Technical Note

An Implementation of Open Source-Based Software as a Service (SaaS) to Produce TOA and TOC Reflectance of High-Resolution KOMPSAT-3/3A Satellite Image

Kwangseob Kim and Kiwon Lee *

Department of Electronics and Information Engineering, Hansung University, Seoul 02876, Korea; engintruder@hansung.ac.kr
* Correspondence: kilee@hansung.ac.kr

Abstract: The majority of cloud applications are created or delivered to provide users with access to system resources or prebuilt processing algorithms for efficient data storage, management, and production. The number of cases linking cloud computing to the use of global observation satellite data continues to rise, owing to the benefits of cloud computing. This study aims to develop a cloud software as a service (SaaS) that yields reflectance products in high-resolution Korea Multi-Purpose Satellite (KOMPSAT)-3/3A satellite images. The SaaS model was designed as three subsystems: a Calibration Processing System (CPS), a Request System for CPS supporting RESTful application programming interface (API), and a Web Interface Application System. Open-source components, libraries, and frameworks were used in this study’s SaaS, including an OpenStack for infrastructure as a service. An absolute atmospheric correction scheme based on a Second Simulation of a Satellite Signal in the Solar Spectrum (6S) radiative transfer code with atmospheric variable inputs was used to generate the top-of-atmosphere (TOA) and top-of-canopy (TOC) reflectance products. The SaaS implemented in this study provides users with the absolute atmospheric calibration functionality to apply their KOMPSAT-3/3A satellite image set through a web browser and obtain output directly from this service. According to experiments to check the total performance time for images, bundled with four bands of red, green, blue, and near-infrared, it took approximately 4.88 min on average for the execution time to obtain all reflectance results since satellite images were registered into the SaaS. The SaaS model proposed and implemented in this study can be used as a reference model for the production system to generate reflectance products from other optical sensor images. In the future, SaaS, which offers professional analysis functions based on open source, is expected to grow and expand into new application fields for public users and communities.

Keywords: cloud computing; KOMPSAT-3/3A; open source; reflectance; software-as-a-service

1. Introduction

The concept of cloud computing arose from the idea of renting rather than buying computer resources such as hardware and software. The service model is classified as software services of software as a service (SaaS): platform as a service (PaaS), which provides the foundation for running applications, and infrastructure as a service (IaaS), which provides memory, CPU, storage, and network, depending on what elements apply to the cloud service. Meanwhile, cloud computing can be classified as a private cloud, a public cloud, a community cloud, or a hybrid cloud [1,2].

Many attempts have been made to link and apply cloud computing environments in managing, storing, and distributing large-capacity geo-observation satellite information. In the side of the business for earth observation (EO) image, the beneficial points of a cloud computing scheme or environment to an EO data processing system are as follows [3,4]: scalable image processing capabilities in online or on-demand mode, processors for effective colocation handling actual image databases, the software uses optimized for hardware
requirements for end-users, comfort integration and deployment of new algorithms and processing tools, expanding remote sensing technology by improved application sharing, and direct linking remote sensing experts to normal users. With architecture, concept, and characteristics, Yang and Huang [5] introduced a cloud computing infrastructure for geoscience application challenges. Wang et al. [6] presented examples of high-performance EO satellite information management, processing, and analysis in cloud environments, as well as a summary of the architecture and interfaces based on the cloud service model.

IaaS is a type of cloud computing that provides virtualized computing resources over the Internet. Third-party providers can deploy and manage infrastructure components such as hardware, software, servers, and storage on behalf of their users in an IaaS environment. PaaS is a cloud computing model for delivering applications over the Internet. In this model, cloud providers deliver to their users or developers the hardware and software components that are typically required to implement applications. For example, the Google Earth Engine (GEE) is PaaS in close collaboration with Google Cloud to bring the Landsat and Sentinel-2 collections. Users can access prebuilt satellite information on GEEs in web browsers in an application programming interface (API) manner to perform the analysis they want to use the Earth Engine data catalog [7,8]. SaaS is a software distribution model in which vendors or service providers host applications and make them available to users over a network environment, including the Internet. The application service provider and on-demand computing software delivery models are closely related to the SaaS model. IaaS and PaaS must be substructures for SaaS implementation, even though cloud service models are divided.

For the cloud-based remote sensing platforms or frameworks for EO applications, many attempts and achievements have been conducted. Yue et al. [9] proposed the machine learning approach to predict computational intensity dividing the spatial domain into balanced subdivisions to obtain better performance. Wang et al. [10] compared the cost of cloud computing-based remote sensing data processing in three paradigms: standalone, message passing interface, and MapReduce. Yan et al. [11] presented a cloud-based product generation system for multi-source remote sensing data. Data cubes connected to a cloud environment for EO were discussed by Kopp et al. [12]. Yao et al. [13] investigated how EO data can be solved using cloud computing and discrete global grid systems. Gomes et al. [14] presented an overview of seven platforms for extensive EO data management and analysis, including GEE, Sentinel Hub, and Open Data Cube. Schramm et al. [15] introduced openEO API–harmonizing for EO cloud services with virtual data cube functionalities. Huang et al. [16] applied an elastic computing paradigm with a containerized spark engine on Kubernetes for on-the-fly fusion of remotely sensed big data. Kline [17] summarized Landsat migration state of the cloud environment in the United States Geological Survey.

Many different types of SaaS for EO are available, all based on PaaS and IaaS infrastructure. Development cases for SaaS have been released to EO data. Using Landsat and rainfall data from a cloud application, Hausner et al. [18] evaluated the cost effectiveness of riparian restoration projects. Shaharum [19] presented the remote ecosystem monitoring assessment pipeline on a cloud computing platform that hosts to monitor changes in the distribution of oil palms; Landsat data with a built-in machine learning algorithm was used. Yang et al. [20] presented a cyber-infrastructure for real-time forest mapping and estimation integrating cloud computing applications included the GEE and the Google Cloud Platform. Antunes et al. [21] introduced a cloud computing approach for object-based image analysis and classification of arbitrarily sizable remote sensing datasets. Sagan et al. [22] investigated the monitoring of inland water quality using EO data and discussed the limitations of machine learning and cloud computing. Paludo et al. [23] used GEE cloud computing to develop a mapping scheme for summer soybean and corn using EO data from Landsat-8, Sentinel-2, and field data. Amani et al. [24] used Sentinel data to show how the GEE platform and neural networks could be used for crop mapping in Canada. To develop loss monitoring applications on cloud-based platform architectures,
Zou et al. [25] categorized the computational characteristics of various algorithms to develop loss monitoring applications on platform architecture cloud-based computing. Ferreira et al. [26] introduced the production system for land use and coverage maps based on the data cubes using image time series analysis and machine learning techniques. Singh et al. [27] studied quantifying COVID-19-enforced global changes in atmospheric pollutants using cloud computing-based remote sensing to provide a testbed for climate sensitivity experiments on GEE. Hardy et al. [28] developed SaaS to derive grassland information from the Sentinel-2 image. From the geospatial service provider’s viewpoint, EO data are processed as the Data as a Service in a cloud environment [29]. However, SaaS has not yet been released, which provides specific functionalities with user interfaces to generate value-added products such as surface reflectance using any type of high-resolution imagery.

The motivation for this study is as follows. To increase the use of high-resolution optical satellite imagery, the user wants to generate surface reflectance, the essential primary product. However, few proprietary software tools support the functionality to produce surface reflectance from KOMPSAT-3/3A images by absolute atmospheric correction. Therefore, if users obtain the product using the function of generating only surface reflectance in a web environment, they efficiently utilize the processing environment or software they want to use. It can increase the utilization of high-resolution satellite image much more efficiently. Thus, the purpose of this work is to develop a cloud-based remote sensing image processing system with intuitive user interfaces for satellite image processing algorithms using a fully open-source stack as SaaS. There are numerous cases for open-source use and applications in EO image processing and geospatial data manipulation in specialized fields or domains [30–33]. Open Geospatial Consortium, Inc. (OGC), which develops standards for interoperability of geospatial information, continues to publish reports linking OGC interface specifications with cloud computing [34–36].

To implement SaaS toward EO applications, one may use cloud services such as Amazon Web Services (AWS), Microsoft Azure, IBM Cloud, or Google Cloud. As for the cloud computing infrastructure as IaaS, OpenStack is the most popular cloud platform to implement specific application services, an open-source software cloud computing platform [37]. Kang and Lee experimented with autoscaling of geo-based image processing in an OpenStack environment and tested the implementation of a remote sensing image analysis service on the open-source cloud platform OpenStack [38,39]. Lee et al. [40] implemented a cloud-based satellite image processing service by open-source stack (https://ksatdb.kari.re.kr/main/main.do, accessed on 26 July 2021). Yoon et al. [41] implemented a geospatial image processing case linked to OGC-Web Processing Service 2.0 to the e-government standard framework. Lee et al. [42] evaluated performance for a geospatial image processing service based on open-source PaaS using Cloud Foundry on OpenStack.

The proposed SaaS system in this study is based on a fully open source stack, and the service supports cross-web browser functionalities and does not need any installation by users. The target service in this SaaS model is to generate top-of-atmosphere (TOA) and top-of-canopy (TOC) reflectance products for Korea Multi-Purpose Satellite (KOMPSAT)-3/3A. The KOMPSAT-3/3A program is an EO mission of the Korea Aerospace Research Institute (KARI).

The spectral band specification of KOMPSAT-3 launched on 17 May 2012, is as follows: 450–900 µm for the panchromatic band, 450–520 µm for multi-spectral band (MS) 1 of blue, 520–600 µm for MS2 of green, 630–690 µm for MS3 of red, and 760–900 µm for MS4 of near-infrared (NIR). The ground sample distance (GSD) at the nadir is 0.70 m for a panchromatic image, 2.8 m for multi-spectral bands, and 5.5 m for NIR data. KOMPSAT-3A, which was launched on 25 March 2015, has the same spectral band specifications as KOMPSAT-3. A panchromatic image has a GSD of 0.55 m at the nadir, 2.2 m for multispectral bands, and 5.5 m for NIR data. The swath width at the nadir is 15 and 12 km for KOMPSAT-3 and KOMPSAT-3A, respectively.

As the remote sensing engine to produce TOA and TOC reflectance, optical radiometric calibration modules within the Orfeo ToolBox (OTB) [43] was adopted because those
modules support an absolute atmospheric correction algorithm using Second Simulation of a Satellite Signal in the Solar Spectrum (6S) as a radiative transfer code to simulate solar radiation reflected by atmospheric and surface systems. In this study, the code is translated into Java and then ported into a SaaS component. The OTB is used in specific remote sensing applications [44,45], but there are a few cases for atmospheric calibration that have been ported to Java versions.

The following are the reasons for the SaaS-provided TOA and TOC reflectance generation programs, which are among the various space-borne image processing functions. The first is because reflectance products generated by absolute atmosphere correction can use primary data for additional scientific analysis, such as vegetation analysis, desired by general users. The second is because there is no commercial software capable of obtaining reflectance results from KOMPSAT-3/3A images. Third, it is difficult for general users to implement algorithms or processing schemes to obtain such reflectance products.

IaaS environments were first defined in this paper as the infrastructure used in the implementation of open source-based SaaS. A list of all open sources that were used in this development was also presented. The operational concept of the SaaS model and its basic architecture with subsystems were described. The user interface and internal processing schemes were presented as a result of the implementation, and the performance required for online execution processing was tested.

2. Open-Source List Applied for SaaS Development

The entire user interface process in the web environment provided by the substructure is open source in this study. Of course, because the SaaS model developed in this study is for the specific purpose of generating surface reflectance, the engine for satellite image information processing is the most important. However, IaaS, which enables these applications, is also important and does not use cloud environments such as AWS or Google Cloud. OpenStack used as IaaS in this study was briefly summarized. In addition, technical specifications related to the applied open source for the absolute atmospheric correction process applied to the generation of surface reflectance products were explained. We described the three subsystems that make up the SaaS developed in this study.

2.1. OpenStack as IaaS

Since its initial release in 2010, OpenStack has been the most active open-source cloud computing project, releasing new versions twice a year. OpenStack can be used to set up a developer’s or operator’s own cloud computing environment, allowing them to create the instance servers they desire. The OpenStack is based on a modular architecture with the components of a controller, a scalable storage system, blocking storage, networks, and Internet Protocol addresses, and database services, as well as the API compatible with Amazon elastic compute cloud (EC2) and Amazon Simple Storage Service (S3) in AWS. OpenStack provides the essential components for public, private, community, or hybrid cloud services. Lock-in risks associated with other proprietary platforms are reduced with OpenStack, and target-based optimization and scalable service expansion are possible. It also offers dashboard or command-line products for process control, storage, and networking.

As an infrastructure service that runs with 28 subprojects based on the Wallaby, the study installed seven projects to configure IaaS. An authentication management identity service, an operating system management image service on virtual machines (VMs), a networking service that creates and manages the virtual network, and a compute service that creates and manages virtual computing resources are all required for configuring an OpenStack service. In addition, the Block storage service and the shared file system service, or services use additional storage and are used for storing and sharing image data during internal processing in this study. The dashboard is a web-based user interface that enables users to request VM creation and management.
Table 1 shows OpenStack components as IaaS used in this implementation. The Nova project, which is responsible for virtualizing IaaS systems, consists of 13 subservices, including API, compute, and scheduler. It uses Kernel-based VM technology to allow a Linux operating system to create independent virtual environments. To drive each system, Nova creates several independent VMs. Because it can adjust the scale depending on future situations, it is built as a separate VM. When deployed as a separate VM, cloud virtual network technology is also required. The neutron project makes the OpenStack network possible. This provides the ability to create and manage networks, subnets, and routers through object abstraction. Except for VMs communicating with the outside world, users can configure a secure network by configuring it as a virtual private network.

### Table 1. OpenStack components as IaaS used in this implementation.

| Subsystem or Service                  | Project Name and Version |
|---------------------------------------|--------------------------|
| Platform                              | OpenStack (Wallaby)      |
| Identity service                      | Keystone 19.0.1          |
| Image service                         | Glance 22.0.1            |
| Networking service                    | Neutron 18.1.1           |
| Compute service                       | Nova 23.0.3              |
| Block storage service                 | Cinder 18.0.1            |
| Dashboard                             | Horizon 19.2.1           |
| Shared file systems service           | Manila 12.0.1            |

#### 2.2. Absolute Atmospheric Correction Module to Generate Reflectance Products as SaaS

Among the open sources that provide atmospheric correction schemes or algorithms, the OTB provides an absolute atmosphere correction method that uses atmospheric variables. The absolute atmospheric correction method showed that the error with the field measurement of RadCalNet portal value was small, according to a comparative experiment using relative atmospheric correction and absolute atmospheric correction [46,47]. The KOMPSAT-3/3A for OTB extension was implemented [48], and the TOA and TOC reflectance products using this extension were validated with RVUS, BTCN, BSCN, and LCFR data in RadCalNet [49,50]. In this study, a generation module of TOA and TOC reflectance products for KOMPSAT-3/3A of Java version was first implemented and then mounted into the SaaS, without requiring the installation of OTB or other executable modules for atmospheric calibration processes.

For a KOMPSAT-3/3A image, the following formulas compute TOA and TOC reflectance products in unit-less according to the atmospheric correction scheme [46]. Internal algorithms are based on the KOMPSAT-3/3A extension for OTB. $L_\lambda$ is the spectral radiance at the sensor aperture of the wavelength (Equation (1)). Gain and Offset are the sensor’s characteristics, and DN is the digital number.

\[
L_\lambda = Gain \times (\text{DN}) + Offset
\]  

(1)

The TOA and TOC reflectance of the band wavelength of KOMPSAT 3/3A uses the Equations (2) and (3).

\[
\rho_{\lambda, \text{TOA}} = \frac{\pi L_\lambda d^2}{(ESUN_{\lambda}) \cos \theta_s}
\]  

(2)

where $\rho_{\lambda}$ represents the TOA reflectance on the satellite image band wavelength, and $d$, $ESUN_{\lambda}$, and $\theta_s$ represent the Earth–sun distances, the mean solar spectral irradiance, and the solar zenith angle, respectively [38].

\[
\rho_{\text{TOC}} = \frac{\rho_{\text{TOA}} - \rho_{\text{atm}}}{1 + S \frac{\rho_{\text{TOA}} - \rho_{\text{atm}}}{\tau_{\mu_s} \tau_{\mu_r} h_S}}
\]  

(3)
Here, $\rho_{TOC}$ and $\rho_{TOA}$ are the reflectance of TOC and TOA, respectively, and $\rho_{atm}$ represents the intrinsic atmospheric reflectance. $S$ is a correction constant, and $T(\mu_s)$, $T(\mu_v)$, and $t_g$ represent downward transmittance, upward transmittance, and the albedo of the atmosphere, respectively [46].

The products of TOA and TOC reflectance for KOMPSAT-3/3A of Java version were quite identical to those of KOMPSAT-3/3A for the OTB extension. In this implementation, inputs required for the absolute atmospheric correction are automatically read in the Open Source Software Image Map metadata file and used for processing immediately. In addition, it integrates necessary input parameters, including geometric information and atmospheric variables on the image acquisition date, into a single file and reads them at once, and KOMPSAT-3/3A data provided in each band from the bundle into one file are processed. Then it stores them separately into a band-specific file. This SaaS makes these steps process automatically.

The 6S Version 2.1 [51], which is the simulation code used in atmospheric compensation algorithms to calculate lookup tables, was applied. Simulation values such as atmospheric reflectance, atmospheric spherical reflectance, and transmittance can be obtained by entering satellite and atmospheric-related values for surface reflectance processing. Calculating radiance values using gain and bias values for atmospheric and surface reflectance calculations, calculated from the image acquisition date, and calculated values that apply atmospheric correction values extracted from the 6S model to surface reflectance calculations are among the implemented modules on the internal level. Atmospheric reflectance processing involves calculating radiance values from the original image and then calculating them as atmospheric reflectance values. Surface reflectance, TOC reflectance, is produced as three steps of atmospheric correction after producing TOA reflectance.

2.3. Open-Source Stack for Configuration and Deployment of SaaS

The SaaS designed in this study consists of three subsystems: a Calibration Processing System (CPS), a Request System for CPS (RSCS), and a Web Interface Application System (WIAS). CPS performs actual reflectivity processing, and WIAS configures the user interface that users access, whereas RSCS is a Representational State Transfer (REST) full API server that manages the relay role and concurrent processing data from WIAS and CPS. Because it is all composed of separate VMs, internal communication is performed during processing. The system operating system used Ubuntu Server. Table 2 shows the open source list for CPS, RSCS, and WIAS included in the SaaS.

The Java programming language is used to develop the CPS. Most languages have a framework that makes it easier to use by incorporating various functions into development. The Spring framework, which is based on the inversion of control and the Spring Boot, is used in this SaaS. The Spring framework (https://spring.io/projects/spring-framework, accessed on 26 July 2021), an open-source application framework for the Java platform, is a lightweight solution that provides comprehensive capabilities for developing enterprise-class applications. This is a lightweight container that contains Java objects and manages them directly.
Table 2. List of open source for three subsystems of CPS, RSCS, and WIAS included in SaaS.

| Subsystem or Implementation Task | Applied Environment, Open-Source Projects, Execution Modules, and Version |
|----------------------------------|--------------------------------------------------------------------------------------------|
| Calibration Processing System (CPS) | **Server operating system** Ubuntu server cloud image 20.04 LTS  
**Web server container** Apache Tomcat 9.0.5  
**Runtime environment** Open JRE 1.8  
**Programming language** Java 1.8  
**Web framework** Spring Boot (Spring framework) 2.4.0 (5.3.1)  
**I/O library for remote sensing** GDAL 2.4.0  
**Simulation of a satellite signal** 6S 2.1 |
| Request System for Calibration Processing System (RSCS) | **Server operating system** Ubuntu 20.04 LTS  
**Web server container** Apache Tomcat 9.0.5  
**Runtime environment** Open JRE 1.8  
**Programming language** Java 1.8  
**Web framework** Spring Boot (Spring framework) 2.4.0 (5.3.1)  
**Security library** Spring Security 5.4.1  
**Processing messaging protocol** Spring Boot Websocket 2.4.0 (WebSocket RFC 6455)  
**Database (System Information Storage)** PostgreSQL 10  
**Database access mapping** Spring Data Java Persistence API (JPA) 2.4.1  
**Processing scheduling library** Quartz Scheduler 2.3.2 |
| Web Interface Application System (WIAS) | **Server Application** Apache HTTP Server 2.4  
**Programming language** HTML 5  
**Scripting language** JavaScript-based on ECMA 2015  
**Styling language** CSS 3  
**Request messaging protocol library** SocketJS 1.5.0  
**Document manipulation library** jQuery 4.5.3  
**Document styling library** Bootstrap 3.5.1  
**Web mapping library** OpenLayers 6.4.3 |

For communication with RSCS, the Open Java Runtime Environment (JRE) was used as the runtime environment for Java-based project operation. It was built to run on Apache Tomcat containers. The open-source software and libraries used directly to generate reflectance products are Geospatial Data Abstraction Library (GDAL) and 6S. GDAL is a C++-based library that requires a Java Native Interface (https://gdal.org/, accessed on 26 July 2021). GDAL is responsible for input and output processing of satellite image sets, including RGB composition for web visualization and tile map creation for Tile Map Service (TMS).

The RSCS as the RESTful API system, which serves as a web interface and an internal processing system relay, is also built on a Java environment, so the basic configuration is the same as the CPS. The system is in charge of processing and managing requests. As a result, communication and message exchange technologies, as well as systemic security and data storage technologies, are employed. User management was handled by Spring security, and real-time communication was handled by WebSocket technology. PostgreSQL was used to store and manage user information and processing information, and a Quartz scheduler was used to manage schedules for processing. When a user requests data processing, the RSCS requests processing to the CPS, and the processing situation is configured to check the Quartz scheduler for a certain period and send the situation to the user via WebSocket.

The WIAS is a web-based application that users access and has built-in HTML, JavaScript, and Cascading Style Sheets. The Apache HTTP Server was used to run the web-based system and the jQuery; Bootstrap library was used to configure the interface. It also made use of OpenLayers, which includes web mapping technology. For real-time communication with RSCS, Web-Socket technology used the SocketJS library. SocketJS includes technology that allows clients to automatically determine whether WebSocket support is available or not when communicating with the RSCS-based Spring framework.
3. Concept and Architecture of SaaS

The SaaS provides applications directly through the web, so the user environment is essential. External data are sometimes required for the process of obtaining results through SaaS. The SaaS implemented in this study is provided as an open environment that users with KOMPSAT-3/3A images can use. The KOMPSAT-3/3A imageries are not free or open data, and searching and ordering are carried out through the website (https://ksatdb.kari.re.kr/arirang/map/map.jsp, accessed on 26 July 2021) by KARI. If the user has a Level 1G (geo-metrically corrected version) bundle image, use the SaaS system (https://hansung.engintruder.com/rsapp, accessed on 26 July 2021). Any users can use this service for free immediately after applying for an account. The SaaS requires input data about atmospheric conditions on the image acquisition date to obtain surface reflectance. Users can search for these data on the AERONET portal (https://aeronet.gsfc.nasa.gov, accessed on 26 July 2021), and if there are any, they can download them and directly input a value. If there are no such data, it will be treated as a default value provided by the SaaS (Figure 1).

Figure 1. SaaS application view from the user side for KOMPSAT-3 (K3) and KOMPSAT-3A (K3A) bundle image sets.

Figure 2 shows cloud computing technology’s overall configuration and processing flow to provide SaaS on OpenStack-based IaaS. The SaaS is a kind of web application where users use processing functions to generate reflectance products without installation and process storage on the cloud environment. For configuration, four VMs, including shared storage, were created using IaaS. Three out of four are VMs for RSCS and CPS processing and VM for shared storage, and they cannot be accessed directly from the outside. Application servers are deployed through Spring Boot inside the RSCS and CPS VMs for processing middleware and process request. VMs for data storage connect to all VMs in network-based storage, sharing as a space for storing data and processed data for user processing. Micro-services architecture is a system configuration that uses VMs or containers for each small functionality. It can be flexibly configured on the basis of future users and utilization to provide services more safely.
Figure 2. Linkage of OpenStack-based IaaS and SaaS.

The generation system for reflectance products for KOMPSAT-3/3A consists of a calibration system named CPS and RSCS supporting RESTful APIs. The RESTful API system includes security, request, and data management functions, whereas the calibration system is a modular set that handles atmospheric calibration. Both systems were built using the Java language and operate on Open Java Development Kit (JDK) and Open JRE 1.8 and later. Open JDK is used in development because it includes well-organized tools needed for development, and JRE is used in the operation because it includes an environment for running Java, and it is included in the JDK. JDK and JRE come in a variety of versions, and Open JDK and JRE can be installed and used on any operating system without the need for a license. Each language has a framework that is being used for utilization and scalability. In Java, Spring frameworks are widely used worldwide, and Spring boot, which was created to increase convenience among derived technologies, is a type of framework that has recently been widely used, so we build on it.

Both systems can run on any operating system because the Spring framework runs in a Java virtual environment. However, the CPS requires prior processing for configuring interfaces that allow flexible communication with Java, such as GDAL and 6SV routine, which are used for atmospheric calibration. The operating system is configured to detect and set the operating system automatically, which varies from one operating system to the next. Additionally, the operating system must perform some preliminary work. Therefore, the CPS was designed and built to support Linux and Windows, considering future server and cloud system developments.

As shown in Figure 3, the processing modules for CPS consists of six individual ones. As the main features, the parameter management module is a module that manages all the values required for processing. For atmospheric calibration, there are many values that users need to input themselves. There are two types of metadata: metadata and atmospheric condition. The date, gain, bias, solar irradiance, azimuth angle, and elevation angle are all included in the metadata.
Meanwhile, atmospheric condition contains ozone, water vapor, atmospheric pressure, and aerosol type. The KOMPSAT-3/3A metadata is provided as a separate file formatted in the Extensible Markup Language. The SaaS implemented in this study automatically detects the value required for processing and satellite sensors when a user enters a metadata file with four bands of blue, green, red, and NIR for atmospheric calibration and uses the Relative Spectral Response (RSR) files stored inside the system as objects. The GEOM file creation function determines the entered value for the processing by generating a GEOM file as each input file name. TOA processing can begin after both satellite sensor data and metadata have been entered, and TOC processing can begin only after additional atmospheric values have been entered into the TOA processing input. At this point, the atmospheric value requires user input, but because the default value is set, it can be processed without input. Especially, TOC processing requires input values associated with atmospheric conditions. This necessitates the user’s direct input, which can be replaced with an AERONET file. “Setting of atmosphere value” is a unit function that updates the value based on the input value. TOA and TOC reflectance are two different types of reflectance modules. Absolute atmospheric correction and the generation of reflectance products are referred to as reflectance.

There are three internal utilities in the CPS: AERONET, 6S, and GDAL. The utility module of AERONET automatically enters the atmospheric value required for TOC processing using a file provided by the Aerosol Robotic Network portal (https://aeronet.gsfc.nasa.gov/, accessed on 26 July 2021). The 6S module first creates the required input values to perform the solar radiation simulation and performs them to send the resulting values. For simulation performance, metadata for satellite sensor model, parameters about atmospheric conditions, aerosol type, and band-specific RSR values are entered, and the resulting values are used to calculate TOC reflectance: atmospheric reflectance, atmospheric spherical albedo, and transmittance. Finally, the GDAL utility module provides the functions to read a file for data processing or output a calculated value to a file containing geographical elements. It also serves formats for GeoTIff files, which are downloadable file formats for users and further mapping services.

Figure 4 shows the main features and processing modules for WIAS and API with the data flow. With the Command Line Interface, the CPS is ready to use right away. It is
designed to connect external interfaces to the RESTful API system so that it can be used structurally and extensively to build a service. When a system is deployed with RESTful APIs, it is available on all platforms because it is provided using the HTTP standard protocol. To process the atmospheric correction scheme and generate reflectance products, the system is designed and built with request and data request modules. Spring security is also configured to allow data access based on security level via its ability to access and authenticate data. Basic requests are provided in basic RESTful requests such as GET, POST, and DELETE.

Figure 4. Features and processing modules for WIAS and Restful API with data flow.

GET End Points are APIs for requesting data. A TMS request is provided that allows mapping in a data list, downloading, and pyramid style. Authentication and permission requests are POST requests, and authentication using IDs and passwords creates a valid token based on JSON Web Token (JWT) for a certain period, and services can be used through those tokens. An API for uploading or request processing of data is also requested as a POST type. In response to requests for processing status, asynchronous socket communication was used, taking into account user convenience. Even if the user does not wait until the end of the processing request, the server sends a message to the client if the status changes, and this asynchronous operation can be used to perform other tasks simultaneously. Finally, a request to erase data was configured as a DELETE API request. Database Management System was used to manage data in integrity, and an organizational form and PostgreSQL were used. PostgreSQL stores and processes information on status and security, allowing users to view and download historical records. RESTful API systems are all JavaScript Object Notations (JSON) in which data response formats can evolve into various types of systems in the future.

4. Implemented Results

Users on the web can access the SaaS as the implemented results to provide the application to generate TOA and TOC reflectance products at the website (https://hansung. engintruder.com/rsapp, accessed on 26 July 2021). Client views of the SaaS communicate with the APIs to perform each function offering from the application. The SaaS consists of four views: Login, Management Data, Processing, and Data. Login view uses authentication and permission APIs to authenticate users to access services. The Management Data view may upload data, inquire about processing status, check processing and data list, and delete data. A Processing view can register and request input values for processing
requests, and the Data view performs requests that can check data in a TMS. On the client side, some web technologies were used: jQuery for dynamic script simplification based on HTML5, Bootstrap for user interface, OpenLayers for map web mapping, and WebSocket for real-time communication.

Figure 5 shows the results for the JWT certification process. This means SaaS initial activation by sign-up and sign-in processes with OpenStreetMap (OSM) [52], a free and editable worldwide map released with an open-content license. The user enters the ID and password and clicks a sign-in button to invoke authentication and authorization of the API. If the request is successful, the JWT value is sent and stored in the client Cookie for a set period of time. A token value is sent for all subsequent requests to verify authentication. When a user logs out, the Cookie value is removed and the Token value is also removed, effectively disabling authentication. In a Single Page Application, the service consists of a map of the entire screen that performs the processes for data lists and processing requests through the model view.

Figure 5. SaaS activation: sign-up and sign-in processes on OpenStreetMap (OSM) as the base map.

Figure 6 shows the processes of uploading data and registering the KOMPSAT-3/3A bundle image set and internal tiling in the SaaS. First, the system selects the sensor type and then uploads a four-band image set including red, green, blue, and NIR bands and metadata files required for the subsequent processing via file multi-selection. In the case of KOMPSAT-3A, it shall be uploaded without changing the name of the file provided with data, and visualization shall be automatically processed after uploading. The server can use the Socket Messaging API to send the situation to the client whenever the processing is finished in real time. Uploading, tiling, and completion are the three stages that make up the SaaS registration process. Both visualization and download are possible once the project is complete.

Figure 7 shows the generation processes of reflectance products in the SaaS with entering atmospheric variables and listing products with the features of data visualization and downloading results. The feature of data visualization is for the result of data being produced in the form of a pyramid through tiling, which is automatically processed when uploading and is brought up in the TMS scheme. When a user clicks the view button, the user will be taken to the corresponding satellite information center position, where the user will be presented with the appropriate data based on the zoom level. There are 16 zoom levels available, ranging from 0 to 16. The datum can be downloaded by SaaS users by clicking the download button next to it. When data are requested to be downloaded, the server compresses the data for download automatically. When the compression is completed, the download is automatically executed, and the files processed on the server are saved for 5 days and then automatically deleted if there is no request.
Figure 6. Registration of KOMPSAT-3/3A bundle image set in the SaaS and internal tiling process.

Figure 7. Generation processes of TOA and TOC reflectance products in the SaaS: entering atmospheric variables and listing products.

Figure 8 represents the product view, downloading, and deletion processes in the SaaS and the input modal view for atmospheric calibration processing requests. Calibration processing requests are divided into TOA and TOC reflectance. For TOA reflectance, only 1.4 and 1.5 versions need to be selected to set the gain and bias values according to the launch date. The 1.4 and 1.5 versions were released in November 2017 and June 2019, respectively. The calibration processing module already has both gain and bias values for each version, so it is automatically entered when only the version is selected. For TOC reflectance, input data regarding atmospheric conditions are required.
four interfaces to enter atmospheric input values: Atmospheric Pressure, Water Vapor, Ozone Amount, and Aerosol Optical Depth. While, as an optional function, users can upload separate 1-month data as the AERONET file. Without users’ actions, the system automatically checks the date of the image acquisition and uses the atmospheric parameters for TOA and TOC reflectance calculations. While the atmospheric model values support four values: No Aerosol, Continental, Maritime, Urban, and Desert. Like the uploading process, information about the processing step is sent from the server in real time through the Socket Messaging API. The processing sequentially proceeds according to Ready, Processing, Tiling, and Complete processes, and when all processing is completed, mapping visualization is possible through View in the data list. Users can also download processed data.

Figure 8. Product view, downloading, and deletion processes in the SaaS.

Experiments were conducted to verify the performance of cloud services. Figure 9 shows the result of the performance test of SaaS execution time. The experimental data used both KOMPSAT-3/3A image sets. For KOMPSAT-3, the size of the data used in the experiment were 97–108 MB for one band, with bundle data containing four bands ranging from 308 to 432 MB, and the SaaS produced a bar-reflection diagram for four bands for these bundle image data. The size of KOMPSAT-3/3A bundle images applied in the experiment was between 436 and 444 MB. The computer specifications used in the experiment were Intel(R) Xeon(R) CPU E5-2630 v4 = 2.20 GHz (core 20), 64 GB of memory, and SM.2 SSD 512 GB of storage. Of course, this experiment can perform better if the physical computer specifications that make up the cloud environment are better than these conditions.

Figure 9. Result of the performance test of the SaaS execution time.

5. Discussion

Cloud computing technologies are considered a key technology for the high-performance processing of large EO satellite imageries. In most cases handling a huge volume of EO image sets, most cloud services are deployed in proprietary cloud environments. However, there is also a demand for micro-web services that provide specific functionalities or processing algorithms while dealing with an image set, bundled red, green, blue, and NIR bands. The SaaS, which builds and operates in an open source-based cloud
environment, is suitable for online service systems that meet these needs. As more micro-
SaaS service systems and more users in the public domains, the services that provide
these capabilities may link to a hybrid cloud service covering large-scaled EO satellite
information. This study develops the SaaS model to produce TOA and TOC reflectance
images for KOMPSAT-3/3A satellite images, built under an open source-based cloud
environment. Therefore, this kind of micro-SaaS, which provides these reflectance products
of other high-resolution satellite sensors and other processing algorithms, is needed in
the future.

There are several points for the broad utilization of these services. The first is that
high-resolution images such as KOMPSAT-3/3A satellite images are primarily commercial,
unlike Landsat and Sentinel-2 image sets. In the case of WorldView series images, surface
reflectance produced by atmospheric compensation algorithms of Maxar has been provided
as value-added products. However, in the case of KOMPSAT-3/3A images, such products
are not yet provided, so users must use tools or develop processing schemes to generate
surface reflectance products for their application purposes. However, any tool with built-in
algorithms for reflectance products of KOMPSAT-3/3A satellite images through absolute
atmospheric calibration algorithms has not been officially released yet. The OTB extension
with the result validation by [46–48] is the only case till now. This implementation of
this SaaS model does not need OTB, and users on a web browser can utilize this function
for reflectance products of KOMPSAT-3/3A in a cloud environment. The second is that
the processing to generate reflectance products is performed automatically in a cloud
environment, but users may input data directly if they do not want default parameters.
In that case, users must retrieve and enter data regarding atmospheric conditions in the
AERONET data directly. However, if the measuring station providing the AERONET
data is far from the area of interest of the KOMPSAT-3/3A image that users want, the
result accuracy of reflectance products that may be lower than the exact atmospheric value
cannot be entered. Therefore, the user should check the location of the AERONET
station when using KOMPSAT-3/3A images. The third is for interfacing issues for cloud services.
The SaaS allows users to process data directly in a web environment, which is internally
connected to IaaS. This SaaS can also be linked to other PaaS or IaaS and can be a mixed
cloud service. In this case, applying the OGC-based interoperable interfaces is necessary
to enable more stability and efficiency between cloud systems and cloud components.
This also contributes to the system’s expansion. The service system’s user accessibility
is the fourth factor to consider. Currently, the SaaS in this study is in the form of a trial
version, with no restrictions on who can access it. It even has the capability of downloading
reflectance products, which are the result of user processing. If the data providers or
organizations with authorities of KOMPSAT-3/3A data properties use and operate the
system, it may be limited for the end-user to utilize and distribute the processing results
online freely. The fifth is about system maintenance. Because the SaaS is built on an
open-source basis, the other related open-source resources must also check the version if
one open-source component is updated. In other words, even if a critical element improves
performance and tries to renew this feature, an entire system renewal may not take place
immediately if other open sources associated with this feature do not support this new
version. This is also directly related to data processing schemes or algorithms to provide
reliable output continuously. The sixth is regarding SaaS extension. This SaaS is based on
modules and components, so it can support more optical satellite sensor models that want
reflectance results as well as generating functions for reflectance products of Landsat-7/8
image sets in the future. Finally, for user-entered data, this SaaS system does not require or
own the same authority as intellectual property rights or copyrights. The user owns the
reflectance products, which are the result of the processing. The limitation of the SaaS is
that it cannot be used if Web access is not easy because it works in a Web environment.
Furthermore, because KOMPSAT-3/3A satellite images are not supplied free of charge,
they can only be used by users who own them. If copyright issues regarding KOMPSAT-
6. Conclusions

The importance of cloud computing technology in the development of EO satellite image management and distribution systems is currently being highlighted. SaaS, which has built-in IaaS and PaaS environments and provides services directly to users, is one of three types of cloud service models. IaaS and PaaS are a set of technologies that are used to configure infrastructure or platforms and are hidden from users. They are part of the developer’s domain.

In this study, we designed and implemented the SaaS model, allowing users to obtain real-time calculated reflectance results through direct web services without installing separate tools for high-resolution KOMPSAT-3/3A satellite images. During the development process, open source, including OpenStack for the IaaS environment, was applied to the entire environment from IaaS. The Spring framework to provide comprehensive capabilities for the Java platform was also applied. This research is significant because it is the first time SaaS with expertise in processing high-resolution EO satellite images has been implemented. It also lists open-source stacks that are relevant to the development in a systematic manner. Reflectance products derived from absolute atmospheric correction processing using atmospheric condition data are important data that can be used as primary data for deep learning, vegetation index calculation, and precise image classification. This service can produce the most accurate results if they have the correct input to achieve more accurate results because this SaaS provides a user interface that allows users to enter the necessary data themselves. The SaaS is designed and implemented as an application structure which may be applied to other types of multi-spectral image sets of high-resolution satellite sensor models with spatial resolution of less than 5 m, as well as Landsat-8 or Sentinel-2. We will continue to enhance the scheme applied to this SaaS and verify the results obtained from the SaaS.

The advantage of SaaS is that users can take advantage of certain features through a web browser and store the outputs gained from it in their local storage without installing them separately. Certain services can be provided regardless of the operating system or the computer environment used by the user, according to service providers. Furthermore, system maintenance is conducted in the background, which is beneficial for long-term service maintenance. The SaaS model provides such analysis functions and can be applied to public institutions and industries that provide satellite information analysis services.

Author Contributions: Conceptualization, K.L.; methodology, K.L.; software, K.L., K.K.; investigation and analysis, K.L.; writing, K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was financially supported by Hansung University for Kiwon Lee. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1045923) for Kwangseob Kim.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank the anonymous reviewers for their careful reading of our manuscript and their comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Different Types of Cloud Computing Service Models. Available online: https://www.bluepiit.com/blog/different-types-of-cloud-computing-service-models/ (accessed on 26 June 2021).
2. Choosing the Right Cloud Service: IaaS, PaaS, or SaaS. Available online: https://rubygarage.org/blog/iaas-vs-paas-vs-saas (accessed on 26 June 2021).
3. Advantages of Cloud Computing in Remote Sensing Applications. Available online: https://www.l3harrisgeospatial.com/Learn/Blogs/ArtMID/10198/Advantages-of-Cloud-Computing-in-Remote-Sensing-Applications (accessed on 26 June 2021).

4. ENVI in the Cloud. Available online: https://www.l3harrisgeospatial.com/Learn/Whitepapers/ArtMID/17811/ENVI-in-the-Cloud (accessed on 26 June 2021).

5. Yang, C.; Huang, Q. Spatial Cloud Computing: A Practical Approach; CRC Press: Boca Raton, FL, USA, 2014.

6. Wang, L.; Yan, J.; Ma, Y. Cloud Computing in Remote Sensing; CRC Press: Boca Raton, FL, USA, 2020.

7. Yue, P.; Zhou, H.; Gong, J.; Hu, L. Geoprocessing in cloud computing platforms—A comparative analysis. Int. J. Digit. Earth 2013, 6, 404–425. [CrossRef]

8. Amani, M.; Ghorbanian, A.; Ahmadi, S.A.; Kakooei, M.; Armin Moghimi, A.; Mirmazloumi, S.M.; Moghaddam, S.H.A.; Mahdavi, S.; Gahremanloo, M.; Parsian, S.; et al. Google Earth Engine Cloud Computing Platform for Remote Sensing Big Data Applications: A Comprehensive Review. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 2020, 13, 5326–5350. [CrossRef]

9. Yue, P.; Zhou, H.; Gong, J.; Hu, L. Geoprocessing in cloud computing platforms—A comparative analysis. Int. J. Digit. Earth 2013, 6, 404–425. [CrossRef]

10. Yang, Z.; Li, W.; Chen, Q.; Wu, S.; Liu, S.; Gong, J. A scalable cyberinfrastructure and cloud computing platform for forest monitoring. Remote Sens. 2020, 12, 62. [CrossRef]

11. Kopp, S.; Becker, P.; Doshi, A.; Wright, D.J.; Zhang, K.; Xu, H. Achieving the Full Vision of Earth Observation Data Cubes. Data 2019, 4, 94. [CrossRef]

12. Kopp, S.; Becker, P.; Doshi, A.; Wright, D.J.; Zhang, K.; Xu, H. Achieving the Full Vision of Earth Observation Data Cubes. Data 2019, 4, 94. [CrossRef]

13. Schramm, M.; Pebesma, E.; Milutin Milenković, M.; Luca Foresta, L.; Jeroen Dries, J.; Jacob, A.; Wagner, W.; Mohr, M.; Neteler, M.; Kadunc, M.; et al. The openEO API—Harmonising the Use of Earth Observation Cloud Services Using Virtual Data Cube Functionalities. Remote Sens. 2021, 13, 1125. [CrossRef]

14. Huang, W.; Zhou, J.; Zhang, D. On-the-Fly Fusion of Remotely-Sensed Big Data Using an Elastic Computing Paradigm with a Containerized Spark Engine on Kubernetes. Sensors 2021, 21, 2971. [CrossRef]

15. Schramm, M.; Pebesma, E.; Milutin Milenković, M.; Luca Foresta, L.; Jeroen Dries, J.; Jacob, A.; Wagner, W.; Mohr, M.; Neteler, M.; Kadunc, M.; et al. The openEO API—Harmonising the Use of Earth Observation Cloud Services Using Virtual Data Cube Functionalities. Remote Sens. 2021, 13, 1125. [CrossRef]

16. Hausner, M.B.; Huntington, J.L.; Nash, C.; Morton, C.; McEvoy, D.J.; Pilliod, D.S.; Hegewisch, K.C.; Daudert, B.; Abatzoglou, T.J.; Grant, G. Assessing the effectiveness of riparian restoration projects using Landsat and precipitation data from the cloud-computing application ClimateEngine.org. Ecol. Eng. 2018, 120, 432–440. [CrossRef]

17. Kline, K. USGS Landsat Migration to the Cloud, Presentation Material in CEOS WGISS-49 Meeting. 2021. Available online: https://ceos.org/meetings/wgiss-51/ (accessed on 26 June 2021).

18. Shaharum, N.S.N.; Shafri, H.Z.M.; Ghani, W.A.W.A.K.; Samsatli, S.; Prince, H.M.; Yusuf, B.; Hamud, A.M. Mapping the spatial distribution and changes of oil palm land cover using an open source cloud-based mapping platform. Int. J. Remote Sens. 2019, 40, 7459–7476. [CrossRef]

19. Yang, Z.; Li, W.; Chen, Q.; Wu, S.; Liu, S.; Gong, J. A scalable cyberinfrastructure and cloud computing platform for forest aboveground biomass estimation based on the Google Earth Engine. Int. J. Digit. Earth 2019, 12, 995–1012. [CrossRef]

20. Antunes, R.R.; Blaschke, T.; Tiede, D.; Biais, E.S.; Costa, G.A.O.P.; Happ, P.N. Proof of concept of a novel cloud computing approach for object-based remote sensing data analysis and classification. Gisci. Remote Sens. 2019, 56, 536–553. [CrossRef]

21. Sagan, V.; Peterson, K.T.; Maimaitijiang, M.; Sidike, P.; Sloan, J.; Greeling, B.A.; Maalouf, S.; Adams, C. Monitoring in-land water quality using remote sensing: Potential and limitations of spectral indices, bio-optical simulations, machine learning, and cloud computing. Earth-Sci. Rev. 2020, 205, 103187. [CrossRef]

22. Paludo, A.; Becker, W.R.; Richetti, J.; Cavalcante, L.; Silva, D.A.; Johann, J.A. Mapping summer soybean and corn with remote sensing on Google Earth Engine cloud computing in Parana state—Brazil. Int. J. Digit. Earth 2020, 13, 1624–1636. [CrossRef]

23. Singh, M.; Singh, B.B.; Singh, R.; Upendra, B.; Kaur, R.; Gill, S.S.; Biswas, M.S. Quantifying COVID-19 enforced global changes in atmospheric pollutants using cloud computing based remote sensing. Remote Sens. Appl. Soc. Environ. 2021, 22, 100489. [CrossRef]

24. Hardy, T.; Kooistra, L.; Franceschini, M.D.; Richter, S.; Vonk, E.; Eertwegh, G.; Deijl, D. Sen2Grass: A Cloud-Based Solution to Generate Field-Specific Grassland Information Derived from Sentinel-2 Imagery. AgriEngineering 2021, 3, 118–137. [CrossRef]
29. Cloudeo. Available online: https://www.cloudeo.group/ (accessed on 26 June 2021).
30. Brovellia, M.A.; Minghinia, M.; Moreno-Sanchezb, R.; Oliveira, R. Free and open source software for geospatial applications (FOS4G) to support Future Earth. Int. J. Digit. Earth 2017, 10, 386–404. [CrossRef]
31. Hütt, C.; Waldhoff, G.; Bareth, G. Fusion of Sentinel-1 with Official Topographic and Cadastral Geodata for Crop-Type Enriched LULC Mapping Using FOSS and Open Data. ISPRS Int. J. Geo-Inf. 2020, 9, 120. [CrossRef]
32. Coetzee, S.; Ivánová, I.; Mitasova, H.; Brovelli, M.A. Open Geospatial Software and Data: A Review of the Current State and a Perspective into the Future. ISPRS Int. J. Geo-Inf. 2020, 9, 90. [CrossRef]
33. Duarte, L.; Teodoro, A.C. GIS Open-Source Plugins Development: A 10-Year Bibliometric Analysis on Scientific Literature. Geomatics 2021, 1, 206–245. [CrossRef]
34. Keighan, E.; Pross, B.; Caumont, H. (Eds.) Testbed 10 Performance of OGC® Services in the Cloud: The WMS, WMTS, and WPS Cases; OGC 14-028r1; Open Geospatial Consortium Inc.: Wayland, MA, USA, 2014.
35. Percivall, G. (Ed.) The Role of Geospatial in Edge-Fog-Cloud Computing—An OGC White Paper, OGC 18-004r1. Available online: https://docs.opengeospatial.org/wp/18-004r1/18-004r1.html (accessed on 26 July 2021).
36. OGC Testbed-14: Federated Clouds Engineering Report, OGC 18-090r1. Available online: http://docs.opengeospatial.org/per/18-090r1.html (accessed on 26 July 2021).
37. The Most Widely Deployed Open Source Cloud Software in the World. Available online: https://www.openstack.org/ (accessed on 26 July 2021).
38. Kang, S.; Lee, K. Testing Implementation of Remote Sensing Image Analysis Processing Service on OpenStack of Open Source Cloud Platform. J. Korean Assoc. Geogr. Inf. Stud. 2013, 16, 141–152. [CrossRef]
39. Lee, K.; Kim, K. Auto-Scaling of Geo-Based Image Processing in an OpenStack Cloud Computing Environment. Remote Sens. 2016, 8, 662. [CrossRef]
40. Lee, K.; Kang, S.; Kim, K.; Chae, T.-B. Cloud-based Satellite Image Processing Service by Open Source Stack: A KARI Case. Korean J. Remote Sens. 2017, 33, 339–350. [CrossRef]
41. Yoon, G.; Kim, K.; Lee. K. Linkage of OGCP WPS 2.0 to the e-Government Standard Framework in Korea: An Implementation Case for Geo-Spatial Image Processing. ISPRS Int. J. Geo-Inf. 2017, 6, 25. [CrossRef]
42. Lee, K.; Kim, K. A Performance Evaluation of a Geo-Spatial Image Processing Service Based on Open Source PaaS Cloud Computing Using Cloud Foundry on OpenStack. Remote Sens. 2018, 10, 1274. [CrossRef]
43. Orfeo Toolbox. Open Source Processing of Remote Sensing Images. Available online: https://www.orfeo-toolbox.org/ (accessed on 26 July 2021).
44. Duarte, L.; Silva, P.; Teodoro, A.C. Development of a QGIS Plugin to Obtain Parameters and Elements of Plantation Trees and Vineyards with Aerial Photographs. ISPRS Int. J. Geo-Inf. 2018, 7, 109. [CrossRef]
45. De Luca, G.; Silva, J.M.N.; Cerasoli, S.; Araújo, J.; Campos, J.; Di Fazio, S.; Modica, G. Object-Based Land Cover Classification of Cork Oak Woodlands using UAV Imagery and Orfeo Toolbox. Remote Sens. 2019, 11, 1238. [CrossRef]
46. Lee, K.; Kim, K.; Lee, S.; Kim, Y. Determination of the Normalized Difference Vegetation Index (NDVI) with Top-of-Canopy(TOC) Reflectance from a KOMPSAT-3A Image Using Orfeo Toolbox (OTB) Extension. ISPRS Int. Geo-Inf. 2020, 9, 257. [CrossRef]
47. Lee, K.; Kim, K. Validation of Surface Reflectance Product of KOMPSAT-3A Image Data Using RadCalNet Data. Korean J. Remote Sens. 2020, 36, 167–178, (In Korean with English abstract). [CrossRef]
48. Lee, K.; Kim, K. An Experiment for Surface Reflectance Image Generation of KOMPSAT 3A Image Data by Open Source Implementation. Korean J. Remote Sens. 2019, 35, 1327–1339, (In Korean with English abstract). [CrossRef]
49. Kim, K.; Lee, K. Validation of Surface Reflectance Product of KOMPSAT-3A Image Data Application of RadCalNet Bao-tou (BTCN) Data. Korean J. Remote Sens. 2020, 36, 1509–1521, (In Korean with English abstract). [CrossRef]
50. Kim, K.; Lee, K. A Validation Experiment of the Reflectance Products of KOMPSAT-3A Based on RadCalNet Data and Its Applicability to Vegetation Indexing. Remote Sens. 2020, 12, 3971. [CrossRef]
51. Satellite Agriculture & Land Surface Applications. Available online: https://salsa.umd.edu/6spage.html (accessed on 26 July 2021).
52. OpenStreetMap. Available online: https://www.openstreetmap.org (accessed on 26 July 2021).