzee, and A. Iwaniak. 2013. "Orchestrating OGC Web Services to Produce Thematic Maps in a Spatial Information Infrastructure." *Computers, Environment and Urban Systems* 37: 107–120. doi:10.1016/j.compenvurbsys.2012.08.001.

Robinson, K. F., and A. K. Fuller. 2017. "Participatory Modeling and Structured Decision Making." In: *Environmental Modeling with Stakeholders*, 83–101. New York, NY: Springer International Publishing.

Rodrigues, M. 2016. "GIS-based Modeling of a Rescaled Surface of Land Development Pressure in the Macaronesian Islands." *GIScience & Remote Sensing* 53 (3): 320–336. doi:10.1080/15481603.2016.1139774.

Sahoo, S. S., A. Sheth, and C. Henson. 2008. "Semantic Provenance for Escience: Managing the Deluge of Scientific Data." *IEEE Internet Computing* 12 (4): 46–54. doi:10.1109/MIC.2008.86.

Sandker, M., B. Campbell, and A. Suwarno. 2008. "What are Participatory Scoping Models?." *Ecology and Society* 13 (1). doi:10.5751/ES-02478-1301r02.

Santoro, M., S. Nativi, and P. Mazzetti. 2016. "Contributing to the GEO Model Web Implementation: A Brokering Service for Business Processes." *Environmental Modelling & Software* 84: 18–34. doi:10.1016/j.envsoft.2016.06.010.

Schaeffer, B. 2008. "Towards a Transactional Web Processing Service." Proceedings of the GI-Days, Münster.

Scheer, D., W. Konrad, H. Class, A. Kissinger, S. Knopf, and V. Noack. 2017. "Regional-Scale Brine Migration along Vertical Pathways Due to CO 2 injection–Part 1: The Participatory Modeling Approach." *Hydrology and Earth System Sciences* 21 (6): 2739. doi:10.5194/hess-21-2739-2017.

Seidl, R. 2015. "A Functional-Dynamic Reflection on Participatory Processes in Modeling Projects." *Ambio* 44 (8): 750–765. doi:10.1007/s13280-015-0670-8.

Selvam, S., G. Manimaran, P. Sivasubramanian, N. Balasubramanian, and T. Seshunarayana. 2014. "GIS-based Evaluation of Water Quality Index of Groundwater Resources around Tuticorin Coastal City, South India." *Environmental Earth Sciences* 71 (6): 2847–2867. doi:10.1007/s12665-013-2662-y.

Shahparakia, F., M. Abbaspourb, M. Shafie-Pourc, and M. Mahmoudid. 2017. "A Hybrid Deterministic-Statistical Model Integrating Economic, Meteorological and Environmental Variables to Air Pollution." *Environmental Energy and Economic Research* 1: 25–44.

Sun, Z., P. Yue, and L. Di. 2012. "GeoPWTManager: A Task-Oriented Web Geoprocessing System." *Computers & Geosciences* 47: 34–45. doi:10.1016/j.cageo.2011.11.031.

Videira, N., P. Antunes, and R. Santos. 2017. "Engaging Stakeholders in Environmental and Sustainability Decisions with Participatory System Dynamics Modeling." In *Environmental Modeling with Stakeholders*, 241–265. New York, NY: Springer International Publishing.

Voinov, A., and E. J. B. Gaddis. 2008. "Lessons for Successful Participatory Watershed Modeling: A Perspective from Modeling Practitioners." *Ecological Modelling* 216 (2): 197–207. doi:10.1016/j.ecolmodel.2008.03.010.

Voinov, A., and F. Bousquet. 2010. "Modelling with stakeholders." *Environmental Modelling & Software* 25 (11): 1268–1281. doi:10.1016/j.envsoft.2010.03.007.

Voinov, A., N. Kolagani, M. K. McCall, P. D. Glynn, M. E. Kragt, F. O. Ostermann, … P. Ramu. 2016. "Modelling with Stakeholders–Next Generation." *Environmental Modelling & Software* 77: 196–220. doi:10.1016/j.envsoft.2015.11.016.

Wang, J., M. Chen, G. N. Lü, S. S. Yue, K. Chen, and Y. N. Wen. 2018, "A Study on Data Processing Services for the Operation of Geo-Analysis Models in the Open Web Environment". *Earth and Space Sciences*, DOI: 10.1029/2018EA000459.

Wang, P., Z. Ding, C. Jiang, M. Zhou, and Y. Zheng. 2016. "Automatic Web Service Composition Based on Uncertainty Execution Effects." *IEEE Transactions on Services Computing* 9 (4): 551–565. doi:10.1109/TSC.2015.2412943.

Ward, D. P., A. T. Murray, and S. R. Phinn. 2000. "A Stochastically Constrained Cellular Model of Urban Growth." *Computers, Environment and Urban Systems* 24 (6): 539–558. doi:10.1016/S0198-9715(00)00008-9.

Welsh, W. D., J. Vaze, D. Dutta, D. Rassam, J. M. Rahman, I. D. Jolly, … J. Teng. 2013. "An Integrated Modelling Framework for Regulated River Systems." *Environmental Modelling & Software* 39: 81–102. doi:10.1016/j.envsoft.2012.02.022.

Wen, Y. N., M. Chen, G. N. Lu, and H. Lin. 2013. "Prototyping an Open Environment for Sharing Geographical Analysis Models on Cloud Computing Platform." *International Journal of Digital Earth* 6 (4): 356–382. doi:10.1080/17538947.2012.716861.

Wen, Y. N., M. Chen, S. S. Yue, P. B. Zheng, G. Q. Peng, and G. N. Lu. 2016. "A Model-Service Deployment Strategy for Collaboratively Sharing Geo-Analysis Models in an Open Web Environment." *International Journal of Digital Earth* 10 (4): 405–425. doi:10.1080/17538947.2015.1131340.

Whelan, G., K. Kim, M. A. Pelton, K. J. Castleton, G. F. Laniak, K. Wolfe, … M. Galvin. 2014. "Design of a Component-Based Integrated Environmental Modeling Framework." *Environmental Modelling & Software* 55: 1–24. doi:10.1016/j.envsoft.2014.01.016.

Xie, Z., L. Pearlstine, and D. E. Gawlik. 2012. "Developing a Fine-Resolution Digital Elevation Model to Support Hydrological Modeling and Ecological Studies in the Northern Everglades." *GIScience & Remote Sensing* 49 (5): 664–686. doi:10.2747/1548-1603.49.5.664.

Yue, P., Y. Wei, L. Di, L. He, J. Gong, and L. Zhang. 2011. "Sharing Geospatial Provenance in a Service-Oriented Environment." *Computers, Environment and Urban Systems* 35 (4): 333–343. doi:10.1016/j.compenvurbsys.2011.02.006.

Yue, S., M. Chen, Y. Wen, and G. Lu. 2016. "Service-Oriented Model-Encapsulation Strategy for Sharing and Integrating Heterogeneous Geo-Analysis Models in an Open Web Environment." *ISPRS Journal of Photogrammetry and Remote Sensing* 114: 258–273. doi:10.1016/j.isprsjprs.2015.11.002.

Yue, S., Y. Wen, M. Chen, G. Lu, D. Hu, and F. Zhang. 2015. "A Data Description Model for Reusing, Sharing and Integrating Geo-Analysis Models." *Environmental Earth Sciences* 74 (10): 7081–7099. doi:10.1007/s12665-015-4270-5.

Zhai, R.T., C.R. Zhang, J.M. Allen, W.D. Li, M.A. Boyer, K. Segerson, and K.E. Foote. 2018. "Predicting land use/cover change in Long Island Sound Watersheds and its effect on invasive species: a case study for glossy buckthorn." *Annals of GIS* 24 (2): 83–97. doi:10.1080/19475683.2018.1450786.

Zhang, C. X., M. Chen, R. R. Li, C. Y. Fang, and H. Lin. 2016. "What's Going on about Geo-Process Modeling in Virtual Geographic Environments (Vges)." *Ecological Modelling* 319: 147–154. doi:10.1016/j.ecolmodel.2015.04.023.

Zhang, F., M. Chen, D. P. Ames, C. R. Shen, S. S. Yue, Y.N. Wen, and G. N. Lü. 2018. "Design and Development of a Service-oriented Wrapper System for Sharing and Reusing Distributed Geoanalysis Models on the Web." *Environmental Modelling and Software* doi:10.1016/j.envsoft.2018.11.002.

Zhang, M., X. Bu, and P. Yue. 2017. "GeoJModelBuilder: An Open Source Geoprocessing Workflow Tool." *Open Geospatial Data, Software and Standards* 2 (1): 8.

Zhao, P., T. Foerster, and P. Yue. 2012. "The Geoprocessing Web." *Computers & Geosciences* 47: 3–12. doi:10.1016/j.cageo.2012.04.021.