Environmental Research 191 (2020) 110225

Contents lists available at ScienceDirect

Environmental Research

journal homepage: www.elsevier.com/locate/envres

Service-oriented interface design for open distributed environmental simulations

Fengyuan Zhang[a,b,c], Min Chen[a,b,c,*], Songshan Yue[a,b,c,d], Yongning Wen[a,b,c], Guonian Lü[a,b,c], Fei Li[a,b,c,e,f]

[a] Key Laboratory of Virtual Geographic Environment (Ministry of Education of PRC), Nanjing Normal University, Nanjing, 210023, China
[b] Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing, 210023, China
[c] State Key Laboratory Cultivation Base of Geographical Environment Evolution (Jiangsu Province), Nanjing, 210023, China
[d] Key Laboratory of Geospatial Technology for the Middle and Lower Yellow River Regions (Henan University), Ministry of Education, Kaifeng, 475004, China
[e] Research Center for Environment and Health, Zhongnan University of Economics and Law, Wuhan, 430073, China
[f] School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan, 430073, China

ARTICLE INFO

Keywords:
Environmental simulation
Service-oriented model interface
Distributed resource assembly
PM2.5
Open web environment

ABSTRACT

Modeling and simulations are important methods in environmental research. Currently, massive simulation resources from different domains have been developed to simulate various dynamic phenomena and processes to address different environmental problems. These heterogeneous simulation resources (e.g., models, data, and servers) can be wasted if they are not shared and reused effectively. Recently, experts may exchange resources and conduct simulations in the open web environment via these shared and distributed services. However, some challenges remain, such as the heterogeneity and reusability of simulation resources. The goal of this study was to analyze typical scenarios involved in simulation tasks and design a set of service-oriented interfaces for different simulation resources. These interfaces, including the model description interface, model encapsulation interface, server management interface and sim-task operation interface, can be used to describe, encapsulate, manage and invoke environmental simulation resources, which can further contribute to resource assembly for environmental simulation tasks. This study evaluated the case of PM2.5 concentration distribution simulation by meteorological data, land cover data and a random forest model in 2014. Using the designed interface, this study conducted the simulation and explored the influence of different interpolation methods (inverse distance weighting (IDW) and kriging) for meteorological data in the random forest-based PM2.5 concentration simulation. For this case, the results show that kriging is a more suitable interpolation method than IDW for meteorological data in the simulation, and this interface design can organize simulation resources, configure tasks, and balance task loads in different servers on the open web.

1. Introduction

Environmental issues analysis always involves many dynamic phenomena and processes, such as soil pollution, flood, traffic noise and climate change (Serreze, 2011; Chen et al., 2015; Tóth et al., 2016; Chen and Lin, 2018; Conde-Cid et al., 2019; Nourani et al., 2019; Sun et al., 2019; Wang et al., 2019). Many studies have proven that modeling and simulations are effective ways to explore dynamic phenomena and processes to support further research and policy decisions to address environmental issues (Demeritt and Wainwright et al., 2005; Parsons, 2011; Chen et al., 2013; Lin et al., 2013a, 2013b; Albanesi and Albanesi, 2014; Lin and Chen, 2015; Lü et al., 2019; Chen et al., 2020; Koo et al., 2020). To date, massive model resources have been developed in different domains for various environmental issues, such as atmospheric (Todorova et al., 2010), hydrology (Basnyat et al., 2000; Li et al., 2013), soil (Seibert et al., 2009; Li et al., 2017) and ecology (Li et al., 2018). In the processes of environmental simulation, there are models (such as interpolation, sample and iteration) to make up these processes (Skamarock and Klemp, 2008; Neitsch et al., 2009). In addition to these model resources, simulation also requires data and server resources. Making multiple copies of these simulation resources is wasteful; instead, they should be shareable and reusable by others (Granell et al.,

* Corresponding author. Key Laboratory of Virtual Geographic Environment (Ministry of Education of PRC), Nanjing Normal University, Nanjing, 210023, China.
E-mail address: chenmin0902@163.com (M. Chen).

https://doi.org/10.1016/j.envres.2020.110225
Received 11 December 2019; Received in revised form 11 September 2020; Accepted 13 September 2020
Available online 19 September 2020

2013; Hutton et al., 2016; Zhang and Zhu, 2018; Knox et al., 2019). Thus, simulation resource sharing and reuse have become essential prerequisites for researchers to assemble models, data and server resources for environmental simulation tasks (Tenopir et al., 2011; Belete et al., 2017; Yi et al., 2018; Yue et al., 2019; Gan et al., 2020; Gichamo et al., 2020). However, the model, server and data resources are often heterogeneous, which poses a challenge to resource sharing (Yue et al., 2015; Wen et al., 2017; Zhang et al., 2019b).

To confront this challenge, an increasing amount of research has appeared. First, researchers have concentrated on component-based sharing, such as modular modeling language (MML), the Open Model Interface (OpenMI) Component Modeling Interface (CMI) and Basic Modeling Interface (BMI) from Community Surface Dynamic Modeling System (CSDMS)) (Maxwell and Costanza, 1997; Gregersen et al., 2007; Peckham et al., 2013; Peckham, 2014). Second, with the development of the web, web services, such as Web Services Description Language (WSDL), Open Geospatial Consortium Web Processing Service (OGC WPS), and The Cloud Services Innovation Platform (CSIP), have become a new trend for resource sharing (Christensen et al., 2001; Open Geospatial Consortium, 2007; Michaelis and Ames, 2009; David et al., 2014; Deng et al., 2019; Gao et al., 2019; Omidipoor et al., 2019). Third, several related projects have begun to engage in resource sharing and reuse by application or platform. For example, HydroShare is a platform that shares and reuses model and data resources and describes their hydrological models and data using the Open Archive Initiative's Object Reuse and Exchange (OAI-ORE) standard (Tarboton et al., 2014; Horsburgh et al., 2016); The Community Surface Dynamic Modeling System (CSDMS) has a website for model repositories to share related resources of models (Peckham et al., 2013; Overeem et al., 2013); Tethys is a Python-based scripting application development toolkit for modeling water resources. It provides a Python scripting environment through which users can share functions that process data, perform map rendering and manage distributed resources (Jones et al., 2014; Swain et al., 2016; Kadlec and Ames, 2017; Qiao et al., 2019).

However, at the outset, it is still difficult to share various kinds of different models. As shown in Table 1, different models have different features or granularity. For example, component-based sharing is suitable for constructing an open source model as a component, such as Landlab models, the Storm Water Management Model (SWMM) and the random forest model, but may be inappropriate for black box style models, such as Ground Water System (GMS) and MIKE11; some of them are large systems with many sub-modules, such as the Soil and Water Assessment Tool (SWAT) model and Weather Research Forecast (WRF) model, while others are simply data processors or algorithms, such as the Random Forest model. Thus, the heterogeneity among various model resources restricts resource sharing and reuse. Second, current research has focused on sharing simulation resources; however, these approaches do not satisfy the requirements for assembling different types of simulation resources in simulation tasks. For example, component-based sharing (such as Landlab, which follows the BMI standard) and service-oriented sharing (such as the Biogeographic WPS, which follows the WPS) are suitable standards for model sharing; however, users need to assemble other resources by themselves. A previous study has explored model-sharing methods on the web; however, their reuse and the assembly of other resources in one environment simulation task remains difficult (Zhang et al., 2019b).

The goal of this study was to design a series of service-oriented interfaces to support simulation resource reuse among experts and contribute to resource assembly for environmental simulation tasks on the web. We designed the PM2.5 simulation in different interpolation methods of meteorological data to verify the benefits of our interface design. The remainder of this article is organized as follows. Section 2 analyzes the types of resources and usage scenarios for simulation tasks and the design process. Section 3 describes the detailed designs of different interfaces. Section 4 discusses an experiment of PM2.5 concentration simulation based on different interpolation methods (IDW and kriging) for meteorological data in Beijing using the designed interfaces. Section 5 discusses the advantages and remaining technology challenges and then summarizes this research. Finally, Section 6 discusses the conclusions and future work.

2. Resource analysis and methods design

2.1. Resource analysis

Environmental simulation tasks typically include several types of resources. If model users want to use a model in one environmental simulation task, they need the related model documentation, model executable files, simulation data and servers (Yue et al., 2020). As shown in Fig. 1, in this study, we classify the simulation resources as model resources, data resources and server resources. Generally, model resources are sourced from the model authors or developers and always have two parts: model descriptions and model usage materials. The model description is the introduction of the model, which includes metadata of models (such as the name, version, input/output data configuration, execution environment and dependencies). The model descriptions are always embodied in types of media with unstructured formats, such as user's guides, homepages, and articles. Model usage materials can be a model program or script for simulation executing, including executable files, components, and scripts. Data resources, which are simulation data in the task, include the input data (such as DEM, land cover, and weathers.) needed to execute the models and are

Table 1
Differences among models.

Models	Types	Inputs	Outcomes	Drawbacks	References
SWAT	Executable file/plug	DEM, Land use, Soil, Weather (Files)	Stream flow, etc. (Files)	Hard to wrap as component	Neitsch et al. (2011)
WRF	Executable file	Observation Data/WRF Terrestrial Data/Gridded Data (Files)	Temperature (.nc file)	Cannot run on Windows platform	Skamarock and Klemp (2008)
GMS	Executable file	DEM, Drilling Data	Iso-surfaces, etc.	Commercial	Gogu et al. (2001)
MIKE11	Executable file	Watershed, River shape, etc.	Water quality, etc.	Hard to wrap as component	Thompson et al. (2004)
FDS	Executable file	Grids, 3D model, materials, etc. (Files)	Smoke view, statistic data (Files)	Hard to wrap as component	McGrattan and Forney (2006)
FVCOM	Executable file	Tides/winds/heat flux/date (Files)	Waves Direction (Files)	Hard to wrap as component	Chen et al. (2003)
SWMM	Executable file/GUI/ components (OpenMI)	Pipe network/rainfall/DEM (Files)	Report/output (Files)	Hard to wrap as component	Rossman (2010)
Landlab	Source Code/Components (BMI)	Any (Data stream/parameters)	Any (Data stream/ parameters)	Hard to assemble with other simulation resources	Hobley et al. (2017)
Random Forest	Source Code/Components	Training Data/Source Data (Data stream/parameters)	Predicting Data (Data stream)	Hard to assemble with other simulation resources	Farnaaz and Jabbar (2016)
Biogeographic WPS	Web Service	land-cover, DEM, etc. (Files)	Related results (Files)	Lack of input/output description	Graul and Zipf (2008)

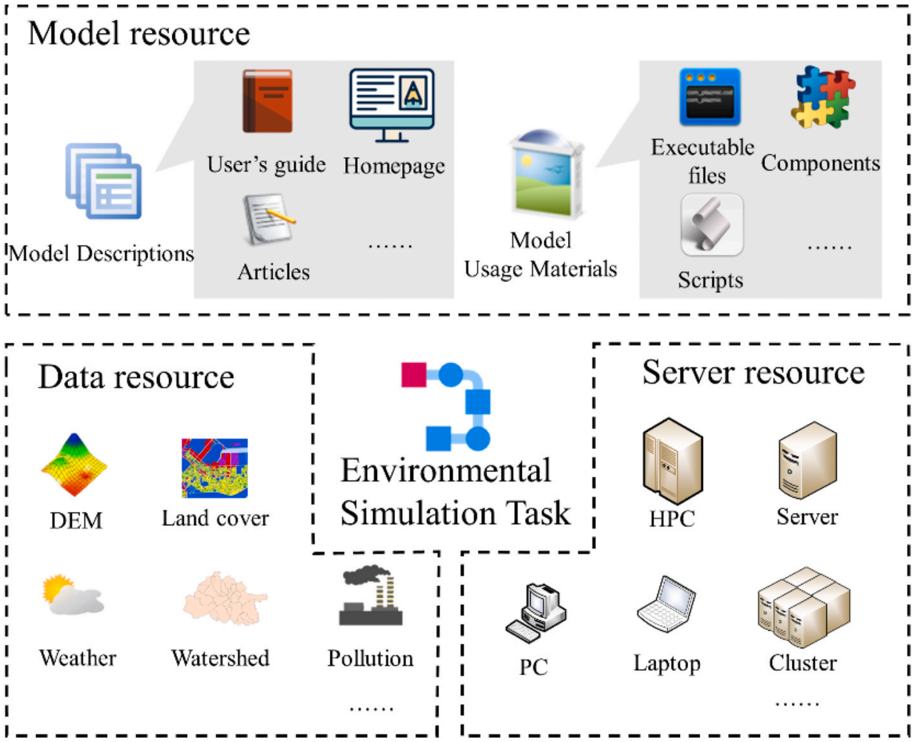


Fig. 1. Simulation resources in environmental simulation tasks.

typically provided by the users. Server resources are used to load and execute the model resources and differ based on requirements; these may include PCs, servers, high performance computers (HPCs) and so on, with different operating systems, such as Windows, Linux, or MacOS.

2.2. Usage scenario design

As shown in Fig. 2, this study aims to design a set of service-oriented interfaces for the roles (model descriptors, model providers, server providers and model users) and related simulation resources (model, data and server resources) that can support resource sharing and reuse for environmental simulation tasks. In this scenario, with the help of sharing and reusing processes on the web, the participants use the interfaces to contribute their resources for the simulation task.

In environmental simulations, different participants could provide different resources and finish the simulation (Voinov and Bousquet, 2010; Voinov et al., 2016). Model description, as one part of the model resource, always comes from model descriptors, who may be model authors or model founders. They always provide these descriptions by media, and others who want to understand the models need to acquire the unstructured descriptions from these media. Model usage materials—another type of model resource—are typically available from the model providers. Using various programming and wrapper techniques, the model providers also provide model usage materials in multiple formats such as components (*.dll), executable files (*.exe), scripts (*.py, *.R) and web services. Server resources usually come from server providers. Server providers also need to provide information regarding which server resources are suitable for model execution. Model users play the role of organizing simulation tasks. These users may be investigating an environmental issue and have the related data resource that is needed to run an environmental simulation.

A real scenario may not include separate participants for each role mentioned above because participants can play more than one role. In many cases, the model descriptions and usage material may be shared by the same providers. They can define the model and build the usage file at the same time. In addition, as computers become more advanced, model users often possess sufficient resources to run models themselves because most models do not need HPC resources. Thus, model users may

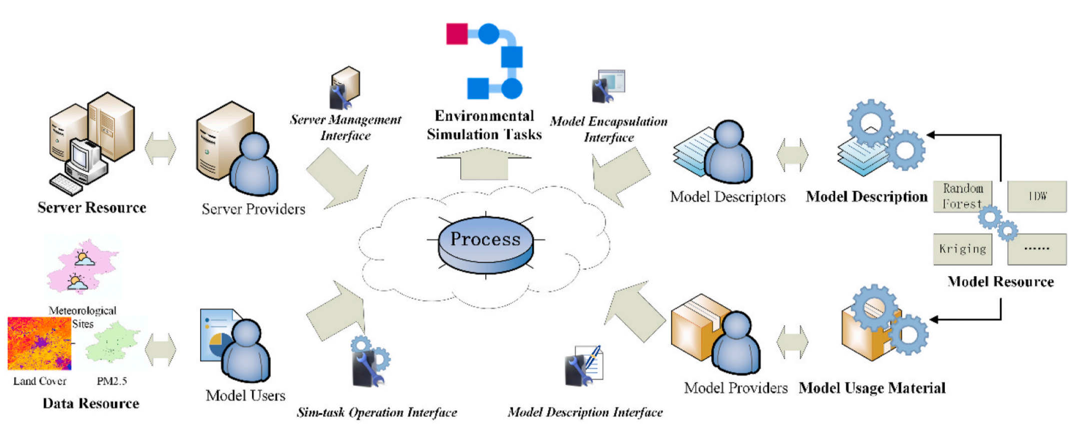


Fig. 2. Resource sharing and reuse in a simulation task.

also play the server provider role and make their own PCs available as server resources.

With the above roles, this article designs four interfaces to support related resource sharing and reuse. The interfaces are listed below:

- Model description interface: support model descriptors to share model descriptions.
- Model encapsulation interface: support model providers to share model usage material.
- Server management interface: support server providers to share server resources.
- Sim-task operation interface: support model users to share data resources, and reuse model, data, and server resources.

2.3. Interface-based process design

As shown in Fig. 3, the sharing and reuse process design for the environmental simulation task has four interfaces (1–4) and involves five tools (a to f). The model description interface (1) aims to describe model resources using unstructured information. The model encapsulation interface (2) is used to wrap the heterogeneous materials used for model resources. The server management interface (3) helps users manage their server resources, and the sim-task operation interface (4) can be used to assemble simulation resources and invoke shared models.

These tools are designed to express or manage the simulation resources in environmental simulation tasks and can take different forms, including files, URLs, and software. The tools include the Model Description Language (MDL) document (a), wrapped model programs (b), model deployment packages (c), service loaders (d) and model services (f). These tools already have documentation that specifies the details (Yue et al., 2016; Wen et al., 2017; Zhang et al., 2019b).

The whole process has several steps, as shown below:

- Model resources are usually described by the model description interface (1) and formatted into MDL documents (a). The MDL document focuses on describing a model; it includes a detailed description of model properties and execution parameters (Yue et al., 2016). Users can obtain the model description information by parsing the MDL documents.
- The model usage material can be a wrapped model program (b) using the model encapsulation interface (2). The wrapped model program is the entry point for running a model and exposes universal types of interactions and behaviors. This wrapper can be an executable file or script that can be executed in a target computing node.
- The MDL document (a) and wrapped model program (b) can be packed as a model deployment package (c) with a designed role. The model deployment package is a package file that contains the file necessary to run the model, and it can be deployed in a service loader (d) as a model service.
- Service loader (d) is software that can be deployed and published as web services or executed in a local area network. Readers seeking more information about sharing and reuse for service deployment and service loader can refer to Zhang et al. (2019b). Service loader and related server resources can be exposed through the server management interface (3).
- Model services (e) are service-oriented model resources on the web. They expose an API through which model users can upload data and expose them as services and access the services using the sim-task operation interface (4). Model users can also assemble various types of resources for the environmental simulation tasks by invoking a model service via the interface.

3. Detailed interface design

3.1. Criteria

As mentioned in Section 3, the process has four interfaces for three types of resources. Different resources have different requirements for resource sharing and reuse. Thus, different interfaces must be designed for different types of resources, and the interface criteria must meet a series of requirements for sharing and reuse. For example, specified model descriptions require the descriptions of multiple fields, including

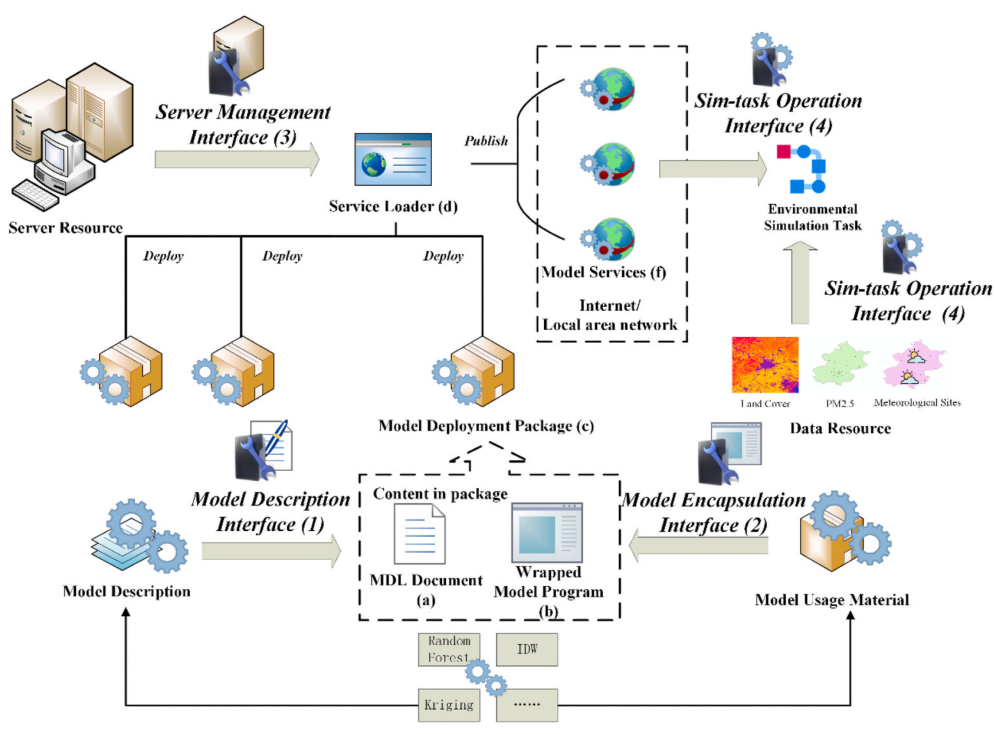


Fig. 3. Sharing and reuse process design for environmental simulation.

the model name, its input/output, its goal, etc., and the interface should be able to retrieve and modify these fields. Moreover, the interface should support interactions from multiple programming languages to make them suitable for heterogeneous environments. Below, we list some of the interface design criteria.

- Cross platform: The tools and interfaces must support heterogeneous platforms such as Windows, Linux and MacOS.
- GUI support: The interfaces should not only support programming interfaces but also GUI-based interactions with the resources and other content.
- Multilanguage support: Insofar as possible, the SDK interfaces need to meet different users' programming preferences (e.g., C#, Python, R, etc.).
- RESTful style: All service-oriented interfaces in the web should follow the easy to use RESTful style.
- Support for sharing restrictions and usage permissions: Some information and functions need to be protected because of security considerations.
- Detail interface description and related parameters: The interfaces and containing function should have detailed descriptions.

Based on the above criteria, this study designs four interfaces. First, this article plans to use different programming languages (such as C#, Python, JAVA, etc.) for the model description interface and model encapsulation interface to build a universal structural description and interaction mechanism. Then, this article includes the design for a service-oriented server manager interface and sim-task operation interface. These interfaces would be published as web services and a corresponding SDK is designed for linking with them. The detailed interface design is shown below.

3.2. Model description interface

Model resources typically have many properties. Such properties are basic information, attribute information, behavior information and environment information. The basic information aims to indicate the model resource. The attribute information includes descriptive keywords and sentences that describe what the model is, which category it belongs to, and what problems the model can solve. Hence, the behavior information can include running behaviors (such as initialization, parameter adjustment and data exchange) and related data templates. Behavior information also needs to support model encapsulation. Different models also require different environments; this need is supplied by the environmental information, which can be useful in seeking a suitable server.

The model description interface includes four modules for model resource description and some functions to transform them. As shown in Fig. 4, the model description interface, starting with the *IModelClass*, has basic functions, three subinterfaces to describe models and I/O functions. The subinterfaces consist of *IAttributeSet*, *IBehavior* and *IRuntime*, which describe the attribute information, behavior information and environment information, respectively.

Basic functions are designed to describe the complete information of the model resources, including the model name, UID, and execution style. The model name is the name of the model resource; A UID is a Unique Identification string assigned to specific model resources. The execution style is a field that lists the different execution styles of models, which have three styles: state simulation, simple calculation and time series. This field can be considered as a code template for model encapsulation. Related research about execution style can be found in Yue et al. (2015). This field also has four functions for formatting and parsing Extensible Markup Language (XML) files or streams. In addition to the above attributes, users also have three interactive functions that can retrieve the three subinterfaces (*IAttributeSet*, *IBehavior*, *IRuntime*) in *IModelClass*.

IAttributeSet aims to classify model resources by category and provide a brief introduction. First, *IAttributeSet* can state category information about a model resource based on different classification systems. For example, the FDS model can be classified as *GIScience & Remote Sensing-> Geographic Simulation* in the classification system named *Geography Subject* in the OpenGMS platform. Thus, the *principle* should be *Geography Subject*, and the *path* should be *GIScience & Remote Sensing - Geographic Simulation*. In addition, the model introduction may be

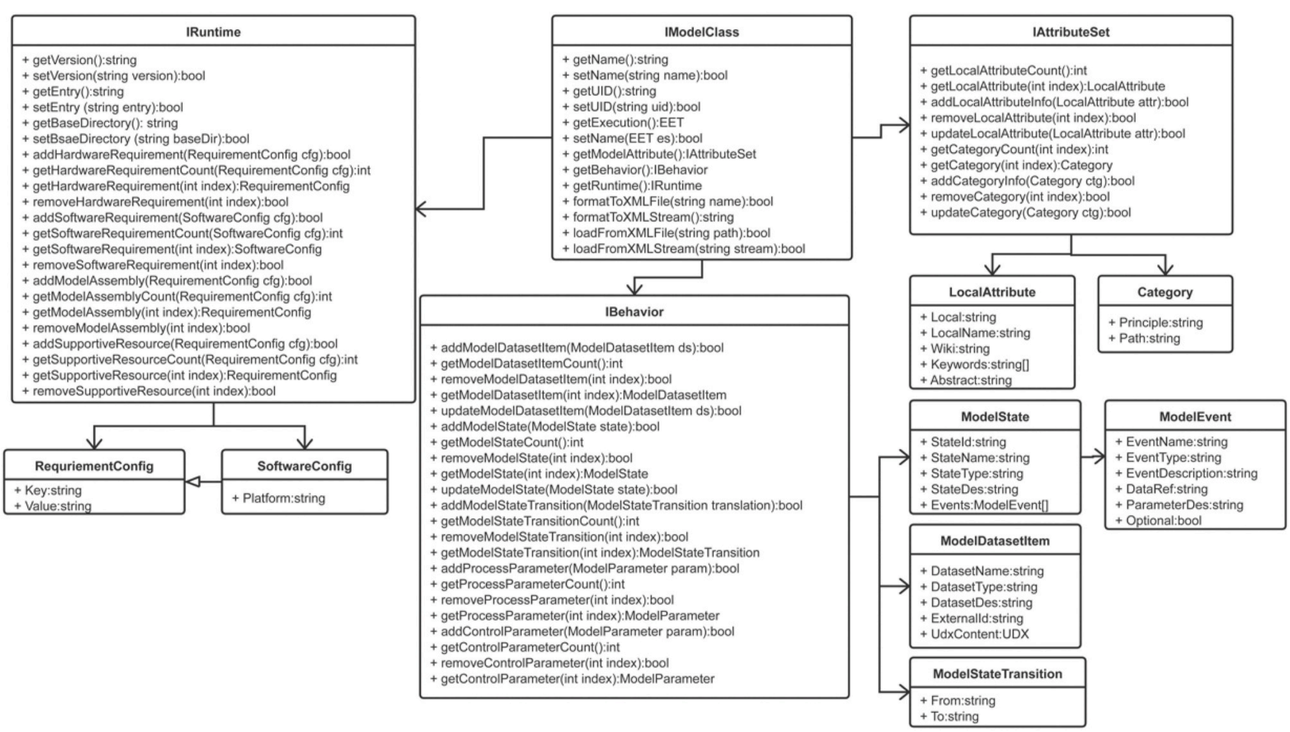


Fig. 4. Model description interface.

available in many different languages. In *IAttributeSet*, the *local* attribute can be set to reflect different languages. The *local* settings for different languages are specified in ISO 639–1 and ISO 639–2 (Byrum, 1999). In addition to the *local* property, this interface can also set a *localName* for the model name in the corresponding language. *Wiki*, *keywords* and *abstract* hold the outside homepage link for model introduction (URL), model keywords and a brief introduction to the model, respectively.

IBehavior is an interface for describing the model running behavior. Running a simulation always involves features such as steps, parameters and input/output data. The IBehavior interface includes three properties for describing these features: dataset item, parameters and state/event. The data item information is described by fields such as *datasetName*, *datasetType*, *datasetDescription*, *externalId*, and *UdxContent*. The *datasetType* includes both the internal and external types.

The internal dataset item describes the data schema inside the *UdxContent* using the Universal Data eXchange (UDX) model, and the external dataset item links the outside data item schema through the *externalId* field. UDX is a data model that can be used to describe heterogeneous data (Yue et al., 2015). The *IBehavior* interface model descriptors include two types of global parameters: control parameters and process parameters. Control parameters is initialized at the beginning of model execution. Process parameters contain values that are important in allowing model users to obtain the key value of simulation during running. States and events are designed to show the various model steps and the input/output (I/O) behavior of the running model via the *IBehavior* interface. In *IBehavior*, I/O can be described as states and events. Related definitions can be referred to Zhang et al. (2019b).

IRuntime is an interface that shows information about model execution dependencies and its runtime environment. Execution dependency information includes the version number, entry file and basic directions through the fields: *version*, *entry* and *baseDir*.

Running environment information includes hardware and software configuration, assembly information, and support resources. These are described by the fields *hardwareConfigures*, *softwareConfigures*, *assemblies*, and *supportiveResources*, which include *key* and *value* pairs that describe the runtime environment. *HardwareConfigures* and *softwareConfigures* are designed to describe the hardware and software environment of server resources, such as central processing unit (CPU), memory, python, etc. *Assemblies* aim to reflect the public or private library dependencies of models at runtime. Public assemblies are public libraries or related files stored in the model loader (e.g., service container) that can be shared with all models stored in that loader, while private assemblies are libraries or related files stored in the model package. *SupportiveResources* record the necessary software or libraries attached to the package. Users can install them from the software included in the package.

3.3. Model encapsulation interface

Model encapsulation means wrapping a model as a standard model. The main task involved in wrapping a model is data interaction mapping, which can be performed by following the states/events in MDL. The states and events aim to map the model's native logic to computer logic in the encapsulation for standardizing the models as components in a service loader.

To meet the requirements mentioned above, we design the *IModelServiceContext* for model providers to map methods during model encapsulation. As shown in Fig. 5, the *IModelServiceContext* interface provides functions for switching status, interacting with data and parameters, and posting messages. First, the status switching functions change the status of the model based on states and events. All simulation processes start at *Initialize* and end with *Finalize*. *Enter State* and *Leave State* are state switches during model execution, and during the current state, the model needs to fire events that involve I/O using *Fire Event*. Then, data and parameter interactions allow obtaining and setting the model's data and parameters, such as data requests and responding to data and parameter interactions. Each data transmission has three parts: *Flag*, *MIME* and *Body*. *Flag* has three statuses that show whether the data are ready or have an error. The *MIME* type describes the format and type of data. *Body* reflects the value of data. There are also two types of global parameters: process parameters and control parameters declared in *IBehavior*. *IBehavior* also has message posting for exchanging messages with the model users, including normal messages, warning messages and error messages.

Fig. 5. Model encapsulation interface.

3.4. Server management interface

The interface *IServerAdmin* should shows the basic information and active hardware information needed to deploy a new model service and manage server resources. Shown as Fig. 6, the basic information in *IServerAdmin* includes the service loader version, platform (e.g., Windows, Linux, MacOS, etc.) and host. The active hardware information includes current CPU, message resources and hard disk usage. In addition, this interface can also deploy new model services using model package files. When users deploy model services, they need to set properties such as accessibility, which determines restrictions on model service access by others. The manager can also acquire other resources stored by the server, such as model service logs, running instances and data caches. Although server resources are published as services on the web, the server management interface should include user access restrictions because server resources contain private information and functionality, such as CPU type, memory size and deployment capabilities. An interface that does not guard against access to such information and functions would be accessible by anyone, which is dangerous. Thus, all server management interfaces are protected by a designed app key. The app key is a secret key generated by the service loader and used by the administrator and other management personnel to check and control the servers. All requests to the server management interface must append the key.

The server management interface is designed for use by a manager of server resources. The interface is designed to be published as web API

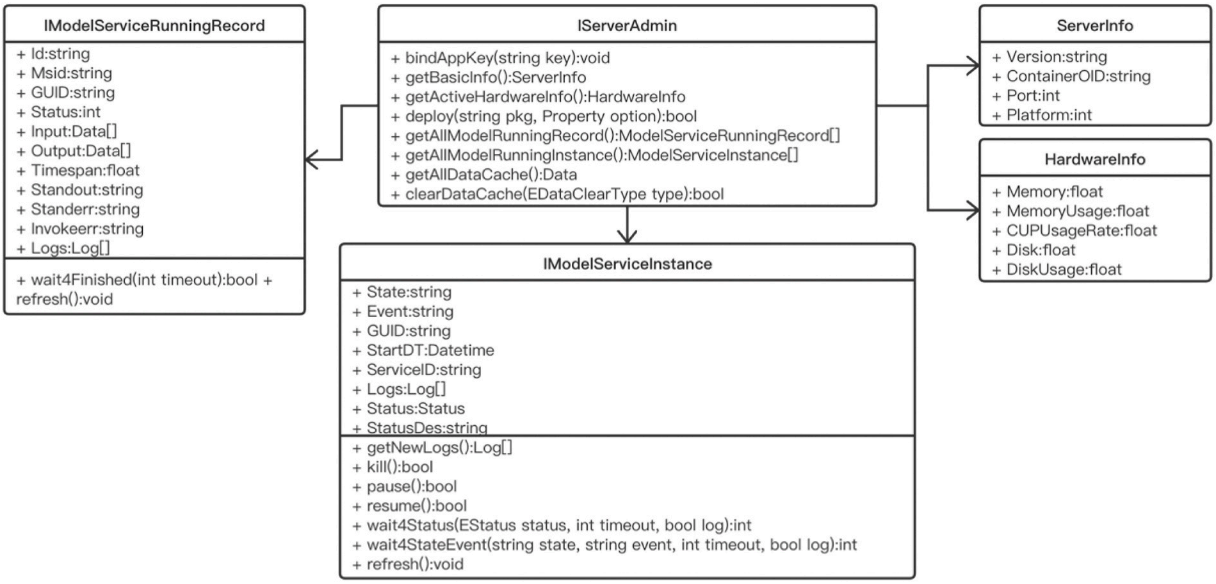


Fig. 6. Server management interface.

through the HTTP protocol. As previously mentioned, service-oriented model resources can be flexible on the web, and component-based styles are more familiar to model users. This research designs a service-oriented interface that can also support component-based development. The SDK uses the web request module to wrap model services as a component. As shown in Fig. 7, the components in the user's computer link to the corresponding service via the network; for example, component A links with service A and component B links with service B. User can use an application to invoke the service by interacting with the corresponding component.

Based on the server management interface, in this study, we built a GUI in web apps and software to manage other server resources (Zhang et al., 2019b). As shown in Fig. 8, web applications and software can manage server resources and contain related model resources using the interface.

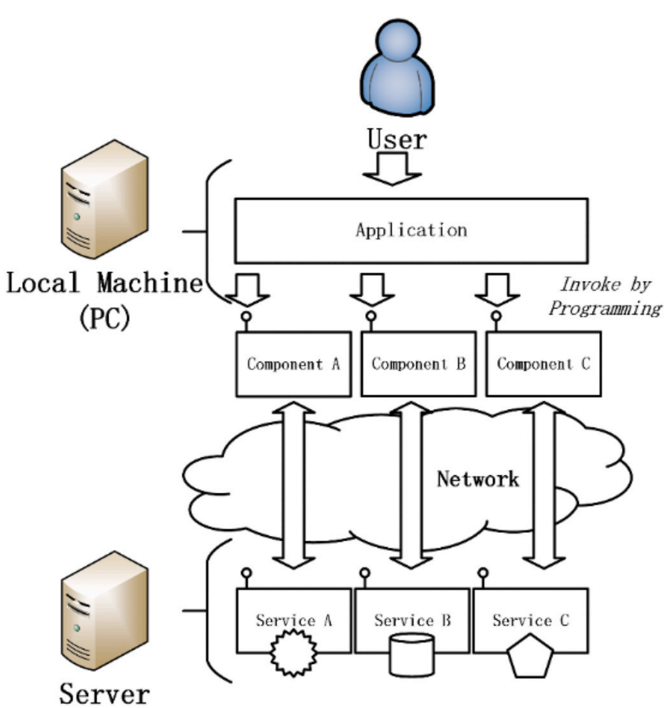


Fig. 7. Service-oriented style component.

3.5. Sim-task operation interface

For the models that the server management interface publishes as services on the web, a service loader publishes a web API so that users can access the models and related services using the sim-task operation interface. The sim-task operation interface includes many subinterfaces that apply to different resources or functions, including *IServer*, *IServiceAccess, IModelService, IData, IModelResourcesRecord, and IModelServiceInstance*. The relationships between two subinterfaces are shown in Fig. 9. Before users invoke a model service, they need to bind the target server resource, using the *IServer* interface and specifying an IP address and port. Then, through *IServer*, model users can obtain the handlers for related resources. The *IServiceAccess* interface provides functions for checking, binding and generating related resources, such as binding a model service and uploading a data file. Users can retrieve other interfaces to access these resources, including *IModelService*, *IData*, *IModelResourcesRecord* and *IModelServiceInstance*. *IModelService* is used to obtain information and invoke the model. IData allows access to the input or output data. *IData* indicates the data resources in the service loader and downloads them to the local machine. *IModelServiceRecord* shows information about the running model and *IModelServiceInstance* provides a model that exposes a handler so users can interact with the running model.

4. Experiment

As an example, we designed an experiment that involves PM2.5 concentration distribution inversion by a random forest algorithm in Beijing in 2014. Many studies have suggested that the PM2.5 concentration is related to land cover and meteorological indicators (Guo et al., 2017; Meng et al., 2018; Li and Zhang, 2019). The data in this task include land cover data, meteorological data and PM2.5 concentration data. The experiment uses the designed interfaces to wrap the model resources and publish them as services on the web. Then, we assemble the model service, data and server resources for the simulation task and analyze the results.

4.1. Study area

Beijing (39°56′N, 116°20′E) encompasses an area that exceeds

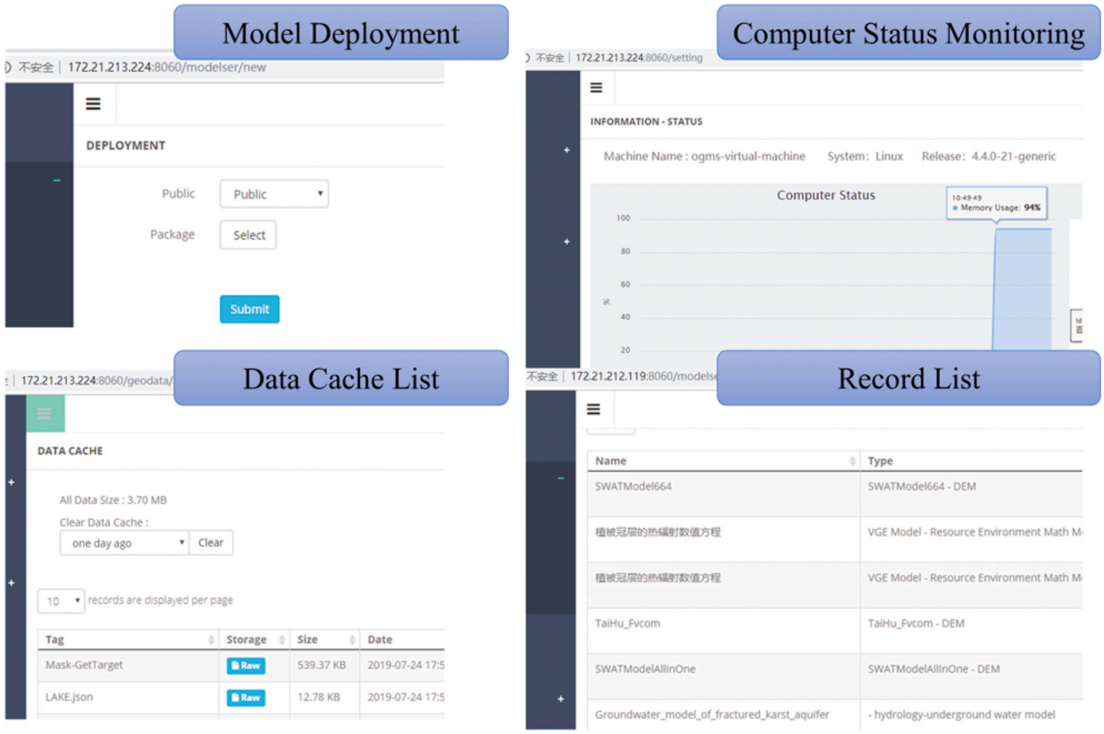


Fig. 8. Server management application interface.

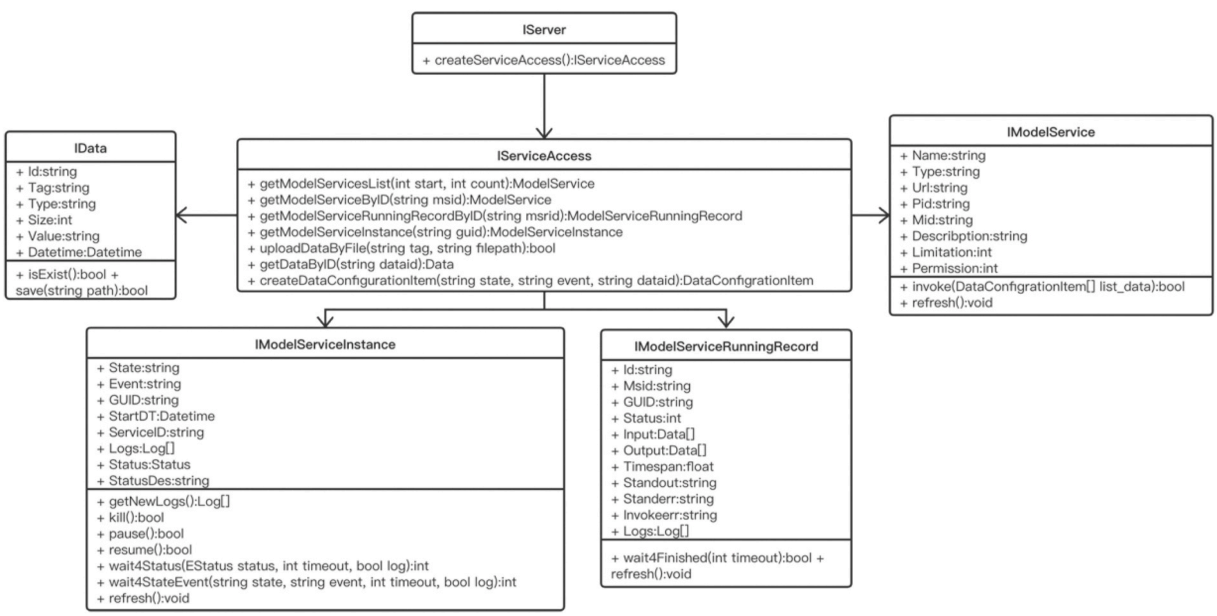


Fig. 9. Sim-task operation interface.

16,410 km^2 with a population of more than 21 million people (to 2019). Beijing is located along the northern edge of the North China Plain and has mountains in the western and northern regions—Xi mountain and Jundu mountain. Summer in Beijing is hot and rainy, whereas winter in Beijing is characterized by a cold and dry climate. In recent years, Beijing has undergone fast urbanization and industrialization (Han et al., 2014). The fast development of Beijing has led to massive energy consumption, and the resulting pollutant emissions (such as PM2.5 pollution) increase year by year, which has adverse impacts on air quality, human health and the eco-environment (Li et al., 2011, 2014).

4.2. *Dataset*

The land cover data are obtained from the Land Cover Climate Change Initiative (CCI) and downloaded from the European Space Agency (EPA) (http://maps.elie.ucl.ac.be/CCI/viewer/index.php). The Land Cover CCI comprises annals data; we choose the year 2014. The resolution of the data is 300 m, and as shown in Fig. 10, land cover is classified as 18 types (in Beijing), such as urban area, forest, etc. The legend of land cover is shown in supplementary information, Appendix A. The data are shown in the OpenGMS platform (http://geomodeling.njnu.edu.cn/dataItem/5f198f5a1d03e5614d87862b).

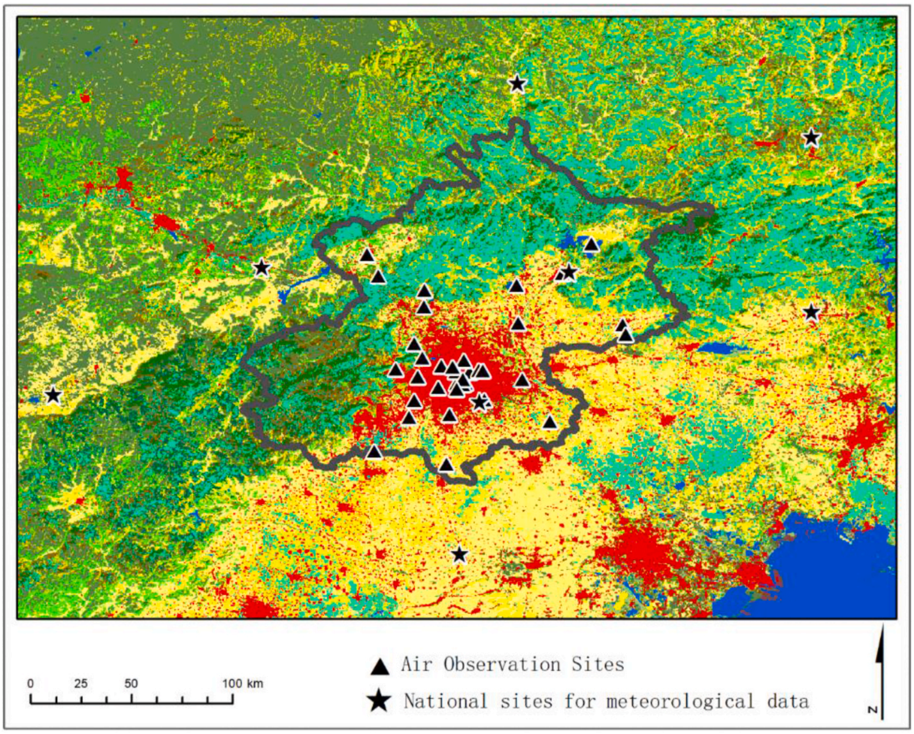


Fig. 10. The distribution of air observation sites, national sites for meteorological data, and land cover in and around Beijing.

PM2.5 concentration data are derived from national sites monitoring data. As shown in Fig. 10, the data are obtained from 35 air monitoring sites distributed in Beijing by hour, from 0 am 1/1/2014 to 11 pm 12/31/2014. The data have been processed as daily data with the average data of all valid data in one day and formatted in one shape file. The data are shown in the OpenGMS platform (http://geomodeling.njnu.edu.cn/dataItem/5ebacc831d03e539d6a4dd2e).

As show in Fig. 10, the meteorological data consist of monitoring data from eight national sites (Beijing, Miyun, Huailai, Fengning, Chengde, Zunhua, Langfang, and Weixian) and cover daily data for the period 1/1/2014 to 12/31/2014. The meteorological data fields include max/min/mean temperature (°C), precipitation (mm), mean/max wind speed (m/s), sunshine duration (h), min/mean relative humidity (RH) (%), max/min/mean pressure (hPa) and large/small evaporation (mm). Small evaporation refers to the evaporation from the 20 cm pan (E20), and the large evaporation refers to the evaporation from the E601 evaporator. (Mao and Jiang, 2009; Xaymurat and Huang, 2011). Due to data missing, we decided to abandon the data from 12/1/2014 to 12/31/2014. The data are shown on the OpenGMS platform (http://geomodeling.njnu.edu.cn/dataItem/5ebace5f1d03e539d6746f2f).

4.3. Methods

The goal of this experiment is to predict the PM2.5 concentration distribution in Beijing by land cover data and meteorological data with different interpolation methods. As shown in Fig. 11, with the aim of matching meteorological data with monitoring sites, this study employs IDW interpolation and kriging-spherical interpolation to simulate the distribution of meteorological data. This study analyzes the results obtained by the application of different interpolation methods to meteorological data to show the influence of interpolation methods on a random forest.

This study extracts meteorological data, land cover data and PM2.5 concentration data in a daily dataset. Due to the limited dates of meteorological data, the PM2.5 data are limited from 1/1/2014 to 11/30/2014. We then split the PM2.5 concentration data into training data and validation data by different air monitoring sites for cross-validation. We chose 20 sites for training and 15 sites for validation; the splitting of air monitoring sites is randomly performed by computer. Before simulation, this study deleted invalid data in PM2.5 concentration data, including values of 0 and values that exceed 1000.

A random forest is a machine learning algorithm that has been demonstrated useful in previous studies for PM2.5 prediction (Hu et al., 2017; Stafoggia et al., 2019). Using the Random Forest algorithm, we train a predictive model and obtain the prediction results. We validate the trained model using the verification data and obtained the assessment results by measuring the R-squared (R^2) and root mean square error (RMSE) (Gong et al., 2014).

In this study, the related functions are encapsulated as model services using the roadmap described earlier. The model services that need to be wrapped are listed in Table 2. Related models, tools and libraries are provided in the OpenGMS platform (http://geomodeling.njnu.edu.cn/).

4.4. Task design and servers' deployment

In this study, we deployed 3 servers to support task operation. The details of these servers are shown in Table 3. These servers are equipped with different environments to satisfy different requirements of model services. For example, the model service *RandomForest_train* needs Sklearn and GDAL; thus, server C needs to install Sklearn and GDAL for *RandomForest_train*, and users in the same web environment can invoke it. Furthermore, this design can balance task loads among different servers. After deployment, with the help of the sim-task operation interface, the steps of the whole experiment are implemented as follows:

a) Process the daily meteorological data from 1/1/2014 to 12/31/2014 using the *IDW_Extract* or *Kriging_Extract* models.
b) Use the *SitesPicking* model to randomly split the sites into a training site and a validation site.

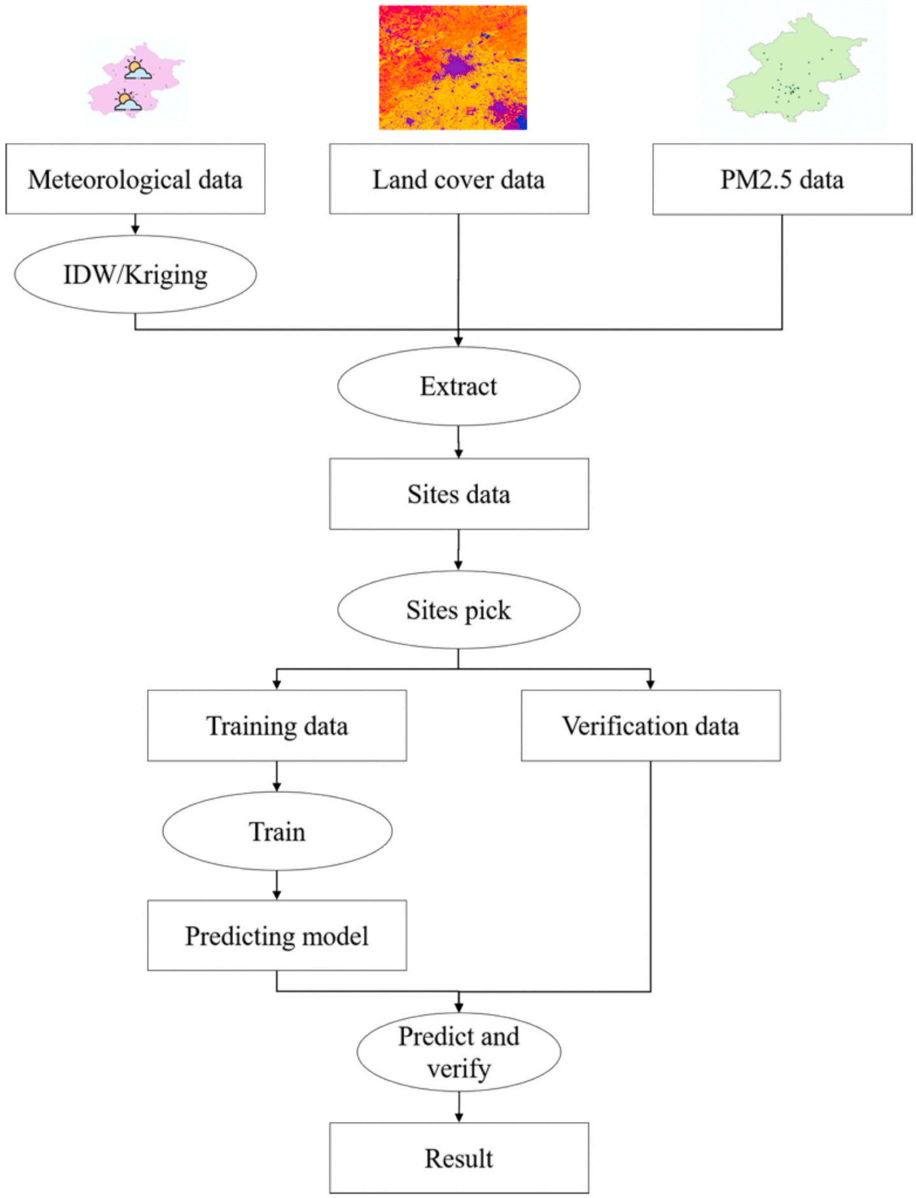


Fig. 11. Roadmap for the experiment.

c) Combine meteorological data with PM2.5 concentration data and land cover data, and then split them into a training dataset and a validation dataset by the *RandomForest_Preprocess* model.
d) Train the model using the *RandomForest_train* model to construct the predicting model.
e) Apply the trained model to predict the PM2.5 concentration using the *RandomForest_predict_validation* model, and validate the results by R^2 and RMSE.

Due to the splitting of random sites for training and validation, this study ran the simulation 10 times by each interpolation method to explore the stability of the simulation, and they shared same training sites and validation sites each time. After model description, encapsulation and deployment, users can utilize 167 lines of codes for whole simulation and model switching, whereas interpolation methods can only change serval lines.

5. Results

The whole experiment comprised 10 independent simulations for each interpolation method, and each simulation had different air observation sites for training and validation; they are listed in the supplementary information, Appendix B. The R^2 for each simulation is listed in the supplementary information, Appendix C. The boxplot of the R^2 for each simulation is shown in Fig. 12(a), and the boxplot of the RMSE is shown in Fig. 12(b). The R^2 of IDW ranges from 0.808 to 0.904, and the kriging ranges from 0.815 to 0.921. The RMSE of IDW ranges from 23.375 to 33.576, and the kriging ranges from 21.234 to 33.108. The average values of R^2 and RMSE are 0.866 and 27.894 for IDW, and those of R^2 and RMSE are 0.878 and 26.572 for kriging. The average and median values of R^2 of kriging is higher than those of IDW, and the RMSE of kriging is lower than IDW for these values in general. With different training sites and validation sites in 10 independent simulations, kriging performed better than IDW in nine simulations. The results show that the simulations are better and more stable with kriging than with IDW.

The importance of predictor variables in all simulations based on IDW and kriging is listed in the supplementary information, Appendix D. The importance of predictor variables in all simulations shown in Fig. 13, and the average importance ranking is shown in Fig. 14. As

Table 2
List of wrapped model services.

Model	Description	Dependency	URLs
IDW_ Extract	Use IDW to interpolate the distribution of meteorological data and extract to PM2.5 sites	ArcPy,	http://geomodeling.njnu.edu.cn/modelItem/f656053d-6b50-480a-be00-eace059c3cd8
Kriging_ Extract	Use kriging to interpolate the distribution of meteorological data and extract to PM2.5 sites	ArcPy	http://geomodeling.njnu.edu.cn/modelItem/6334a2c2-daba-4cab-9638-164537125d59
SitesPicking	Randomly split air observation sites for cross-validation	None	http://geomodeling.njnu.edu.cn/modelItem/77f44f3d-2419-4759-842b-a8a45781a5ae
RandomForest_Preprocess	Preprocess the data before Random Forest	None	http://geomodeling.njnu.edu.cn/modelItem/03fdde78-4ce7-49c3-aa3e-d2f7439c538c
RandomForest_train	Use random forest to train samples	Sklearn	http://geomodeling.njnu.edu.cn/modelItem/ee6bb964-ff2c-48ba-b033-55000f6e6578
RandomForest_predict_validation	Use trained model to predict the data and validate the results	Sklearn, GDAL	http://geomodeling.njnu.edu.cn/modelItem/bdd0364a-5398-401b-816e-92753ee6eb37

Table 3
List of server resources.

Server	Model service	Environment
Server A	IDW_ Extract, Kriging_Extract	ArcPy
Server B	SitesPicking	None
Server C	RandomForest_train, RandomForest_predict_validation, Preprocess	Sklearn, GDAL

shown in Figs. 13 and 14, both IDW and kriging have similar key predictors in simulation. For example, evaporations (small[evamin], large [evamax]) and RH (average[rhmean], minimum[rhmin]) are top-ranking variables in both IDW average and kriging-based simulation. The other predictors (including maximum/minimum/average temperature[tmpmax, tmpmin, tmpmean], sunshine duration[sd], maximum/average wind speed[windmax, windmean], precipitation[pcp], maximum/minimum/average pressure[prsmax, prsmin, prsmean], and land cover[landuse]) rank similarly. In contrast to IDW, wind speed ranks higher than precipitation, and average temperature ranks higher than minimum temperature and sunshine duration in kriging.

6. Discussion

6.1. Methods and dataset

The goal of this research was to design set of service-oriented interfaces for sharing simulation resources to enable simulations to be executed on the web. With this interface design and the use of meteorological data and land cover data, this research finished the PM2.5 concentration distribution simulation with relatively high mean R^2 values (0.866 and 0.878) in different interpolation methods for meteorological data (IDW and kriging). IDW and kriging are interpolated methods for sites data, which have been demonstrated to be useful in meteorological data interpolation (Meng et al., 2018; Xue et al., 2019). As a result, the kriging interpolation would be better for random forests in PM2.5 distribution simulation with higher R^2 and lower RMSE values. However, the importance ranking between these two kinds of simulation seems to be similar, which may be caused by the small scale of the study area and fewer meteorological sites.

Land cover data contributed less to the distribution prediction in one year. Compared with daily PM2.5 data, land cover data are fixed predictors that are only distributed in space. However, some studies have suggested that land cover data are useful in PM2.5 prediction on a larger scale (China and U.S.) and longer time (2005–2016 or 2005–2015) by the Random Forest model (Chen et al., 2018; Meng et al., 2018). Furthermore, the land cover cell cannot consider the surrounding environment. It expresses only the value in itself, not related surrounding cells. Thus, land cover data applying a regional scale or a larger study area or longer time could provide better results. Land cover cells, which contain more surrounding information, could also have a higher contribution in the simulation.

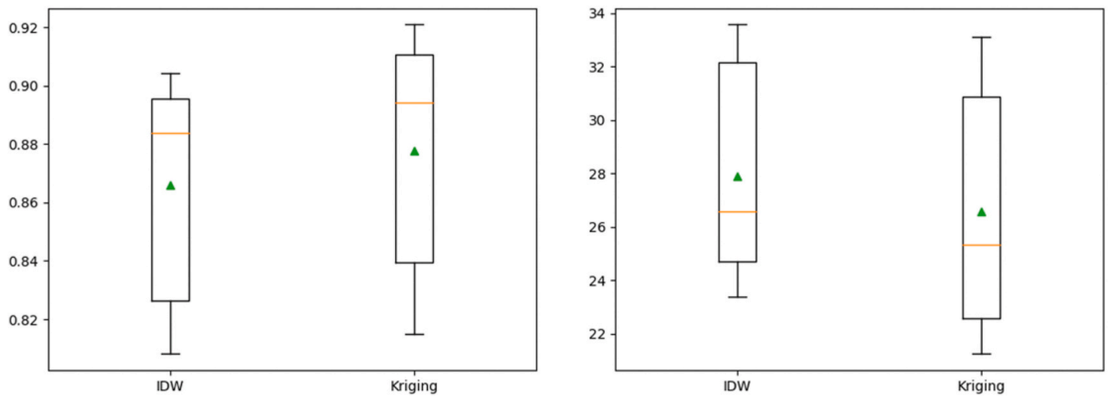
The evaporation and RH are shown to be substantially more important in the prediction. Previous studies have revealed that RH has an excellent correlation with PM2.5 concentration and is one of the key indicators in static-stability air days from 2013 to 2015 in Beijing (Zhang et al., 2015; Yin et al., 2016; Wang et al., 2018). Ma et al. (2017) has suggested that when the RH is very high, the air is stagnant and poor air flowability hinders the spreading of PM2.5. For evaporation, it has been proved that the rate of evaporation is mainly dependent on the airflow velocity in wind tunnel measurements and to RH and temperature (Raimundo et al., 2014). Another study has indicated that higher RH can cause lower evaporation (McCulley et al., 2006). Thus, the evaporation may be inversely proportional to the PM2.5 concentration, and high evaporation and low evaporation can be reasonable predictors for PM2.5 concentration simulation.

6.2. Flexibility of simulation configuration

Regarding focus, some concentrate on sharing single resources and reuse, aiming to reduce maintenance costs and simplify popularizing the models. In this study, we designed interfaces for sharing and reuse of three types of resources. Adhering to these interfaces can make self-maintenance more difficult; however, it also allows users to give a clear description of simulation and assemble these resources into environmental simulation tasks in a manner similar to building a Lego model. For example, in the experiment provided above, users can replace the Random Forest model by others, such as a neural network algorithm or other models built by experts. Alternatively, in another case, users can replace the *IDW_Extract* model by the *Kriging_Extract* model and obtain different results. Furthermore, by the choice of other models, providers could also furnish models with different granularities, thereby facilitating model wrapping of finer granularity that is more flexible for simulation task building. In addition, model providers may disassemble the comprehensive model as several small models that could be reused with other models to build simulation tasks.

6.3. Roles' classification

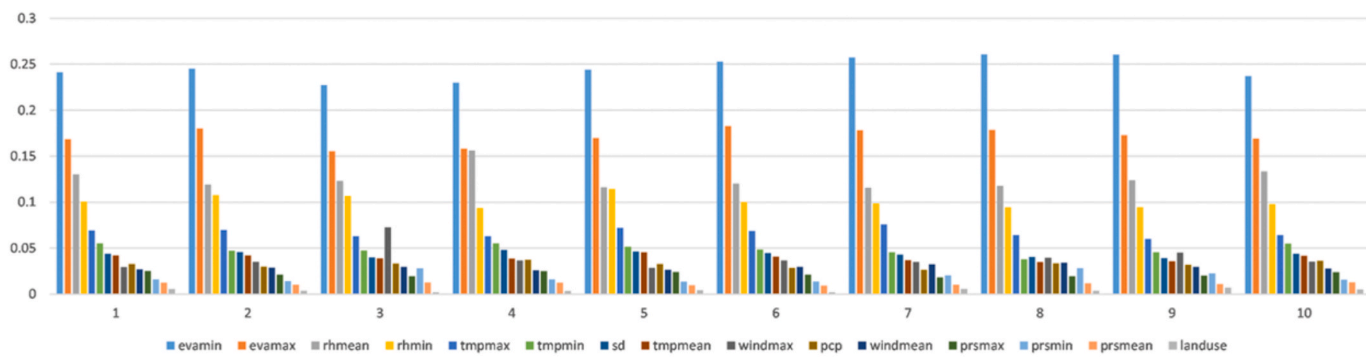
The advantage of roles' classification in the simulation task is that the participants may come from different domains or their own resources match only their strengths (Chen et al., 2019). For example, some scholars and researchers at universities or research institutions are environmental modelers who published papers in academic journals to describe their models; however, they lack programming skills. Thus, they cannot provide both the description and model usage material, whereas if they collaborate with a group good at programming but less skilled at model description, the collaboration can provide the complete model resource. The other benefit of roles' classification is that it allows distributed simulation resources to be reused effectively. Roles' classification can guide the granularity of classifying simulation resources for a simulation task. After classification, these shared resources can act as components or service around the world for other users and may be



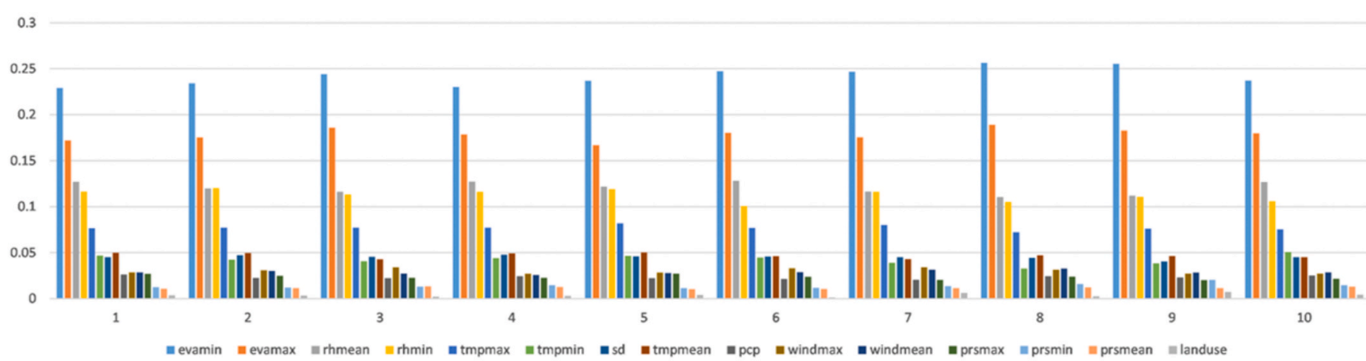
(a) Box plot of the R^2 for IDW and Kriging

(b) Box plot of the RMSE for IDW and Kriging

Fig. 12. Box plot of the R^2 and RMSE for IDW and Kriging.



(a) Importance of all variables in all simulations based on IDW



(b) Importance of all variables in all simulations based on Kriging interpolation

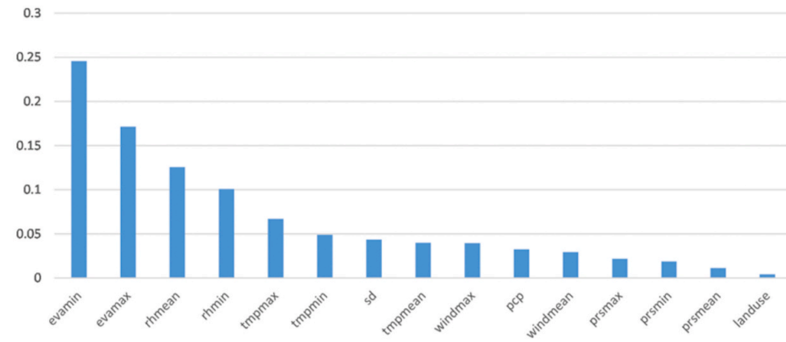
Fig. 13. Importance rankings for all variables based on IDW and kriging interpolation of meteorological data in ten independent simulation.

reused multiple times. For example, model providers may share model resources on the web in Nanjing, and server providers may provide server resources in Beijing. Therefore, model users can build simulation tasks with these shared resources around the world.

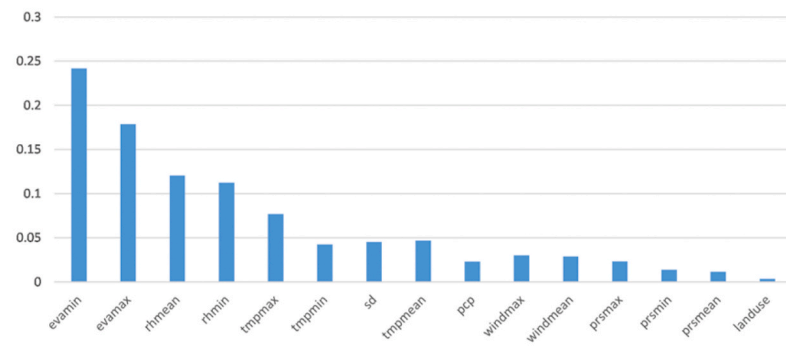
6.4. Strengths of service-oriented style

Compared with the service-oriented style, the component-based style can achieve better running performance and is friendlier and more transparent. The performance of the component-based style can be more effective for interactions than the service-oriented style because it does not consume time exchanging data and messages over the web (Goodall et al., 2011). Furthermore, the component-based style is friendlier and more transparent for model users because it would be easier to invoke components that run on a local machine than to invoke web services, and if the component is open source, it can be modified by users to address new requirements for simulation.

However, increasingly more related groups and institutions are working on model services wrappers and publication standards based on the Simple Object Access Protocol (SOAP) and RESTful styles. It is due to the service-oriented style that can be more convenient, lightweight and flexible. When using the service-oriented style, model users do not need to copy files to their local machines; they simply invoke the web service URLs and obtain a service target model. Thus, service-oriented sharing is more suitable for distributed resources and saves hard disk space, memory, and local CPU usage (Huhns and Singh, 2005). Furthermore, providers typically impose no programming language restrictions. For example, Python-based components should be invoked by Python, and other programming languages would have more trouble to invoke them. Thus, the service-oriented style can reuse distributed resources on the



(a) Average importance rankings of all variables based on IDW of meteorological data



(b) Average importance rankings of all variables based on Kriging interpolation of meteorological data

Fig. 14. Average importance rankings of all variables based on IDW and kriging interpolation of meteorological data.

web to the greatest extent possible and avoid resource wasting. As the performance and reliability of the Internet increase, the current shortages in service-oriented style offerings will disappear. Service-oriented style is becoming the main trend for model sharing in environmental problem-solving (Goodall et al., 2011; Jiang et al., 2017; Zhang et al., 2019a).

Nonetheless, service-oriented architectures still have some limitations, including performance and security. Initially, compared with the component-based and tight-coupling styles, service-oriented architecture typically has performance disadvantages for running models. Web requests always include network latency during interactions between users and models or models and models. In contrast, communication between components or processes in the component-based style and tight-coupling style running on a single computer typically require less time. Second, service-oriented architecture exposes the related simulation resources via the web so that users can access them. We need to balance the desire to build an open environment for simulation resource sharing with the need to protect the security of the simulation resources involved. Future research should include stronger resource security control during resource access.

7. Conclusion and future work

Based on the features of these simulation resources, this research analyzed the scenarios and designed a set of interfaces for simulation resource sharing and reuse for simulation tasks. The designed service-oriented interfaces can classify the simulation resources from the simulation task, and support simulation resource reuse, task configuration and task load balancing in simulation tasks for solving environmental issues on the web. With this interface design, the PM2.5 distribution simulation, which is based on different meteorological data interpolations, was completed, and the simulation results were compared by R^2 and RMSE. After simulation, we determined that Kriging is more suitable than IDW for meteorological data in PM2.5 concentration simulation by the Random Forest model. However, simulation sharing and reuse is ongoing, and additional research topics will be addressed in the future:

(1) Model information descriptions need more detailed interfaces. As different disciplines have developed, models are becoming increasingly complex, and their potential information is becoming increasingly rich. Such information should be described structurally, such as with a basic calculation grid, domains, time spans, etc. Richer structural descriptions of model resources could support further environmental simulation applications, such as model coupling and integration, data processing service assembly, and collaborative modeling. For example, users who want to integrate two models could compare the grid information between the input and the output of these models from their structural descriptions.

(2) Distributed simulation resources around the world still lack a central node to manage them. Even if shared on the web, these simulation resources could exist in different web environments. When these resources are shared on the web, model users can potentially assemble and invoke them by themselves. However, when the resources exist in different web environments, it may be difficult for model users to access them. Hence, a model user may not be able to access a suitable resource because of the lack of a central management node.

Credit author statement

Fengyuan Zhang, Methodology, Writing - original draft, Writing - reviewing and editing. Min Chen, Conceptualization, Reviewing and Editing, Funding acquisition. Songshan Yue, Methodology, Software. Yongning Wen, Project administration. Guonian Lü; , Supervision. Fei Li, Validation, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We appreciate the detailed suggestions and comments from the anonymous reviewers. We express heartfelt thanks to the other members of the OpenGMS team. This work was supported by the NSF for Excellent Young Scholars of China (Grant No. 41622108), Key Programme of NSF of China (Grant No. 41930648), the National Natural Science Foundation of China (Grant No. 41701441), the Priority Academic Program Development of Jiangsu Higher Education Institutions (Grant No. 164320H116), and Open Fund of Key Laboratory of Geospatial Technology for the Middle and Lower Yellow River Regions (Henan University), Ministry of Education (Grant No. GTYR201807).

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.envres.2020.110225.

References

Albanesi, M.G., Albanesi, R., 2014. A decision-making support system for land use estimation based on a new anthropentropy predictive model for environmental preservation–theory, model and web–based implementation. Int J Adv Intell Syst 7 (1), 85–102.
Basnyat, P., Teeter, L.D., Lockaby, B.G., Flynn, K.M., 2000. The use of remote sensing and GIS in watershed level analyses of non-point source pollution problems. For. Ecol. Manage. 128 (1), 65–73.
Belete, G.F., Voinov, A., Laniak, G.F., 2017. An overview of the model integration process: from pre-integration assessment to testing. Environ. Model. Software 87, 49–63.
Byrum, J.D., 1999. ISO 639-1 and ISO 639-2: international standards for language codes. ISO 15924: international standard for names of scripts. https://eric.ed.gov/?id=ED441415.
Chen, C., Liu, H., Beardsley, R.C., 2003. An unstructured grid, finite-volume, three-dimensional, primitive equations ocean model: application to coastal ocean and estuaries. J. Atmos. Ocean. Technol. 20 (1), 159–186.
Chen, M., Lin, H., 2018. Virtual geographic environments (VGEs): originating from or beyond virtual reality (VR). Int. J. Digit. Earth 11 (4), 329–333. https://doi.org/10.1080/17538947.2017.1419452.
Chen, M., Lin, H., Hu, M., He, L., Zhang, C., 2013. Real-geographic-scenario-based virtual social environments: integrating geography with social research. Environ. Plann. Plann. Des. 40 (6), 1103–1121.
Chen, Y., Zhou, H., Zhang, H., Du, G., Zhou, J., 2015. Urban flood risk warning under rapid urbanization. Environ. Res. 139, 3–10.
Chen, G., Li, S., Knibbs, L.D., Hamm, N.A., Cao, W., Li, T., Guo, J., Ren, H., Abramson, M. J., Guo, Y., 2018. A machine learning method to estimate PM2. 5 concentrations across China with remote sensing, meteorological and land use information. Sci. Total Environ. 636, 52–60.
Chen, M., Yue, S., Lü, G., Lin, H., Yang, C., Wen, Y., Tao, H., Xiao, D., Jiang, H., 2019. Teamwork-oriented integrated modeling method for geo-problem solving. Environ. Model. Software 119, 111–123.
Chen, M., Voinov, A., Ames, D.P., Kettner, A.J., Goodall, J.L., Jakeman, A.J., Barton, M. C., Harpham, Q., Cuddy, S.M., DeLuca, C., Yue, S., Wang, J., Zhang, F., Wen, Y., Lü, G., 2020. Position paper: open web-distributed integrated geographic modelling and simulation to enable broader participation and applications. Earth Sci. Rev. 103223.
Christensen, E., Curbera, F., Meredith, G., Weerawarana, S., 2001. Web services description language (WSDL) 1.1. . http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.460.467&rep=rep1&type=pdf.
Conde-Cid, M., Fernández-Calviño, D., Nóvoa-Muñoz, J.C., Núñez-Delgado, A., Fernández-Sanjurjo, M.J., Arias-Estévez, M., Álvarez-Rodríguez, E., 2019. Experimental data and model prediction of tetracycline adsorption and desorption in agricultural soils. Environ. Res. 177, 108607.
David, O., Lloyd, W., Rojas, K., Arabi, M., Geter, F., Ascough, I.I.J.C., Green, T., Leaveley, G., Carlson, J., 2014. Model-as-a-service (MaaS) Using the Cloud Services Innovation Platform (CSIP).
Demeritt, D., Wainwright, J., 2005. Models, modelling, and geography. Questioning Geography. Fundamental Debates 206.
Deng, J., Desjardins, M.R., Delmelle, E.M., 2019. An interactive platform for the analysis of landscape patterns: a cloud-based parallel approach. Spatial Sci. 25 (2), 99–111.
Farnaaz, N., Jabbar, M.A., 2016. Random forest modeling for network intrusion detection system. Procedia Computer Science 89 (1), 213–217.
Gan, T., Tarboton, D.G., Dash, P., Gichamo, T.Z., Horsburgh, J.S., 2020. Integrating hydrologic modeling web services with online data sharing to prepare, store, and execute hydrologic models. Environ. Model. Software 104731.
Gao, F., Yue, P., Zhang, C., Wang, M., 2019. Coupling components and services for integrated environmental modelling. Environ. Model. Software 118, 14–22.
Gichamo, T.Z., Sazib, N.S., Tarboton, D.G., Dash, P., 2020. HydroDS: data services in support of physically based, distributed hydrological models. Environ. Model. Software 125, 104623.
Gogu, R., Carabin, G., Hallet, V., Peters, V., Dassargues, A., 2001. GIS-based hydrogeological databases and groundwater modelling. Hydrogeol. J. 9 (6), 555–569.
Gong, G., Mattevada, S., O'Bryant, S.E., 2014. Comparison of the accuracy of kriging and IDW interpolations in estimating groundwater arsenic concentrations in Texas. Environ. Res. 130, 59–69.
Goodall, J.L., Robinson, B.F., Castronova, A.M., 2011. Modeling water resource systems using a service-oriented computing paradigm. Environ. Model. Software 26 (5), 573–582.
Granell, C., Schade, S., Ostländer, N., 2013. Seeing the forest through the trees: a review of integrated environmental modelling tools. Computers, Comput. Environ. Urban Syst. 41, 136–150.
Graul, C., Zipf, A., 2008. Biogeographic Research and the Spatial Web: Towards Interoperable Analysis Tools for Global Change Based on the Web Processing Service (WPS). Digital Earth Summit on Geoinformatics. Potsdam, Germany.
Gregersen, J.B., Gijsbers, P.J.A., Westen, S.J.P., 2007. OpenMI: open modelling interface. J. Hydroinf. 9 (3), 175–191.
Guo, H., Wang, Y., Zhang, H., 2017. Characterization of criteria air pollutants in Beijing during 2014–2015. Environ. Res. 154, 334–344.
Han, L., Zhou, W., Li, W., Li, L., 2014. Impact of urbanization level on urban air quality: a case of fine particles (pm2.5) in Chinese cities. Environ. Pollut. 194, 163–170.
Hobley, D., Adams, J.M., Siddhartha Nudurupati, S., Hutton, E.W., Gasparini, N.M., Istanbulluoglu, E., Tucker, G.E., 2017. Creative computing with Landlab: an open-source toolkit for building, coupling, and exploring two-dimensional numerical models of Earth-surface dynamics. Earth Surf. Dyn. 5, 21–46.
Horsburgh S., J., Morsy M., M., Castronova M., A., Goodall L., J., Gan, T., Yi, H., Stealey J., M., Tarboton G., D., 2016. HydroShare: Sharing Diverse Environmental Data Types and Models as Social Objects with Application to the Hydrology Domain. J. Am. Water Resour. Assoc. 52 (4), 873–889. https://doi.org/10.1111/1752-1688.12363.
Hu, X., Belle, J.H., Meng, X., Wildani, A., Waller, L.A., Strickland, M.J., Liu, Y., 2017. Estimating PM2. 5 concentrations in the conterminous United States using the random forest approach. Environ. Sci. Technol. 51 (12), 6936–6944.
Huhns, M.N., Singh, M.P., 2005. Service-oriented computing: key concepts and principles. IEEE Internet Comput 9 (1), 75–81.
Hutton, C., Wagener, T., Freer, J., Han, D., Duffy, C., Arheimer, B., 2016. Most computational hydrology is not reproducible, so is it really science? Water Resour. Res. 52 (10), 7548–7555.
Jiang, P., Elag, M., Kumar, P., Peckham, S.D., Marini, L., Rui, L., 2017. A service-oriented architecture for coupling web service models using the Basic Model Interface (BMI). Environ. Model. Software 92, 107–118.
Jones, N., Nelson, J., Swain, N., Christensen, S., Tarboton, D., Dash, P., 2014. Tethys: a Software Framework for Web-Based Modeling and Decision Support Applications.
Kadlec, J., Ames, D.P., 2017. Using crowdsourced and weather station data to fill cloud gaps in MODIS snow cover datasets. Environ. Model. Software 95, 258–270.
Knox, S., Tomlinson, J., Harou, J.J., Meier, P., Rosenberg, D.E., Lund, J.R., Rheinheimer, D.E., 2019. An open-source data manager for network models. Environ. Model. Software 122, 104538.
Koo, H., Iwanaga, T., Croke, B.F., Jakeman, A.J., Yang, J., Wang, H.H., Sun, X., Lü, G., Li, X., Yue, T., Yuan, W., Liu, X., Chen, M., 2020. Position paper: sensitivity analysis of spatially distributed environmental models-A pragmatic framework for the exploration of uncertainty sources. Environ. Model. Software 104857. https://doi.org/10.1016/j.envsoft.2020.104857.
Li, X., Zhang, X., 2019. Predicting ground-level PM2. 5 concentrations in the Beijing-Tianjin-Hebei region: a hybrid remote sensing and machine learning approach. Environ. Pollut. 249, 735–749.
Li, W., Zhou, S., Wang, X., Xu, Z., Yuan, C., Yu, Y., Zhang, Q., Wang, W., 2011. Integrated evaluation of aerosols from regional brown hazes over northern China in winter: concentrations, sources, transformation, and mixing states. J. Geophys. Res. 116 (D9), D09301.
Li, Y., Gong, J., Zhu, J., Song, Y., Hu, Y., Ye, L., 2013. Spatiotemporal simulation and risk analysis of dam-break flooding based on cellular automata. Int. J. Geogr. Inf. Sci. 27 (10), 2043–2059.
Li, W., Shao, L., Shi, Z., Chen, J., Yang, L., Yuan, Q., Yan, C., Zhang, X., Wang, Y., Sun, J., Zhang, Y., Shen, X., Wang, Z., Wang, W., 2014. Mixing state and hygroscopicity of dust and haze particles before leaving Asian continent. J. Geophys. Res. 119 (2), 1044–1059.
Li, F., Zhang, J., Jiang, W., Liu, C., Zhang, Z., Zhang, C., Zeng, G., 2017. Spatial health risk assessment and hierarchical risk management for mercury in soils from a typical contaminated site, China. Environ. Geochem. Health 39 (4), 923–934.
Li, F., Zhang, J., Liu, C., Xiao, M., Wu, Z., 2018. Distribution, bioavailability and probabilistic integrated ecological risk assessment of heavy metals in sediments from Honghu Lake, China. Process Saf. Environ. Protect. 116, 169–179.
Lin, H., Chen, M., 2015. Managing and sharing geographic knowledge in virtual geographic environments (VGEs). Spatial Sci. 21 (4), 261–263.
Lin, H., Chen, M., Lu, G., 2013b. Virtual geographic environment: a workspace for computer-aided geographic experiments. Ann. Assoc. Am. Geogr. 103 (3), 465–482.

Lin, H., Chen, M., Lu, G., Zhu, Q., Gong, J., You, X., Wen, Y., Xu, B., Hu, M., 2013a. Virtual geographic environments (VGEs): a new generation of geographic analysis tool. Earth Sci. Rev. 126, 74–84.

Lü, G., Batty, M., Strobl, J., Lin, H., Zhu, A.X., Chen, M., 2019. Reflections and speculations on the progress in Geographic Information Systems (GIS): a geographic perspective. Int. J. Geogr. Inf. Sci. 33 (2), 346–367.

Ma, Q., Wu, Y., Zhang, D., Wang, X., Xia, Y., Liu, X., Tian, P., Han, Z., Xia, X., Wang, Y., Zhang, R., 2017. Roles of regional transport and heterogeneous reactions in the PM2. 5 increase during winter haze episodes in Beijing. Sci. Total Environ. 599, 246–253.

Mao, X., Jiang, W., 2009. Comparisons of several evaporation/precipitation datasets for the Bohai Sea based on salinity simulation. J. OCEAN UNIV. 8 (3), 209–214.

Maxwell, T., Costanza, R., 1997. A language for modular spatio-temporal simulation. Ecol. Model. 103 (2–3), 105–113.

McCulley, J.P., Aronowicz, J.D., Uchiyama, E., Shine, W.E., Butovich, I.A., 2006. Correlations in a change in aqueous tear evaporation with a change in relative humidity and the impact. Am. J. Ophthalmol. 141 (4), 758–760.

McGrattan, K.B., Forney, G.P., 2006. Fire Dynamics Simulator (Version 5): User's Guide (NIST Special Publication 1019-5). National Institute of Standards and Technology, US Department of Commerce, Gaithersburg, MD.

Meng, X., Hand, J.L., Schichtel, B.A., Liu, Y., 2018. Space-time trends of PM2. 5 constituents in the conterminous United States estimated by a machine learning approach, 2005–2015. Environ. Int. 121, 1137–1147.

Michaelis, C.D., Ames, D.P., 2009. Evaluation and implementation of the OGC web processing service for use in client-side GIS. GeoInformatica 13 (1), 109–120.

Neitsch, S.L., Arnold, J.G., Kiniry, J.R., Williams, J.R., 2011. Soil and Water Assessment Tool Theoretical Documentation Version 2009. Texas Water Resources Institute.

Nourani, V., Gökçekuş, H., Umar, I.K., 2019. Artificial intelligence based ensemble model for prediction of vehicular traffic noise. Environ. Res. 108852.

Omidipoor, M., Jelokhani-Niaraki, M., Samany, N.N., 2019. A web-based geo-marketing decision support system for land selection: a case study of tehran, Iran. Spatial Sci. 25 (2), 179–193.

Open Geospatial Consortium, Inc, 2007. OpenGIS® web processing service (WPS) specification. http://portal.opengeospatial.org/files/?artifact_id=24151.

Overeem, I., Maureen, M.B., James, P.M.S., 2013. Strategies for integrated modeling: the community surface dynamics modeling system example. Environ. Model. Software 39, 314–321.

Parsons, M.A., 2011. Making data useful for modelers to understand complex Earth systems. Earth Sci. India 4 (4), 197–223.

Peckham, S.D., 2014. The CSDMS Standard Names: Cross-Domain Naming Conventions for Describing Process Models, Data Sets and Their Associated Variables.

Peckham, S.D., Hutton, E.W., Norris, B., 2013. A component-based approach to integrated modeling in the geosciences: the design of CSDMS. Comput. Geosci. 53, 3–12.

Qiao, X., Li, Z., Ames, D.P., Nelson, E.J., Swain, N.R., 2019. Simplifying the deployment of OGC web processing services (WPS) for environmental modelling–Introducing Tethys WPS Server. Environ. Model. Software 115, 38–50.

Raimundo, A.M., Gaspar, A.R., Oliveira, A.V.M., Quintela, D.A., 2014. Wind tunnel measurements and numerical simulations of water evaporation in forced convection airflow. Int. J. Therm. Sci. 86, 28–40.

Rossman, L.A., 2010. Storm Water Management Model User's Manual. National Risk Management Research Laboratory, Office of Research and Development, US Environmental Protection Agency, Cincinnati, p. 276, version 5.0.

Seibert, J., Grabs, T., Köhler, S., Laudon, H., Winterdahl, M., Bishop, K., 2009. Linking soil-and stream-water chemistry based on a riparian flow-concentration integration model. Hydrol. Earth Syst. Sci. 13 (12), 2287–2297.

Serreze, M.C., 2011. Climate change: rethinking the sea-ice tipping point. Nature 471 (7336), 47.

Skamarock, W.C., Klemp, J.B., 2008. A time-split nonhydrostatic atmospheric model for weather research and forecasting applications. J. Comput. Phys. 227 (7), 3465–3485.

Stafoggia, M., Bellander, T., Bucci, S., Davoli, M., De Hoogh, K., De'Donato, F., Gariazzo, C., Lyapustin, A., Michelozzi, P., Renzi, M., Scortichini, M., Shtein, A., Viegi, G., Kloog, I., Schwartz, J., 2019. Estimation of daily PM10 and PM2. 5 concentrations in Italy, 2013–2015, using a spatiotemporal land-use random-forest model. Environ. Int. 124, 170–179.

Sun, A.Y., Zhong, Z., Jeong, H., Yang, Q., 2019. Building complex event processing capability for intelligent environmental monitoring. Environ. Model. Software 116, 1–6.

Swain, N.R., Christensen, S.D., Snow, A.D., Dolder, H., Espinoza-Dávalos, G., Goharian, E., Jones, N.L., Nelson, E.J., Ames, D.P., Burian, S.J., 2016. A new open source platform for lowering the barrier for environmental web app development. Environ. Model. Software 85, 11–26.

Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Heard, J., Ames, D., Goodall, J.L., Band, L., Merwade, V., Couch, A., Jennifer, A., Hooper, R., Valentine, D., Maidment, D., 2014. HydroShare: advancing collaboration through hydrologic data and model sharing.

Tenopir, C., Allard, S., Douglass, K., Aydinoglu, A.U., Wu, L., Read, E., Manof, M., Frame, M., 2011. Data sharing by scientists: practices and perceptions. PloS One 6 (6), e21101.

Thompson, J.R., Sørenson, H.R., Gavin, H., Refsgaard, A., 2004. Application of the coupled MIKE SHE/MIKE 11 modelling system to a lowland wet grassland in southeast England. J. Hydrol. 293 (1–4), 151–179.

Todorova, A., Syrakov, D., Gadjhev, G., Georgiev, G., Ganev, K.G., Prodanova, M., Miloshev, N., Spiridonov, V., Bogatchev, A., Slavov, K., 2010. Grid computing for atmospheric composition studies in Bulgaria. Earth Sci. India 3 (4), 259–282.

Tóth, G., Hermann, T., Da Silva, M.R., Montanarella, L., 2016. Heavy metals in agricultural soils of the European Union with implications for food safety. Environ. Int. 88, 299–309.

Voinov, A., Bousquet, F., 2010. Modelling with stakeholders. Environ. Model. Software 25 (11), 1268–1281.

Voinov, A., Kolagani, N., McCall, M.K., Glynn, P.D., Kragt, M.E., Ostermann, F.O., Pierce, S.A., Ramu, P., 2016. Modelling with stakeholders–next generation. Environ. Model. Software 77, 196–220.

Wang, X., Wei, W., Cheng, S., Li, J., Zhang, H., Lv, Z., 2018. Characteristics and classification of PM2. 5 pollution episodes in Beijing from 2013 to 2015. Sci. Total Environ. 612, 170–179.

Wang, X., Huang, R., Li, L., He, S., Yan, L., Wang, H., Wu, X., Yin, Y., Xing, B., 2019. Arsenic removal from flooded paddy soil with spontaneous hygrophyte markedly attenuates rice grain arsenic. Environ. Int. 133, 105159.

Wen, Y., Chen, M., Yue, S., Zheng, P., Peng, G., Lu, G., 2017. A model-service deployment strategy for collaboratively sharing geoanalysis models in an open web environment. Int. J. Digit. Earth. 10 (4), 405–425.

Xaymurat, A., Huang, M., 2011. Comparative analysis of observed data between E-601 evaporator and 20 cm pan in Tianshan River Basin. Journal of China Hydrology 31 (4), 76–80.

Xue, T., Zheng, Y., Tong, D., Zheng, B., Li, X., Zhu, T., Zhang, Q., 2019. Spatiotemporal continuous estimates of PM2. 5 concentrations in China, 2000–2016: a machine learning method with inputs from satellites, chemical transport model, and ground observations. Environ. Int. 123, 345–357.

Yi, H., Idaszak, R., Stealey, M., Calloway, C., Couch, A.L., Tarboton, D.G., 2018. Advancing distributed data management for the HydroShare hydrologic information system. Environ. Model. Software 102, 233–240.

Yin, Q., Wang, J., Hu, M., Wong, H., 2016. Estimation of daily PM2. 5 concentration and its relationship with meteorological conditions in Beijing. J. Environ. Sci. 48, 161–168.

Yue, S., Wen, Y., Chen, M., Lu, G., Hu, D., Zhang, F., 2015. A data description model for reusing, sharing and integrating geo-analysis models. Environ. Earth. Sci. 74 (10), 7081–7099.

Yue, S., Chen, M., Wen, Y., Lu, G., 2016. Service-oriented model-encapsulation strategy for sharing and integrating heterogeneous geo-analysis models in an open web environment. ISPRS J. Photogrammetry Remote Sens. 114, 258–273.

Yue, S., Chen, M., Yang, C., Shen, C., Zhang, B., Wen, Y., Lü, G., 2019. A loosely integrated data configuration strategy for web-based participatory modeling. GIScience Remote Sens. 56 (5), 670–698.

Yue, S., Chen, M., Song, J., Yuan, W., Chen, T., Lü, G., Shen, C., Ma, Z., Xu, K., Wen, Y., Song, H., 2020. Participatory intercomparison strategy for terrestrial carbon cycle models based on a service-oriented architecture. Future Generat. Comput. Syst. 112, 449–466.

Zhang, G., Zhu, A.X., 2018. The representativeness and spatial bias of volunteered geographic information: a review. Spatial Sci. 24 (3), 151–162.

Zhang, C., Di, L., Sun, Z., Lin, L., Eugene, G.Y., Gaigalas, J., 2019a. Exploring cloud-based web processing service: a case study on the implementation of cmaq as a service. Environ. Model. Software 113, 29–41.

Zhang, Z., Zhang, X., Gong, D., Quan, W., Zhao, X., Ma, Z., Kim, S.J., 2015. Evolution of surface O3 and PM2. 5 concentrations and their relationships with meteorological conditions over the last decade in Beijing. Atmos. Environ. 108, 67–75.

Zhang, F., Chen, M., Ames, D.P., Shen, C., Yue, S., Wen, Y., Lü, G., 2019b. Design and development of a service-oriented wrapper system for sharing and reusing distributed geoanalysis models on the web. Environ. Model. Software 111, 498–509.