Zhang et al. Open Geospatial Data, Software and Standards (2017) 2:8
DOI 10.1186/s40965-017-0022-7

Open Geospatial Data, Software and Standards

SOFTWARE Open Access

GeoJModelBuilder: an open source geoprocessing workflow tool

Mingda Zhang[3], Xiaoqian Bu[3] and Peng Yue[1,2*]

Abstract

Background: Scientific workflows have been commonly used in geospatial data analysis and Cyberinfrastructure. They allow distributed geoprocessing algorithms, models, data, and sensors to be chained together to support geospatial data analysis, and environmental monitoring, and integrated environmental modelling.

Results: This paper presents an open source geoprocessing workflow tool, GeoJModelBuilder. It leverages open standards, Sensor Web, geoprocessing commands and services, OpenMI-compliant models together.

Conclusion: The implementation provides a flexible, reusable, interoperable, and user-friendly way for geoprocessing in an open environment.

Keywords: Scientific workflow, Geoprocessing services, Environmental monitoring, Integrated environmental modelling, Open standards

Background

With the rapid development of scientific computing and web technologies, an increasing number of geospatial information services are continuously available on the Web. Open standards like the Open Geospatial Consortium (OGC) interface standards, such as Web Processing Service (WPS), Web Feature Service (WFS), and Web Coverage Service (WCS), further promote sharing and interoperation of geospatial information and processing functions. For example, GeoPW, developed by Wuhan University, is a WPS toolkit [1]. Almost 200 geoprocessing services are available in GeoPW, which makes it easy to utilize geoprocessing services to complete processing tasks for the Web client. However, an atomic geoprocessing service is limited and sometimes fails to meet demands of users for some large-scale and complex processing tasks. The way of composing geoprocessing services and constructing geoprocessing workflows is important for fulfilling complex processing tasks [2, 3]. The workflow tool ArcGIS Model Builder can enable the composition of various geoprocessing functions and realize complex processing workflows. However, ArcGIS Model Builder is limited to its own proprietary environments and can only compose geoprocessing functions in ArcGIS. In this paper, we suggest that an open model builder could at least provide the following capabilities: 1) supporting open standards like OGC standards; 2) allowing environmental monitoring and live geoprocessing by incorporating sensor observations; 3) taking the best of local and remote data and computing resources; 4) connecting GIS functionalities with environmental models for better decision making. In the past several years, Wuhan University has been focusing on developing and enriching some capabilities to an existing geoprocessing workflow tool, named GeoJModelBuilder [4–7]. In this paper, we will give a full review of current capabilities in the software, and present some latest progress like model as services to be integrated in the software and the scripting approach for broad connection to various geospatial resources like GRASS (Geographic Resources Analysis Support System) algorithms and geoprocessing services.

Versatile computing environments and heterogeneous resources are distinguishing characteristics in big data era [8, 9]. Under the circumstance, one of the key issue in scientific workflows is that they often need to coordinate these various resources. GeoJModelBuilder is a flexible, extensible, interoperable, loosely-coupled workflow tool, which is designed to support the Web environment as

* Correspondence: pyue@whu.edu.cn
[1]School of Remote Sensing and Information Engineering, Wuhan University, 129 Luoyu Road, Wuhan, Hubei 430079, China
[2]Collaborative Innovation Center of Geospatial Technology, 129 Luoyu Road, Wuhan, Hubei 430079, China
Full list of author information is available at the end of the article

well as local algorithms and models. "Model as a Service" (MaaS) approach has been used in the Model Web, which takes the Web as the environment to integrate models [10]. The MaaS approach can provide an engineering approach towards the implementation of integrated modelling systems layered on environmental information infrastructures [11]. GeoJModelBuilder can use both the Web and local models for integrated environmental modelling. For complex environmental models such as numerical time-marching models, the models could be exposed as WebSocket services on the Web. In the local environment, they can be accessed through the OpenMI interfaces. For simple models like traditional geospatial analysis algorithms, they could be exposed on the Web through the WPS standard interface. Thus environmental models can be wrapped as services following Web Service standards, and coupled through service-oriented workflows [12, 13]. The OGC WPS standard specifies standard operations such as *GetCapabilities*, *DescribeProcess,* and *Execute*, for accessing geoprocessing functions on the Web. When time-step based model interactions are required in complex environmental models, the current WPS specification is not sufficient to access complex environmental models. In this case, GeoJModelBuilder can accommodate the Open Modelling Interface (OpenMI), a standard to describe modelling components and runtime data exchanges between them [14], and makes the OpenMI components accessible through the WebSocket protocol. Thus, the OGC services, WebSocket services, and OpenMI-compliant models are plugged in and play in the GeoJModelBuilder to implement integrated modelling and environmental monitoring. Furthermore, although open standards like OGC standards facilitate the interoperability, they do not solve the problem once for all. The scripting approach by exporting workflows to scripts and gluing various open source packages such as GRASS is promising and accommodated into GeoJModelBuilder. There are advantages for scripting languages to glue different components and models. Multiple geoprocessing algorithms and packages can be incorporated into GeoJModelBuilder by a scripting approach, thus realizing heterogeneous resources integration. This increases the capabilities of GeoJModelBuilder to access both Web and local resources when executing workflows. GeoJModelBuilder is implemented as a desktop tool. Its graphical user interfaces (GUI) are integrated with the NASA World Wind [15]. NASA World Wind is an open source virtual globe rendering engine, which allows users to quickly and easily create interactive visualizations of map and geographical information. GeoModelBuilder can operate on operating systems such as Windows or Unix/Linux.

The remainder of the paper is organized as follows. Section II introduces the implementation of GeoJModelBuilder, including relevant technologies, architecture and applications. The conclusion is provided in III.

Implementation

Relevant technologies

GeoJModelBuilder is an open source geoprocessing workflow tool, capable of supporting geoprocessing services, environmental monitoring, GRASS algorithms, and integrated environmental modelling (IEM). A set of open standards and technologies are leveraged to develop the software.

1. It adopts the OGC standards include Web Processing Service (WPS), Web Feature Service (WFS), Web Coverage Service (WCS) and Web Map Service (WMS). Sharing and interoperation of geospatial information and processing functions can be improved by using these services. By sending GetCapabilities requests to OGC WPS, WFS, and WCS services, GeoJModelBuilder is able to load all geospatial resources in the platform and facilitates the operation of dragging and dropping geoprocessing functions for users.
2. The workflow tool is coupled with the NASA World Wind, which supports the visualization of input data, results, sensors, and services in an interactive way. In GeoJModelBuilder, data layers in World Wind can be bound to workflows as input data. Execution results as maps can be visualized in World Wind. If provenance of workflows are recorded, the causal dependency connections between data layers in World Wind and workflows could be used to trace the lineage of data products.
3. Sensor Web Enablement (SWE) defines a series of standards for information models and service interfaces, such as Sensor Observation Service (SOS), Sensor Event Service (SES), Sensor Planning Service (SPS), and Web Notification Service (WNS). SOS, SES, SPS and WNS are applied in GeoJModelBuilder to support environmental monitoring. Abnormal observations will automatically trigger execution of workflows by judging consistency with presupposed criteria.
4. WebSocket is a computer communications protocol, providing full-duplex communication channels over a single Transmission Control Protocol (TCP) connection. The WebSocket protocol is able to make more interaction between a browser and a web server, which realizes a real two-way ongoing conversation [16]. WebSocket coupled with OpenMI instantiates MaaS approach as well as implements cooperation between components and services. The approach enables GeoJModelBuilder to integrate time-step based environmental models.

5. GRASS is a free and open source Geographic Information System (GIS) software commonly used for geospatial data management and analysis [17]. Incorporating GRASS components with GeoJModelBuilder enriches classes of geoprocessing functions and solves more complex geoprocessing problems.
6. The services and scripts are embedded into GeoJModelBuilder to execute geoprocessing algorithms in a given order and monitor execution situations in real time. The division of abstract and concrete layers of workflows allows the geoprocessing component could be instantiated using either services or scripts/commands. For example, the execution could be deferred to either services or GRASS scripts/commands to support the composition of both geoprocessing services and local software packages.

Architecture

The architecture of GeoJModelBuilder consists of four modules: resource management module, environmental monitoring module, geoprocessing workflow management module which includes workflow designing, binding and execting and data visualization and provenance module. Services and components can be dynamically bound to geoprocessing models in resources management module. Geoprocessing modeling and environmental monitoring can be achieved by the environmental monitoring module. Through dragging, dropping and linking geoprocessing functions on the operation panel, workflows can be generated friendly. The workflow models can be executed by workflow executors, which are able to execute both services and scripts. Map results can be visualized and traced by the data exhibition and provenance module. Figure 1 shows the architecture and interactions among these modules.

Service and component management

This module's functionality is to manage distributed services and components. OGC standard-compliant services such as WPS, WFS, and SOS, and local geoprocessing algorithms in GRASS can be viewed as fundamental blocks to construct workflows. In order to establish connections between local and distributed resources, messages are exchanged using the eXtensible Markup Language (XML). Variables to invoke GRASS scripts are extracted and described in XML files. Models, data and binding information are saved in different XML files in order to ensure flexibility.

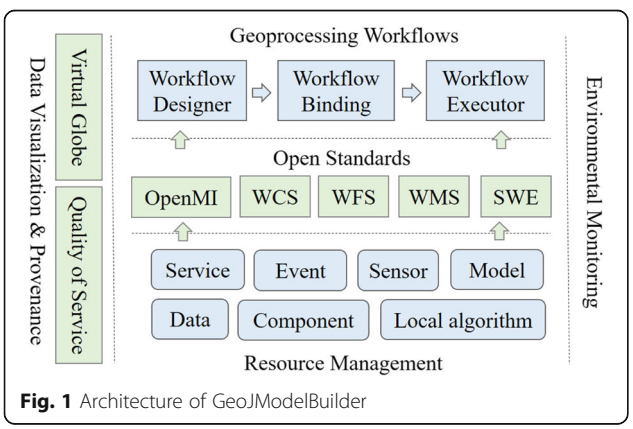

Fig. 1 Architecture of GeoJModelBuilder

Workflows in GeoJModelBuilder are designed as two layers, abstract and concrete layers, to seperate the business logics from resource usage. The workflow binding refers to the mapping from an abstract workflow to concrete resources, where underlying data and resources are bound to nodes in the abstract workflow. Each workflow node can be bound dynamically to specific services and components based on the type and parameter mapping. The seperation of the abstract and concrete layers has advantages of logical consistency, physical separation and dynamic adaptation.

Figure 2 shows the process to bind services and components. In the left panel, resources are generally divided into two parts, Processes and Events. Geoprocessing services or GRASS components have been bound to each geoprocessing nodes listed in hierarchical trees of the left panel by default. By clicking "Binding Model" and selecting specific models, users can select the other bindings and build up new bindings.

Environmental monitoring

The environmental monitoring module benefits from interoperable geospatial Web Services and OpenMI components. Versatile geospatial resources accessed through Web Service interfaces, and numerous environmental models following OpenMI can be plugged in GeoJModelBuilder flexibly. One core function of OpenMI is the ILinkableComponent interface. Components can be linked using the interface, which captures all information about the link between two linkable components. SWE-Standard services such as SOS, SES, SPS and WNS are implemented in GeoJModelBuilder in order to fulfill event-driven sensor planning and geoprocessing. The event-driven mechanism enables push-based active environmental monitoring and automatic dissemination of abnormal events. SOS plays the role of event producer, while SES plays the role of an event processing engine. By means of subscribing for an event, users will get the notification from WNS if abnormal observations occur, they can either choose to access existing observations or task sensor systems by a SPS for new observations.

Time-dependent values and runtime interactions are often required in environmental models. Time-step computations of models simulate time-dependent phenomena and need to interact continuously based on time-step

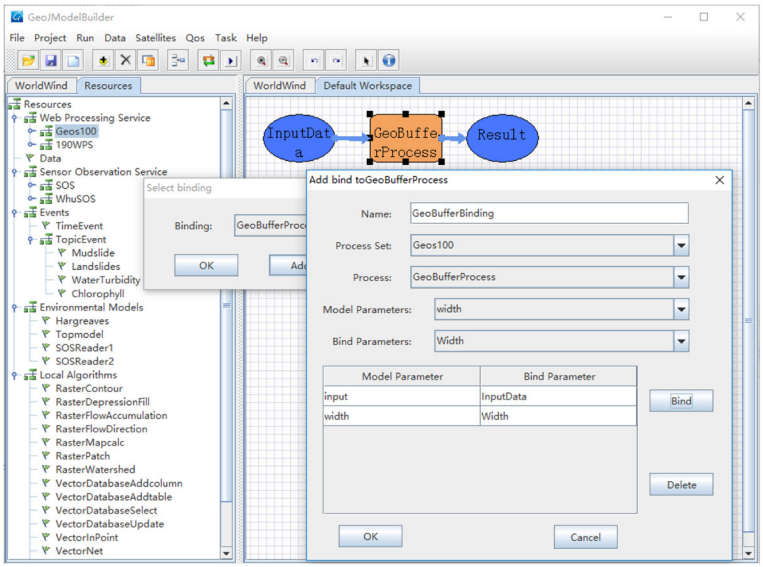

Fig. 2 Resources binding

computations during runtime. The HTTP protocol causes much overhead in the interaction of models on the Web. OpenMI components are published as WebSocket-based geoprocessing services, thus realizing the time-step computations and MaaS approach. Besides, a middleware is added to exchange data between services and components. In the environmental monitoring module, SWE services, WebSocket services and OpenMI-compliant models are incorporated into GeoJModelBuilder to implement integrated modelling and environmental monitoring.

Geoprocessing model designer

The geoprocessing model designer module provides users a graphic user interface (GUI) to compose distributed services and components, coupled with construct scientific workflows in a user-friendly way.

Activity is a functional unit in execution, which is abstracted as an Input-Process-Output (IPO) form. Activities are connected by data flows. Data flows not only reflect data exchanges, but also imply execution sequences of activities. For example, the data flow of activity A and activity B means that the output of activity A is the input of activity B. Besides, it also implies the execution sequence is from activity A to activity B. In GeoJModelBuilder, an atomic activity is generated by dragging and dropping a geospatial algorithm visually. Workflows are expressed using inputting data, parameters of processes, and linked activities in a specific order.

Workflow executor

The model executor module's functionality is to execute the geoprocessing workflow. It transforms abstract workflows into concrete workflows by binding data and geoprocessing utilities. The framework of model executor is showed in Fig. 3. It uses service executor when composes OGC standard services, and uses script executor when composes local geoprocessing components from either open source tools or commercial packages. Workflow execution engine in GeoJModelBuilder is implemented to execute workflows, monitor the runtime status, and record the provenance.

Data visualization and provenance

The aim of data visualization is to leverage NASA World Wind to visualize input data, results, sensors and geoprocessing services. In addtion to the tab used to construct workflows, there is another tab to show the virtual globe in GeoJModelBuilder in the right panel. Figure 4

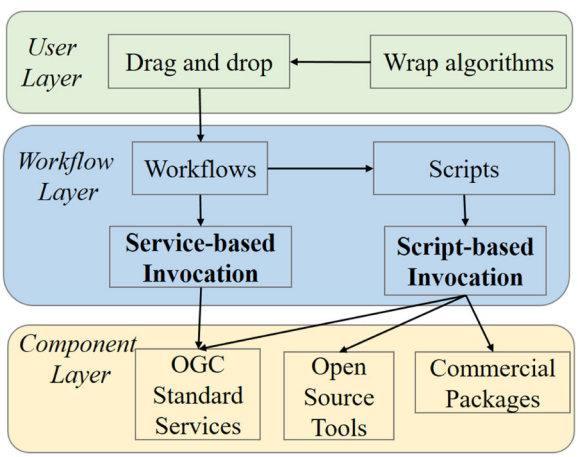

Fig. 3 Workflow execution mechanism

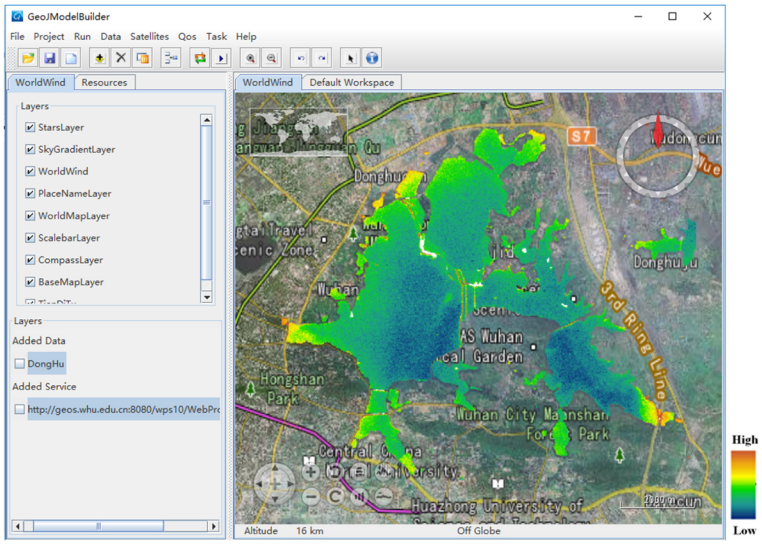

Fig. 4 Data visualization

shows the virtual globe and the geoprocessing result loaded into the virtual globe. The layers are listed in the left panel, which involves existed layers loaded into the NASA World Wind and additional layers loaded by users. Geospatial data, in-situ and remote sensors can also be visualized in the virtual globle.

If the abstract workflow is bound to specific services to generate a concrete workflow, it is convenient to check provenance of data and Quality of Service (QoS) in GeoJModelBuilder. General QoS attributes, QoS-aware optimization, and provenance checking are added into GeoJModelBuilder. Therefore, it allows to provide high quality of services for geospatial applications [7].

Results and discussion

The section introduces the integration method of geoprocessing models and application of the workflow system based on three specific cases. The aim of Case 1 is to implement turbidity extraction from a GF-1 remote sensing image of East Lake in Wuhan City, Hubei Province, China. It demonstrates the integration of geoprocessing web services in GeoJModelBuilder. Case 2 is on the IEM of TOPography hydrological MODEL (TOPMODEL) and Hargreaves model to predict watershed runoff. It uses the MaaS approach, thus illustrating the integration of time-step based model computations. The aim of Case 3 is to extract drainage networks from Digital Elevation Model (DEM) Data, and demonstrate the scripting approach, using GRASS algorithms as the processing units.

Use case 1

A number of placed water quality sensors in East Lake can continuously collect chlorophyll concentration and water turbidity data. When water turbidity exceeds the threshold, remote sensing observation covering East Lake is collected, a turbidity extraction processing flow is triggered automatically. Figure 5 demonstrates the whole process to extract turbidity and the implementation in GeoJModelBuilder. The observation imagery will go through orthorectification, radiometric calibration, atmospheric correction, clipping, Normal Differential Water Index (NDWI) calculation, mask building, silt inversion, and mapping processes to provide decision support on whether pollution happens or the range of pollution. These algorithms are accessible through interoperable services provided by GeoPW. It is possible to incorporate services in a distributed information infrastructure and improve the automatic discovery by using the service and workflow technologies.

Use case 2

TOPMODEL is a hydrologic model which can be leveraged to predict watershed runoff [18]. It takes topographic index, precipitation, and evapotranspiration as inputs. The topographic index can be obtained from Digital Elevation Model (DEM) using geospatial functions offered by GRASS, which can be exposed as a WPS. Hargreaves method is able to calculate evapotranspiration, which needs daily temperatures as time-dependent inputs [19]. Hargreaves model is implemented conforming to OpenMI-compliant components, while TOPMODEL is implemented conforming to WebSocket services. A middleware technology is used to bridge components and services, coupled with models, initialized according to the network address of specified TOPMODEL, the data address of topographic index and other model parameters. In the simulation, the middleware

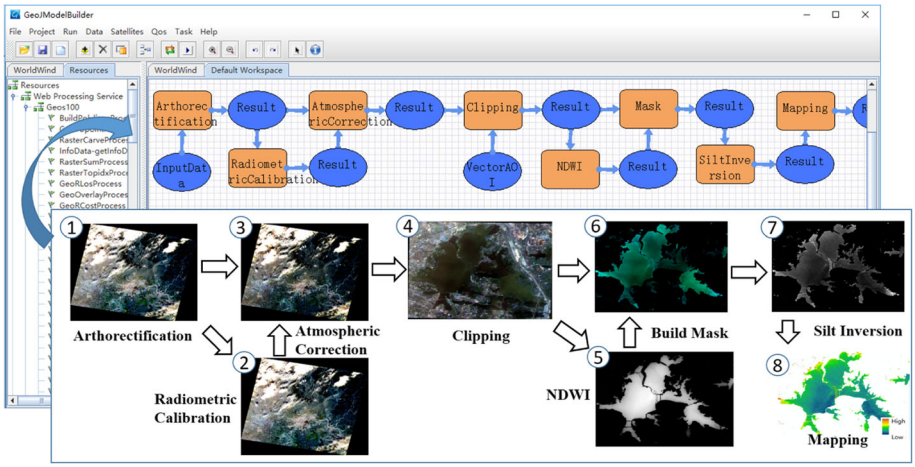

Fig. 5 The case on water turbidity

transfer precipitation and evapotranspiration to the TOPMODEL service according to time steps, and then acquire processing results. The constructed workflow and analysis results are demonstrated in Fig. 6. Data of this case is provided by CUAHSI Hydro-Desktop platform and from the year of 2008. The time step of the watershed runoff simulation is 1 day. Therefore, geospatial analysis functions, WebSocket services and OpenMI-compliant models are coupled together to implement integrated modelling and environmental monitoring in GeoJModelBuilder.

Use case 3

A hydrological analysis application of extracting drainage networks from DEM data is utilized to demonstrate how GeoJModelBuilder uses algorithms from GRASS as workflow components. GRASS provides Python interfaces that allow its algorithms to be called in a command-based style. GeoJModelBuilder adopts a scripting approach to integrate GRASS. As Fig. 7 shows, methods of depressions filling, flow direction calculation, flow accumulation, and drainage network extraction are selected to compose the workflow. These algorithms are accessible through local components provided by GRASS. The scripting approach follows a four-phase procedure: providing XML descriptions of hydrologic models in GRASS, constructing workflows of extracting drainage networks based on XML descriptions, exporting the workflow into a Python script, executing the script to get the final result. The procedures are integrated in GeoJModelBuilder. GeoJModelBuilder is able to execute python scripts to

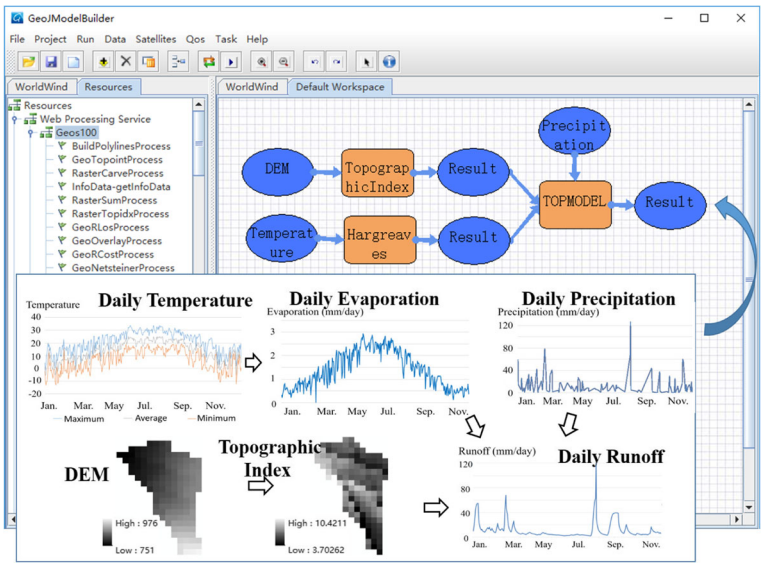

Fig. 6 The case on watershed runoff

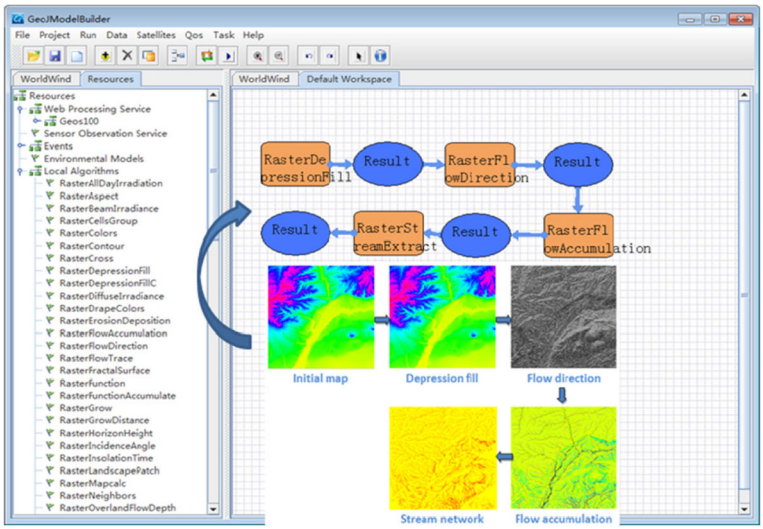

Fig. 7 The case on drainage extraction

invoke those algorithms in GRASS. The goal of utilizing a heterogeneous tool like GRASS is to enrich GeoJModelBuilder with more geoprocessing capabilities. In addition, the ability to support Python scripts could allow GeoJModelBuilder to "glue" more heterogeneous tools, which could be regarded an improved feature of GeoJModelBuilder.

Conclusions

This paper describes the architecture, methods, and application of GeoJModelBuilder. It is an open source workflow tool coupling geoprocessing Web Services, Sensor Web Services, local processing software, OpenMI-compliant models, and NASA World Wind to support geoprocessing modeling and environmental monitoring. The architecture of GeoJModelBuilder includes four modules: resource management module, environmental monitoring module, geoprocessing workflow module and data visualization and provenance module. It is designed as two layers consists of abstract and concrete layer to take advantages of resources for independence and dynamics adaption. Service integration, component integration, coupled with MaaS integration are supported in GeoJModelBuilder. Thus the tool provides a flexible, reusable, interoperable, and user-friendly way for geoscientific application in Cyberinfrastructure[1].

Availability and requirements

GeoJModelBuilder, which is available through sourceforge at http://sourceforge.net/projects/geopw. It is an open source software, and developed by Wuhan University. The software is written in JAVA and can run on Windows or Unix/Linux operating systems.

Endnote

[1]An online demo video is provided on YouTube (https://www.youtube.com/watch?v=BamSP8-al3Y).

Acknowledgements

We are grateful to anonymous reviewers for their constructive comments and suggestions. The work was supported by National Natural Science Foundation of China (91438203), Major State Research Development Program of China (2016YFB0502301), Hubei Science and Technology Support Program in China (2014BAA087), and Program for New Century Excellent Talents in University in China (NCET-13-0435).

Authors' contributions

PY designed the software architecture and methodologies. MZ is the leading developer to implement the software. XB implemented the scripting approach. MZ, XB, and PY wrote the paper together. PY acted as the corresponding author. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Author details

[1]School of Remote Sensing and Information Engineering, Wuhan University, 129 Luoyu Road, Wuhan, Hubei 430079, China. [2]Collaborative Innovation Center of Geospatial Technology, 129 Luoyu Road, Wuhan, Hubei 430079, China. [3]State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing (LIESMARS), Wuhan University, 129 Luoyu Road, Wuhan, Hubei 430079, China.

Received: 19 December 2016 Accepted: 1 April 2017
Published online: 10 April 2017

References

1. Yue P, Gong J, Di L, Yuan J, Sun L, Sun Z, Wang Q. GeoPW: laying blocks for the geospatial processing web. Trans GIS. 2010;14(6):755–72.
2. Yue P, Baumann P, Bugbee K, Jiang L. Towards intelligent GIServices. Earth Sci Inf. 2015;8(3):463–81.

3. Yue P, Guo X, Zhang M, Jiang L, Zhai X. Linked Data and SDI: The Case on Web Geoprocessing Workflows. ISPRS J Photogramm Remote Sens. 2016;114:245–57.
4. Zhang M, Yue P. GeoJModelBuilder: A java implementation of model-driven approach for geoprocessing workflows. In: InAgro-Geoinformatics (Agro-Geoinformatics), 2013 Second International Conference on, 08 October 2013. 2013.
5. Yue P, Zhang M, Tan Z. A geoprocessing workflow system for environmental monitoring and integrated modelling. Environ Model Softw. 2015;69:128–40.
6. Bu X, Yue P, Wang L, Zhang M. A scripting approach for integrating software packages and geoprocessing services into scientific workflows. In: Agro-Geoinformatics (Agro-geoinformatics), Fourth International Conference on, 2015. 2015.
7. Yue P, Tan Z, Zhang M. 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), 2014. 2014.
8. Yue P, Zhang C, Zhang M, Zhai X, Jiang L. An SDI Approach for Big Data Analytics: The Case on Sensor Web Event Detection and Geoprocessing Workflow. IEEE J Selected Topics Appl Earth Observations and Remote Sensing. 2015;8(10):4720–8.
9. Yue P, Ramachandran R, Baumann P, Khalsa S, Deng M, Jiang L. Recent Activities in Earth Data Science. IEEE Geoscience and Remote Sensing Magazine. 2016;4(4):84–9.
10. Nativi S, Mazzetti P, Geller GN. Environmental model access and interoperability: The GEO Model Web initiative. Environ Model Softw. 2013;39:214–28.
11. Geller GN, Turner W. The model web: a concept for ecological forecasting. In: IEEE International Geoscience and Remote Sensing Symposium, 2007. 2007.
12. Granell C, Díaz L, Gould M. Service-oriented applications for environmental models: Reusable geospatial services. Environ Model Softw. 2010;25(2):182–98.
13. Bastin L, Cornford D, Jones R, Heuvelink GB, Pebesma E, Stasch C, Williams M. Managing uncertainty in integrated environmental modelling: The UncertWeb framework. Environ Model Softw. 2013;39:116–34.
14. Moore RV, Tindall CI. An overview of the open modelling interface and environment (the OpenMI). Environ Sci Pol. 2005;8(3):279–86.
15. Pirotti F, Brovelli MA, Prestifilippo G, Zamboni G, Kilsedar E, Piragnolo M, Hogan P. An open source virtual globe rendering engine for 3D applications: NASA World Wind. Open Geospatial Data, Software and Standards. 2017. doi:10.1186/s40965-017-0016-5.
16. Hickson I. The websocket api. W3C Working Draft, 2011. 2011.
17. Geographic Resources Analysis Support System (GRASS) Software. Open Source Geospatial Foundation http://grass.osgeo.org. Accessed 12 Dec 2016.
18. Beven KJ, Kirkby MJ. 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. 1979;24(1):43–69.
19. Hargreaves GH, Samani ZA. Estimating potential evapotranspiration. J Irrig Drain Div. 1982;108(3):225–30.