# DIY GEOSPATIAL WEB SERVICE CHAINS: GEOCHAINING MAKE IT EASY

Huayi Wu[a], Lan You[b, a, *], Zhipeng Gui[a, c]

[a] LIESMARS, Wuhan University, 129 Luoyu Road, Wuhan, 430079, China
[b] School of Mathematics and Computer Science, Hubei University, 11 Xueyuan Road, Wuhan, 430062, China
[c] Center of Intelligent Spatial Computing for Water/Energy Science, 4400 University Dr., Fairfax, 22030, VA, USA

**Commission IV, WG IV/5**

**KEY WORDS:** Geospatial web service chain, Web Services Composition, Visual Modelling, Execution, Monitoring, WS-BPEL

**ABSTRACT:**

It is a great challenge for beginners to create, deploy and utilize a Geospatial Web Service Chain (GWSC). People in Computer Science are usually not familiar with geospatial domain knowledge. Geospatial practitioners may lack the knowledge about web services and service chains. The end users may lack both. However, integrated visual editing interfaces, validation tools, and one-click deployment wizards may help to lower the learning curve and improve modelling skills so beginners will have a better experience. GeoChaining is a GWSC modelling tool designed and developed based on these ideas. GeoChaining integrates visual editing, validation, deployment, execution etc. into a unified platform. By employing a Virtual Globe, users can intuitively visualize raw data and results produced by GeoChaining. All of these features allow users to easily start using GWSC, regardless of their professional background and computer skills. Further, GeoChaining supports GWSC model reuse, meaning that an entire GWSC model created or even a specific part can be directly reused in a new model. This greatly improves the efficiency of creating a new GWSC, and also contributes to the sharing and interoperability of GWSC.

## 1. INTRODUCTION

Following the rapid development of Services Oriented Architecture (SOA) and Web Services technology, the Open Geospatial Consortium (OGC) established the OGC Web Services framework (OWS) and a series of implementation specifications (e.g., GML, WMS, WFS, WCS, WPS, CSW). In addition, more and more stable geospatial web services are available online. In such circumstances, geographic information sharing and interoperability in distributed and heterogeneous computing environments is now possible. However, the functions provided by isolated services are limited. For more powerful functionality, these shared services require composition. The composition model of a series of distributed geospatial services working collaboratively to accomplish a complicated task with certain specifications is by calling a Geospatial Web Service Chain (GWSC). A GWSC can effectively solve the computationally intensive problems and complex procedure issues in Geographic Information Processing (GIP), with reduced development costs by promoting software reuse. GWSC accompanies the trend in GIP from traditional desktop-based stand-alone processing to collaborative distributed computing platforms and will likely become the mainstream solution for Distributed Geographic Information Processing (DGIP) in the near future (Yang, 2009).
In the IT field, various web services composition specifications have been proposed (e.g., WS-BPEL (OASIS, 2007), WS-CDL (W3C, 2004a), WSFL (IBM, 2001)). These specifications provide a good foundation for modelling GWSC, but the problem is that they are closely associated with IT implementations and focus too much on technological details. To create and utilize a composition model, users not only need to master professional GIS knowledge, but also have thorough knowledge of XML-related specifications (e.g., XPATH, XQuery, XSLT), web services specifications (e.g., WSDL, SOAP, UDDI, WSIF) and also understand composition specification itself, dramatically increasing learning costs and modelling difficulties, hindering the exploitation of GWSC technologies to some extent.

In recent years, a number of studies on GWSC modelling were published. Nadine (2003) and Friis-Christensen (2009) compared and analysed a variety of platform architecture models for GWSC modelling. With the bloom of semantic web and ontology studies, semantic reasoning and AI planning based automatic/semi-automatic modelling became a hot topic for research. Lemmens (2007) investigated how to establish low-level semantic description for automatic modelling GWSC. Lutz (2007) proposed an ontology and rules based semi-automatic modelling framework, creating GWSC models using regression planning. Others (Di, 2006; Yue, 2007; Chen, 2009) proposed ontology-driven modelling schemes, in which OWL-S (W3C, 2004b) based abstract models are created in semi-automatic ways, and transformed into WS-BPEL for execution ultimately. SWING (Andrei, 2008) is semantic based modelling framework, which cover the entire lifecycle of GWSC. SAW-GEO (Gobe, 2007) presents a prototype system, which combines together both the IT specification's visual modelling scheme and semantic reasoning scheme. But in terms of theoretical and technological maturity, semantic based schemes still do not provide practical solutions to cope with the fact that mass resources lack semantic description. Therefore, a manual visual modelling scheme is still the only viable solution at current stage. Furthermore, to assist users with different levels and professional background learn how to create and utilize GWSC quickly, such a platform should be established, one

* Corresponding author. Email: youlan1025@gmail.com, Tel: +86-27-88661740

where specification technical details are hidden, and integrates modelling, deployment, execution and all critical functions together.

Thus, this paper presents a new integrated GWSC modelling framework - GeoChaining. GeoChaining has following characteristics:

1. Functions are highly integrated, encompassing GWSC's entire lifecycle (i.e., conception, creation, deployment, execution, monitoring, and refinement).
2. Provides visual design and persistence functions, and therefore supports model reuse.
3. Integrates a services/data catalogue and clients, providing abundant resources, and so makes resource searching, publishing and modelling much easier.
4. Shows the GWSC's execution status dynamically, making status monitoring and error tracking more intuitive.
5. Integrates the data visualization function to exhibit raw data and data results in virtual earth, thereby making data checking and result analysis much easier.

The remainder of this paper is organized in 3 sections. Section 2 introduces the architecture of a GeoChaining platform; the components and their functions of each tier are presented in detail. Section 3 gives an example of GWSC modelling and execution using GeoChaining. Section 4 is a brief conclusion.

## 2. SYSTEM ARCHITECTURE

GeoChaining is designed with a three-tier architecture, i.e., with client, server and resource tiers, as illustrated in Fig. 1. The functions and collaboration of tiers are as follows:

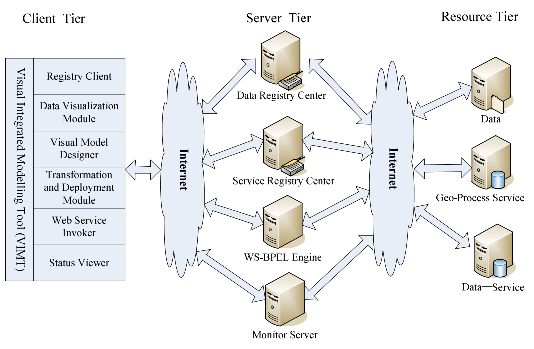


Figure 1. System architecture of GeoChaining

### 2.1 Client Tier: Visual Integrated Modelling Tool

The client tier is a Visual Integrated Modelling Tool (VIMT) for composing GWSCs, which integrates visual editing, validation, transformation, deployment and invocation functions together. The execution status dynamic display, geodata publication and visualization are also supported. Since a VIMT is the interactive bridge between the user and server side, abundant Graphic User Interfaces (GUIs) are provided. The six major modules of the VIMT are registry client, data visualization module, visual model designer, transformation & deployment module, web services invoker and status viewer.

**2.1.1 Registry Client**: The Registry client provides a resource view by interacting with data/services registry on the server tier, where all resources are organized and cached as a tree. The tree structure reflects the hierarchical classification of resources. When users find needed resources, they can use the resources as GWSC model elements by simply dragging related tree nodes into the visual editor. Since there is no need to search resources by using other tools, modelling time is reduced. Through the registry and upload functions, local data can also be published for processing and sharing. The tree view for the registry client is shown as area 1 in Fig.2.

**2.1.2 Data Visualization module**: Data visualization is critical for data analysis and the GIP workflow design. By integration with GeoGlobe (Gong, 2010; Wu, 2010), users can layer the raw and results geodata over the base map in a 3D virtual earth viewer. The data visualization GUI is shown as area 2 in Fig.2.

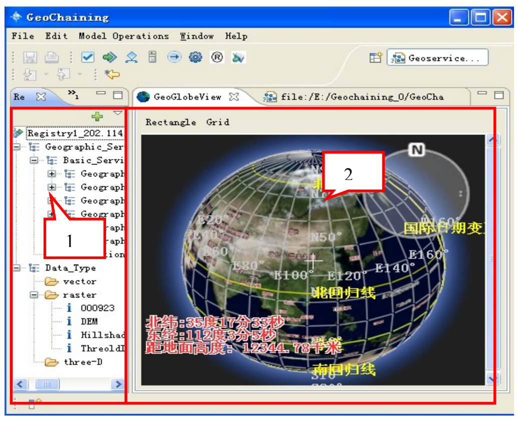


Figure 2. The GUI of registry client and data visualization

**2.1.3 Visual model designer**: In order to reduce the modelling GWSCs difficulty so users without workflow and web services related knowledge can also easily create models, GeoChaining adopts a simple data-flow based GWSC meta-model as model language, i.e., DDBASCM (Gui, 2008). Users can finish model design only using several simple "drag-and-drop" operations. These operations include model and model element creation, properties editing, validation, model diagram enhancement. The main part of the Designer GUI is illustrated in Fig. 3, where area 1 is model repository, area 2 is editor panel, area 3 is model element palette.

**2.1.4 Transformation & Deployment module**: Based on service description information of the deployed WS-BPEL process, the input parameters GUI are created dynamically. After correct input, the parameters are assembled into a web service request message and sent to the WS-BPEL engine for triggering GWSC's execution.

**2.1.5 Web Service Invoker**: For ensuring normative and interoperability, and also to promote GWSC model execution stability, GWSC models created in the design stage are transformed into executable WS-BPEL process models, and deployed into the WS-BPEL engine for execution and management.

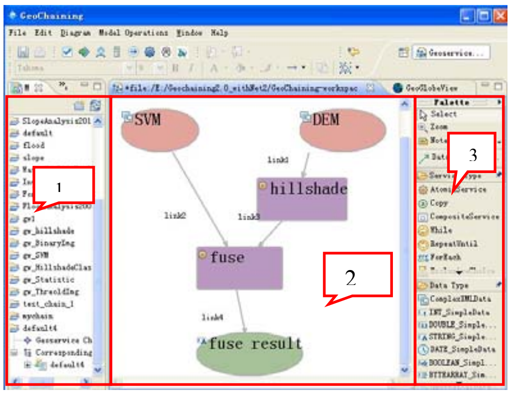


Figure 3. GUI of Visual Model Designer with demo model

**2.1.6 Status viewer**: This module is in charge of GWSC execution status dynamic exhibition. Using the instance ID of the activated WS-BPEL process, the status viewer acquires the real time execution status for all component services from the status monitor, and displays them in the GUI.

### 2.2 Server Tier

The Server tier consists of a data registry, services registry, WS-BPEL engine, and an execution status monitor. The Data/services registries manage metadata of registered resources. They provide query and register interfaces, as well as a data storage function. The WS-BPEL engine controls GWSC execution, and is in charge of communication with component services. The execution status monitor checks the execution status of all component services.

**2.2.1 Data registry**: This module provides query and publish interfaces. Through a VIMT, users can query registered data and also register and upload their private data onto the server as data resources.

**2.2.2 Services registry**: This module was developed by extending freebXML (an open source ebXML registry implementation, http://ebxmlrr.sourceforge.net). The CSW interfaces are implemented in the registry. The services metadata are organized based on ISO 19119 classification system.

**2.2.3 WS-BPEL engine**: We use the open source ActiveBPEL engine (http://www.activevos.com/) as the WS-BPEL engine, which handles the execution and management of the deployed WS-BPEL processes.

**2.2.4 Execution status monitor**: Although the ActiveBPEL engine provides some statuses (Faulted, finished, executing, ready_to_execute) for the WS-BPEL process and its component services, these statuses are coarse, especially for services with long running time. In order to provide more detailed progress information as a basis for GWSC model exception capture and optimization, the status monitor was developed. The Status monitor gets the execution status by invoking the status interface provided by the services, which means the services themselves must be measurable.

### 2.3 Resource Tier

The Resource tier contains all data, data/processing services either registered in registry or not. However, in order to be found and utilized easily, all resources must be registered.

The general modelling procedures are as follows: in VIMT, users query geodata and geoservices from the data/services registry, and then use these resources to visually design their GWSC models. After correctness validation, GWSC models are transformed and deployed in the WS-BPEL engine on the server tier. When invocation is triggered by the client, a currently deployed WS-BPEL process for a certain GWSC model is instantiated and executed. During this stage, execution status and progress details of all the component services are dynamically monitored and fed back into the VIMT. After successful execution, users can load and visualize the data result in VIMT just using one-click.

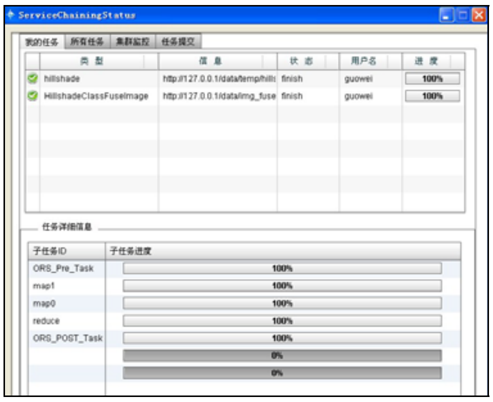
## 3. EXPERIMENT

In order to validate the feasibility of GeoChaining platform's design, we design a GWSC model about 3D features classification and rendering analysis. This GWSM model contains two component services, i.e., a hill shade rendering service and a 3D feature classification and fusing service. Input raw data are DEM and classified image, while the output data result is the fused image, showing classification and rendering results for 3D features. In this experiment as the study area, we use data covering the Poyang lake area. The GWSC model diagram created is shown in Fig. 3, where two services are expressed as purple rectangles, raw data and data results are expressed as pink ellipses and a green ellipse respectively.

The modelling and execution procedures are as follows:

1. Create a new GWCM model file, and set model name.
2. Query resources from registry using registry client, and drag tree nodes stand for two demo services from the registry client tree viewer into visual model designer, and set the names shown on diagram.
3. Drag two demo data sets into visual model designer, and set names;
4. Drag the model element representing the geodata from model element palette into the visual model designer as the result data node, and edit the name.
5. Establish the data dependency relationship between services and data by directly connecting them.
6. Validate, transform and deploy the GWSC model.
7. Invoke GWSC using the web service invoker.
8. Monitor execution status, as illustrated in Fig. 4.
9. Visualize the data results, as shown in Fig. 5.

From the experiment, we see that the entire modelling procedures are simple, all functions are easy to operate, and the user experience is rich. For modelling with the GWSC model, users only need to know what functions are provided by selected services and what data they need or generate, and the degree of matching between selected data and services.



| | 类型 | 信息 | 状态 | 用户名 | 进度 |
|---|---|---|---|---|---|
| ✓ | hillshade | http://127.0.0.1/data/temp/hill | finish | guowei | 100% |
| ✓ | HillshadeClassFuseImage | http://127.0.0.1/data/img_fuse | finish | guowei | 100% |

| 子任务ID | 子任务进度 |
|---|---|
| ORS_Pre_Task | 100% |
| map1 | 100% |
| map0 | 100% |
| reduce | 100% |
| ORS_POST_Task | 100% |
| | 0% |
| | 0% |

Figure 4. Execution status of demo model

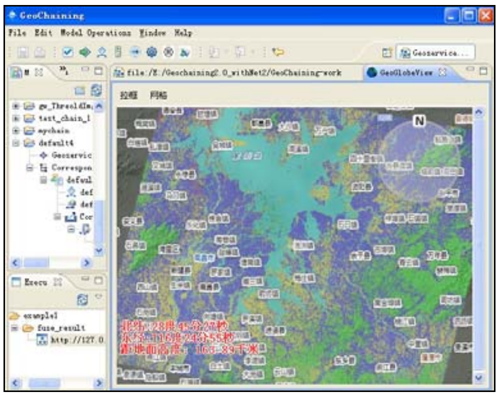


Figure 5. Data results visualization

## 4. CONCLUSION

Modelling GWSC by directly using IT specifications has a high learning threshold, even with a visual editor's help. In general, GWSC creators and users must spend lots of time studying related knowledge and specifications before modelling. Additionally, a series of independent software products also must to be mastered and utilized to view and manage events in entire lifecycle of a GWSC. These issues severely hinder GWSC technologies development and widespread application.

GeoChaining is a feasible solution. Encompassing the entire GWSC lifecycle, the various functions are highly integrated, including resource query, visual design, validation, deployment, execution, status monitoring, data publishing and visualization. Technical specification details are hidden, solving the complexity problem that inhibits non-specialist use. Users with different backgrounds and levels of expertise can now easily design, deploy and execute their own GWSC models in the same GUI by using simple "drag-and-drop" and "one-click" operations. Furthermore, users can intuitively monitor execution in real time way and visualize geodata in 3D virtual earth. Thus, potential problems and bottle-necks can be easily identified, an important factor for further model optimization. GeoChaining can significantly lower the learning threshold and improve modelling efficiency by reducing the extra effort to find modelling resources and learning related software and specifications.

## 5. ACKNOWLEDGEMENT

The work described in this paper is supported by the Natural Science Foundation of China under grant 40971211, 41021061 and 41023001. Thanks to Mr. Steve McClure for language assistance.

## 6. REFERENCES

Andrei, M. et al., 2008. SWING: An Integrated Environment for Geospatial Semantic Web Services. In: *ESWC 2008*. LNCS 5021, S. Bechhofer, Ed., ed.: Springer: Berlin, pp. 767-771.

Chen, J., et al., 2009. Use of grid computing for modeling virtual geospatial products. *International Journal of Geographical Information Science*, 23(5), pp. 581-604.

Di, L.,et al., 2006. Ontology-driven automatic geospatial-processing modeling based on web-service chaining. In: *Proceedings of the Sixth Annual NASA Earth Science Technology Conference*, 7pp.

Friis-Christensen, A., et al., 2009. Service chaining architectures for applications implementing distributed geographic information processing. *International Journal of Geographical Information Science*, 23(5), pp. 561-580.

Gobe, H., et al.. 2007. Semantically-assisted geospatial workflow design. In: *Proceedings of the 15th international symposium on Advances in geographic information systems*, Seattle, Washington, 7pp.

Gong, J., et al., 2010. Multi-source geospatial information integration and sharing in Virtual Globes, *Science China - Technological Sciences*, 53:1-6 Suppl.

Gui, Z., et al., 2008. A data dependency relationship directed graph and block structures based abstract geospatial information service chain model. In: *Proceedings 4th International Conference on Networked Computing and Advanced Information Management*, IEEE Computer Society, Gyeongju, Korea, pp. 21-27.

IBM, 2001. Web Services Flow Language (WSFL 1.0). http://www.itee.uq.edu.au/~infs7201/Assessments/Assignment%201%20Material/WSFL.pdf

Lemmens, R. et al., 2007. Enhancing Geo-Service Chaining through Deep Service Descriptions. *Transactions in GIS*, 11(6), pp. 849-871.

Lutz, M. et al., 2007. A Rule-Based Description Framework for the Composition of Geographic Information Services. *GeoSpatial Semantics 2007*. LNCS 4853, Springer: Berlin, pp. 114-127.

Nadine, A., 2003. Chaining geographic information web services. *IEEE Internet Computing*, 7(5), pp. 22-29.

OASIS, 2007. Web Services Business Process Execution Language Version 2.0 (WS-BPEL 2.0). http://docs.oasisopen.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

W3C, 2004a. Web Services Choreography Description Language Version 1.0. In: N. Kavantzas, D. Burdett, G. Ritzinger. (Eds.) W3C. http://www.w3.org/TR/ws-cdl-10/

W3C, 2004b. OWL-S: Semantic Markup for Web Services. http://www.w3.org/Submission/OWL-S

Wu, H., et al., 2010. A Virtual Globe-based 3D Visualization and Interactive Framework for Public Participation in Urban Planning Processes, *Computer, Environment and Urban System*, vol.34. no.4, pp.291-298.

Yang, C., et al., 2009. Introduction to distributed geographic information processing research. *International Journal of Geographical Information Science*, 23(5), pp. 553-560.

Yang, C., et al., 2011. Using spatial principles to optimize distributed computing for enabling the physical science discoveries, *Proceedings of National Academy of Science (PNAS)*, 108(14), pp.5488-5491

Yue, P. et al., 2007. Semantics-based automatic composition of geospatial Web service chains. *Computer & Geosciences*, 33(5), pp. 649-665.