Towards Decentralized Integration of Environmental Models on the Web: A Resource Alignment Service

Peishi Jiang[a], Mostafa Elag[a], Praveen Kumar[a,*], Liu Rui[b], Luigi Marini[b]

[a] *Ven Te Chow, Hydrosystem Laboratory, Civil and Environmental Engineering, University of Illinois, Urbana, IL, USA*
[b] *National Center for Supercomputing Applications, University of Illinois, Urbana, IL, USA*

Abstract

...

Keywords: decentralized integration, web service, semantic interoperability, the Resource Alignment Service (RAS)

1. Introduction

Integrated environmental modeling is essential in simulating a complex ecosystem (Argent, 2004). This is because individual scientists or small research groups usually have different practise in modeling in terms of programming language, variable names, units and other factors, causing models from various disciplines hard to be accessed, reused and coupled. Thanks to the advance of web technology, models are able to be exposed as web services (or called as Model as a Service (MaaS)), and integrating these web serviced models over web thus becomes promising (Goodall et al., 2011). The traditional integrated modeling approach often requires models to be encapsulated in a specific framework or standard. Meanwhile, coupling models over the web provides an opportunity that models can be independent of any framework or standard by only exposing their functionalities as web services. In this study, we aim to integrate web serviced models in geoscience by utilizing a Resource Alignment Service (RAS), which is able to ensure the semantic consistency of information flowing between web-serviced resources (i.e., model and data).

Traditionally, scientists usually apply a specific framework or standard for coupling their research-required models to tackle the heterogeneity issue of models. To name a few, Community Surface Dynamics Modeling System (CSDMS) (Peckham et al., 2013), OpenMI standard (Moore & Tindall, 2005) and Earth System Modeling Framework (Hill et al., 2004). Such tight coupling approach controls both the data structures and processes of all parts of the model. For example, in CSDMS community, each model is wrapped by a BMI (Basic Model Interface) wrapper and coupled with other models under a common framework, as shown in Figure 1(a). The OpenMI provdies a standard[1] to convert standalone models into OpenMI-compliant models, and then couples them by using a set of utilities (see Figure 1(b)). Despite of the succesful applications of

*Corresponding author
Email address: kumar1@illinois.edu (Praveen Kumar)
[1] http://www.opengeospatial.org/node/1822

the tight coupling approach, a factor that cannot be ignored is the difficulty of coupling models that do not follow the same standards or frameworks. There is a need, therefore, to decentralize model integration.

The service-oriented, loose-coupling approach gives the potential to integrate models independent of any framework over the web. Being exposed as web services, models can be conserved in their own hardware environments. As shown in Figure 1(c), the decentralized integration of web serviced models is independent of any integration framework and operating system. Thus, modelers only need to focus on the standardization of the web interface and the protocol of data exchanging (Goodall et al., 2013). Web serviced models can "talk" to each other if the semantic consistency of quantities exchanged between them is ensured by a utility or web service (e.g., the Resource Alignment Service used in this study).

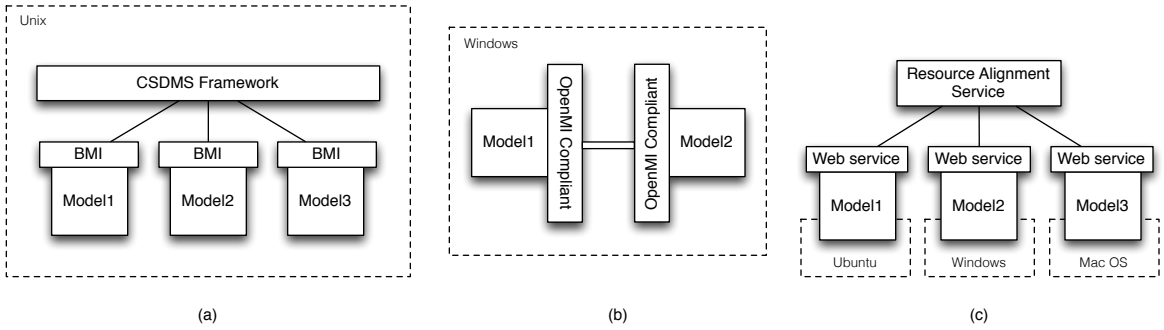

Figure 1: A comparison of the traditional tight integration centralized in a specific framework and the decentralized integration through web services. (a) the model integration in CSDMS framework operated in a Unix machine; (b) the integration of OpenMI-compliant models operated in a Windows machine; (c) the decentralized integration of web serviced models operated in various operating system.

Until recently, the application of the service-oriented architecture in environmental modeling starts (Castronova et al., 2013; Mineter et al., 2003; Granell et al., 2010; Goodall et al., 2011, 2013). Following the concept of Software as a Service (SaaS), models and data can also be exposed as web services, called as Model as a Service (MaaS) and Data as a Service (DaaS) respectively. Goodall et al. (2011) exposed water resource models as web services by using the Open Geospatial Consortium (OGC) Web Processing Service (WPS) and demonstrated how it could be encapsulated as OpenMI-compliant models. Then, Castronova et al. (2013) furthered the idea of servicing an OpenMI-compliant model based on WPS by considering the case of time-dependent models. In addition, an OpenMI-ESMF web service wrapper was developed to couple a climate model implemented via ESMF web service with an OpenMI-compliant hydrologic model running on a personal computer (Goodall et al., 2013). Furthermore, the concept of "Model Web" was put forward to draw the picture of a world of interoperable MaaS and DaaS (Geller & Turner, 2007; Geller & Melton, 2008). Nativi et al. (2013) then adopted the idea of "Model Web" in the Group on Earth Observation (GEO) Model Web initiative. Nevertheless, none of the existing efforts in coupling serviced models, either the application of OpenMI standard in integrating web serviced models or the proposed concept of "Model Web", has solved the semantic interoperability issue entirely independent of a modeling integration framework.

Therefore, in this paper, we develop a web service named Resource Alignment Service (RAS) to ensure semantic consistency of information exchanged between the decentralized web serviced models over the web. The capability of RAS includes (1) semantic mediation between variable names; (2) conversion of mismatched units; (3) temporal alignment of resource time horizon and (4) spatial alignment of resources spatial attributes. Models are implemented by the WPS, which provides an interface standard for exposing both the model execution and the obtain of a model's input(s) and output(s) as web services. Then, RAS is utilized to align the quantities exchanged between the WPS-implemented serviced models.

In the remainder of the paper, Section 2 describes the design requirement of the decentralized integration of web serviced models. In Section 3, the architecture of the Resource Alignment Service (RAS) is presented, including the an brief overview of GeoSemantic framework, the functionality and the components of RAS. Section 4 demonstrates the application of RAS in coupling WPS-implemented models with the input data from Clowder, an online data repository. First we create a workflow of heterogeneous collection of data and WPS-implemented models. Then, we show how RAS can seamlessly align the semantics of quantities exchanging between these resources. Finally, a brief summary is given in Section 5, with mentions on the future work.

2. Design Requirement and Overview

A service-oriented architecture is adopted for the decentralized integration of web serviced models. Hence, it is crucial to standardize the communication protocols of web services. In this paper, the OGC Web Processing Service (WPS) specification is implemented for setting up the interface of the serviced models.

2.1. Background

Service-oriented computing is the application of service-oriented architecture in computation, where models or computing components are distributed on various remote server and provide services to other clients via a communication protocol (Huhns & Singh, 2005; Erl, 2004). The client can be either a user or other web service consuming the functionality exposed over the internet. The service-oriented computing approach implies a loose coupling manner. Following a specific communication protocol, the client makes a request to the web service operating on a remote server over the web. Then, after receiving the request, the service interpretes the incoming information, carries out a certain processing and sends a response back to the client for further application. Based on such architecture, the web serviced models can be conserved in any hardware environment (e.g., operating system) and set up in any programming language. It is also convenient for modelers to maintain and update their models at the backend, without the entire web-based architecture being disturbed. Only the compliance of using a common web service standard and commnication protocol is required for the commnication between client and service successfully.

Written in a certain language (e.g., eXtensive Markup Language (XML) or JavaScript Object Notation (JSON)), the request data needs to follow a specific web service standard. Two common web service standards are the REpresentational State Transfer (REST) specifications and the Simple Object Access Protocol

(SOAP). SOAP has been widely applied for setting up web service, usually combined with the Web Service Description Language (WSDL). However, a web service in SOAP/WSDL format is much more complex than a RESTful web service application (Mulligan & Gračanin, 2009). This is because SOAP is designed for strutured information and therefore requires the incoming data to follow a certain prototype. Meanwhile, a RESTful application only relies on a stateless communication protocol (e.g., the Hypertext Transfer Protocol (HTTP)). By using a set of HTTP methods (GET, POST, PUT, DELETE, HEAD, Fielding et al. (1999)), a RESTful application is able to interact with the resources exposed by a web service. In this study, we apply the RESTful application to design our servce-oriented computing for its simplicity and easy-to-use. Nevertheless, due to their general service application, neither SOAP or REST is designed for a specific application (e.g., geoscience). It is also necessary, therefore, to apply a web service standard following REST that is able to define how data is transfered over the web.

In geoscience, much work has been done for the standardization of data transfered between web service. For example, Water Markup Language (WaterML) and the Open Geospatial Consortium (OGC) standards are used for transmitting hydrologic time series data and geospatial data respectively. WaterML is initially applied as the protocol of hydrologic data transmission in WaterOneFlow web services held by Consortium of Universities for the Advancement of Hydrologic Scienc, Inc - Hydrologic Information System (CUAHSI-HIS, Valentine et al. (2007)). Now the second version of WaterML is adopted as one of the OGC standards. Furthermore, various types of standards are published by the OGC, which includes but not limited to: (1) specifications for data format such as geography markup language (GML)and keyhole markup language (KML); (2) specifications for data publication such as web coverage service (WCS), web mapping service (WMS) and web feature service (WFS); and (3) specifiations for geoprocessing such as web processing service (WPS).Swain et al. (2015) provides a good review of the current open source solutions for water resources web application, most of which apply various OGC standards. In this study, as a geoprocessing standard, OGC WPS is implemented by the web serviced model, which has proven to be successfully applied in other modeling studies requiring data exchanging between services (Castronova et al., 2013; Goodall et al., 2011; Schaeffer, 2008; Vitolo et al., 2012).

2.2. Serving models using the OGC WPS specification

The OGC WPS provides an approach for standardizing data processing, including three methods: GetCapabilities, DescribeProcess and Execute. The GetCapabilities method allows clients to receive a list of existing processes or models from the remote server implementing the WPS. The DescribeProcess method replies clients with the metadata of the inputs and outputs of a specific model or process. Finally, the execution of a model or process can be conducted through the Execute method based on the input values from clients. All the three methods can be requested by either HTTP POST or HTTP GET method. In this study, models are implemented by WPS, and we utilized the DescribeProcess and Execute method for retrieving the metadata specifics of the models and the model execution respectively.

PyWPS[2], a Pythoon package for implementing WPS, is utilized for servicing models. The advantage is that PyWPS provides a module for modeler to apply WPS standard directly on any model. Set up on a specific web server (e.g., Apache[3] and Nginx[4]), PyWPS is able to expose models as web services applying WPS specification.

3. RAS Architecture

The Resource Alignment Service (RAS) is a product of GeoSemantic framework (Elag et al., 2015), a project aiming to resolve the semantic interoperability issue of various resources based on semantic web technology (Berners-Lee et al., 2001). In this section, we first briefly introduce the GeoSemantic framework and how the RAS works under this framework. Then, the workflow of the RAS is presented, including the trigger (or the request inputs), utilizing information from GeoSemantic framework, passing information through its inherent aligning functions and returning back the alignment result. The components of the RAS are detailed at last.

3.1. GeoSemantic Framework

GeoSemantic framework is an ongoing project whose goal is to allow the structured information seamlessly exchanging between various resources by achieving their semantic interoperability (Elag et al., 2015). Through semantic web technology, GeoSemantic enhances the capability of finding resources by searching a standard names grpah, discovering the implicit relationships between resources by reasoning, and annotating a new or an existing resource by adding semantic tags. A general architecture of it is presented in Figure 2, with emphasis on RAS's functionality.

As emphasized in Figure 2, RAS ensures the semantic consistency of information flowing between various web serviced resources. For example, align data stored in Clowder[5] (an online resource collections developed under GeoSemantic framework) to the inputs of a web serviced model. In this study, we focus on the utilization of RAS in coupling the WPS-implemented models to help decentralized integration of web serviced models.

3.2. RAS Workflow

RAS is developed as a service being used for ensuring the semantic consistency of information exchanging between online resources. The entire functionality of it includes semantic mediation of variable names, conversion of mismatched units and file type, and tempo-spatial alignment. A Python web application package, Flask[6], is utilized to set up the service for RAS.

[2]http://pywps.wald.intevation.org/
[3]http://httpd.apache.org/
[4]http://nginx.org/
[5]https://clowder.ncsa.illinois.edu/clowder/
[6]http://flask.pocoo.org/

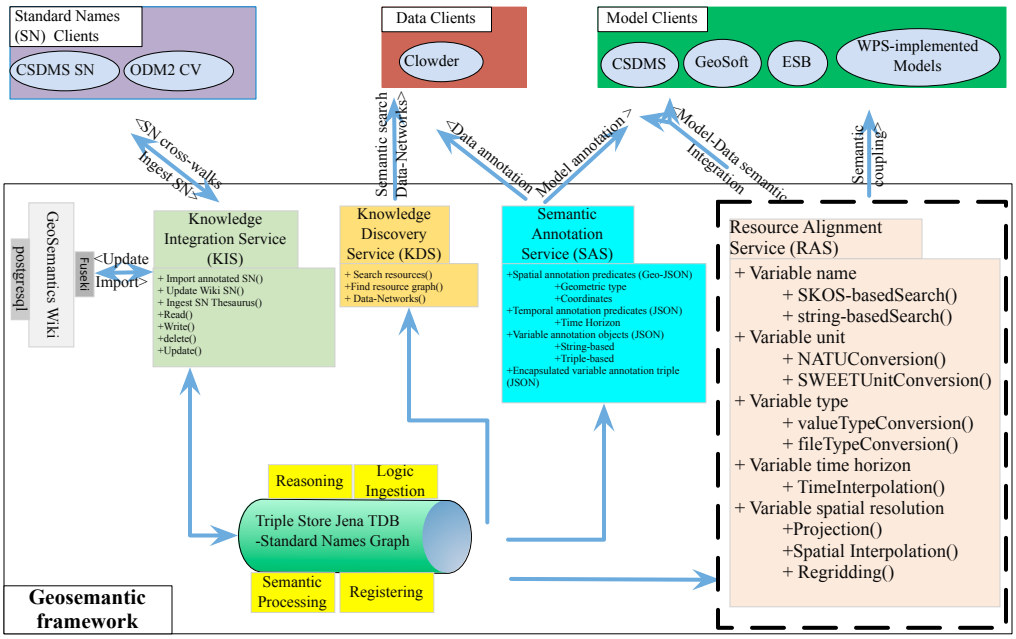

Figure 2: The architecture of GeoSemantic framework. Four web services are developed to allow users to import or ingest the standard names into the triple store, discover the inherent relationships between the resources from the triple store, annotate metadata of resources semantically and align two heterogeneous resources.

A basic workflow of RAS is presented in Figure 3. Through HTTP POST method, RAS takes the metadata of two lists of variables, the formats of the two lists, and the values of the variable list to be aligned as posted data. The response of RAS is also in JSON format with two fields: *aligned* and *unaligned*. *aligned* field includes all the variables being aligned successfully, with the aligned values. Similarly, the rest variables which fail in the alignment are reported in *unaligned*, with their failure reasons.

During the alignment process, RAS first parses the incoming metadata of the two lists according to the provided list formats. The current supported list formats are the XML-based response from the WPS's DescribeProcess call and a JSON-based variable template designed for the metadata of random data or varaible from non WPS-implemented models. The JSON-based variable template will be described in the next section. Then, the two variable lists are resorted according to the criteria that whether a variable in one list is able to be "matched" with a variable in another list through semantic mediation in the standard name graph. The "matched" pairs of variables are used for further alignment, while the "unmatched" variables are classified as the *unaligned* in RAS's response. The semantic consistency of each pair of the "matched" variables is checked in terms of the units, the value type, the temporal and spatial property and the file type. The aligning will be conducted if there is any inconsistency based on the given variabl values. If the alignment of one pair of variable fail (e.g., conversion from one file type to another is nor supported), then the variables are grouped in the *unaligned* with the reason reported.

A simple applicaiton situation is, for example, that the outputs of a WPS-implemented model A need to

be used as the inputs of another WPS-implemented model B. The metadata of the two models can be obtained through the WPS DescribeProcess request, say *metadata_A* and *metadata_B* respectively. The information, which is posted to RAS's service, then includes the following, as shown in Figure 4:

- the metadata of model A: *metadata_A*
- the format of *metadata_A*: "wps output"
- the metadata of model B: *metadata_B*
- the format of *metadata_B*: "wps input"
- the values or URIs of model A's outputs

After the semantic consistency check and the alignment process, RAS will response back to the client with *aligned* and *unaligned* information in JSON format.

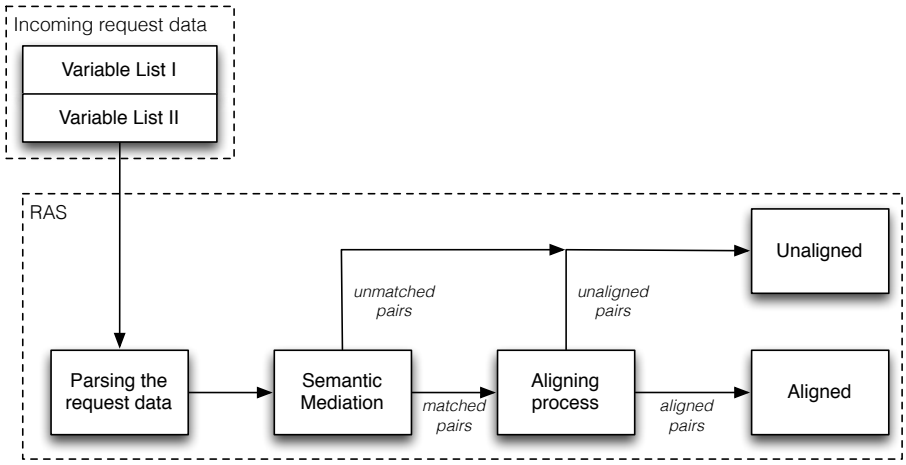

Figure 3: The workflow of RAS.

3.3. RAS Components

Figure 5 presents the RAS's components used for the alignment between two lists of variables in terms of variable name, variable unit, variable value type, file type, variable temporal and spatial properties. The alignment is carried out for each pair of variables, say, aligning from variable I to variable II.

variable name. The semantic mediation of two variable names is conducted through either a SKOS-based search or a string-based search. SKOS-based search is applied for variable names in a form of URI. The procedures are (1) finding all the matched (i.e., *skos:exactMatch* or *skos:closeMatch*) variable names for variable I by querying the standard names graph by using SPARQL[7], an Resource Description Framework

[7]http://www.w3.org/TR/rdf-sparql-query/

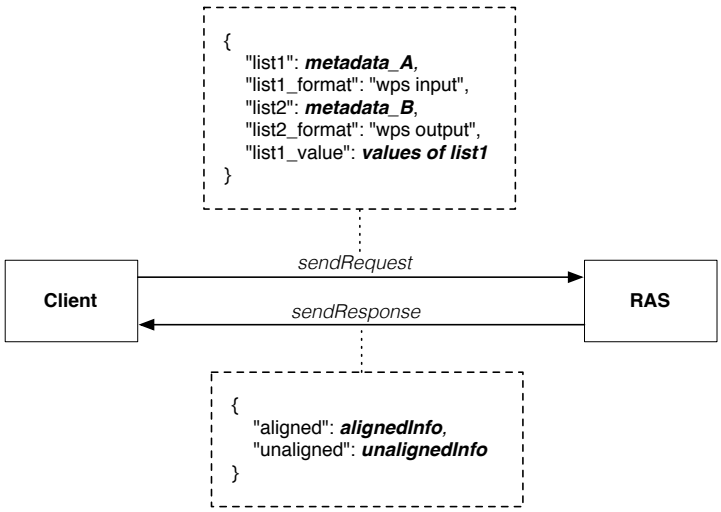

Figure 4: An example of the request and the response information of RAS for aligning between the outputs of WPS-implemented model A and the inputs of WPS-implemented model B. (Note: the bold texts are implicit information.)

(RDF) query language; (2) checking whether variable II is in the list of the matched variables of variable I (return matched if true). *rdflib*[8], a Python library for working with RDF, is utilized for RDF querying. In addition, if the types of variable names are string, then the mediation is carried out based on string-based search. The variable name in string is first transformed in a URI format by assigning it with a model name space, and then SKOS-based search is used for semantic mediation.

variable unit. Similar to the alignment of two variable names, the conversion of two variable units is based on UDUNITS conversion and SWEET unit conversion, according to the types of units: string and URI, respectively. If the unit types are string, UDUNITS[9] (a package support converting units of physical quantities) is employed to perform the conversion. On the other hand, a unit ontology of Semantic Web for Earth and Environmental Terminology (SWEET) ontologies is applied in the case that both the unit formats are URI.

variable type. The alignment of the variable types of two variables is based on both the value and file types conversion. The value type alignment is straightforward through type conversion in Python (e.g., from *float* to *integer*). In terms of conversion in file types, currently the supported file types are comma separated value (CSV) files and Network Common Data Form (NetCDF[10]) files, with the header information in the file if any.

variable time horizon. The temporal property alignment of two variables is conducted based on linear temporal interpolation, given the start time, the end time and the time step of each variable. Through temporal

[8]https://github.com/RDFLib/rdflib
[9]http://www.unidata.ucar.edu/software/udunits/
[10]http://www.unidata.ucar.edu/software/netcdf/

interpolation approach, the actual time period of the data of variable II aligned from that of variable I is the intersection of the given time periods of the two variables. It means if the time periods (from the start time to the end time) of the two variables do not overlap, there is no data interpolated for variable II.

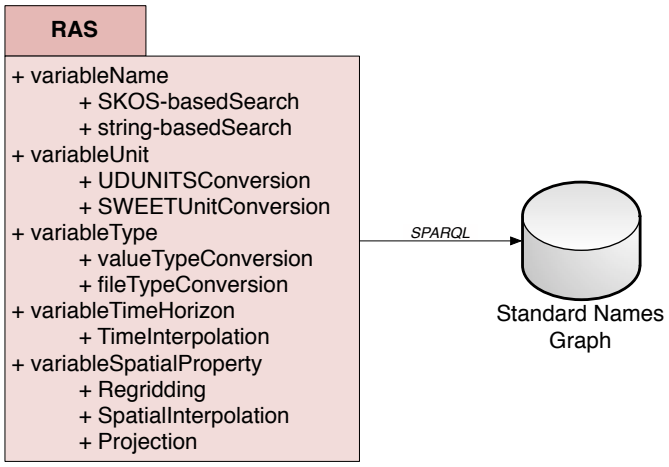

Figure 5: The components of the Resource Alignment Service (RAS).

References

Argent, R. M. (2004). An overview of model integration for environmental applicationscomponents, frameworks and semantics. *Environmental Modelling & Software*, *19*, 219–234.

Berners-Lee, T., Hendler, J., Lassila, O. et al. (2001). The semantic web. *Scientific american*, *284*, 28–37.

Castronova, A. M., Goodall, J. L., & Elag, M. M. (2013). Models as web services using the open geospatial consortium (ogc) web processing service (wps) standard. *Environmental Modelling & Software*, *41*, 72–83.

Elag, M., Kumar, P., Marini, L., & Peckham, S. (2015). Semantic interoperability of long-tail 426 geoscience resources over the web, .

Erl, T. (2004). *Service-oriented architecture: a field guide to integrating XML and web services*. Prentice Hall PTR.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999). *Hypertext transfer protocol–HTTP/1.1*. Technical Report.

Geller, G. N., & Melton, F. (2008). Looking forward: Applying an ecological model web to assess impacts of climate change. *Biodiversity*, *9*, 79–83.

Geller, G. N., & Turner, W. (2007). The model web: a concept for ecological forecasting. In *Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007. IEEE International* (pp. 2469–2472). IEEE.

Goodall, J. L., Robinson, B. F., & Castronova, A. M. (2011). Modeling water resource systems using a service-oriented computing paradigm. *Environmental Modelling & Software*, *26*, 573–582.

Goodall, J. L., Saint, K. D., Ercan, M. B., Briley, L. J., Murphy, S., You, H., DeLuca, C., & Rood, R. B. (2013). Coupling climate and hydrological models: Interoperability through web services. *Environmental Modelling & Software*, *46*, 250–259.

Granell, C., Díaz, L., & Gould, M. (2010). Service-oriented applications for environmental models: Reusable geospatial services. *Environmental Modelling & Software*, *25*, 182–198.

Hill, C., DeLuca, C., Suarez, M., Da Silva, A. et al. (2004). The architecture of the earth system modeling framework. *Computing in Science & Engineering*, *6*, 18–28.

Huhns, M. N., & Singh, M. P. (2005). Service-oriented computing: Key concepts and principles. *Internet Computing, IEEE*, *9*, 75–81.

Mineter, M. J., Jarvis, C., & Dowers, S. (2003). From stand-alone programs towards grid-aware services and components: a case study in agricultural modelling with interpolated climate data. *Environmental Modelling & Software*, *18*, 379–391.

Moore, R. V., & Tindall, C. I. (2005). An overview of the open modelling interface and environment (the openmi). *Environmental Science & Policy*, *8*, 279–286.

Mulligan, G., & Gračanin, D. (2009). A comparison of soap and rest implementations of a service based interaction independence middleware framework. In *Simulation Conference (WSC), Proceedings of the 2009 Winter* (pp. 1423–1432). IEEE.

Nativi, S., Mazzetti, P., & Geller, G. N. (2013). Environmental model access and interoperability: The geo model web initiative. *Environmental Modelling & Software*, *39*, 214–228.

Peckham, S. D., Hutton, E. W., & Norris, B. (2013). A component-based approach to integrated modeling in the geosciences: The design of csdms. *Computers & Geosciences*, *53*, 3–12.

Schaeffer, B. (2008). Towards a transactional web processing service. *Proceedings of the GI-Days, Münster*, .

Swain, N. R., Latu, K., Christensen, S. D., Jones, N. L., Nelson, E. J., Ames, D. P., & Williams, G. P. (2015). A review of open source software solutions for developing water resources web applications. *Environmental Modelling & Software*, *67*, 108–117.

Valentine, D., Zaslavsky, I., Whitenack, T., & Maidment, D. R. (2007). Design and implementation of cuahsi waterml and wateroneflow web services. In *Proceedings of the Geoinformatics 2007 Conference, San Diego, California* (pp. 5–3).

Vitolo, C., Buytaert, W., & Reusser, D. (2012). Hydrological models as web services: An implementation using ogc standards. In *Proceedings of the 10th International Conference on Hydroinformatics, HIC, Hamburg, Germany*.