Environmental Modelling & Software 69 (2015) 128–140

Contents lists available at ScienceDirect

Environmental Modelling & Software

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

A geoprocessing workflow system for environmental monitoring and integrated modelling

Peng Yue*, Mingda Zhang, Zhenyu Tan

State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, 129 Luoyu Road, Wuhan, 430079, China

ARTICLE INFO

Article history:
Received 22 August 2014
Received in revised form
23 February 2015
Accepted 20 March 2015
Available online 7 April 2015

Keywords:
Geoprocessing workflow
Geospatial services
Integrated modelling
Event
OpenMI
GIS

ABSTRACT

Environmental information infrastructure benefits from mainstream information technologies including workflow and service technologies. These technologies allow distributed geoprocessing algorithms, models, data, and sensors to be accessed through Web Services, which later can be chained together to support environmental monitoring and integrated modelling. Existing approaches on integrated environmental modelling, such as OpenMI, have advantages in enabling interoperability between modelling components. It is possible to integrate both of them to take the best from both approaches. The paper introduces the design and implementation of a geoprocessing workflow tool, named GeoJModelBuilder, which is able to integrate interoperable Sensor Web, geoprocessing services, and OpenMI-compliant model components into workflows. In this way, sensors, data, geoprocessing functions, and models could be integrated in a flexible, reusable, interoperable, and user-friendly way. The system has been published as an open source software and illustrated in cases on environmental monitoring and integrated modelling.

1. Introduction

Integrated environmental modelling (IEM) has been identified as an important discipline to provide approaches and tools to understand the social–economic–environmental systems (Rizzoli and Davis, 1999; Parker et al., 2002; Jakeman and Letcher, 2003; Laniak et al., 2013a). It provides a structural way to develop and organize multidisciplinary knowledge through models. The development of IEM covers issues on applications, science, technology, and community (Laniak et al., 2013b). In the past decades, significant efforts have been devoted towards the technological development of IEM systems (Granell et al., 2013a). With the advancement of Spatial Data Infrastructure (SDI) or Cyberinfrastructure, "Model as a Service" (MaaS) approach is being used in the Model Web initiative to provide an engineering approach towards the implementation of integrated modelling systems layered on environmental information infrastructure (Geller and Turner, 2007; Nativi et al., 2013). In this way, different models could be implemented as services, often accessible through Web Service interfaces, and coupled through service-oriented workflows (Granell et al., 2010; Bastin et al., 2013; Nativi et al., 2013).

Workflows and Web Services are widely employed in environmental information infrastructures (Béjar et al., 2014; Laniak et al., 2013b). These technologies allow distributed models, data, and sensors to be accessed through Web Services, which later can be chained together to support environmental monitoring and integrated modelling. A vision on integrated environmental modelling has shown that interoperability across modelling components and leverage of Web Services for the next generation modelling framework are critical technological drivers (Laniak et al., 2013b). In the geospatial Web Services area, the Open Geospatial Consortium (OGC) is the major organization working on developing interoperable geospatial Web Services standards by adapting or extending the common Web Service standards (OGC, 2015). Examples of OGC Web Service standards include Web Feature Service (WFS), Web Map Service (WMS), Web Coverage Service (WCS), Sensor Observation Service (SOS), Catalogue Services for Web (CSW), and Web Processing Service (WPS). These standards could be used in the software design and implementation of IEM systems (Laniak et al., 2013b). For example, an environmental model could be accessed by WPS (Nativi et al., 2013). Although the geospatial community tends to treat models as any algorithm accessible through the WPS interface, there are still some challenges on adopting WPS for accessing environmental models. The WPS specification has some limitations in dealing with complex semantics and time-step computations of models. However, some proposals have been

* Corresponding author.
E-mail address: pyue@whu.edu.cn (P. Yue).

http://dx.doi.org/10.1016/j.envsoft.2015.03.017
1364-8152/

available to address these challenges, such as adding formalized models as the payload in a WPS 2.0 extension (Nativi et al., 2013), profiling of the WPS for models (Bastin et al., 2013), and maintaining state on the WPS server (Castronova et al., 2013a). Therefore, the workflow tool in this paper still adopts the WPS to support its promise for MaaS. In cases where WPS is not adequate for providing complex environmental models, the tool also accommodates the OpenMI, a standard for describing modelling components and runtime data exchange between them (Vanecek and Moore, 2014). Thus the tool will provide a workflow environment to bridge models in the geospatial and environmental communities.

The paper introduces the design and implementation of a geoprocessing workflow tool, named GeoJModelBuilder, for environmental monitoring and integrated modelling. Here the term geoprocessing has a broader meaning than the analysis function in traditional Geographical Information Systems (GIS), and can refer to any sort of geospatial processing, analysis functions, or models. For example, it could include Earth system prediction models or a simple spatial buffer analysis function. Both traditional geospatial analysis algorithms and environmental models are integrated using geoprocessing workflows. Compared to existing scientific workflow tools, the tool is geo-enabled in that it integrates interoperable OGC Sensor Web, geoprocessing services, and OpenMI-compliant model components into workflows. In this way, it addresses the two key topics on the technological development of IEM systems, interoperability and Web, envisioned by Laniak et al. (2013b). The event-driven technologies adopted by Sensor Web are leveraged with geoprocessing workflow to support environmental monitoring. In contrast to "passive" workflow enactment in traditional workflow engines, the advocate of the new "Event" node in the workflow can support "active" environmental monitoring, thus enriching the service-oriented paradigm with events for environmental model software architectures. The system is coupled with virtual globes to allow users to interact with IEM in a user-friendly graphical user interface (GUI). Thus the tool allows models, sensors, computational and data resources distributed on the Web to be integrated in a flexible, reusable, interoperable, and user-friendly way. The result has been published as open source software and illustrated in cases on environmental monitoring and integrated modelling.

The remainder of the article is organized as follows. Section 2 introduces related work in the literature. Two motivating examples are described in Section 3. Section 4 presents the architectural design of GeoJModelBuilder. Details of software implementation are described in Section 5. Section 6 provides the evaluation. Conclusions and pointers to future work are given in Section 7.

2. Related work

Conventional environmental modelling systems include GIS software (Laniak et al., 2013b). Since 2004, the Environmental Systems Research Institute (ESRI) has released a geoprocessing framework in ArcGIS software. The framework includes not only a set of analysis functions but also a geoprocessing ModelBuilder tool (ESRI, 2015). The tool allows users visually combine different analysis model components and generate new geoprocessing models. The spatial modelling tool in the ERDAS IMAGINE software, named Spatial Modeler (Intergraph, 2015), has a similar model chaining function. It can combine GIS analysis components with ERDAS's commercial image processing components to support integrated modelling. These proprietary GIS software systems have limited capabilities in supporting open integrated modelling in the Web environment. The analysis components and the modelling capability are often used in their own proprietary environments. There are also some open source scientific workflow tools, such as Kepler (Ludäscher et al., 2006), Taverna (Oinn et al., 2004), and Vistrails (Callahan et al., 2006), which have been used for IEM (Bastin et al., 2013; Granell et al., 2013a; van Zyl et al., 2012). Nevertheless, these workflow tools are originally designed to be general or for non-geo domains. For example, Kepler is intended for workflow applications across different domains, Taverna focuses on bioinformatics activities, and VisTrails initially addresses data visualization. Significant efforts have to be devoted to adapt them to distributed geoprocessing service environment (Pratt et al., 2010; de Jesus et al., 2012; McFerren et al., 2012). GeoJModelBuilder is designed in nature for geo-enabled workflows by providing built-in support for open standard-compliant geospatial Web Services and data. In addition, it leverages OGC Sensor Web standards and virtual globes for event-driven workflow enactment in a user-friendly environment, which are not done yet in existing IEM tools.

Component-based modelling has been widely used in existing IEM tools (Granell et al., 2013a). There are already various modelling frameworks available, such as Common Component Architecture (CCA), Earth System Modelling Framework (ESMF), and Open Modelling Interface (OpenMI). Among them, the OpenMI standard aims to be a global framework for coupling diverse models, and has been subjected to extensive studies recently (Laniak et al., 2013b). From the perspective of component-based software engineering (Sommerville, 2004), models (a given functionality, executables, or algorithmic code) could be treated as software components working together using workflows (Bastin et al., 2013; Granell et al., 2013b). Although there are some arguments that model integration is more than software coupling (e.g., data sets should be treated as components on the same level as models) (Voinov and Shugart, 2013), the software module coupling tools such as workflow tools do show promise for right solutions (Voinov and Shugart, 2013; Laniak et al., 2013b; Granell et al., 2013a). From the "Model as a Service" perspective, models could be accessed through Web Services (Nativi et al., 2013). The advancement of service-oriented computing and Web-Service-based implementation follow the component-based software engineering principles by realizing services as reusable, loosely coupled, and independent components. Thus, the vision of Model Web could be implemented by treating models as services and workflows as composition of Web Services (Granell, 2014).

OGC Web Services, W3C SOAP-based Web Services, and RESTful services are available for implementation of services. Some efforts have been devoted to make them work together, such as defining WSDL for OGC services (Sonnet, 2004), and using WSDL 2.0 as the bridge between REST (REpresentational State Transfer) and W3C Web Service (W3C, 2007; Lucchi et al., 2008). Other efforts investigate approaches on service chaining, or service composition, for example, the use of the Web Services Business Process Execution Language (WSBPEL, BPEL for short) (OASIS, 2007) to support geospatial service chains (Friis-Christensen et al., 2009; Yu et al., 2012). Three types of service chaining are defined in the OGC Abstract Service architecture (Percivall, 2002) depending on the level of user control: user-defined (transparent), workflow-managed (translucent), and aggregate (opaque). In transparent chaining, the human user is responsible for invoking service components and controlling the chains such as passing around processing results. Translucent chaining allows a user to define a service chain using a workflow language such as BPEL. The execution of service chains is managed by workflow engines. In opaque chaining, the human user invokes a service that carries out the chain. The user has no awareness of the individual services in the chain. This type of chaining is often addressed by automatic service composition (Yue et al., 2012).

The OGC WPS specification provides an approach to make traditional analysis functions accessible through standard

interfaces on the Web (Lanig et al., 2008). Although the WPS proposal was proved to be workable and suitable for many geoprocessing functionalities (Michaelis and Ames, 2009), there are still several opportunities for enhancement. The interaction with WPS is independent on previous interactions (stateless) and does not support asynchronization. The environmental models, however, often require runtime interaction, which needs to maintain state. Asynchronous geospatial processing with the WPS is needed for long-time taking geoprocessing (Westerholt and Resch, 2014). REST is also gaining popularity to alleviate the complexity of WPS specification (Granell et al., 2014; Castronova et al., 2013a). The OGC is working on WPS Version 2.0 interface standard, in which asynchronization and REST will be supported (OGC, 2014). However, this movement has to be careful, since an investigation on the complexity of OWS schemas shows that some OWS schemas are too complex, making the implementation or system upgrade to a higher version of schemas an arduous task (Tamayo et al., 2012). The OGC Sensor Web Enablement (SWE) initiative started in 2003, and proposed a set of service specifications to support interoperable usage of sensor resources including discovery, access, as well as tasking (Botts et al., 2007; Yue et al., 2014a). The first generation of SWE specifications between 2006 and 2007 include Observations and Measurements (O&M), Transducer Model Language (TML), Sensor Model Language (SensorML), Sensor Observation Service (SOS), Sensor Planning Service (SPS), Sensor Alert Service (SAS), and Web Notification Service (WNS). Development to the next generation of SWE is going on (Bröring et al., 2011). It adds the eventing mechanism by providing Event Pattern Markup Language (EML) and evolving SAS to Sensor Event Service (SES) (Echterhoff and Everding, 2011). While SWE has been used in a wide variety of projects, it is too generic and domain specific profiles have to be developed to enhance interoperability within domains (Bröring et al., 2011).

Recent studies on workflow-based chaining suggest the interplay of Sensor Web and geoprocessing services. Previous approaches are based on traditional Service-Oriented Architecture (SOA), which mainly applies to the request–response mode. In other words, these approaches follow the "passive" mode of workflow enactment. They do not meet the requirements of environmental observation and monitoring in response to events, which are necessary for "active" enactment of workflows. By following the Event Driven Architecture (EDA) (Michelson, 2006), OGC SES lets clients subscribe for observations, and performs filtering of sensor data based on the filter criteria defined in subscriptions (Echterhoff and Everding, 2008). Once a match is discovered, a notification is sent to the subscriber. EDA can be used for timely response on various events and for coordination with business process integration in ubiquitous enterprises (Michelson, 2006). Yu et al. (2008) used an event-driven method in geospatial workflows. An event-driven mechanism was used to achieve the feed-forward cycle in self-adaptive Earth predictive systems (Yu et al., 2010). GeoJModelBuilder defines a new type of workflow entities, named "Event", which can be dragged and dropped in the workflow designer to support "active" environmental monitoring.

3. Motivating examples

Two environmental applications are used as examples to help understand the workflow approach. The first case is relatively simple, assumes that models/algorithms are accessible through Web services, and follows the MaaS approach. It will be used to demonstrate the event driven "active" environmental monitoring. The second case involves both environmental models and spatial analysis algorithms. The technical solution to this case will adopt OpenMI due to current limitations of WPS. Users will have the same experience as the first case in the GUI, since the tool hides the difference in the underlying technical implementation.

3.1. Use case 1

Suppose Mr. Wang is an employee of the National Disaster Reduction Center (NDRC) in China. He is responsible for monitoring the water quality of East Lake in Wuhan City, Hubei Province. He has placed a number of water quality sensors in East Lake, which can continuously collect chlorophyll concentration and water turbidity data. When detecting anomaly (e.g., Water turbidity exceeds the threshold), Mr. Wang will collect suitable remote sensing observation covering East Lake, process the data based on a turbidity extraction processing flow (Fig. 1). The observation imagery will go through orthorectification, radiometric calibration, atmospheric correction, clipping, Normal Differential Water Index (NDWI) calculation, mask building, silt inversion, and mapping

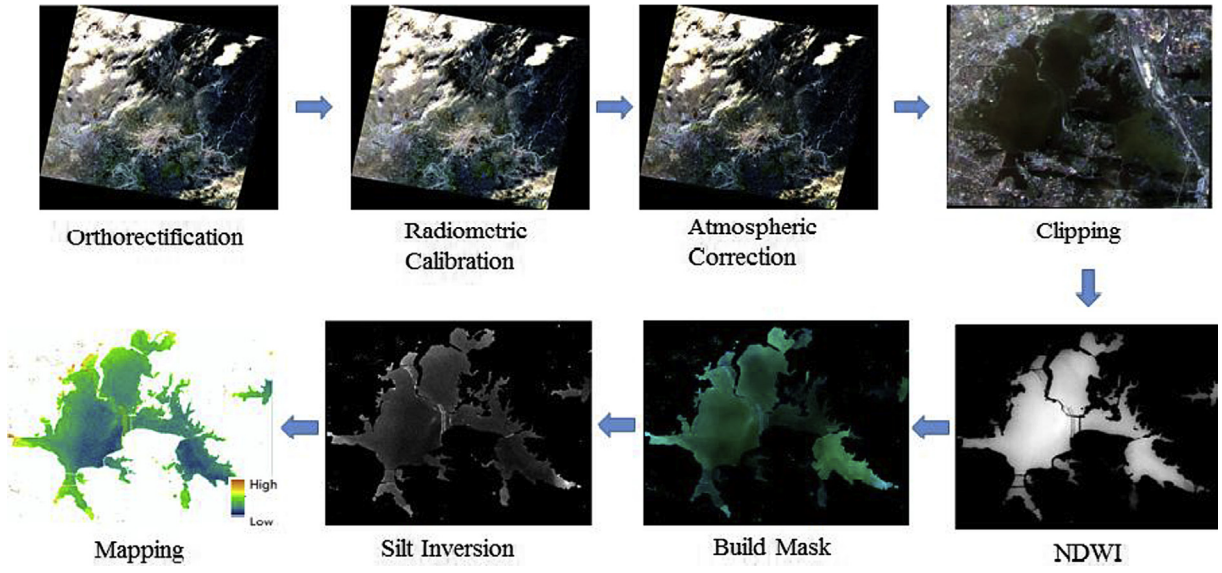

Fig. 1. A scientific workflow for turbidity extraction from remote sensing imagery.

processes to provide decision support on whether pollution happens or the range of pollution. Traditionally, experts have to check in-situ observation data regularly, and if anomaly is detected, find remote sensing imagery and process the data step by step using an image processing software. Using the service and workflow technologies, it is then possible to integrate data and computational resources in a distributed information infrastructure. These resources are accessible through interoperable services, enabling flexible composition of them to meet different modelling requirements. The composition results could be archived to generate turbidity maps on demand. With the standard Sensor Web technologies, in-situ observation data is sent to the data center, accessed through the SOS interface, and checked automatically against the threshold. Once an event is detected, the turbidity extraction processes could be triggered automatically. Thus the workflow approach improves the automation of the environment monitoring. In case users want to adjust sensor tasking or input parameters, and select alternative services, the tool also allows the user interaction with workflows. Thus the tool supports both transparent and translucent chaining.

3.2. Use case 2

The second example is used to illustrate the integration of Web resources and environmental models, assuming data and geospatial analysis algorithms are accessible through services in environmental information infrastructures. The environmental models includes TOPography based hydrological MODEL (TOPMODEL) and Hargreaves model, which are taken from an existing case in HydroModeler (Castronova et al., 2013b). The TOPMODEL is a hydrologic model that can be used to predict watershed runoff (Beven and Kirkby, 1979). It takes topographic index, precipitation, and evapotranspiration as inputs (Fig. 2). The topographic index can be derived from DEM (Digital Elevation Model) using spatial analysis functions, such as *r.topidx* offered by GRASS (Geographic Resources Analysis Support System) (GRASS, 2015), which can be exposed as a WPS (Yue et al., 2010). The DEM data in the watershed area could be extracted by a polygon from a DEM file covering a wide area. Evapotranspiration can be calculated using the Hargreaves method (Hargreaves and Samani, 1982), which requires daily temperatures as time-dependent inputs. Both precipitation and temperature data could be stored in data centers and accessed through the SOS interface. The TOPMODEL and Hargreaves model are accessed through OpenMI to facilitate model coupling. The challenge is then how to integrate sensor observations, geoprocessing services, and environmental model components in an integrated environment.

4. GeoJModelBuilder: system design

GeoJModelBuilder is designed around how to manage and coordinate geospatial sensors, data, analysis functions, and models in a workflow environment (Fig. 3). It takes advantages of interoperable geospatial Web Services and OpenMI. Thus different geospatial resources, accessed through Web Service interfaces, and environmental models following the OpenMI could be plugged in flexibly. The adherence to standards makes the system interoperable, reusable, extensible, and evolvable. As a workflow tool, GeoJModelBuilder shares many similar functions with other scientific workflows tools including workflow designer, workflow binding with services and models, and workflow execution engine. In particular, GeoJModelBuilder focuses on a workflow environment that can integrate geospatial services and environmental models. Fig. 3 shows the system architecture of GeoJModelBuilder. The following features are considered particularly when developing GeoJModelBuilder.

4.1. Integrating geoprocessing services

Using standard interfaces (e.g. OGC WPS) and protocols (e.g., HTTP GET/POST and SOAP), distributed geoprocessing services can be easily discovered and composed into a workflow to solve complex geospatial problems. GeoJModelBuilder allows users to drag and drop various geoprocessing services into the panel of the workflow designer. Once each geoprocessing function is accessible through standard interfaces, they can be loaded into GeoJModelBuilder easily and participate into workflows.

GeoJModelBuilder provides full support to the OGC WPS standard. OGC WPS specifies a standard interface for discovering and executing distributed geoprocessing functions. The three mandatory operations included in the standard WPS interface are GetCapabilities, DescribeProcess, and Execute (Schut, 2007). The GetCapabilities operation allows a client to request and receive a service capabilities document that describes the operations and processes of a specific WPS implementation. The DescribeProcess operation allows a client to get detailed information, such as input and output parameter types, about specific processes. The Execute operation allows the client to run a specific process in a WPS server. In addition to the service interface, more specific features are defined: encodings of request/response for process execution, embedded data and metadata in inputs/outputs for process execution, references to Web-accessible data inputs/outputs, long-running processes support, process status information and processing errors report, and storage of process outputs. Like other OGC service specifications, WPS uses the HTTP protocol, in

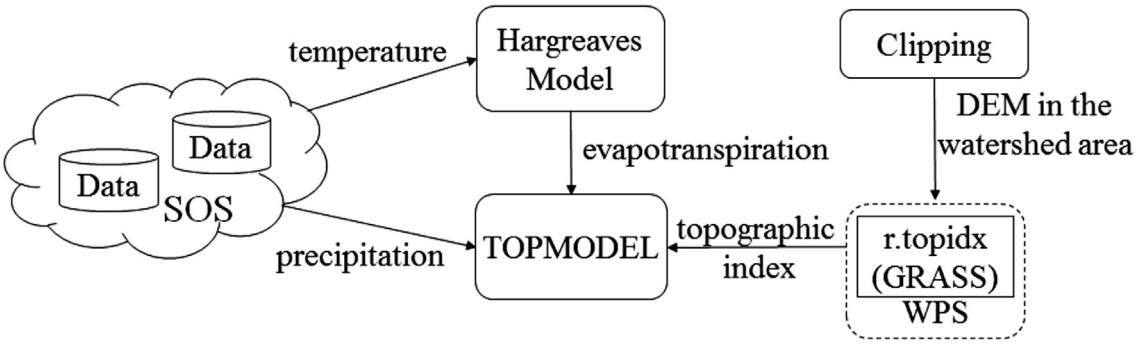

Fig. 2. An example for integration of Web resources and environmental models: watershed runoff simulation.

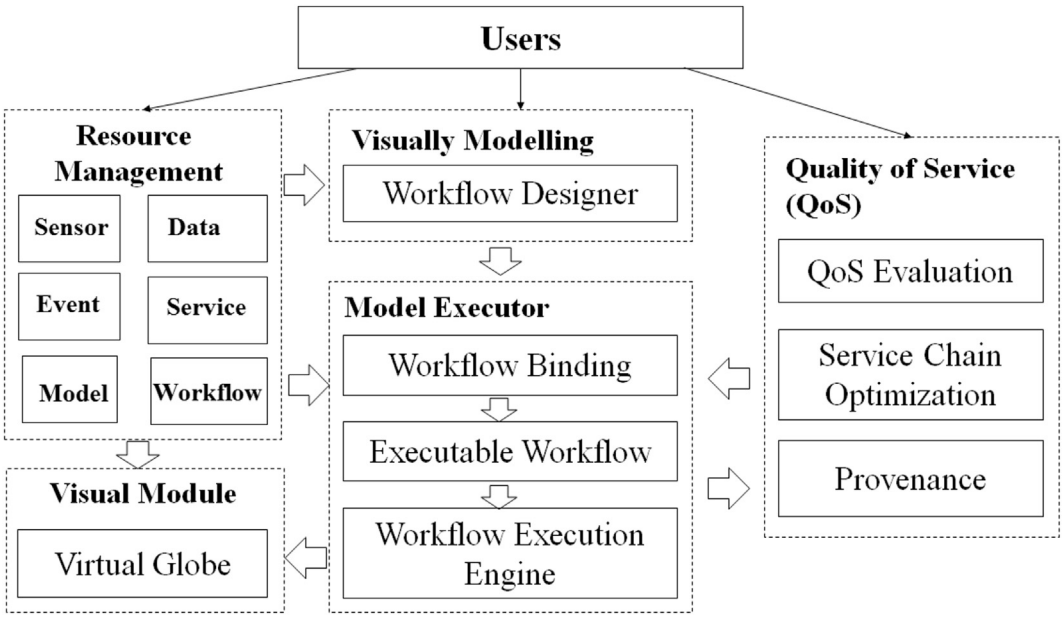

Fig. 3. GeoJModelBuilder system architecture.

particular the GET and POST operations, to exchange XML-based request and response messages.

The integration of workflow and geoprocessing services separates the abstract and concrete workflows. The abstract workflow is resource-independent. Workflow binding refers to the mapping from the abstract workflow to the underlying resources, where concrete data and services are bound to nodes in the abstract workflow based on the type and parameter mapping. Abstract workflows are designed in the workflow designer using the drag-and-drop mode. Each workflow node can be bound dynamically to specific services to generate an executable concrete workflow. The derived data products can be visualized into virtual globes.

4.2. Integrating OpenMI

Integrating OpenMI and geospatial services enhances capabilities of the workflow tool to take the best of both approaches. The tool can not only conduct sophisticated environmental modelling, but also provide modellers with powerful capabilities for data processing, management, and visualization.

OpenMI specifies a set of interfaces for component access as well as how the data is being exchanged between components. Fig. 4 shows a sketch of major OpenMI interfaces and their implementation classes in the OpenMI SDK (OATC, 2015). One core of OpenMI is the interface to access a model component, the ILinkableComponent interface. Components are linked using the ILink interface, which captures all information about the link between two linkable components. Such information includes semantics of each exchanged value (IValueSet) produced by a linkable component and passed over a link: what does it represent (IQuantity), where does it apply (IElementSet), when does it apply (ITime), and how is it processed (IDataOperation) (OATC, 2007). It is noted that the IElementSet was the only spatial construction in the OpenMI standard, while in the latest standard it is an extension of the "ISpatialDefinition" interface (Vanecek and Moore, 2014). Fig. 5 shows an example of the link between two components, where an exchange item (IExchangeItem) is the combination of IQuantity and IElementSet, representing a quantity on an element set. Data exchange in OpenMI is triggered by a component, so the OpenMI SDK provides a class Trigger for this role.

Geospatial services and OpenMI could be either loosely-coupled or tightly coupled, depending on whether the services are providing time-dependent values or not. The topographic index is derived from DEM, which does not need to be re-computed during the time step computation of TOPMODEL. It can be assigned to the model at the beginning. The linkable components in OpenMI provide two methods: *initialize* and *getValues*. The component is initialized by the predefined values, assigned by results of geoprocessing services. The *getValues* method retrieves data from another linkable component according to time steps. It is the fundamental run-time method for pull-based communication in model execution.

While the exchange of data between geoprocessing services and OpenMI using predefined values in the *initialize* method could be characterized as a loosely-coupled approach, embedding SOS as a component in OpenMI could be characterized as a tightly-coupled approach. In this case, the TOPMODEL needs time-dependent precipitation observations from SOS at each time step. An OpenMI component, named SOSReader (Fig. 4), is implemented to retrieve observations in specific timestamps from the SOS, which can be used as time-dependent inputs to other models. SOSReader provides an output exchange item. The location of the sensor in SOS can be put in the element set, and the observation (observable property of the sensor) can be assigned to the quantity in the output exchange item. When the *getValues* method is invoked in the SOSReader, it will retrieve observations at specific times using standard-based service calls.

4.3. Integrating Sensor Web and events

OGC Sensor Web Enablement standards enable Web-based discovery and accessing of sensor networks and sensor observations using open standard protocols and interfaces (Botts et al., 2007). Real-time sensor observations and sensor system information can be retrieved with Sensor Observation Service (SOS). Sensor Planning Service (SPS) is intended to task collection assets with standard interfaces. Sensor Event Service (SES) performs filtering of sensor data (streams) based upon the filter criteria defined in clients' subscription. Web Notification Service (WNS) is designed for asynchronous communications. These services

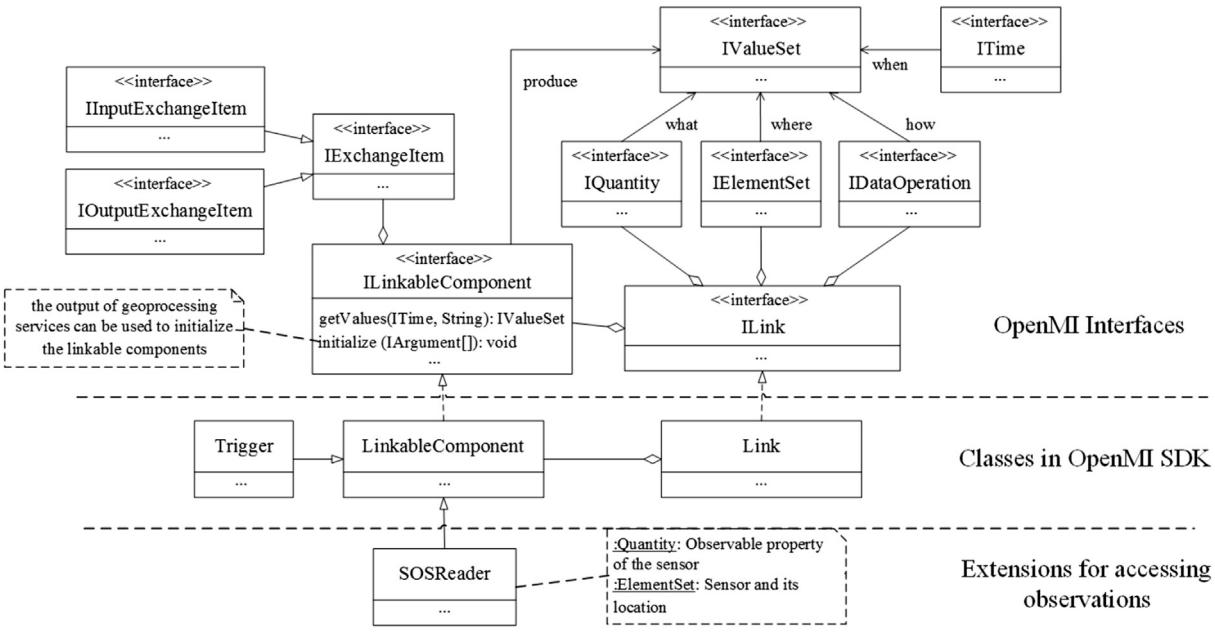

Fig. 4. UML diagram of OpenMI.

are adopted in GeoJModelBuilder for event-driven sensor planning and geoprocessing.

An event is defined as anything that happens at an instant or over an interval of time (Echterhoff and Everding, 2011). From the Sensor Web perspective, an event could be any observation, or an observation that satisfies certain criteria. The event management consists of event generator, event channel, and event processing engine (Fig. 6). User subscribes for an event with filter criteria and ways to receive notifications. The filter will be encoded as an event pattern (rule for filtering and analysis of events) and stored in the event processing engine. The sensors in the filter would be registered and used by event generator. Event generator sends the GetObservation request to a SOS regularly. Observations from SOS are parsed and recoded in the event generator, which plays the role of event producer. These observations are treated as events, and put into the event channel as a queue, waiting to be processed later by the event processing engine, in which the filtering is applied the events. The event processing engine plays a role of SES. When the event pattern is matched, an event processing action will be taken (e.g., notify to the subscriber). Thus the event management allows push-based active environmental monitoring and automatic dissemination of abnormal observations (events). Once users get the notification from WNS, they can either choose to access existing observations or task sensor systems by SPS for new observations. GeoJModelBuilder is coupled with virtual globes. Thus it provides a user-friendly environment for sensor planning. For example, the simulation of satellite orbits in a virtual globe could help users determine when and which satellite is available to a specific area, thus help the tasking of sensor systems. These observations, according specified workflows, are processed by geoprocessing services.

4.4. Integrating QoS

When the abstract workflow is bound to specific services to generate a concrete workflow, Quality of Service (QoS) could be applied in selecting a service from multiple services with the same functionality. Conventionally, a geoprocessing chaining component is capable of searching, chaining, and running services. Here the component is enriched with provenance and quality related capabilities. First, it is able to do QoS evaluation using a selected set of QoS attributes. It complements general QoS attributes

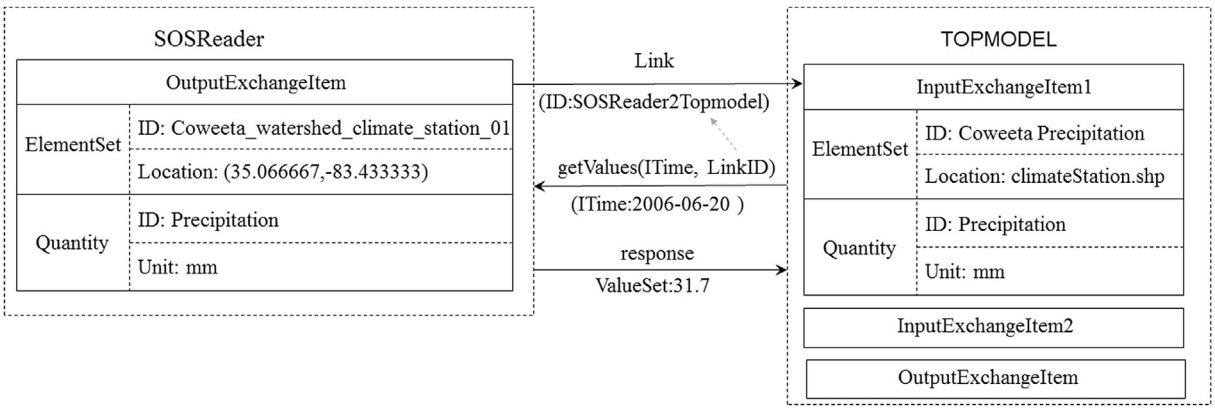

Fig. 5. An example link.

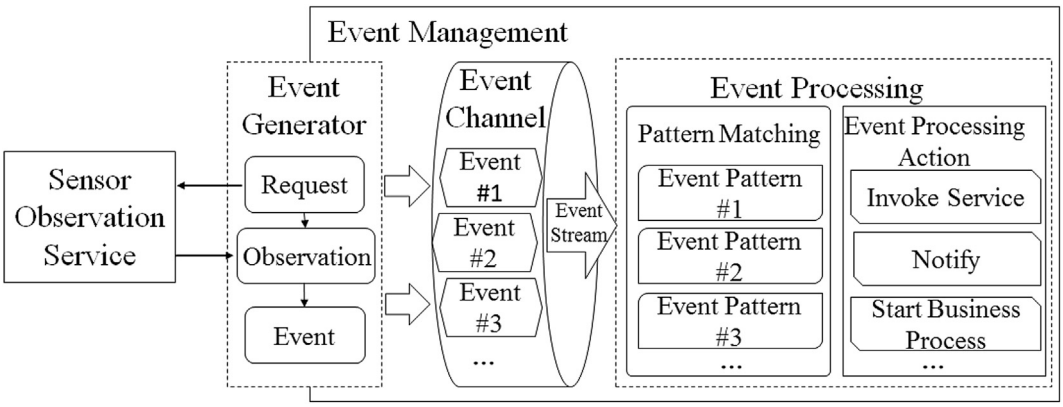

Fig. 6. Event management.

(e.g., performance, availability, reliability) with geospatial attributes and measurements from the service proxy. An important geospatial attribute incorporated for QoS is the quality of output data products, which can be evaluated using provenance. The service proxy tests and monitors geoprocessing services. It can also summarize users' rating results. The test reports can be published in the catalogue service. Second, it can do QoS-aware optimization of geoprocessing service chains. Third, it is able to trace the execution of chains, and adjusting selected services during the run-time. The tracing results, captured as the provenance (He et al., 2015), can also contribute to feedback evaluation of quality of services or chains. By evaluating the QoS and enabling QoS-aware optimization of geoprocessing chains, GeoJModelBuilder could provide high quality of services for geospatial applications.

QoS can be understood in the following two perspectives (Hoyer and Hoyer, 2001): the first is how measurable characteristics of services satisfy a fixed specification, i.e. conformance to specification; the second is the services' capability to meet customer expectations. The first one is related to the objective aspect of QoS, while the second one concerns about the subjective aspect of QoS. Both objective and subjective aspects are addressed in GeoJModelBuilder. In particular, GeoJModelBuilder allows users to selectively set preferences on QoS attributes, which later can be used in the evaluation model of QoS. More details for QoS aware geoprocessing is provided in Yue et al. (2014b).

5. Implementation

GeoJModelBuilder is written as a Java application for cross-platform support. Java Swing and NASA World Wind (NASA, 2014) technologies are used to realize the graphical user interface (GUI) of GeoJModelBuilder. The Java OpenMI software development kit (SDK) is integrated into GeoJModelBuilder for invoking OpenMI-compliant model components (OATC, 2015). Workflows are drawn on the JPanel as graphics using native libraries from the Java runtime environment. Basic operations, such as zoom, redo, and undo, are supported. NASA World Wind is embedded into the GeoJModelBuilder for visualization of input data, results, sensors, and geoprocessing services. Users can click each geoprocessing service in the panel to evaluate its QoS. Example geoprocessing services include those from GeoPW, which provides near two hundred geoprocessing services on the Web (Yue et al., 2010). It provides a skeleton for wrapping software packages (e.g., GRASS, Dynamic Link Library) as geoprocessing services following the OGC WPS standard, and support WPS 0.4 and 1.0 (http://geopw.whu.edu.cn/geopw.html).

The graphical user interface of GeoJModelBuilder is shown in Fig. 7. It shows two interfaces: one is the interface for resource management and modelling, and the other one is a virtual globe based interface. In the first user interface, geospatial resources including geoprocessing services, observation services, environmental models, and events are organized as hierarchical trees on the left panel. When users right-click the "*Geoprocessing*" node and choose "*add*", a dialog to add new services will appear. Once users input the WPS service address, service version, and a name, all geoprocessing processes in the WPS will be added to the resource tree. All children nodes in the tree can be dragged into the workspace and managed as building blocks to construct workflows. In the second user interface, the virtual globe is coupled with GeoJModelBuilder in the right panel. The layers are listed in the left panel, which includes *existed layers* pre-loaded into the NASA World Wind and *added layers* loaded by users. For example, map tiles from the MapWorld (TianDiTu, 2014) are loaded into the virtual globe as a layer for better visualization of geospatial data in China.

Geospatial data and in-situ or remote satellites used in workflows can be visualized in the GeoJModelBuilder. Data from either local file systems or Web could be loaded into the virtual globe. Fig. 8a shows in-situ sensors deployed in the East Lake, Wuhan, Hubei, China. Fig. 8b shows the process of real-time simulation of a satellite. After user selects the *SPOT* check box, three SPOT (satellite for observation of Earth) satellites will be shown on the virtual globe. Furthermore, the movement of satellites according to the predefined orbit parameters could be visualized, which helps users determine when observations of satellites can cover specific areas. Fig. 8c shows the interface for sensor planning. Fig. 8d loads the result of the turbidity extraction.

The two examples in Section 3 are demonstrated in GeoJModelBuilder. Fig. 9 shows the workflow for the turbidity extraction. The workflow consists of three kinds of nodes: events represented graphically as octagons, geoprocessing services represented as rounded boxes, and data represented graphically as circles. Events could be subscribed by the *Event Definition* dialog, which appears when users double-click the octagon named *WaterTurbidity* (Fig. 9). In this dialog, the sensor identifier and observation property are required, and more filter conditions can be specified in advanced options. Subscribers can receive notifications via SMS (Short Message Service) or email when an event happens. For example, when the water turbidity exceeds 60 NTU (Nephelometric Turbidity Units), Mr. Wang will search for the appropriate satellite to get remote sensing image covering the East Lake (*DongHu*) area. GUIs in Fig. 8 can help him task appropriate satellites through SPS. When the observation imagery is ready through SOS, the workflow will run and an example result is shown in Fig. 8d.

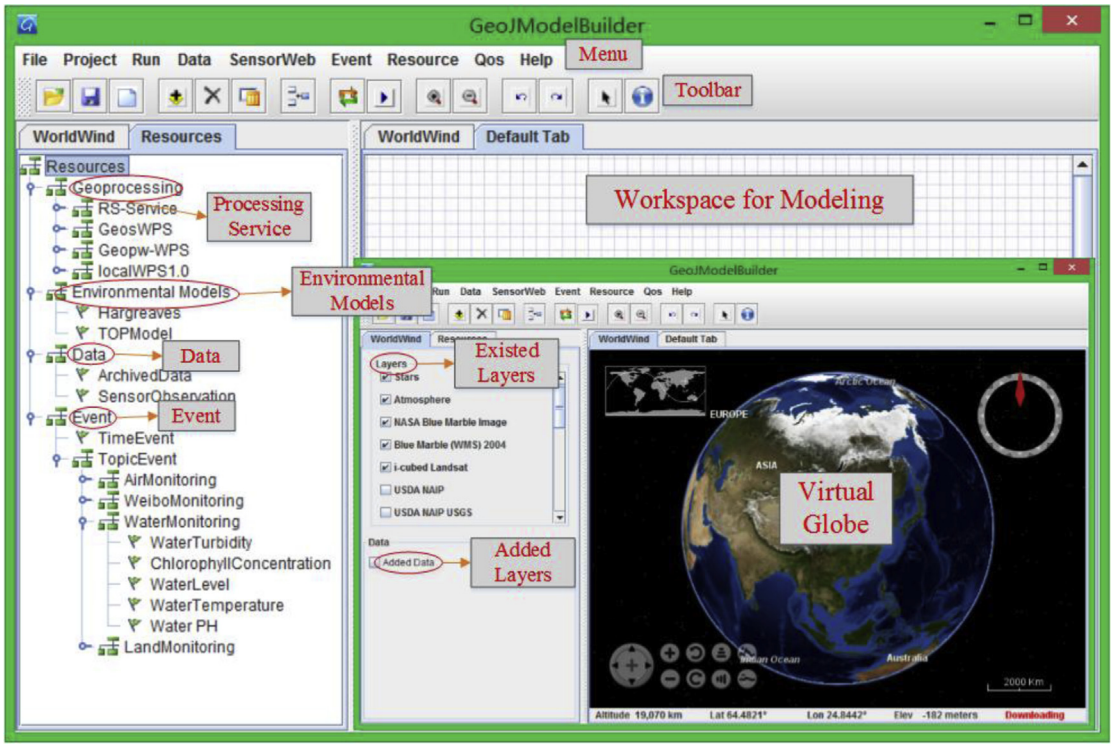

Fig. 7. User interfaces of GeoJModelBuilder.

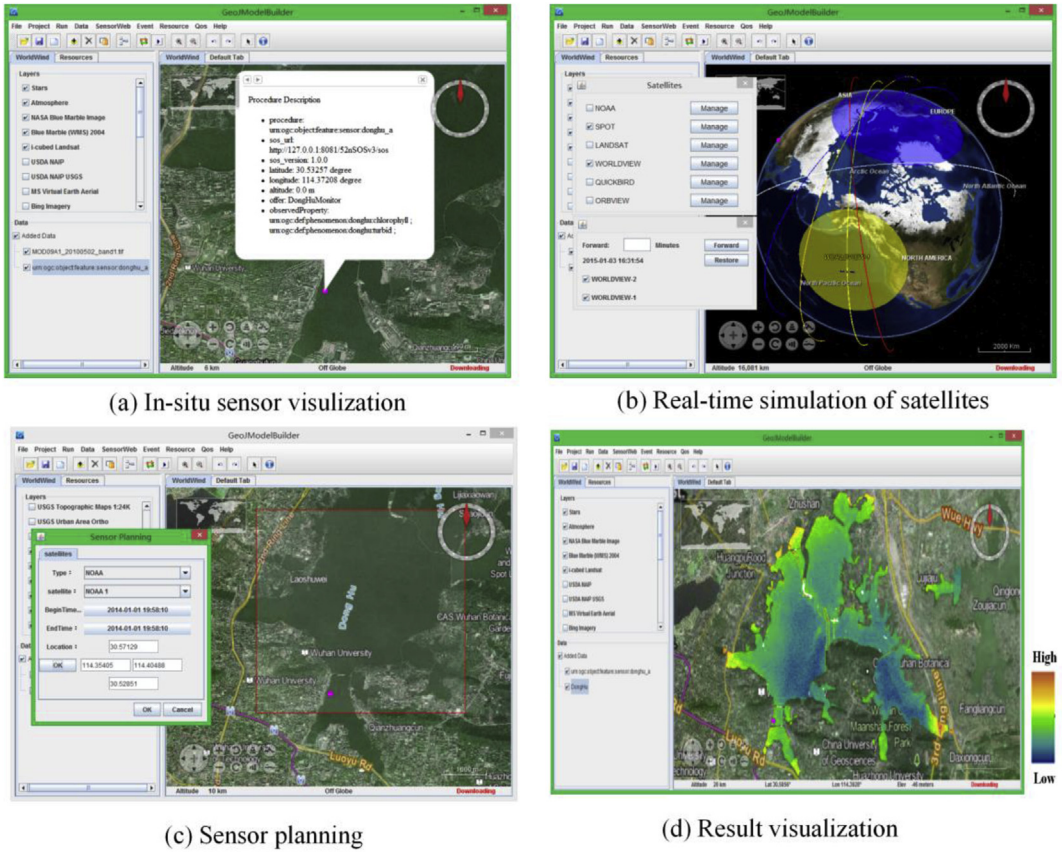

Fig. 8. Capabilities of GeoJModelBuilder in supporting visualization and sensor planning.

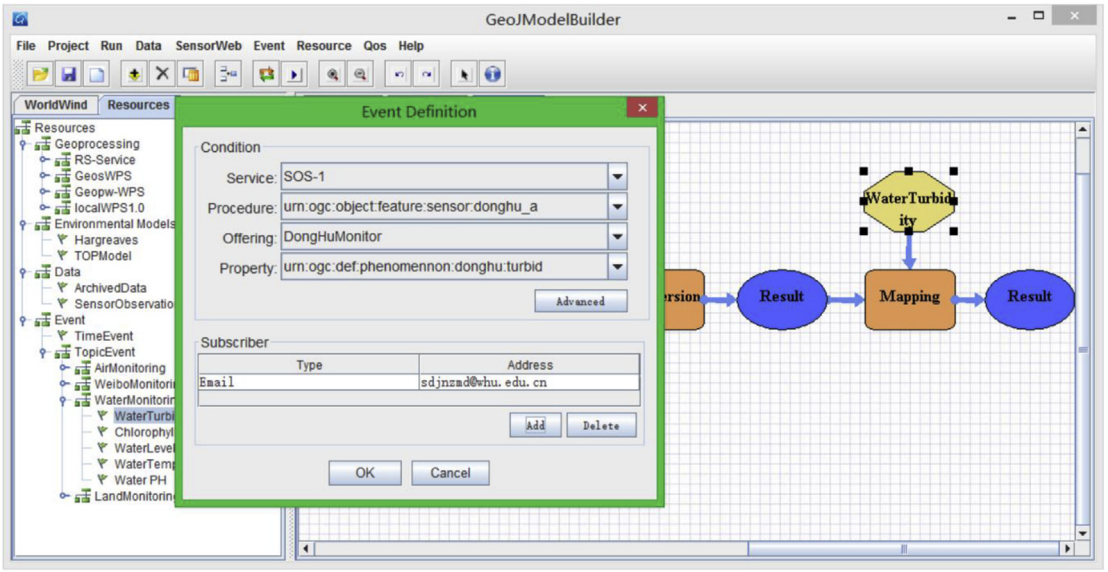

Fig. 9. The GUI for the turbidity extraction example.

Fig. 10 shows the workflow for the watershed runoff simulation. Using the Java OpenMI SDK (OATC, 2015), three OpenMI-compliant components (Hargreaves, TOPMODEL and SOSReader) are implemented in Java. These components can be imported into GeoJModelBuilder through the configuration files. Since the trigger component is similar to a time event, the GUI of GeoJModelBuilder reuses the concept of event, and the time step computation of models is triggered by a time event instead of a trigger. The OpenMI-compliant model components are represented as rounded boxes with a purple colour. *SOSReader* gets time-dependent data from SOS and provides it to other hydrologic models (Hargreaves and TOPMODEL). Two geoprocessing services (*RasterClip* and *TopographicIndex*) work together to provide the topographic index in the area of interests.

Fig. 11a shows how to bind the workflow activity *Clipping* to a geoprocessing service named *RasterClip*, which has two input parameters: *RasterInput* and *ShpInput*. Two input parameters are needed for this model: an image named "*imageInput*" and an AOI file named "*AOIFile*". Data binding is to assign values to input parameters in the activity. When the binding process is finished, the abstract workflow is instantiated to a concrete workflow.

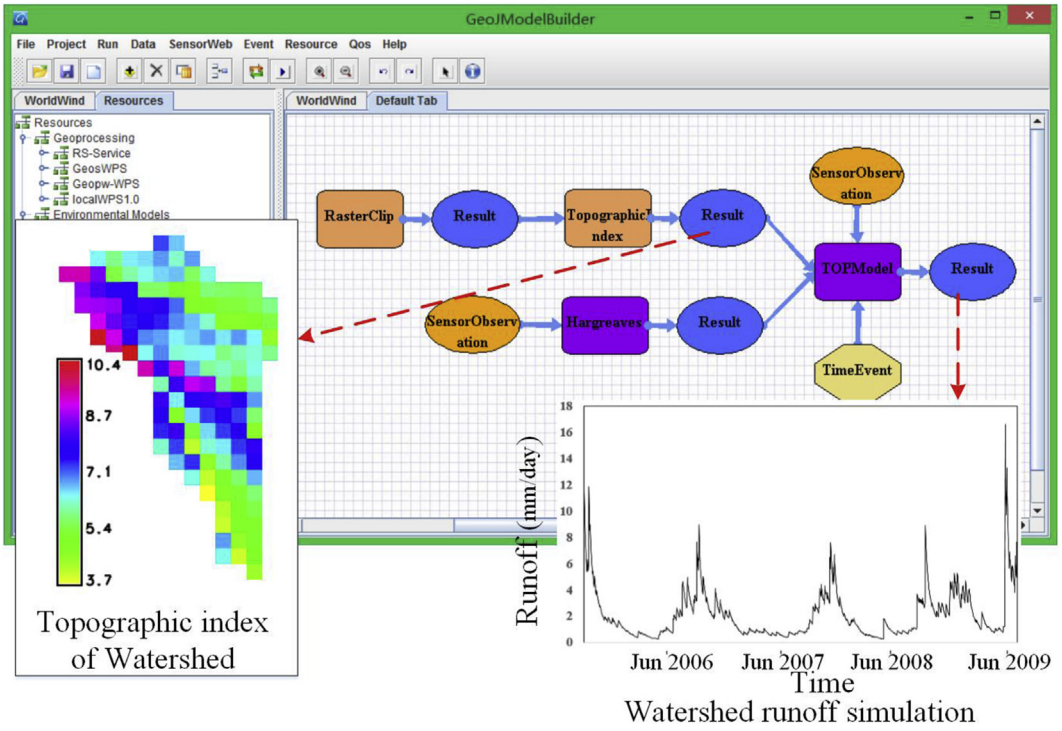

Fig. 10. The GUI for the watershed runoff simulation example.

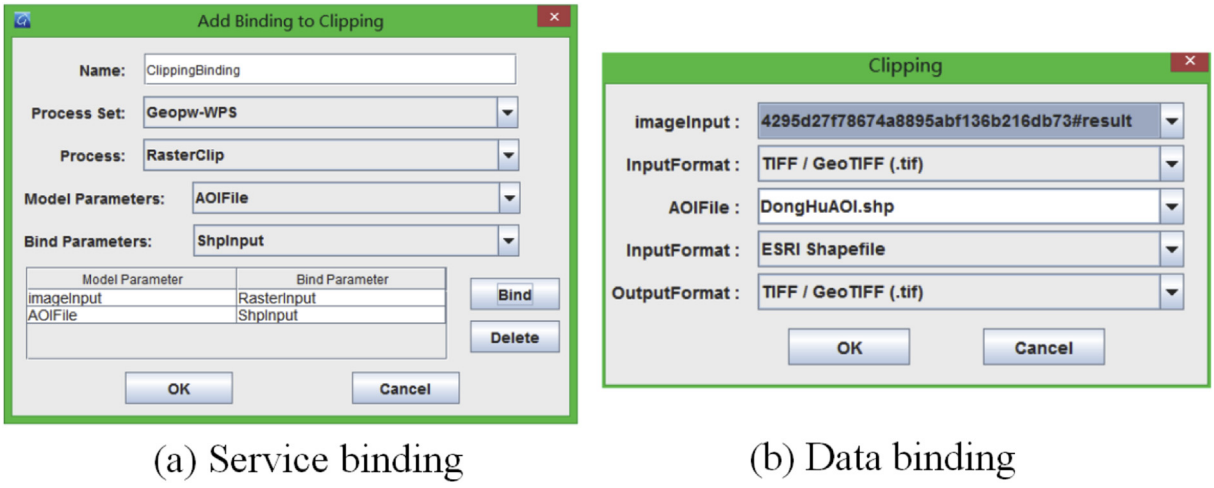

Fig. 11. Workflow binding in GeoJModelBuilder.

Fig. 12 illustrates the wizard for QoS evaluation of individual geoprocessing services. Fig. 13 shows the optimization for geoprocessing workflows. It includes local selection and global optimization. The local selection is to select a set of candidate services, while the global optimization will determine the best combination of services.

GeoJModelBuilder has been published as an open source tool (under the GNU General Public License version 2, http://sourceforge.net/projects/geopw/), and awarded a First Place with the Crystal Bull Award in Europa Challenge 2013, in conjunction with the INSPIRE conference 2013 in Florence, Italy. Its following extension on QoS, named GeoQoS, is published in Europa Challenge 2014, in conjunction with the OSGeo's European Conference on Free and Open Source Software for Geospatial (FOSS4G-Europe 2014) in Bremen, Germany. It is also used for hands-on practice for service development and geoprocessing modelling in the 11th

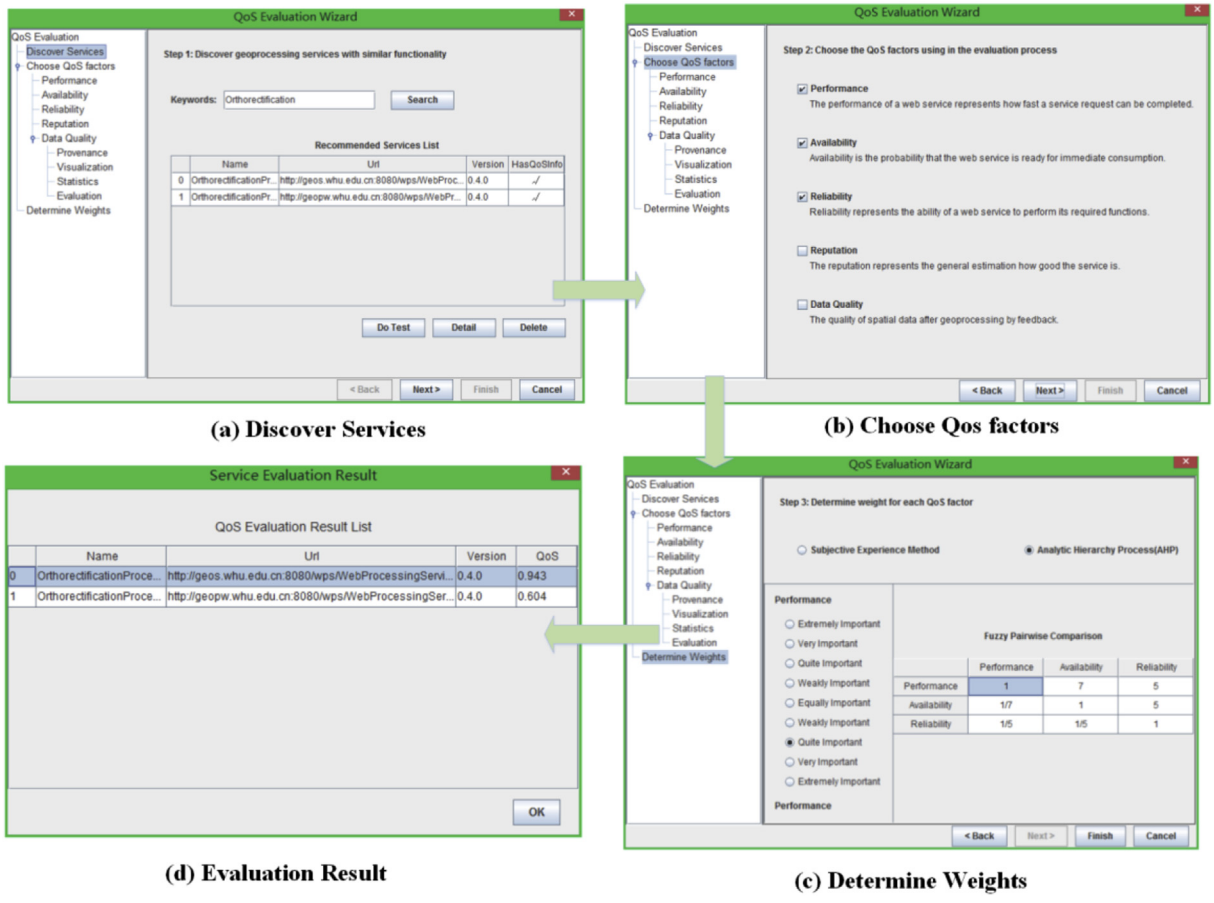

Fig. 12. QoS evaluation of geoprocessing services.

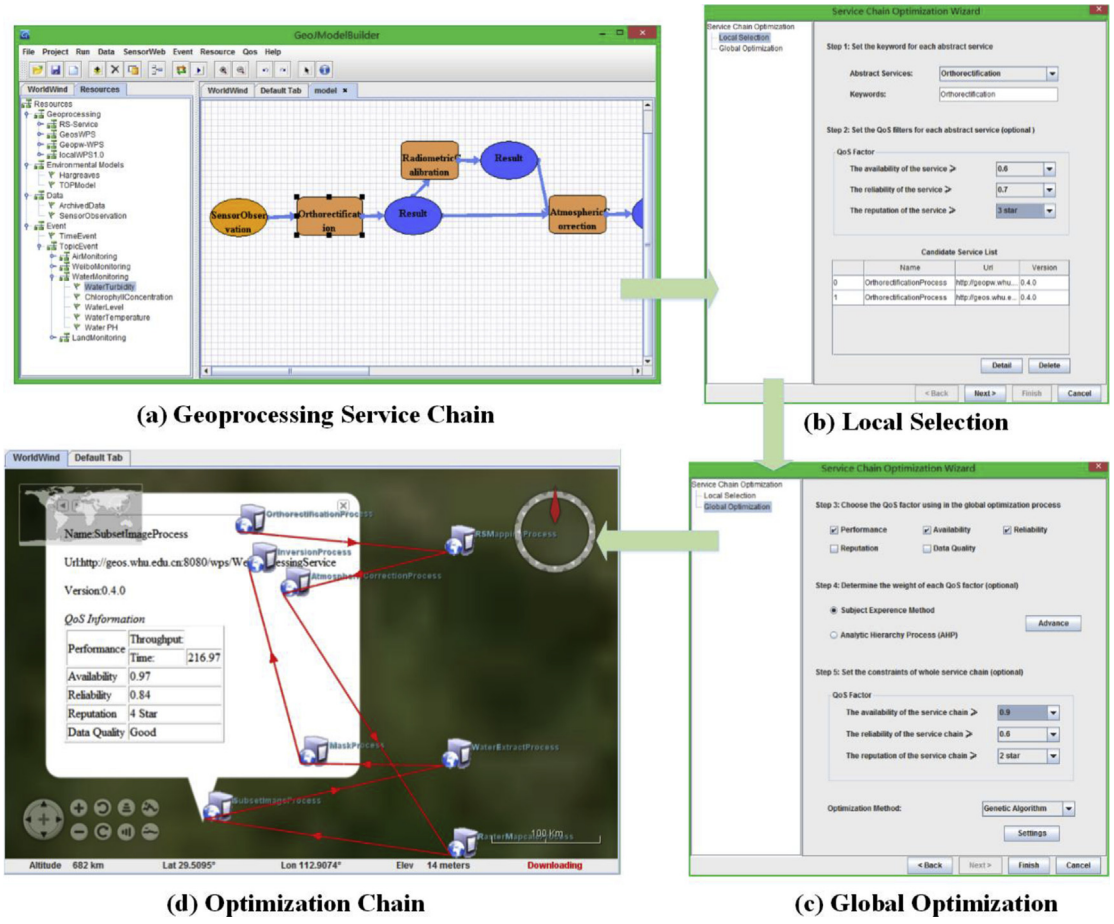

Fig. 13. QoS-aware optimization of geoprocessing workflows.

ISPRS (International Society for Photogrammetry and Remote Sensing) Summer School, held at Wuhan University, Wuhan, China during May 19–28, 2014 (Li et al., 2014). The software is available at SourceForge.net and has been downloaded more than 1000 times by over fifty countries since May 30, 2013. Source code, code documentation, user documentation, and window installers are provided in the SourceForge.net.

6. Result analysis

GeoJModelBuilder provides a workflow engine to execute workflows. The execution of geoprocessing services and OpenMI-compliant model components is different. The geoprocessing services are often invoked once to get the result and does not need to maintain status, while integrated environmental models are usually time-dependent and invoked many times. In a workflow-based service chain, when the former service is executed successfully, the engine will "push" the result to the next service. The OpenMI, instead, is a pull-based architecture, where data are pulled by one model from the previous model in the model chain (Gregersen et al., 2007). When geoprocessing services and environmental models are coupled in a workflow like Fig. 10, the following execution principles are followed: geoprocessing services provide time-independent data (e.g., topographic index) to environmental models. The execution of geoprocessing services and models follows push and pull mechanisms respectively. In this way, all geoprocessing services are executed by the push-based method, OpenMI components are triggered by the *Trigger* component, initialized using the geoprocessing results, and executed in a pull-based method. Considering the promise of the MaaS approaches (Nativi et al., 2013; Bastin et al., 2013; Castronova et al., 2013a), the workflow tool allows models as services to be chained, once models are accessible through WPS. On the other hand, the integration of OpenMI allows the tool to overcome the current limitations of WPS in wrapping models.

The coupling of environmental models and traditional GIS are widely studied (Goodchild et al., 1992; Clark, 1998). In general, two approaches are available for coupling environmental models and GIS: loose and tight approach (Huang and Jiang, 2002). Loose coupling depends on the exchange of data files between GIS and environmental models, while tight coupling is usually developed to extend one system with the capabilities of the other system. As GIS is moving toward geographic information services (GIServices), this paper suggests the hybrid coupling of services and OpenMI. The exchange of geoprocessing results between geoprocessing services and environmental models is a loosely coupling approach. Due to the difference of geospatial analysis functions and environmental models (time independent or dependent inputs, one or many times invocation, unidirectional or bidirectional interaction), the loosely coupling approach allows geoprocessing services and OpenMI-compliant model components to be enacted separately. The tight coupling on wrapping SOS as a modelling component shows promise that resources distributed on the Web could be integrated into OpenMI.

While the tool accommodates different execution mechanisms for models and services, the GUIs hide the underlying technical

details and try to provide consistent user experiences for both service-based environmental monitoring and model-based integrated environmental modelling. Users can drag and drop different geoprocessing services, models, data, and events into the panel to generate workflows, and the tool will deal with the execution of workflows. The event in the workflow supports "active" monitoring of the pollution-prone location in the East Lake. While events are originally designed for the Sensor Web enablement, GeoJModelBuilder harmonizes this concept with the Trigger in OpenMI to provide a uniform view to users. In the GUI, the time step computation of models can be triggered by a time event instead of Trigger, thus bringing consistent experiences to users. In the future, more cases are needed for the improvement of user experiences.

When users receive the notification, they can launch the workflow or plan sensor for retrieving appropriate observation in the tool. GeoJModelBuilder supports both transparent and translucent chaining, and allows users to interact with workflows for value assignment and service binding. It is feasible that GeoJModelBuilder gets the notification directly and launch geoprocessing workflows automatically according to predefined business logics. The future development can add this to enhance the current capabilities of the tool. In addition, the QoS awareness in the current implementation is limited to services, it is possible to extend QoS to quality of models, when more models are available. This could be our future work. Currently, GeoJModelBuilder relies on the SDK for the OpenMI 1.4, which is available as an open source tool and has a Java version, although the OpenMI 2.0 is the newest version and also published as an OGC standard (Vanecek and Moore, 2014). When the OpenMI SDK v2.0 is available, GeoJModelBuilder can be updated to support OpenMI 2.0.

7. Conclusions and future work

GeoJModelBuilder offers a flexible way to support environmental monitoring and integrated modelling. Geoprocessing services and OpenMI-compliant model components can be chained together into workflows easily. By integrating Sensor Web and events with geoprocessing workflows, the workflows are "active" in supporting environmental monitoring and integrated modelling. Coupled with virtual globes, GeoJModelBuilder provides an interactive three-dimensional environment for sensor planning, and visualizing geospatial data or sensors. Users can select suitable geoprocessing services with QoS and optimize geoprocessing chains for best performance of geoprocessing workflows.

Currently GeoJModelBuilder is extended to support OpenMI by integrating OpenMI-compliant model components into GeoJModelBuilder as model resources. It is good to investigate more approaches to accommodate OpenMI in GeoJModelBuilder for broader community use in environment modelling. For example, how OpenMI can be integrated into WPS or Cloud Computing to provide environmental models on the Web. Some proposals have been available to extend WPS to wrap models (Nativi et al., 2013; Bastin et al., 2013; Castronova et al., 2013a). This could be our future work. In addition, we will investigate how to support scripting languages (e.g. Python) by exporting workflow to scripts, which can be executed by gluing various open source packages such as GRASS and R (R, 2015). In this way, we could also leverage High Performance Computing (HPC) technologies such as Message Passing Interface (MPI) (Pacheco, 1997) or MapReduce (Dean and Ghemawat, 2008) for efficient enactment of workflows in versatile computing environments.

Acknowledgements

We are grateful to Patrick Hogan, the program manager of NASA World Wind, for his valuable support the development of GeoJModelBuilder. We also appreciate the three anonymous reviewers and editor for their very constructive comments that helped improve the quality of the paper. The work was supported by National Basic Research Program of China (2011CB707105), National Natural Science Foundation of China (91438203 and 41271397), Program for New Century Excellent Talents in University (NCET-13-0435), and Fundamental Research Funds for the Central Universities (2042014kf0224).

References

Bastin, L., Cornford, D., Jones, R., Heuvelink, G., Pebesma, E., Stasch, C., Nativi, S., Mazzetti, P., Williams, M., 2013. Managing uncertainty in integrated environmental modelling: the UncertWeb framework. Environ. Model. Softw. 39, 116–134.

Béjar, R., Lopez-Pellicer, F.J., Nogueras-Iso, J., Zarazaga-Soria, F.J., Muro-Medrano, P.R., 2014. A protocol for machine-readable cache policies in OGC Web Services: application to the EuroGeoSource information system. Environ. Model. Softw. 60, 346–356.

Beven, K., Kirkby, M., 1979. A physically based, variable contributing area model of basin hydrology/Un modèle à base physique de zone d'appel variable de l'hydrologie du bassin versant. Hydrol. Sci. J. 24 (1), 43–69.

OGC Sensor Web Enablement: overview and high level architecture. In: Botts, M., Percivall, G., Reed, C., Davidson, J. (Eds.), OpenGIS White Pap. 07–165.

Bröring, A., Echterhoff, J., Jirka, S., Simonis, I., Everding, T., Stasch, C., Liang, S., Lemmens, R., 2011. New generation sensor web enablement. Sensors 11 (3), 2652–2699.

Callahan, S.P., Freire, J., Santos, E., Scheidegger, C.E., Silva, C.T., Vo, H.T., 2006. VisTrails: visualization meets data management. In: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data, pp. 745–747.

Castronova, A.M., Goodall, J.L., Elag, M.M., 2013a. Models as web services using the Open Geospatial Consortium (OGC) Web Processing Service (WPS) standard. Environ. Model. Softw. 41, 72–83.

Castronova, A.M., Goodall, J.L., Ercan, M.B., 2013b. Integrated modeling within a hydrologic information system: an OpenMI based approach. Environ. Model. Softw. 39, 263–273.

Clark, M.J., 1998. Putting water in its place: a perspective on GIS in hydrology and water management. Hydrol. Process. 12 (6), 823–834.

de Jesus, J., Walker, P., Grant, M., Groom, S., 2012. WPS orchestration using the Taverna workbench: the eScience approach. Comput. Geosci. 47, 75–86.

Dean, J., Ghemawat, S., 2008. MapReduce: simplified data processing on large clusters. Commun. ACM 51 (1), 107–113.

Echterhoff, J., Everding, T., 2008. OGC Discussion Paper: OpenGIS® Sensor Event Service Interface Specification. Version 0.3.0 OGC 08–133. Open Geospatial Consortium, Inc, 96 pp.

Echterhoff, J., Everding, T., 2011. OGC Event Service-review and Current State. Version. OGC 11–088r1. Open Geospatial Consortium, Inc, 33 pp.

ESRI, 2015. ArcGIS ModelBuilder (accessed 22.01.15.). http://resources.arcgis.com/en/help/main/10.1/index.html#//002w00000001000000.

Friis-Christensen, A., Lucchi, R., Lutz, M., Ostländer, N., 2009. Service chaining architectures for applications implementing distributed geographic information processing. Int. J. Geogr. Inf. Sci. 23 (5), 561–580.

Geller, G., Turner, W., 2007. The model web: a concept for ecological forecasting. In: Proceedings of Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007. IEEE, pp. 2469–2472.

Goodchild, M., Haining, R., Wise, S., 1992. Integrating GIS and spatial data analysis: problems and possibilities. Int. J. Geogr. Inf. Syst. 6 (5), 407–423.

Granell, C., 2014. Robust workflow systems+ flexible geoprocessing services= geo-enabled model web? Geogr. Inf. Syst. Trends Technol. 172–204.

Granell, C., Díaz, L., Gould, M., 2010. Service-oriented applications for environmental models: reusable geospatial services. Environ. Model. Softw. 25 (2), 182–198.

Granell, C., Schade, S., Ostländer, N., 2013a. Seeing the forest through the trees: a review of integrated environmental modelling tools. Comput. Environ. Urban Syst. 41, 136–150.

Granell, C., Díaz, L., Schade, S., Ostländer, N., Huerta, J., 2013b. Enhancing integrated environmental modelling by designing resource-oriented interfaces. Environ. Model. Softw. 39, 229–246.

Granell, C., Díaz, L., Tamayo, A., Huerta, J., 2014. Assessment of OGC web processing services for REST principles. Int. J. Data Min. Model. Manag. 6 (4), 391–412.

GRASS, 2015. Geographic Resources Analysis Support System (GRASS) Software, Version 6.4.2. Open Source Geospatial Foundation (accessed 08.01.15.). http://grass.osgeo.org.

Gregersen, J., Gijsbers, P., Westen, S., 2007. OpenMI: open modelling interface. J. Hydroinf. 9 (3), 175–191.

Hargreaves, G.H., Samani, Z.A., 1982. Estimating potential evapotranspiration. J. Irrig. Drain. Div. 108 (3), 225–230.
He, L., Yue, P., Di, L., Zhang, M., Hu, L., 2015. Adding geospatial data provenance into SDI - a service-oriented approach. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 8 (2), 926–936.
Hoyer, R.W., Hoyer, B.B., 2001. What is quality. Qual. Prog. 34 (7), 53–62.
Huang, B., Jiang, B., 2002. AVTOP: a full integration of TOPMODEL into GIS. Environ. Model. Softw. 17 (3), 261–268.
Intergraph, 2015. ERDAS IMAGINE Spatial Modeler (accessed 22.02.15.). http://www.intergraph.com/assets/pressreleases/2012/10-23-2012b.aspx.
Jakeman, A.J., Letcher, R.A., 2003. Integrated assessment and modelling: features, principles and examples for catchment management. Environ. Model. Softw. 18 (6), 491–501.
Laniak, G.F., Rizzoli, A.E., Voinov, A., 2013a. Thematic issue on the future of integrated modeling science and technology. Environ. Model. Softw. 39, 1–2.
Laniak, G.F., Olchin, G., Goodall, J., Voinov, A., Hill, M., Glynn, P., Whelan, G., Geller, G., Quinn, N., Blind, M., Peackham, S., Reaney, S., Gaber, N., Kennedy, R., Hughes, A., 2013b. Integrated environmental modeling: a vision and roadmap for the future. Environ. Model. Softw. 39, 3–23.
Lanig, S., Schilling, A., Stollberg, B., Zipf, A., 2008. Towards standards-based processing of digital elevation models for grid computing through web processing service (WPS). In: Proceedings of Computational Science and its Applications–ICCSA 2008. Springer, pp. 191–203.
Li, D., Gong, J., Yue, P., 2014. Geoinformatics education in China. Geo-spat. Inf. Sci. 17 (4), 208–218.
Lucchi, R., Millot, M., Elfers, C., 2008. Resource Oriented Architecture and REST. Assessment of Impact and Advantages on INSPIRE. Technical Document. European Communities.
Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., Tao, J., Zhao, Y., 2006. Scientific workflow management and the Kepler system. Concurr. Comput. Pract. Exp. 18 (10), 1039–1065.
McFerren, G., Van Zyl, T., Vahed, A., 2012. FOSS geospatial libraries in scientific workflow environments: experiences and directions. Appl. Geomat. 4 (2), 85–93.
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.
Michelson, B., 2006. Event-driven Architecture Overview. Patricia Seybold Group. http://www.elementallinks.com/2006/02/06/event-driven-architecture-overview/.
NASA, 2014. NASA World Wind. http://worldwind.arc.nasa.gov/index.html.
Nativi, S., Mazzetti, P., Geller, G., 2013. Environmental model access and interoperability: the GEO Model Web initiative. Environ. Model. Softw. 39, 214–228.
OASIS, 2007. Web Services Business Process Execution Language. Version 2.0. Web Services Business Process Execution Language (WSBPEL) Technical Committee (TC).
OATC, 2007. OATC (OpenMI Association Technical Committee) OpenMI 1.4 Standard, 79 pp.
OATC, 2015. Java OpenMI 1.4 SDK. OATC (OpenMI Association Technical Committee) (accessed 23.01.15.). http://sourceforge.net/p/openmi/code/HEAD/tree/branches/OpenMI-Version-1-4-Trunk/MyOpenSource/Alterra/.
OGC, 2014. The OGC Seeks Comments on Candidate OGC Web Processing Service 2.0 Standard (accessed 23.01.15.). http://www.opengeospatial.org/standards/requests/125.
OGC, 2015. Open Geospatial Consortium (accessed 23.01.15.). http://www.opengeospatial.org.
Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., Li, P., 2004. Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20 (17), 3045–3054.
Pacheco, P.S., 1997. Parallel Programming with MPI. Morgan Kaufmann.
Parker, P., Letcher, R., Jakeman, A., Beck, M.B., Harris, G., Argent, R.M., Hare, M., Pahl-Wostl, C., Voinov, A., Jansseh, M., Sullivan, P., Scoccimarro, M., Friend, A., Sonnenshein, M., Barker, M., Matejicek, L., Odulaja, D., Deadman, P., Lim, K., Larocque, G., Tarikhi, P., Fletcher, C., Put, A., Maxwell, T., Charles, A., Breeze, H., Nakatani, N., Mudgal, S., Naito, W., Osidele, O., Eriksson, I., Kautsky, U., Kautsky, E., Naeslund, B., Kumblad, L., Park, R., Maltagliati, S., Girardin, P., Rizzoli, A., Mauriello, D., Hoch, R., Bin, S., 2002. Progress in integrated assessment and modelling. Environ. Model. Softw. 17 (3), 209–217.
OGC 02–112. In: Percivall, G. (Ed.), 2002. The OpenGIS Abstract Specification, Topic 12: OpenGIS Service Architecture. Version 4.3. Open Geospatial Consortium, Inc.
Pratt, A., Peters, C., Siddeswara, G., Lee, B., Terhorst, A., 2010. Exposing the Kepler scientific workflow system as an OGC web processing service. In: Proceedings of iEMSs (International Environmental Modelling and Software Society), 2010.
R, 2015. The R Project for Statistical Computing (accessed 08.01.15.). http://www.r-project.org/.
Rizzoli, A.E., Davis, J.R., 1999. Integration and re-use of environmental models. Environ. Model. Softw. 14 (6), 493–494.
Schut, P., 2007. OpenGIS® Web Processing Service, Version 1.0.0. Open Geospatial Consortium, Wayland, MA. Technical Report No. 05–007r7.
Sommerville, I., 2004. Software Engineering. Pearson Education.
Sonnet, J., 2004. OWS 2 Common Architecture: WSDL SOAP UDDI. Open Geospatial Consortium Inc.
Tamayo, A., Granell, C., Huerta, J., 2012. Measuring complexity in OGC web services XML schemas: pragmatic use and solutions. Int. J. Geogr. Inf. Sci. 26 (6), 1109–1130.
van Zyl, T.L., Vahed, A., McFerren, G., Hohls, D., 2012. Earth observation scientific workflows in a distributed computing environment. Trans. GIS 16 (2), 233–248.
Vanecek, S., Moore, R., 2014. OGC® Open Modelling Interface Interface Standard. Version 2.0. OGC 11–014r3. Open Geospatial Consortium, Inc, 122 pp.
Voinov, A., Shugart, H.H., 2013. 'Integronsters', integral and integrated modeling. Environ. Model. Softw. 39, 149–158.
W3C, 2007. Web Services Description Language (WSDL) 2.0. World Wide Web Consortium (W3C). http://www.w3.org/TR/2007/REC-wsdl20-adjuncts-20070626/.
Westerholt, R., Resch, B., 2014. Asynchronous geospatial processing: an event-driven push-based architecture for the OGC Web Processing Service. Trans. GIS. http://dx.doi.org/10.1111/tgis.12104.
Yu, G., Di, L., Moses, J.F., Li, P., Zhao, P., 2008. Geospatial workflow in a sensor Web environment: transactions, events, and asynchrony. In: Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE.
Yu, G., Di, L., Zhang, B., Wang, H., 2010. Coordination through geospatial web service workflow in the sensor web environment. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 3 (4), 433–441.
Yu, G., Zhao, P., Di, L., Chen, A., Deng, M., Bai, Y., 2012. BPELPower—a BPEL execution engine for geospatial web services. Comput. Geosci. 47, 87–101.
Yue, P., Gong, J., Di, L., Yuan, J., Sun, L., Sun, Z., Wang, Q., 2010. GeoPW: laying blocks for the geospatial processing web. Trans. GIS 14 (6), 755–772.
Yue, P., Gong, J., Di, L., He, L., 2012. Automatic geospatial metadata generation for earth science virtual data products. Geoinformatica 16 (1), 1–29.
Yue, P., Jiang, L., Hu, L., 2014a. Google fusion tables for managing soil moisture sensor observations. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (11), 4414–4421.
Yue, P., Tan, Z., Zhang, M., 2014b. GeoQoS: delivering quality of services on the geoprocessing web. In: Proceedings of OSGeo's European Conference on Free and Open Source Software for Geospatial (FOSS4G-Europe 2014), 15–17 July 2014. Germany, Bremen.