International Journal of Geo-Information

MDPI

Article

A Web Service-Oriented Geoprocessing System for Supporting Intelligent Land Cover Change Detection

Huaqiao Xing [1,2,*], Jun Chen [2,*], Hao Wu [2] and Dongyang Hou [3,4]

1 School of Surveying and Geo-Informatics, Shandong Jianzhu University, Jinan 250101, China
2 National Geomatics Center of China, Beijing 100830, China; wuhao@nsdi.gov.cn
3 School of Geosciences and Info Physics, Central South University, Changsha 410083, China; houdongyang1986@163.com
4 College of Geography and Environment, Shandong Normal University, Jinan 250014, China
* Correspondence: xinghuaqiao@126.com (H.X.); chenjun@nsdi.gov.cn (J.C.)

Received: 5 November 2018; Accepted: 16 January 2019; Published: 20 January 2019

Abstract: Remotely sensed imagery-based change detection is an effective approach for identifying land cover change information. A large number of change detection algorithms have been developed that satisfy different requirements. However, most change detection algorithms have been developed using desktop-based software in offline environments; thus, it is increasingly difficult for common end-users, who have limited remote sensing experience and geographic information system (GIS) skills, to perform appropriate change detection tasks. To address this challenge, this paper proposes an online geoprocessing system for supporting intelligent land cover change detection (OGS-LCCD). This system leverages web service encapsulation technology and an automatic service composition approach to dynamically generate a change detection service chain. First, a service encapsulation strategy is proposed with an execution body encapsulation and service semantics description. Then, a constraint rule-based service composition method is proposed to chain several web services into a flexible change detection workflow. Finally, the design and implementation of the OGS-LCCD are elaborated. A step-by-step walk-through example for a web-based change detection task is presented using this system. The experimental results demonstrate the effectiveness and applicability of the prototype system.

Keywords: land cover; intelligent change detection; web-based geoprocessing system; web service encapsulation; automatic service composition

1. Introduction

It is important to accurately acquire land cover change (LCC) information in a timely manner in order to gain an improved understanding of the environmental changes and interactions between humans and environmental systems [1,2]. With the development of free access policies to remotely sensed imagery (e.g., the Landsat series), multitemporal imagery-based change detection has become an effective approach to identify LCC information [3,4]. In recent years, research on change detection has attracted considerable attention, and many new algorithms or models have been developed and merged. However, most change detection algorithms have individual advantages and disadvantages; thus, it is difficult for users to construct suitable change detection workflows to address specific requirements [5]. More importantly, traditional change detection approaches have often been conducted using desktop-based software (e.g., ENVI, ERDAS, and ArcGIS) in offline environments, which have limitations such as being laborious, inefficient, and time-consuming [6,7]. It is increasingly difficult for common end-users, who have limited remote sensing experience and geographic information system (GIS) skills, to conduct an appropriate change detection approach,

especially over a large area [7,8]. Therefore, it is urgent to develop advanced web-based tools or systems to support intelligent land cover change detection.

In recent years, with the emergence and growing maturity of service-oriented architecture (SOA) and service-oriented computing (SOC) technologies, an increasing number of remotely sensed datasets, processing algorithms, and models have been available and accessible as web services for offering geoprocessing functionality online. Workflow technology has also been widely used to chain atomic web services together to realize complex geoprocessing tasks [9–13]. Compared with the desktop-based geoprocessing approach, online geoprocessing has many strengths, including making functionality accessible to larger user groups, automating recurring tasks, and sharing workflows [6,14]. Additionally, online geoprocessing has the potential to make spatial analysis near-real-time, efficient, and cost-effective. Based on SOA and SOC, a number of web service-oriented geoprocessing systems have been developed over the past decades to support land cover data processing and analysis. For example, Karantzalos et al. (2015) designed and developed a land cover classification web service to efficiently and automatically process high-resolution satellite data [15]. However, a specific web-based land cover change detection system has not yet been fully investigated and implemented. In addition, land cover change detection is a comprehensive procedure that requires careful consideration of the characteristics of selected remotely sensed data and landscape complexity of the studied areas. Despite the large and great number of change detection algorithms and models that have been developed in recent years, no single approach is applicable to all types of imagery, land cover, and geographic regions. After encapsulating the change detection algorithms or models into web services, it is also necessary to provide an appropriate change detection service chain for different conditions.

To address the above challenges, this paper reports on a research effort to develop a web-based system for supporting intelligent and flexible land cover change detection. The proposed system leverages web service encapsulation technology and applies an automatic service composition approach to dynamically generate a suitable change detection service chain. Heterogeneous change detection-related algorithms are encapsulated into web services using a black box approach and semantic descriptions. Then, a constraint-rule-based service composition method is proposed to chain several web services into a flexible change detection workflow. The system provides a convenient interface that enables end-users to upload their remotely sensed data and select change detection requirements. A suitable change detection service chain can be generated and executed using the proposed method, which is integrated into the system. The major advantage of this system is that all of the processing steps can be automatically operated in an open web environment, which allows both professional and nonprofessional users to conduct land cover change detection tasks without installing any desktop-based software.

The remainder of this paper is organized as follows: Section 2 reviews the related works. Section 3 presents the web service encapsulation strategy and the automatic service composition approach. Section 4 introduces the architecture and implementation of the web service-oriented land cover change detection system, and provides a step-by-step walk-through example of using this system. Section 5 presents the evaluations and discussions of the prototype system. Finally, conclusions and future work are given in Section 6.

2. Related Works

2.1. Land Cover-Related Online Geoprocessing Systems

With the development and deep application of the Open Geospatial Consortium (OGC) services, multiple land cover data browsing and downloading services have been made available by scientific institutions and communities. For example, the National Geomatics Center of China (NGCC) has published 22 land cover web services for data browsing, including GlobeLand30 data for the years 2000 and 2010, as well as 10 single land cover types for these two years [16,17]. Additionally, NGCC provides a land cover data downloading service that supports three methods for filling in map sheet

numbers, spatial coordinate ranges, and drawing spatial graphics. As of January 2018, approximately 8000 users from 120 countries have downloaded GlobeLand30 data for use in their own applications using the data browsing and downloading service [18].

The current land cover-related web services are mainly data services that facilitate access to land cover data; they cannot be used directly for online processing or analysis. Nevertheless, an increasing number of users have a strong desire to conduct land cover geoprocessing tasks by invoking existing geoprocessing services and constructing service chains over the Internet. To meet the advanced requirements, some web-based geoprocessing systems have been developed that provide online processing functions or tools to support web-based land cover data processing. These systems are compared and described in Table 1.

Table 1. A detailed list of existing land cover-related online geoprocessing systems and tools.

System/Tool Name	Provided Functions	Development Technologies	URL	Reference Paper
GlobeLand30 Production	Land cover data production	**--Browser side:** Openlayers, jQuery, Ext **--Server side:** PostgreSQL/PostGIS APOLLO Server, PHP, Geoserver, C#	www.globeland30.org	Han et al. (2015) [19]
GlobeLand30 validation	Land cover validation		www.glcval.geo-compass.com	Chen et al. (2016) [20]
GlobeLand30 tagging	Land cover tagging		www.globeland30.org/biaobao/default.aspx	Xing et al. (2015) [21]
GlobeLand30 statistics	Land cover statistics		www.globeland30.org/chinese/stat/index.html	Li et al. (2016) [22]
CropScape	Land cover browsing, statistics	**--Browser side:** Openlayers, Extjs, Ajax-powered rich Internet application	www.nassgeodata.gmu.edu/CropScape	Han et al. (2012) [23]
GeoWiki	Land cover validation	**--Browser side:** Openlayers, Google Earth APIs **--Server side:**PHP	www.geo-wiki.org	Steffen et al. (2012) [24]
LACO-Wiki	Land cover validation	**--Browser side:** Openlayers **--Server side:** ASP.NET, C#, PostgreSQL, Geoserver, GDAL/OGR library	www.laco-wiki.net	Linda et al. (2017) [25]
VIEW-IT	Land cover tagging	**--Browser side:** ArcGIS JavaScript API **--Server side:** ArcGIS Server, PHP, MySQL	Not found	Clark et al. (2011) [26]
Web-based land cover validation tool	Land cover validation	**--Browser side:** Openlayers **--Server side:** IDL, PostGIS, GeoServer	www.landcover-change.jrc.ec.europa.eu/validation/videos/Birdlife_editor.html	Bastin et al. (2013) [27]
Geospatial service for land cover mapping	Land cover mapping	**--Browser side:** Openlayers, GeoExt **--Server side:** Orfeo Toolbox, OpenCV, LibSVM Rasdaman database	Not found	Karantzalos et al. (2015) [15]

These systems in Table 1 address many fields of land cover processing, including web-based land cover mapping, validation, geo-tagging, and statistics. However, a specific geoprocessing system for web-based land cover change detection has not been fully discussed and developed. From the aspect of development technologies, open source technologies and software (e.g., Openlayers, GeoServer, and PostGIS) have often been used for system development because they are free and easily accessible. However, these systems were often focused on specific applications; thus, they provided relatively

fixed service chains or workflows rather than appropriate and intelligent processing solutions for different user requirements or conditions.

2.2. Current Efforts in Geoprocessing Service Composition

To meet complex geoprocessing user requests, a collection of geoprocessing services must be composited into a service chain or workflow when no single service can fulfill those requests [28–30]. In general, current research efforts in geoprocessing service composition can be summarized into two categories: manual service composition and automatic service composition.

In the manual service composition approach, an abstract business workflow model is firstly designed by a service provider in advance utilizing a web service standard language, such as Web Services Business Process Execution Language (WS-BPEL) [31,32]. Then, concrete services are manually selected and bound into a workflow based on the abstract model. Finally, the concrete workflow is sent to a workflow execution engine to generate the final results. A typical example of this is GeoBrain [12], a web-based geospatial information service based on SOA technology in which human users can define, build, and manage geo-service workflows, and translate abstract workflows into a specific service chain instance by binding the related services. Manual service composition methods work well when the service functionalities, data conditions, and user requirements are relatively fixed. However, when the services in the workflow become unavailable or user requirements change frequently, the manual approach will not be able to generate a sufficiently correct service chain. Thus, these methods often lack flexibility, and are difficult to adapt to dynamic environments. Furthermore, the manual service composition approach is largely dependent on expert knowledge. It is difficult for common users to design abstract workflow models and bind the corresponding concrete web services.

In contrast, automatic service composition methods support the automatic discovery, selection, and binding of composite services [9]. By providing a set of component services and a specified requirement (e.g., user request), a suitable service chain can be generated automatically with less human intervention, by using the semantic web [9,33], artificial intelligence (AI) planning techniques, or search technologies [34,35]. Automatic service composition methods have the potential to achieve flexible and adaptable applications by selecting and combining appropriate components based on the user's request and context. For instance, Yue et al. (2007) proposed an approach for automatic geospatial web service composition by employing ontology-based geospatial semantics for enabling the automatic discovery, access, and chaining of geospatial web services [36]. Tan et al. (2015) presented a cloud and agent-based approach for the automatic and intelligent construction of a geospatial service chain in a cloud environment [37]. However, the current studies have focused primarily on the process of automatic service discovery and selection, and have paid less attention to whether the generated service chain could be correctly and successfully executed.

To summarize, the automatic service composition approach is an ideal solution for selecting and composing web services in a dynamic environment. Despite the advantages of the automatic approach, the implementation process is challenging, and a few important issues must be addressed. On one hand, the semantic description of geoprocessing services must be machine readable to allow automatic service identification and utilization. On the other hand, the algorithms for the automatic composition of geoprocessing services must be specified, which should guarantee that the generated service chain could be executed correctly.

3. Methodology

To meet the requirements of intelligent web-based change detection, heterogeneous algorithms and models should first be encapsulated into web services with semantic descriptions; then, these services should be automatically composited into an appropriate service chain or workflow, in order to address user-specific requirements or conditions.

3.1. Heterogeneous Service Encapsulation Strategy

With the help of current OGC standards, geospatial data, processing algorithms, or even computing resources could be encapsulated and published as reusable web services that facilitate end-users to derive information by compositing them into complex workflows over an open and distributed environment [38]. In this paper, we mainly focus on studying the process of algorithm encapsulation.

Unlike general web services, change detection-related web services have unique heterogeneity domain characteristics. First, the input data of change detection services can be from different remote sensors (e.g., Landsat or MODIS) that have different radiometric, spectral, spatial, and temporal resolutions. Second, it is widely accepted that no single change detection algorithm exists that is applicable to all cases. Therefore, a change detection service often has tightly coupling and specific input requirements. Third, change detection algorithms are usually written in different programming languages and libraries (e.g., C#, IDL or GDAL), and run on different operating systems (e.g., Windows or Linux). As a result, execution of a change detection service has special constraints.

Considering the characteristics of change detection web services, this paper discusses the specification and design of two important steps for heterogeneous web services encapsulation—execution body encapsulation and service semantics description—which are shown in Figure 1.

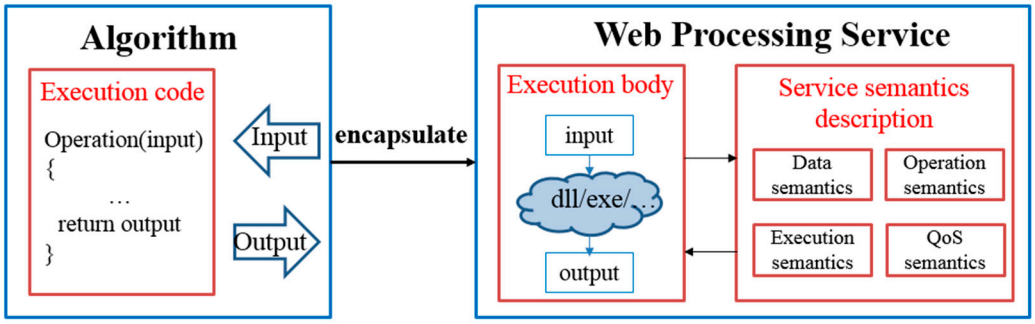

Figure 1. Process of encapsulating change detection algorithms into web processing services.

First, the 'black box' theory is introduced for execution body encapsulation. A black box is a method that can be used without the users knowing how its inner algorithm works [39,40], and the users only need to know the input and output characteristics. Using this approach, the executable code written in any programming language can be encapsulated into a corresponding component object model (COM), such as a dynamic link library (DLL) or an executable program (EXE) that exposes only the interface of the input and output data and hides all of the implementation details.

To facilitate automatic service invocation and composition for change detection, it is important to represent the interface, functionality, constraint, and quality semantics of the change detection web services. In this paper, the service semantics were classified into the following four types: data semantics, operational semantics, execution semantics, and quality of service (QoS) semantics [36]. Data semantics annotate the semantics of the input/output interface in a web service. Operational semantics represent the semantics of service functionality or logical relations with other services. Execution semantics specify the constrained requirements of a service (e.g., the preconditions and effects). QoS semantics consist of a set of metrics for evaluating which service should be selected.

3.2. Constraint Rule-Based Automatic Service Composition

Land cover change detection is a complex geoprocessing task involving several computation steps, such as preprocessing, change information extraction, and postprocessing. Therefore, a collection of interoperable web services should be composited to generate a service chain to realize a complete change detection task.

Automatic service composition is an ideal approach, because the composition process can be operated under the control of computers with minimal manual intervention. The mainstream methods of automatic service composition were proposed based on a data semantic matching approach [41,42]. Assume that the user requirements and web services are represented as RE = <$input_R$, $output_R$> and WS = <input, output>, respectively. The process of automatic service composition can be regarded as a service chain search from $input_R$ to $output_R$. Two semantic matching rules are usually adopted to evaluate the matching degree between a web service and the input or output data. These two rules can be formal described using Semantic Web Rule Language (SWRL), as shown in Table 2.

Table 2. Examples of semantic matching rules and their formal description. SWRL: Semantic Web Rule Language.

Constraint Rules	SWRL-Based Formal Description
hasInputRule	Service:presents (? Service, ?profile)∩ Profile:hasInput(?profile,? input)∩ Process:paraType(?input,? input_req)∩ rdf:type(? input, owl: class)→ rule: hasInputRule (? Service, ? input_req)
hasOutputRule	Service:presents (? Service, ?profile)∩ Profile:hasInput(?profile,? output)∩ Process:paraType(?input,? output_req)∩ rdf:type(?output, owl: class) → hasOutputRule (? Service, ? output _req)

As discussed in Section 2.2, the data semantic matching-based automatic service composition methods are designed only at the conceptual level. Thus, the generated service chain might not be correctly executable, because the constraint conditions during the actual execution are not completely considered. Taking a spectral gradient difference (SGD) service (encapsulated from the SGD algorithm [43]) as an example, the input data type of the SGD service must be remotely sensed imagery. One constraint condition of the SGD service is that the input data format is supposed to be 'GeoTIFF'; that is to say, the SGD service chain might not be correctly executed if the input imagery data format is 'IMG'. To improve the robustness and correctness of the generated service chain during the actual execution process, the data semantic matching-based method is extended by judging whether the input data satisfy the constraint conditions of each atomic service in the generated service chain, as shown in Figure 2.

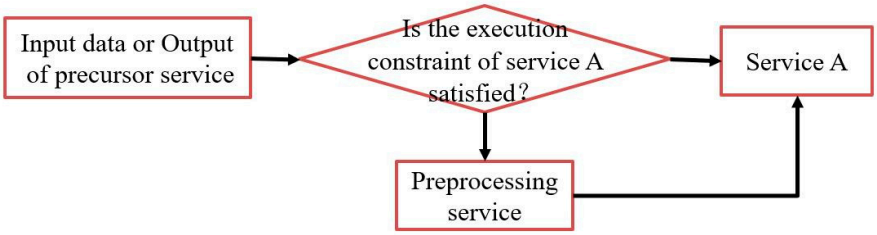

Figure 2. Preprocessing service selection to satisfy the execution constraint of service A.

The improved automatic service composition method includes two steps: constraint rules construction and condition judgements between the service and its input or output. In this paper, the constraint conditions of a service are considered to be data inconsistency. Data format inconsistency, coordinate system inconsistency, and resolution inconsistency are the three main categories of data inconsistency. To eliminate these data inconsistencies automatically, the following three sets of constraint rules were constructed: DFinconRules, CSinconRules, and ReinconRules. SWRL is then used for the formal description of these constraint rules, as shown in Table 3.

Table 3. SWRL-based formal description of constraint rules.

Constraint Rules	SWRL-Based Formal Description
DFinconRules	hasInputRule (?Service, input)∩ rdf:format(?input, format$_i$) ∩ rdf:format(?data, format$_j$)→ rule:DFinconRules (?Service, ?data)
CSinconRules	hasInputRule (?Service, input)∩ rdf:coordinate (?input, coordinate$_i$) ∩ rdf:coordinate (?data, coordinate$_j$) → rule:CSinconRules (?Service, ?data)
ReinconRules	hasInputRule (?Service, input)∩ rdf:resolution (?input, resolution$_i$) ∩ rdf:resolution (?data, resolution$_j$) → rule:RinconRules (?Service, ?data)

With the help of the constraint rules, a robust and executable service chain can be generated by judging whether the input data constraints are satisfied for a specific service. If the input data are inconsistent with the service, corresponding preprocessing services of data format conversion (DFC), coordinate systems conversion (CSC), and resolution conversion (ReC) are selected to guarantee the correct execution of the service chain. The pseudocode of the constraint rule-based automatic service composition process is shown in Algorithm 1.

Algorithm 1. WSChainbyRule (D_{init}, D_{req})

Input: D_{init}, D_{req}; **Output:** *WSChain*
SET D_{temp}=D_{init} *WSChain* =[]
WHILE($D_{temp}D_{req}$){
  FOR EACH ws_i IN *WSList*{
    IF (D_{init} satisfied *hasInputRule*){
      IF (D_{init} is satisfied *DFinconRules*){
        WSChain += *DFC*
      }ELSE IF (D_{init} is satisfied *CSinconRules*){
        WSChain += *CSC*
      } ELSE IF (D_{init} is satisfied *ReinconRules*){
        WSChain += *ReC*
      }
    WSChain +=ws_i
      IF (D_{req} satisfied *hasOutputRule*){
        RETURN *WSChain*
      }ELSE{
        D_{temp}=ws_i.*output*
      }}}

4. System Architecture and Implementation

4.1. Architecture Design

Based on SOA and SOC principles and standards, an online geoprocessing system for land cover change detection (OGS-LCCD) is designed and developed to meet the requirements of intelligent change detection. As a subsystem of GlobeLand30 (www.globeland30.org) [18], the main function of OGS-LCCD is to facilitate professional and nonprofessional end-users to conduct web service-based change detection tasks without using any proprietary desktop-based software. The OGS-LCCD has a

service-oriented architecture, which contains the following four layers: a resource layer, a service layer, a logic layer, and an application layer, as shown in Figure 3.

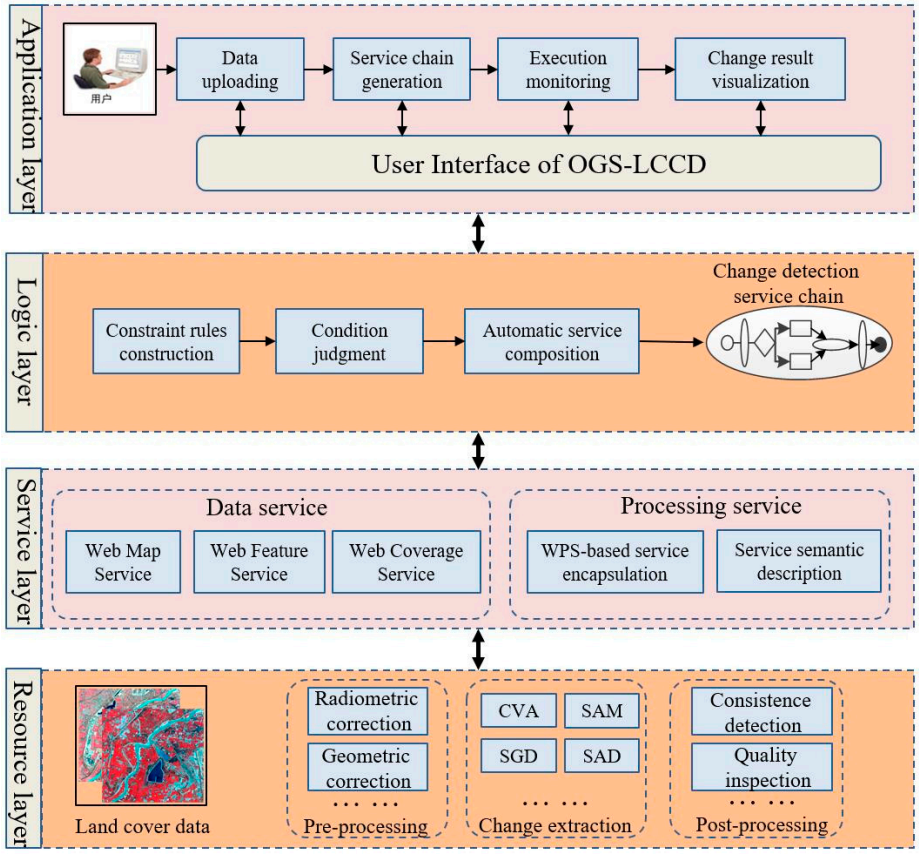

Figure 3. Online geoprocessing system for supporting intelligent land cover change detection (OGS-LCCD) architecture.

The resource layer lies at the bottom of the OGS-LCCD architecture, and aims to provide change detection-related data, processing algorithms, and models for supporting the upper service logical layers. Data in the resource layer mainly include the remotely sensed image, land cover classification data, land cover change data, and other reference data (e.g., sample data). The processing algorithms and models include the preprocessing, change information extraction, and postprocessing algorithms involved in the change detection process steps.

In the service layer, the data and processing algorithms in the resource layer are encapsulated into data services and processing services based on the OGC standards, such as the web map service (WMS), web feature service (WFS), and web processing service (WPS). The data services are used to display the original remotely sensed images and the final change detection results. The web processing services are published from the heterogeneous algorithms using the proposed service encapsulation approach.

The logic layer is the core component of the OGS-LCCD. Its goal is to generate land cover change results according to user requirements. In this process, several processing services in the service layer will be invoked and chained together into a suitable and executable change detection service chain using the proposed constraint-rule-based automatic service composition method.

The application layer lies at the top of the OGS-LCCD architecture, and represents a friendly user interface that enables end-users to upload the original remotely sensed data so that they can achieve the expected land change results online. Through input from the user interface, the optimal service chain can be constructed, and the execution process of the service chain can be monitored. The final land cover change results can be returned to the end-users either in the form of web services or as real data.

4.2. System Implementation

The OGS-LCCD system was implemented using a "balanced" browser/server (B/S) architecture with the Microsoft. NET Framework 4.0 and C# Programming language. The development environment was a computer with an Intel (R) Core (TM) i7-8550U CPU @1.80GHZ, 8.0 GB of RAM, and an Internet connection with a bandwidth of 8 MB/s.

On the server side, various geoprocessing algorithms were encapsulated into web services using the OGC WPS standard 1.0.0. The encapsulation process is shown in Figure 4. The algorithms were original written in either C++/C# with GDAL or ENVI IDL API, and then were wrapped into DLLs using Visual Studio tools and the 'COM_IDL_CONNECT' component. The 52° North API was used to encapsulate these DLLs into web services. The DLLs were programmed in a function inherited from 'Abstract_Algorithm', and service semantics were used to construct an XML document for the 'config_wps' to facilitate the service description. In addition, a service composition engine was developed for service chain generation and execution according to user requirements. Using this service encapsulation process, we have encapsulated more than 30 geoprocessing services in OGS-LCCD.

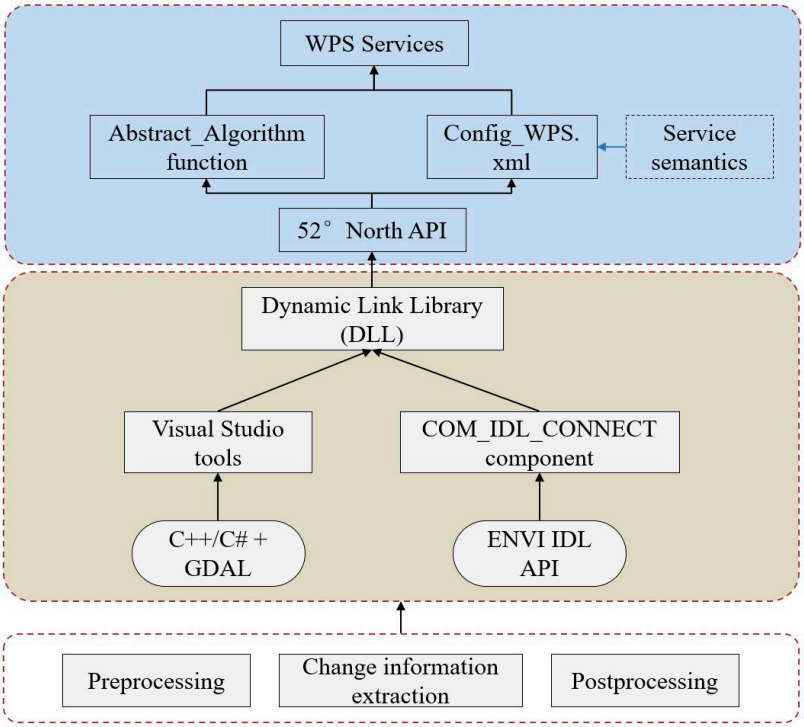

Figure 4. Implementation of web processing service (WPS) service encapsulation from geoprocessing algorithms.

The browser side was represented using a user interface built with HTML, CSS, and JavaScript based on various JavaScript software development kits (SDKs), such as OpenLayers and GeoServer. High-resolution maps were imported into the browser in the form of web services, including Map World, ESRI Map, and OpenStreetMap. These lightweight scripting languages were utilized to enable the OGS-LCCD to be accessed using common browsers (e.g., Internet Explorer, Firefox, Safari, Opera, and Chrome), so that users would not need to install any additional software or plugins.

4.3. Walk-Through Example

In this section, a walk-through example of conducting a web-based land cover change detection task is presented to demonstrate the usefulness of OGS-LCCD. The overall OGS-LCCD operation

involves four steps: data uploading, service chain generation, service chain execution, and results visualization, as illustrated using the unified modeling language (UML) sequence diagram in Figure 5.

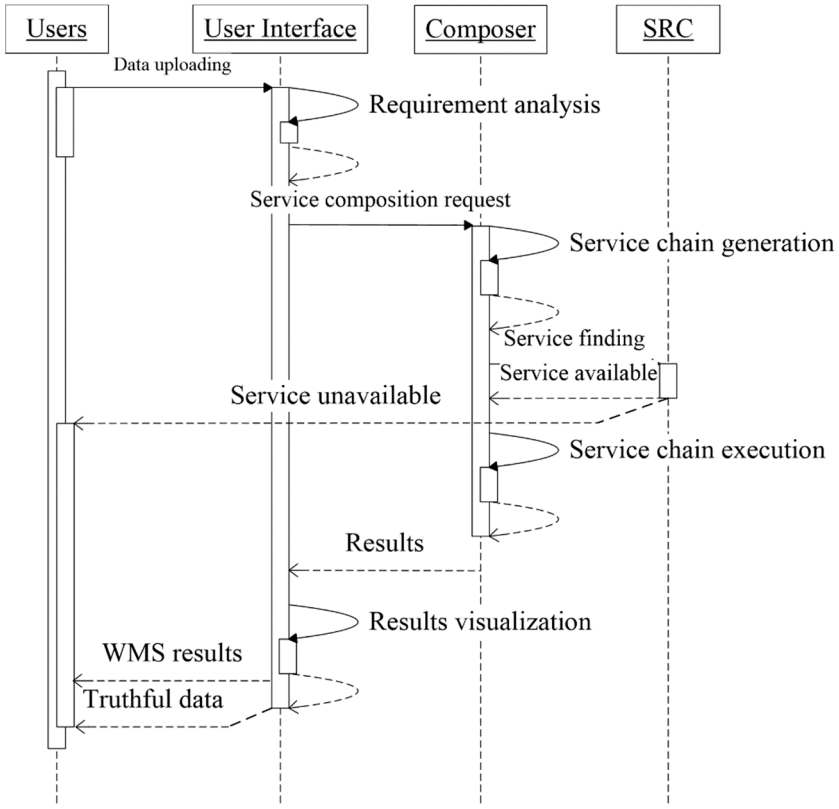

Figure 5. Unified modeling language (UML) sequence diagram of OGS-LCCD operation.

Suppose we want to know where land cover changes have occurred in Wenshang County, Jining, Shandong Province from 2010 to 2018, and we have the Landsat images of Wenshang County acquired in these two years. Using the OGS-LCCD, we can easily upload the Landsat images to the server side of the system. The corresponding geoprocessing service chain will then be automatically generated and executed to derive the land cover change information; the user does not need to install and operate any desktop-based software. The main processing steps are described in the following four sections.

Table 4. JavaScript object notation (JSON) description of input1 image (left) and input2 image (right).

{"Sensor": "Landsat 5 TM",	{"Sensor": "Landsat 8 OLI",
"Acquire_time": "2010/06/22",	"Acquire_time": "2018/06/12",
"Spatial_resolution": "30",	"Spatial_resolution": "30",
"Coverage":{"Type": "Rectangle",	"Coverage":{"Type": "Rectangle",
"UL_lat":"35.82","UL_lon":"116.59",	"UL_lat":"35.82","UL_lon":"116.59",
"UR_lat":"35.82","UR_lon":"116.69",	"UR_lat":"35.82","UR_lon":"116.69",
"BL_lat":"35.74","BL_lon":"116.59",	"BL_lat":"35.74","BL_lon":"116.59",
"BR_lat":"35.74","BR_lat":"116.69",}	"BR_lat":"35.74","BR_lat":"116.69",}
"Radiometric_resolution": "8 bit",	"Radiometric_resolution": "12 bit",
"Format": "IMG",	"Format": "GeoTIFF",
"Data_size":"245242"}	"Data_size":"718239"}

(1) Uploading the remotely sensed data: A jQuery plugin (i.e., uploadify) is used to enable end-users to upload the remotely sensed imagery, and an input text interface is designed for users to select their customized requirements (e.g., the expected return data type). In this example, we upload the remotely sensed imagery acquired in 2010 and 2018 to the service side of the OGS-LCCD from

local desktop computers, as shown in Figure 6. The metadata of these two image files described in JavaScript object notation (JSON) format are shown in Table 4. After uploading the images, we can check and view the images in OGC WMS form, as shown in Figure 7. Additionally, we want to know which areas have been changed between 2010–2018. Therefore, we select 'change area data' as our expected return data type.

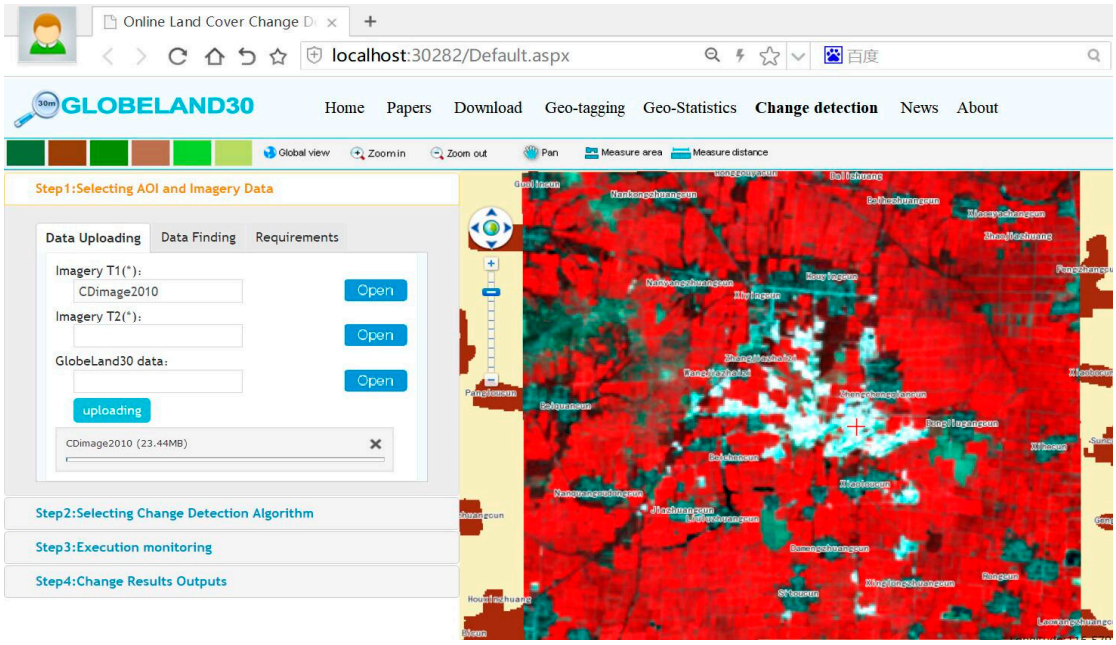

Figure 6. Uploading Landsat 5 data (for year 2010) using OGS-LCCD.

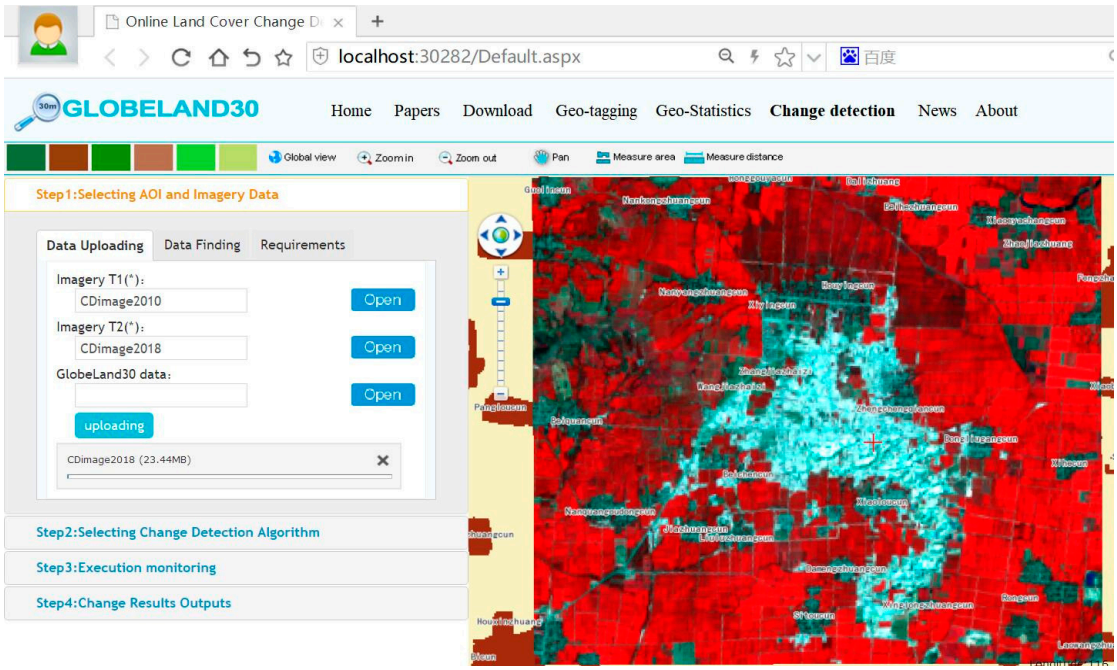

Figure 7. Uploading Landsat 8 data (for year 2018) using OGS-LCCD.

(2) Service chain generation: Based on the uploaded images and the user requirements, the optimal change detection service chain is generated dynamically using the proposed automatic service composition method. In this example, the two images are acquired from the same season and sensor with the same spatial resolution and the different radiometric resolutions. Therefore, the final service

chain can be represented as <DFC (data format conversion) service, RC (radiometric correction) service, CVA (change vector analysis) service, EM (expectation maximization) service>. The semantics of these services in the service chain are shown in Table 5. The visualization service chain is shown in Figure 8.

Table 5. Service semantics of the services in the service chain. DFC: data format conversion, RC: radiometric correction, CVA: change vector analysis, and EM: expectation maximization.

Service Name	Service Semantics
DFC	The DFC service is used to transform image data from the 'IMG' format into the 'GeoTIFF' format. The input and output data type of the DFC service is imagery data.
RC	The RC service is used to convert the digital number (DN) value of image data to surface reflectance. The input and output data type of the RC service is imagery data.
CVA	The CVA service is used to acquire a change magnitude image by computing the difference vectors between two image analysis units. The input type of the CVA service is imagery data, and its output data type is change magnitude data.
EM	The EM service is used to acquire the changed area from a change magnitude map based on an iterative threshold selection method. The input type of the EM service is change magnitude data, and its output data type is change area data.

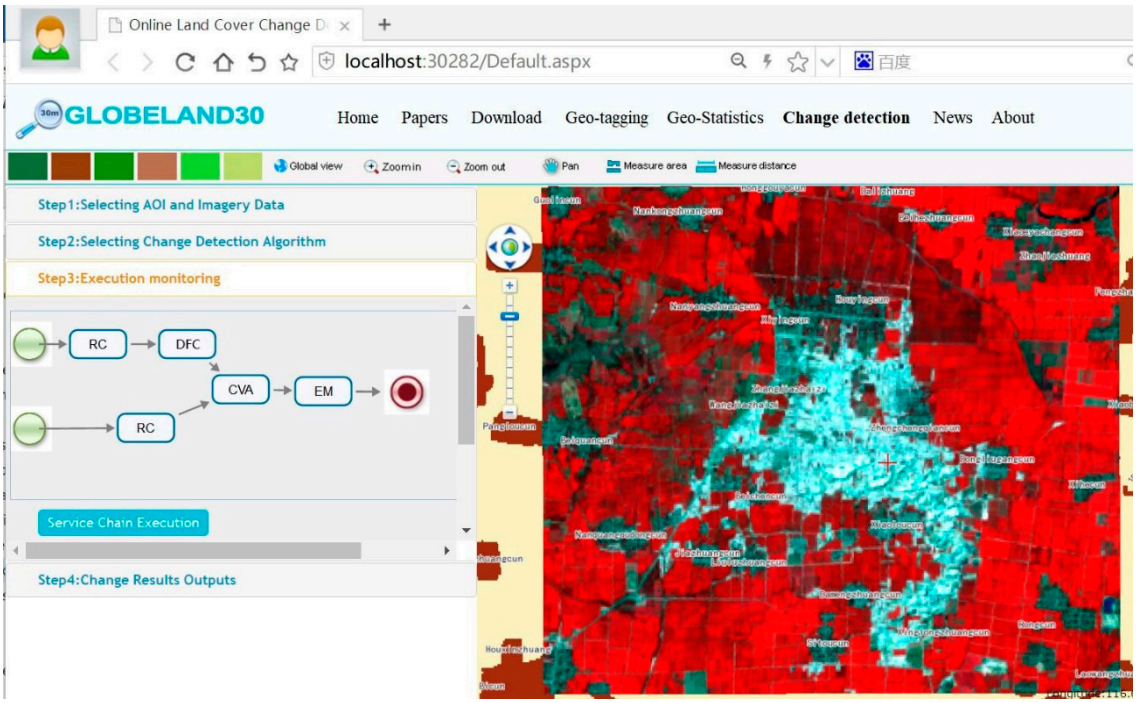

Figure 8. Service chain generation for change detection.

(3) Execution monitoring: In this step, the generated change detection service chain is sent to the service execution engine; then, the service chain execution process can be monitored in real-time through the execution monitoring interface, as shown in Figure 9. In this example, the generated service chain includes four atomic processing services. The execution time of each service is recorded as 36 s, 58 s, 48 s, and 87 s, respectively. The execution status can also be viewed in a progress bar, as shown in Figure 9.

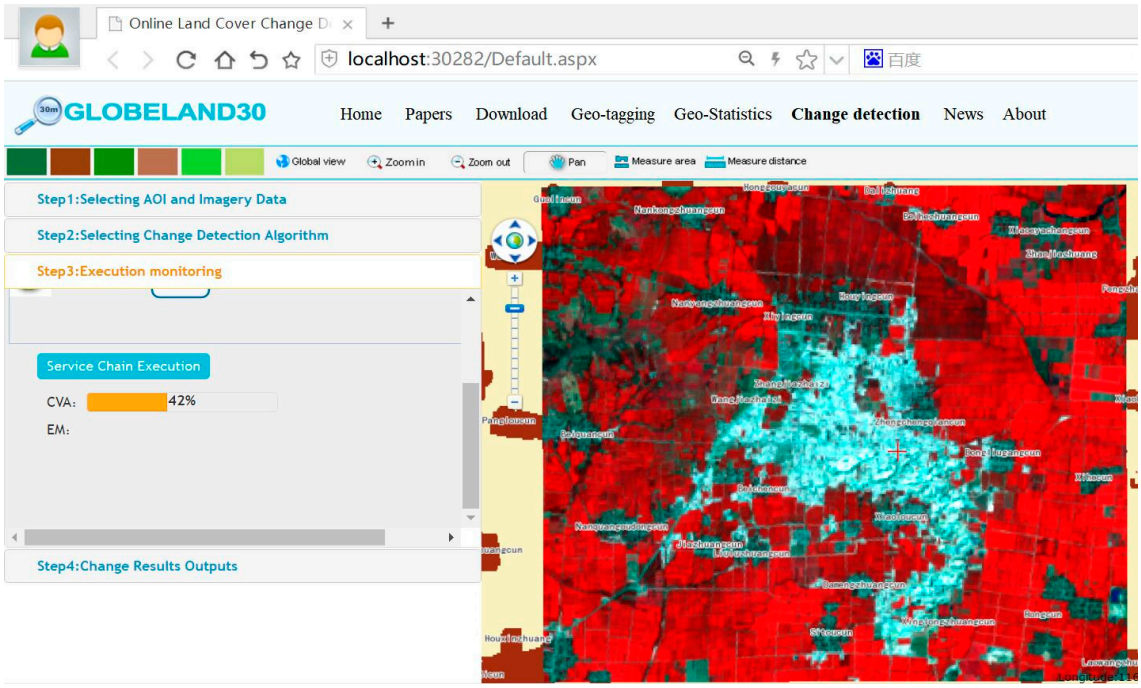

Figure 9. Monitoring of the execution of a change detection service chain.

(4) Web-based results visualization: The final change detection results are received either by downloading from the OGS-LCCD server service or by browsing in the result visualization interface. Using the former approach, the real result data in a format such as 'GeoTIFF' will be returned to users. In the latter, the WMS published from GeoServer software will be browsing on the right side of the OGS-LCCD interface, as shown in Figure 10.

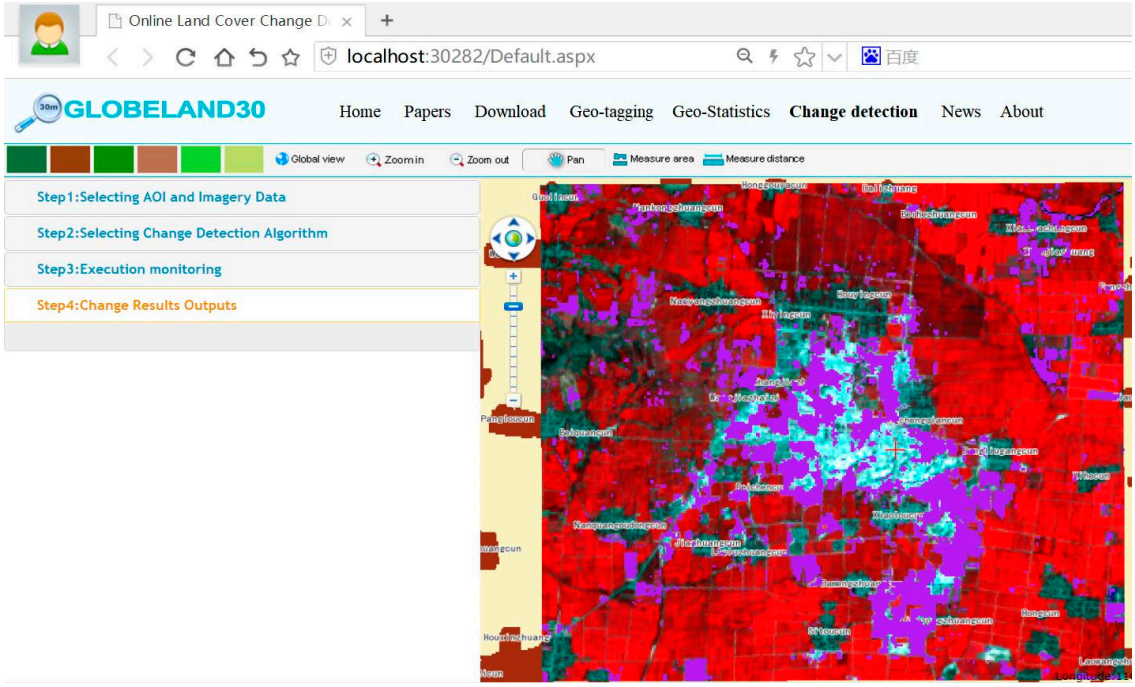

Figure 10. Change detection results (purple regions) visualized using a web map service (WMS).

5. Evaluation and Discussion

5.1. Evaluation

To evaluate the correctness of the proposed service composition method and the efficiency of the OGS-LCCD, two groups of contrastive experiments were performed.

The first experiment was performed between the constraint rule-based automatic service composition method, with a traditional data semantic-based method. The change detection service chains generated using these two methods are shown in Figure 11. These two service chains were also sent to an execution engine. The result was that the service chain that was based on the proposed method (i.e., Figure 11b) was executed successfully, while the service chain based on the traditional method (i.e., Figure 11a) failed to execute.

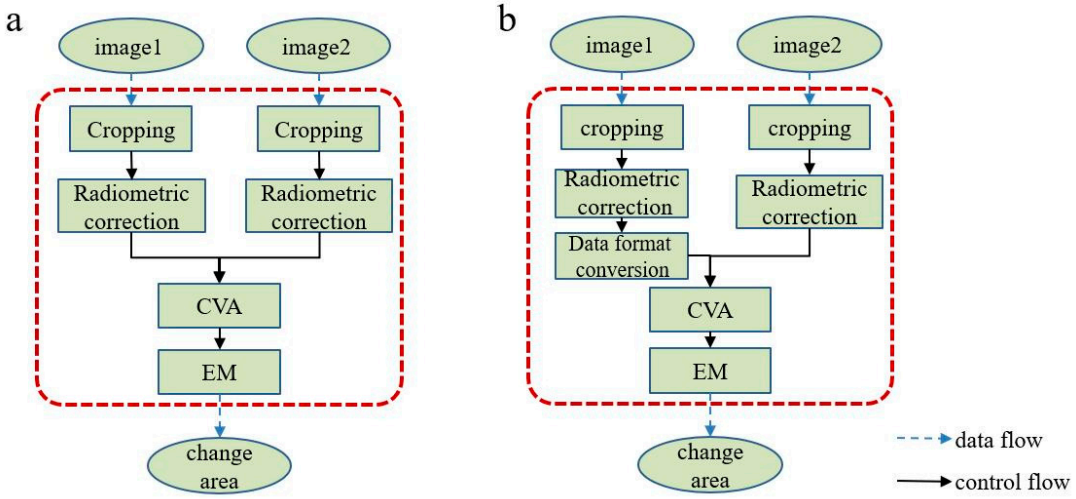

Figure 11. Two generated change detection service chains: (**a**) service chain based on the traditional method; (**b**) service chain based on the proposed method.

The above result occurred for the follow reasons. Although the output data type of the radiometric correction service is equal to the input data type of the CVA service, their data formats are quite different (the former is 'IMG' and the latter is 'GeoTIFF'). In other words, there is a data format inconsistency. If the radiometric correction service and the CVA service are directly connected, the service chain would fail to execute, as shown in Figure 12a. The proposed service composition method detected the data inconsistence, and added the corresponding preprocessing service to the service chain to ensure the successful execution. As shown in Figure 12b, the added data format conversion service guarantees that the data formats of the radiometric correction service and CVA service are consistent.

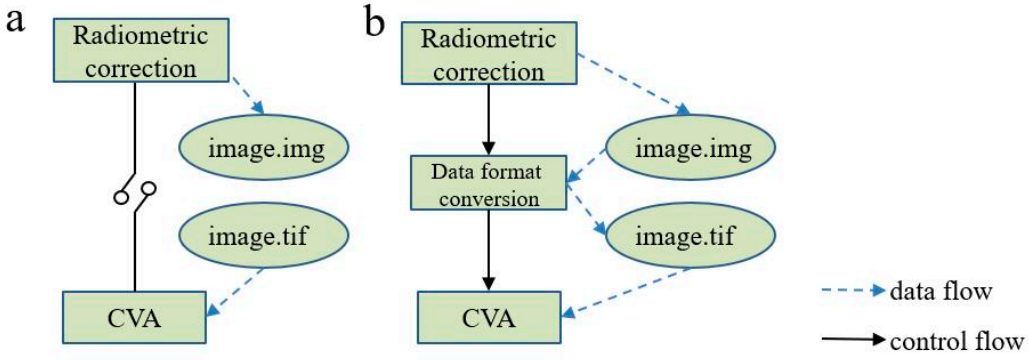

Figure 12. Actual implementation of the two service chains: (**a**) executed failed; (**b**) executed successfully.

Second, stress testing the central processing unit (CPU) and memory utilization were conducted to evaluate the OGS-LCCD system. Twenty threads were selected to simulate multiple end-users simultaneously operating the system. During stress testing, approximately one to 20 threads were activated to generate and execute the service chain. Ten instances of stress testing were conducted to record the maximum and average CPU and memory utilization values. The stress testing results are shown in Figures 13 and 14. As the number of threads increased, the maximum and average CPU and memory utilization values also increased, but the CPU utilization remained lower than 40%. In addition, when the number of active threads was less than 12, the utilization changed more slowly. In contrast, when the number of activated threads was larger (more than 12), the utilization changed more rapidly. The results of these experiments show that the OGS-LCCD can maintain good stability when multiple users are simultaneously performing the web-based change detection tasks.

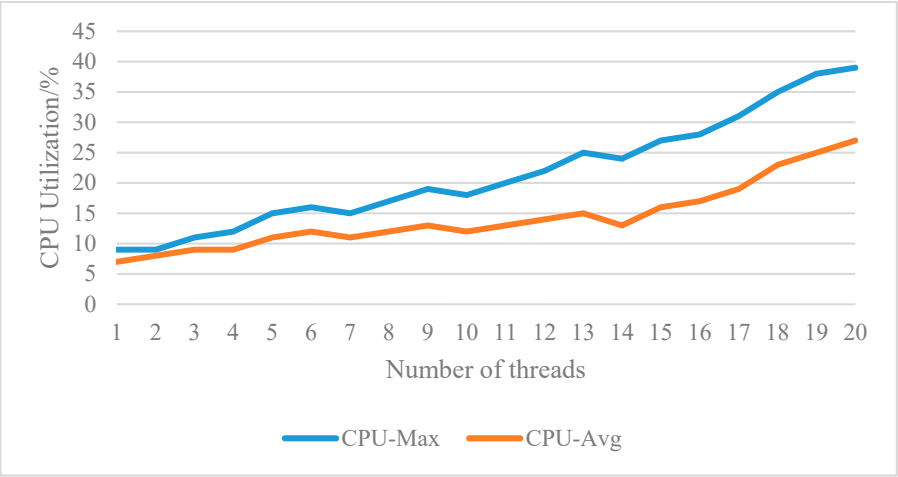

Figure 13. Central processing unit (CPU) utilization (%).

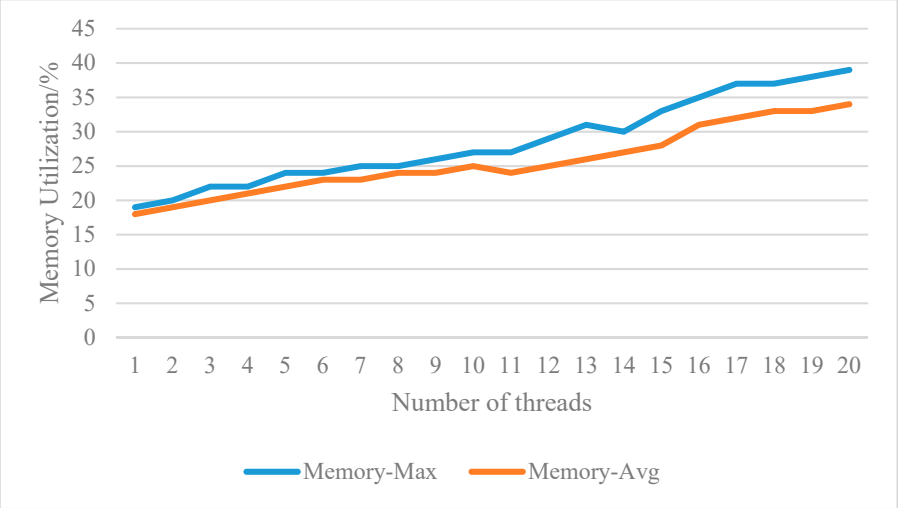

Figure 14. Memory utilization (%).

5.2. Discussion

Based on the results and analysis of the above walk-through example and evaluations, the proposed service composition method and OGS-LCCD exhibit certain advantages and improvements compared with traditional approaches. From the aspect of intelligent change detection, we propose an automatic service composition method to generate a suitable change detection service chain according to user requirements. That is, OGS-LCCD can enable nonprofessional end-users without owning much

land cover expertise to conduct change detection tasks. In addition, with the help of the constraint rules, the proposed service composition method is more robust than the traditional semantic based methods, and can help guarantee that the service chain will be executed correctly. From the aspect of operating efficiency, the OGS-LCCD provides a user-friendly interface to conduct change detection online using a web browser, without installing any desktop and professional software or plugins. The time required to conduct a change detection service chain is significantly reduced, because many complex processing steps are decreased, such as data copying and data format conversion to accommodate different software. In addition, the OGS-LCCD performs well for multiple end-users conducting web-based change detection tasks simultaneously.

Since the OGC released the WPS specification, many open source communities have created WPS platforms, such as the ZOO-Project, 52°North, PyWPS, and Deegree [11,44]. The main goal of these open source platforms is to provide generic and standard-compliant methods for using existing open source libraries and algorithms as WPS-based geoprocessing services. In the proposed system, 52°North APIs are used for change detection and service encapsulation. In contrast to these open-source platforms, OGS-LCCD is a system that is focused primarily on the change detection domain, and is intended to support web-based land cover change detection. In addition, the proposed system provides a constraint-rule-based automatic service composition mechanism that enable end-users to conduct a web-based land cover change detection task with minimal manual intervention.

Despite the advantages of the OGS-LCCD, the development of the web-based system for supporting intelligent land cover geoprocessing is still in its early stages. Similar to general web-based geoprocessing systems, OGS-LCCD possesses several limitations, and some important issues should be improved.

First, remotely sensed imagery data are often large, and uploading these data with the proposed system is onerous and time consuming. With the development of recent cloud computing infrastructure (e.g., Azure Cloud and Alibaba Cloud), it becomes possible to provide large-capacity cloud servers and powerful computing resources. By using such infrastructures, free satellite images (e.g., Landsat image series) can be collected and stored in advance. The end-users would need only to specify the boundaries of their area of interest (AOI) to select the input data rather than upload large amounts of data from their local to server.

Second, change detection models are computationally intensive, and require large amounts of computing resources. The hardware and network configuration of the current system is not enough to handle large-scale computation. To improve the computing efficiency of web-based land cover change detection, the traditional geoprocessing mode should be extended by integrating a state-of-the-art distributed computing mechanism (e.g., Spark or Hadoop) for image data processing [45], which offers powerful and affordable alternatives to run large-scale computations.

Third, land cover change detection over a large area is a complex task; the system should provide additional LCC web services to allow it to address different situations. To address this challenge, multisource and heterogeneous change detection algorithms (e.g., oriented toward long time-series images) will be steadily encapsulated into web services. In addition, the system needs to integrate the strategy of code transmission [38], which facilitates end-users to upload their own algorithms to the server side to conduct personalized change detection tasks.

6. Conclusions and Future Work

This paper presents a web service-based geoprocessing system for supporting intelligent land cover change detection. The OGS-LCCD system provides a novel geoprocessing platform that enables end-users to conduct change detection tasks in an open web environment. The encapsulation of the heterogeneous web services approach and a constraint-rule-based automatic service composition method are proposed to generate an executable change detection service chain for different conditions. The prototype system is then designed and implemented using several available online geoprocessing technologies and open source software.

The walk-through example using OGS-LCCD demonstrates that the proposed system has the potential to enable both professional or nonprofessional end-users to conduct change detection tasks easily using only general web browsers. A comparison experiment between the constraint rule-based automatic service composition method and a traditional data semantic-based method is performed. The results indicate that OGS-LCCD could generate a suitable and executable service chain based on the user's requirements. The stress testing experiments show that OGS-LCCD maintains good efficiency for supporting multiple end-users to conduct web-based change detection tasks simultaneously.

Our future work will focus on improving the efficiency of web-based land cover change detection by integrating a state-of-the-art distributed computing mechanism (e.g., Spark or Hadoop) and collecting frequently used remotely sensed big data on the server side. In addition, we plan to further enrich the geoprocessing web services by encapsulating multisource and heterogeneous change detection algorithms, allowing OGS-LCCD to address more complex land cover change detection situations.

Author Contributions: Conceptualization, H.X. and J.C.; Data curation, D.H.; Methodology, H.X. and J.C.; Resources, J.C. and H.W.; Software, H.W. and D.H.; Validation, J.C. and D.H.; Writing—original draft, H.X. and J.C.; Writing—review & editing, H.X. and D.H.

Funding: This research was funded by the National Science Foundation of China, grant number 41801308 and 41701443; Doctoral Research Fund of Shandong Jianzhu University, grant number XNBS1804; and Natural Science Foundation of Shandong Province, grant number ZR2018MD008.

Acknowledgments: The authors would like to thank the editors and the anonymous reviewers for their constructive comments and suggestions, which greatly helped to improve the quality of the manuscript.

References

1. Feddema, J.J.; Oleson, K.W.; Bonan, G.B.; Mearns, L.O.; Buja, L.E.; Meehl, G.A.; Washington, W.M. The importance of land-cover change in simulating future climates. *Science* **2005**, *310*, 1674–1678. [CrossRef] [PubMed]
2. Chen, J.; Ban, Y.; Li, S. China: Open access to earth land-cover map. *Nature* **2014**, *514*, 434.
3. Hussain, M.; Chen, D.; Cheng, A.; Wei, H.; Stanley, D. Change detection from remotely sensed images: From pixel-based to object-based approaches. *ISPRS J. Photogramm. Remote Sens.* **2013**, *80*, 91–106. [CrossRef]
4. Lu, D.; Li, G.; Moran, E. Current situation and needs of change detection techniques. *Int. J. Image Data Fusion* **2014**, *5*, 13–38. [CrossRef]
5. Andrew, P.T.; Alexis, J.C.; Nicholas, J.T.; Alistair, L.; Peter, F.F. A critical synthesis of remotely sensed optical image change detection techniques. *Remote Sens. Environ.* **2015**, *160*, 1–14.
6. Hofer, B. Uses of online geoprocessing technology in analyses and case studies: A systematic analysis of literature. *Int. J. Digit. Earth* **2015**, *8*, 901–917. [CrossRef]
7. Xing, H.; Chen, J.; Wu, H.; Zhang, J.; Liu, B. An Online Land Cover Change Detection System with Web Service Composition. In Proceedings of the 4th International Workshop on Earth Observation and Remote Sensing Applications, Guangzhou, China, 4–6 July 2016.
8. Xing, H.; Chen, J.; Wu, H.; Zhang, J.; Li, S.; Liu, B. A service relation model for web-based land cover change detection. *Int. J. Photogramm. Remote Sens.* **2017**, *132*, 20–32. [CrossRef]
9. Yue, P.; Di, L.; Yang, W.; Yu, G.; Zhao, P.; Gong, J. Semantic web services-based process planning for earth science applications. *Int. J. Geogr. Inf. Sci.* **2009**, *23*, 1139–1163. [CrossRef]
10. Papazoglou, M.P. Service-oriented computing: Concepts, characteristics and directions. In Proceedings of the Fourth International Conference on Web Information Systems Engineering, Roma, Italy, 10–12 December 2003; pp. 3–12.
11. Zhao, P.; Foerster, T.; Yue, P. The geoprocessing web. *Comput. Geosci.* **2012**, *47*, 3–12. [CrossRef]
12. Di, L. Geobrain—A Web Services Based Geospatial Knowledge Building System. In Proceedings of the NASA Earth Science Technology Conference, Palo Alto, CA, USA, 22–24 June 2004; pp. 22–24.
13. Zhai, X.; Yue, P.; Zhang, M. A sensor web and web service-based approach for active hydrological disaster monitoring. *Int. J. Geo-Inf.* **2016**, *5*, 171. [CrossRef]

14. Tan, X.; Guo, S.; Di, L.; Deng, M.; Huang, F.; Ye, X.; Sun, Z.; Gong, W.; Sha, Z.; Pan, S. Parallel agent-as-a-service (p-aaas) based geospatial service in the cloud. *Remote Sens.* **2017**, *9*, 382. [CrossRef]
15. Karantzalos, K.; Bliziotis, D.; Karmas, A. A scalable geospatial web service for near real-time, high-resolution land cover mapping. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 4665–4674. [CrossRef]
16. Chen, J.; Chen, J.; Liao, A.; Cao, X.; Chen, L.; Chen, X.; He, C.; Han, G.; Peng, S.; Lu, M. Global land cover mapping at 30 m resolution: A pok-based operational approach. *ISPRS J. Photogramm. Remote Sens.* **2015**, *103*, 7–27. [CrossRef]
17. Chen, J.; Li, S.; Wu, H.; Chen, X. Towards a collaborative global land cover information service. *Int. J. Digit. Earth* **2017**, *10*, 356–370. [CrossRef]
18. Chen, J.; Cao, X.; Peng, S.; Ren, H. Analysis and applications of globeland30: A review. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 230. [CrossRef]
19. Han, G.; Chen, J.; He, C.; Li, S.; Wu, H.; Liao, A.; Peng, S. A web-based system for supporting global land cover data production. *ISPRS J. Photogramm. Remote Sens.* **2015**, *103*, 66–80. [CrossRef]
20. Chen, F.; Chen, J.; Wu, H.; Hou, D.Y.; Zhang, W.W.; Zhang, J.; Zhou, X.G.; Chen, L.J. A landscape shape index-based sampling approach for land cover accuracy assessment. *Sci. China* **2016**, *59*, 1–12. [CrossRef]
21. Xing, H.; Chen, J.; Zhou, X. A geoweb-based tagging system for borderlands data acquisition. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 1530–1548. [CrossRef]
22. Li, R.; Kuang, W.H.; Chen, J.; Chen, L.J.; Liao, A.P.; Peng, S.; Guan, Z.X. Spatio-temporal pattern analysis of aritificial surface use efficiency based on Globeland30 (in Chinese). *Scientia Sinica Terrae* **2016**, *46*, 1436–1445.
23. Han, W.; Yang, Z.; Di, L.; Mueller, R. Cropscape: A web service based application for exploring and disseminating us conterminous geospatial cropland data products for decision support. *Comput. Electron. Agric.* **2012**, *84*, 111–123. [CrossRef]
24. Fritz, S.; Mccallum, I.; Schill, C.; Perger, C.; See, L.; Schepaschenko, D.; Marijn, V.D.V.; Kraxner, F.; Obersteiner, M. Geo-wiki: An online platform for improving global land cover. *Environ. Model. Softw.* **2012**, *31*, 110–123. [CrossRef]
25. See, L.; Laso Bayas, J.; Schepaschenko, D.; Perger, C.; Dresel, C.; Maus, V.; Salk, C.; Weichselbaum, J.; Lesiv, M.; McCallum, I. Laco-wiki: A new online land cover validation tool demonstrated using globeland30 for Kenya. *Remote Sens.* **2017**, *9*, 754. [CrossRef]
26. Clark, M.L.; Aide, T.M. Virtual interpretation of earth web-interface tool (view-it) for collecting land-use/land-cover reference data. *Remote Sens.* **2011**, *3*, 601–620. [CrossRef]
27. Bastin, L.; Buchanan, G.; Beresford, A.; Pekel, J.-F.; Dubois, G. Open-source mapping and services for web-based land-cover validation. *Ecol. Inform.* **2013**, *14*, 9–16. [CrossRef]
28. Sheng, Q.Z.; Qiao, X.; Vasilakos, A.V.; Szabo, C.; Bourne, S.; Xu, X. Web services composition: A decade's overview. *Inf. Sci.* **2014**, *280*, 218–238. [CrossRef]
29. Evangelidis, K.; Ntouros, K.; Makridis, S.; Papatheodorou, C. Geospatial services in the cloud. *Comput. Geosci.* **2014**, *63*, 116–122. [CrossRef]
30. Yang, C.; Chen, N.; Di, L. Restful based heterogeneous geoprocessing workflow interoperation for sensor web service. *Comput. Geosci.* **2012**, *47*, 102–110. [CrossRef]
31. Yu, G.; Zhao, P.; Di, L.; Chen, A.; Deng, M.; Bai, Y. Bpelpower—a bpel execution engine for geospatial web services. *Comput. Geosci.* **2012**, *47*, 87–101. [CrossRef]
32. Yue, P.; Zhang, M.; Tan, Z. A geoprocessing workflow system for environmental monitoring and integrated modelling. *Environ. Model. Softw.* **2015**, *69*, 128–140. [CrossRef]
33. Rodriguez Mier, P.; Pedrinaci, C.; Lama, M.; Mucientes, M. An integrated semantic web service discovery and composition framework. *Ieee Trans. Serv. Comput.* **2016**, *9*, 537–550. [CrossRef]
34. Feng, J.Z.; Kong, L.F.; Wang, X.H. Web service automatic composition based on semantic relationship graph. *Comput. Integr. Manuf. Syst.* **2012**, *18*, 427–436.
35. Hashemian, S.V.; Mavaddat, F. A graph-based framework for composition of stateless web services. In Proceedings of the European Conference on Web Services, Zurich, Switzerland, 4–6 December 2006; pp. 75–86.
36. Yue, P.; Di, L.; Yang, W.; Yu, G.; Zhao, P. Semantics-based automatic composition of geospatial web service chains. *Comput. Geosci.* **2007**, *33*, 649–665. [CrossRef]

37. Tan, X.; Di, L.; Deng, M.; Chen, A.; Huang, F.; Peng, C.; Gao, M.; Yao, Y.; Sha, Z. Cloud- and agent-based geospatial service chain: A case study of submerged crops analysis during flooding of the yangtze river basin. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 1359–1370. [CrossRef]
38. Yue, S.; Chen, M.; Wen, Y.; Lu, G. Service-oriented model-encapsulation strategy for sharing and integrating heterogeneous geo-analysis models in an open web environment. *ISPRS J. Photogramm. Remote Sens.* **2016**, *114*, 228–273. [CrossRef]
39. Chen, Z.; Lin, H.; Chen, M.; Liu, D.; Bao, Y.; Ding, Y. A framework for sharing and integrating remote sensing and gis models based on web service. *Sci. World J.* **2014**, *2014*, 57–78. [CrossRef] [PubMed]
40. Wen, Y.; Chen, M.; Yue, S.; Zheng, P.; Peng, G.; Lu, G. A model-service deployment strategy for collaboratively sharing geo-analysis models in an open web environment. *Int. J. Digit. Earth* **2017**, *10*, 405–425. [CrossRef]
41. Cruz, S.A.B.; Monteiro, A.M.V.; Santos, R. Automated geospatial web services composition based on geodata quality requirements. *Comput. Geosci.* **2012**, *47*, 60–74. [CrossRef]
42. Ranisavljević, É.; Devin, F.; Laffly, D.; Nir, Y.L. Semantic orchestration of image processing services for environmental analysis. *ISPRS J. Photogramm. Remote Sens.* **2013**, *83*, 184–192. [CrossRef]
43. Chen, J.; Lu, M.; Chen, X.; Chen, J.; Chen, L. A spectral gradient difference based approach for land cover change detection. *ISPRS J. Photogramm. Remote Sens.* **2013**, *85*, 1–12. [CrossRef]
44. Steiniger, S.; Hunter, A.J.S. The 2012 free and open source gis software map—A guide to facilitate research, development, and adoption. *Comput. Environ. Urban Syst.* **2013**, *39*, 136–150. [CrossRef]
45. Yang, C.; Yu, M.; Hu, F.; Jiang, Y.; Li, Y. Utilizing cloud computing to address big geospatial data challenges. *Comput. Environ. Urban Syst.* **2016**, *61*, 120–128. [CrossRef]