*Remote Sens.***2015**, *7*, 7646-7670; doi:10.3390/rs70607646

OPEN ACCESS

remote sensing

ISSN 2072-4292

www.mdpi.com/journal/remotesensing

Article

Provenance Information Representation and Tracking for Remote Sensing Observations in a Sensor Web Enabled Environment

Zeqiang Chen [1,2] and Nengcheng Chen [1,2,*]

[1] Collaborative Innovation Center of Geospatial Technology, Wuhan 430079, China; E-Mail: czq0119@gmail.com

[2] State Key Laboratory for Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China

* Author to whom correspondence should be addressed; E-Mail: cnc@whu.edu.cn; Tel.: +86-27-6877-9996; Fax: +86-27-6877-8229.

Academic Editors: Lizhe Wang and Prasad S. Thenkabail

Received: 3 January 2015 / Accepted: 1 June 2015 / Published: 9 June 2015

Abstract: The provenance of observations from a Sensor Web enabled remote sensing application represents a great challenge. There are currently no representations or tracking methods. We propose a provenance method that represents and tracks remote sensing observations in the Sensor Web enabled environment. The representation can be divided into the description model, encoding method, and service implementation. The description model uses a tuple to define four objects (sensor, data, processing, and service) and their relationships at a time point or interval. The encoding method incorporates the description into the Observations & Measurements specification of the Sensor Web. The service implementation addresses the effects of the encoding method on the implementation of Sensor Web services. The tracking method abstracts a common provenance algorithm and four algorithms that track the four objects (sensor, data, processing, and service) in a remote sensing observation application based on the representation. We conducted an experiment on the representation and tracking of provenance information for vegetation condition products, such as the Normalized Difference Vegetation Index (NDVI) and the Vegetation Condition Index (VCI). Our experiments used raw Moderate Resolution Imaging Spectroradiometer (MODIS) data to produce daily NDVI, weekly NDVI, and weekly VCI for the 48 contiguous states of the United States, for May from 2000 to 2012. We also implemented inverse tracking. We evaluated the time and space requirements of the proposed method in this scenario. Our results show that this technique provides a solution for determining provenance information in remote sensing observations.

Keywords: provenance; remote sensing observation; Sensor Web; Normalized Difference Vegetation Index (NDVI); Vegetation Condition Index (VCI)

1. Introduction

The Sensor Web is an infrastructure that establishes a bridge between sensor resources and applications through a series of information models and interface specifications defined by the Open Geospatial Consortium (OGC) [1]. The Sensor Web has been extensively used in remote sensing applications, where observations often undergo complex processing between their origin and data product [2–7]. The provenance of remote sensing products is important when tracing their histories, validating their trustworthiness, and analyzing their qualities. Moreover, it is particularly important to track the four aspects (sensor, processing, data, and service) [8–10] that are associated with the following frequently asked questions. Which sensor was used to observe these remote sensing observations, what method or algorithm was applied to these remote sensing observations to generate the products, what remote sensing observations were processed to create these products, and what were the services and processing related to these observations? Here, an observation is an act of measuring or otherwise determining the value of a property [1], a sensor is a mechanical device used to obtain a remote sensing observation, data are similar to products, a processing is a method or algorithm that produces remote sensing data, and a service is a Web service. We have focused on the provenance of these four objects. However, it can be hard to capture provenance information in the Sensor Web enabled environment for remote sensing observations, because there is a lack of representation and tracking methods in loosely-coupled and standards-based service environments [8].

With respect to provenance, researchers have investigated database, workflow/Web, international specification, and sensor network systems. Database methods were the first to be considered. Researchers have focused on transformations performed on a view, a table, or an item in a database, and the inverse provenance method [11–14]. Data provenance within a workflow/Web environment records the workflow process and describes the Web data [15–19]. Some international specifications have been proposed for building open provenance models (OPMs). An OPM defines an open data model from an inter-operability viewpoint, with respect to the community of contributors, reviewers, and users [20]. The W3C Provenance Incubator Group maps a core set of provenance vocabularies and models, including the OPM. W3C developed a general provenance data model called PROV-DM, to offer a basis for interoperability across diverse provenance management systems [21]. The ISO 19115 and ISO 19115-2 Lineage Models were proposed to break the data provenance heterogeneity and integrate resources, such as the Web Coverage Service [22,23]. A remotely sensed data processing provenance system called Karma was based on the OPM. Provenance-aware applications for sensor networks were studied in terms of naming, indexing, and mapping using the OPM method [24].

Great progress has been made with respect to provenance, but existing methods cannot be directly applied to the Sensor Web. The Sensor Web environment is a specification framework, and it must properly integrate provenance representation and tracking. Database techniques focused on what the provenance information was and how the provenance information was stored and queried in a database. However, in the Sensor Web enabled environment, it is more important to consider the standards themselves. There are issues that should be considered by the developer who concretely implements the specification. Workflow/Web methods provided some approaches for organizing provenance information, but have not effectively built provenance models that can be seamlessly integrated with current Sensor Web specifications, because they cannot fuse the provenance model and specifications. International provenance models are open and abstract models that are used to exchange provenance information between systems. However, they do not provide a domain-specific provenance model. For example, in the Sensor Web domain, OPMs and PROV-DM need concrete expressions. This requires a Sensor Web, domain-specific provenance method. Although the Sensor Web has some descriptive specification information for sensors, processes, and observations (Sensor Model Language (SensorML) [25] and Observation & Measurement (O&M) [26,27] specifications), there are no explicit representation models and tracking methods for provenance. For example, it cannot represent the provenance relationships and associated tracking method.

The objective of this paper was to propose a provenance information representation and tracking method for remote sensing observations in the Sensor Web enabled environment. We considered the provenance issue in the Sensor Web enabled environment (and not the common open remote sensing production service, Web Coverage Service) for the following reasons. (1) Sensor Web technologies have been extensively applied to remote sensing applications, and there are existing provenance issues [8]; (2) The provenance issue for the Web Coverage Service has been solved to some degree by the ISO 19115 and ISO 19115-2 Lineage Models [22]; (3) The Web Coverage Service has been applied to the inputs and outputs of the Sensor Web [4,5], so the provenance in the Sensor Web enabled environment is also associated with Web Coverage Service provenance. In this paper, we focused on four aspects (sensors, data, processing, and services) and investigated two procedures: (1) modelling the direct and implicit relationships between the sensors, data, processing, and services as a tuple descriptor, providing a representation and linkage solution for the entire remote sensing observation processing; and (2) developing an indicator that helps users trace historic information, validate trustworthiness, and analyze the quality of remotely sensed observations in their applications. The contributions of this paper are as follows.

(1) We propose a method for representing and tracking provenance information in the Sensor Web enabled environment for remote sensing applications.
(2) We tested this method by applying it to vegetation condition applications.

The remainder of this paper is organized into four sections. We demonstrate a provenance method in Section 2, and describe our method, experiments, and results in Section 3. Finally, Section 4 contains a discussion of our results and conclusions.

2. Provenance Method

In remote sensing applications, final products are created from original observations through a series of complex processing steps. Therefore, the representation of provenance information for remote sensing observations at each stage of the processing is a prerequisite for tracking the provenance of the final products. The representation refers to modeling (provenance description model), encoding (encoding method used for the provenance description model), and implementation (service implementation considering the provenance description model) in the Sensor Web enabled environment. The provenance description model formalizes provenance objects and their relationships, making it easier to represent and track provenance information. The encoding method incorporates the provenance model into Sensor Web specifications. Accordingly, the service implementation is a response to changes made by the encoding method. The ability to track provenance information depends on the content of the representation. Thus, we must: (1) define the provenance description model for remote sensing observations; (2) encode the provenance description model and incorporate it into Sensor Web specifications; (3) implement provenance-compatible Sensor Web specifications into the services; and (4) track provenance information in the Sensor Web enabled environment based on the developed representation. These four problems are referred to as the description model, encoding method, service implementation, and tracking algorithm, and are discussed further in Sections 2.1–2.4.

2.1. Description Model

The description model is the formal representation of the relationships between provenance objects for remote sensing observations. The description model is responsible for building relationships between the four objects with the following conventions. All data in a Sensor Web are encoded with O&M. The data are accessed/stored using the Sensor Observation Service (SOS) [28], and all data flow, discovery, accessing, management, and processing stages use standard Web services. These conventions guarantee that the description model was made in a "pure" Sensor Web enabled environment. As mentioned in Section 1, we focused on the provenances of data, sensor, processing, and service in a Sensor Web enabled environment. The "process" in SensorML can be a sensor or an algorithm, whereas the processing in this paper (**Definition 2.1**) refers to an algorithm. O&M states that an "observation" has a process (e.g., sensor or algorithm) and a result, whereas data in this paper (**Definition 2.1**) refers to the result. This helps to distinguish between a sensor, an algorithm, and a result, and also their provenance. Therefore, we can consider that observations build the relationships between the four objects by either direct or implicit links (**Definition 2.1**). Provenance is the utilization of available information to track historical information. Thus, it is meaningful to distinguish between available and historical information, to derive an explicit provenance scope. This distinction is defined in **Definitions 2.2** and **2.3** from the observation perspective. Finally, the observation provenance is defined in **Definition 2.4**.

Figure 1 shows an example of the observation provenance and **Definition**s **2.1–2.4.**

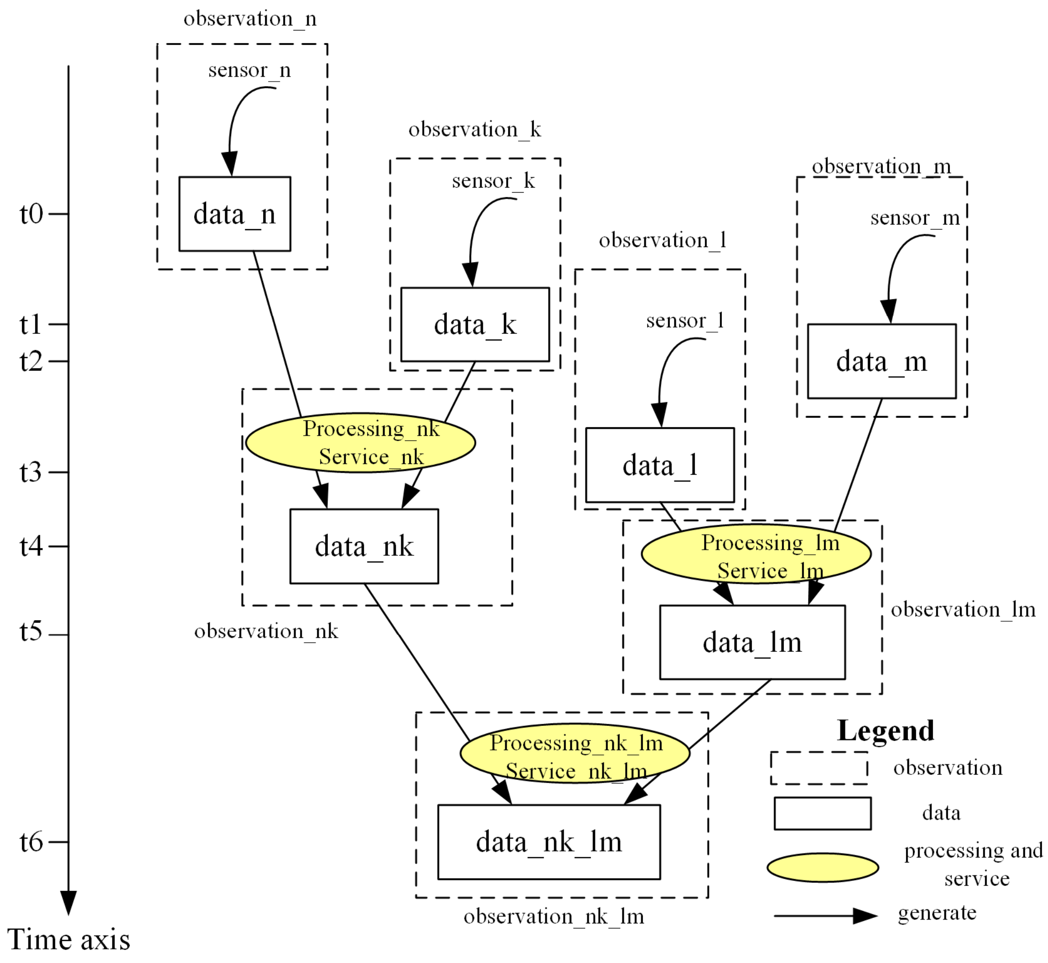


Figure 1. Example observation provenance.

Assume that we have four raw observations (directly obtained from sensors without undergoing any processing): $observation\_n$, $observation\_k$, $observation\_l$, and $observation\_m$ for diffident times. They are obtained using data ($data\_n$, $data\_k$, $data\_l$, and $data\_m$) from four sensors ($sensor\_n$, $sensor\_k$, $sensor\_l$, and $sensor\_m$) at times $t0$, $t1$, $t3$, and $t2$, respectively. Data $data\_n$ and $data\_k$ generate data $data\_nk$ at time $t4$ based on $\Pr ocessin g\_nk$ and $Service\_nk$. Similarly, $data\_l$ and $data\_m$ generate $data\_lm$ at time $t5$ based on $\Pr ocessin g\_lm$ and $Service\_lm$, and $data\_nk$ and $data\_lm$ generate $data\_nk\_lm$ at time $t6$ based on $\Pr ocessin g\_nk\_lm$ and $Service\_nk\_lm$. The purpose of an observation provenance (for example, $observation\_nk\_lm$'s provenance) is to find all related data ($data\_nk$, $data\_lm$, $data\_n$, $data\_k$, $data\_l$, and $data\_m$), their processing ($\Pr ocessin g\_nk\_lm$, $\Pr ocessin g\_nk$, and $\Pr ocessin g\_lm$), and their services ($Service\_nk\_lm$, $Service\_nk$, and $Service\_lm$) at particular time.

Definition 2.1: The **Status Observation** (*SO*) is an observation that occurs at a time point or time interval tn without any process. The time point or time interval is called the **Status Time** (*ST*, where *ST* = *tn*) of the observation. Observation SO_{tn} contains (or points to): observation entity $P_{observation\_tn}$, sensor P_{sensor_t0} or source observations $SO_{tn-1}s$ (that directly result in the *SO*), data P_{data_tn}, processing $P_{processing\_tn}$ and service $P_{service\_tn}$. That is,

$$SO_{tn} = \{P_{data\_tn}, P_{sensor\_t0}, SO_{tn-1}s, P_{service\_tn}, P_{processin g\_tn} \mid R\} \quad (1)$$

where R is the relationship ("|") between P_{sensor_t0}, $SO_{tn-1}s$, P_{data_tn}, $P_{processing\_tn}$ and $P_{service\_tn}$. When $tn = t0$, Equation (1) can be written as

$$SO_{t0} = \{P_{data\_t0}, P_{sensor\_t0} \mid R_1\} \tag{2}$$

where R_1 is a restriction defined as

$$R_1 = \{GP \xrightarrow{P_{sensor\_t0}} P_{data\_t0}\} \tag{3}$$

which indicates that a Geographical Phenomenon (*GP*) with the effect of P_{sensor_t0} results in P_{data_t0}, The symbol "$\rightarrow$" is used here to denote the phrase "results in".

When time tn is later than $t0$ ($tn - 1 \geq t0$), Equation (1) can be written as

$$SO_{tn} = \{P_{data\_tn}, SO_{tn-1}s, P_{service\_tn}, P_{processing\_tn} \mid R_2\} \tag{4}$$

where R_2 is a restriction defined as

$$R_2 = \{SO_{tn-1}s \xrightarrow{P_{processing\_tn}, P_{service\_tn}} SO_{tn}\} \tag{5}$$

which indicates that SO_{tn} is derived from $SO_{tn-1}s$ with the processing $P_{processing\_tn}$ and the service $P_{service\_tn}$.

For example, in Figure 1, observation *observation_n* for time period $[t0, t4)$ and *observation_lm* for time period $[t5, t6)$ are *SOs*.

Definition 2.2: **A Current Status Observation** (*CSO*) is a *SO* at a designated *ST*.

For example, in Figure 1, if the designated *ST* is for time period $[t4, t5)$, *observation_nk* and *observation_l* are both *CSO*s.

Definition 2.3: **A Historic Status Observation** (*HSO*) is a *SO* relative to a *CSO*. Its *ST* is before that of the *CSO*. Furthermore, the *CSO* should be directly or indirectly derived from the *HSO*.

In Figure 1, if *observation_nk_lm* is a *CSO*, *observation_nk*, *observation_lm*, *observation_n*, *observation_k*, *observation_l*, and *observation_m* are its *HSO*s.

Definition 2.4: **Observation Provenance** (*OP*) is the process of determining all of the *HSO*s of a *CSO*, that is

$$OP(CSO) = \{HSO_i, i = 0, 1, \ldots, tn-1\} \tag{6}$$

where *OP*(*CSO*) denotes the data provenance for the *CSO*. The result is the set of the *CSO's HSOs*.

These definitions can be summarized as follows. An *SO* contains the provenance information for a time point or interval. Provenance is a recursive issue. The starting point of the recurrence is a *CSO*, and this loops directly to determine its *HSOs*. If the *CSO* is a root node, then the *HSOs* are element nodes and their relationships are linked to the root node and other element nodes to form a "tree", as shown in Figure 1. Provenance, in this case, refers to a backwards trace over the whole "tree".

As previously mentioned, we consider the provenance issue in a "pure" Sensor Web enabled environment. However, the processed data may not come directly from a sensor, and may be from an impure data source (e.g., a Web Coverage Service or a data system). In this case, the provenance of the data source is not supported by the proposed method, but it can be determined by its own provenance method. For example, if the Web Coverage Service is the data source, the provenance problem can be dealt by the ISO 19115 and ISO 19115-2 Lineage Models [22]. Moreover, we can strategically map the description model to a standard provenance model. If the impure data source can follow or map a standard provenance model, the problem may be solved. We extended the formalized description of a provenance model from PROV-DM [21], as shown in Figure 2, so that the provenance model is machine-readable and capable of sharing and interoperation. The W3C provenance (PROV) family of specifications is based on the PROV-DM conceptual data model. PROV-DM distinguishes core structures, derives provenance information from extended structures, and caters to more specific uses. PROV-DM defines the objects and relationships of Entity, Activity, and Agent. Data and sensors are extensions of Entities, processing is extension of Activity, and services are extensions of Agents.

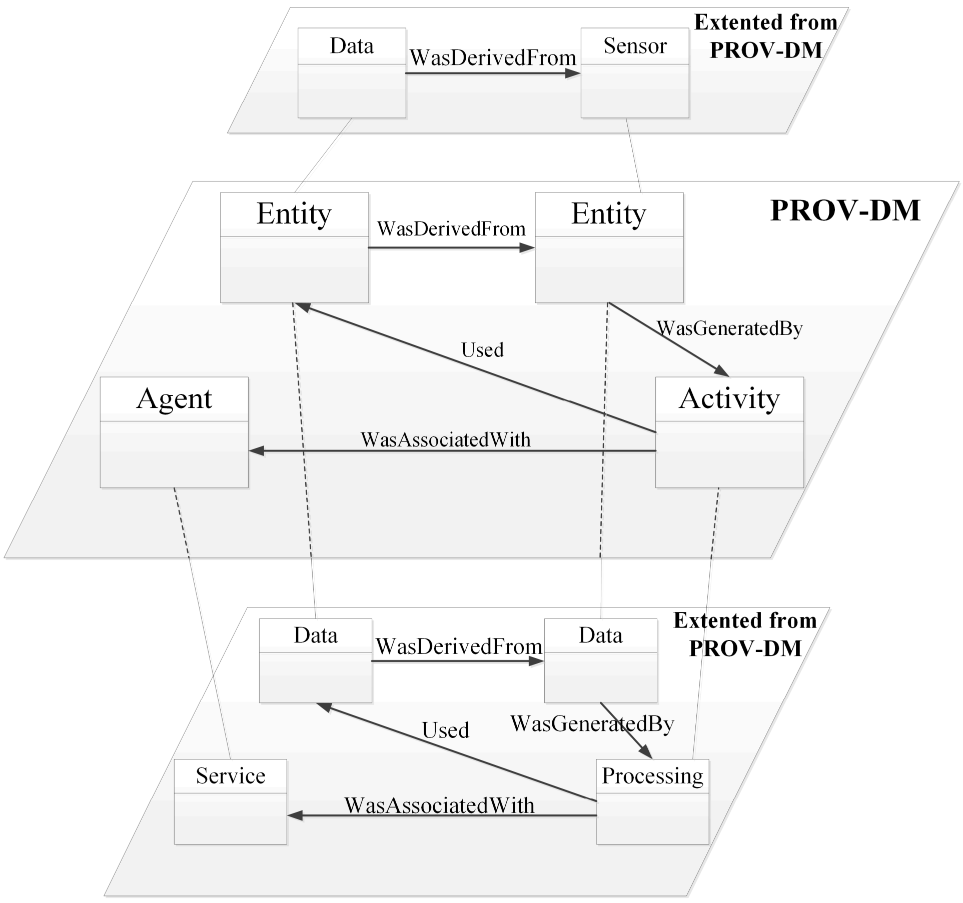


Figure 2. Formalized description of the provenance model extended from PROV-DM.

2.2. Encoding Method

The encoding of the model is used to implement a provenance-aware system. Sensor Web is a Web-based and standard specification-based environment. It is therefore better to incorporate the provenance representation method into existing specifications (e.g., SensorML and O&M), to reduce manpower and material resources, and to maintain consistent sharing and interoperational environments. Considering this, we encode the description model by defining the XML schema for sensors, processing, data, and services. The Sensor Web has some provenance-related information (SensorML and O&M), which describes sensors, processes (algorithms), and observations. SensorML has already modeled sensor and process information, and O&M has modelled all four objects. Thus, in this paper, we mainly use the provenance encoding from O&M 2.0 and SensorML 2.0. In O&M 2.0, the root element is

OM_Obsevation, and the main elements branching from it are: type, metadata, relatedObservation, phenomenonTime, resultTime, validTime, procedure, observedProperty, featureOfInterest, rsultQuality and result. SensorML 2.0 classifies the processes into atomic-non-physical, atomic- physical, composite-non-physical, and composite-physical. They are defined by four elements SimpleProcess, AggregateProcess, PhysicalComponent, and PhysicalSystem.

Based on these existing specifications and encoding methods, the provenance encoding strategies are as follows. The observation object is encoded by the OM_Observation element in the O&M schema, the sensor object is encoded by the PhysicalComponent element or the PhysicalSystem element in SensorML, the data object is encoded by the result element in SensorML, the processing object is encoded by the SimpleProcess element or the AggregateProcess element in SensorML, and the service object is encoded by newly defined elements such as ParentOM, ProcessService, and CurrentOMService, as shown in Figure 3. ParentOM is an element that defines the information track of the parent O&M, which is processed to create the current O&M. The ParentOM XML schema is shown in Figure 4. The input element of the SimpleProcess or AggregateProcess elements in SensorML records the Sensor Web Enablement common data type AnyData. Therefore, the parent O&M, as the input, is not expressed by the input element. It uses ParentOM to record the element. SimpleProcess or AggregateProcess simply show the processing, and do not indicate the provided services. A ProcessService records the processing service, as shown in Figure 5a. It displays processing service capability information and links. The CurrentOMService is similar to the ProcessService and can display current O&M service capability information, as shown in Figure 5b. The result element in O&M does not have a specified type. Given these considerations, the set elements ParentOM, ProcessService, and CurrentOMService are the output elements of the result, which maintain the structure and save the provenance information in O&M, as shown in the red rectangle in Figure 6.

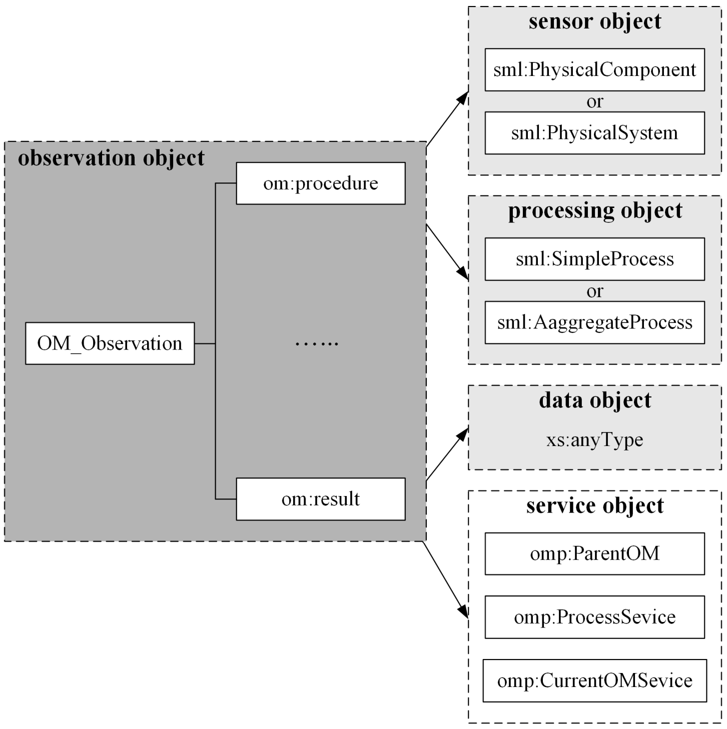


Figure 3. O&M provenance encoding model.

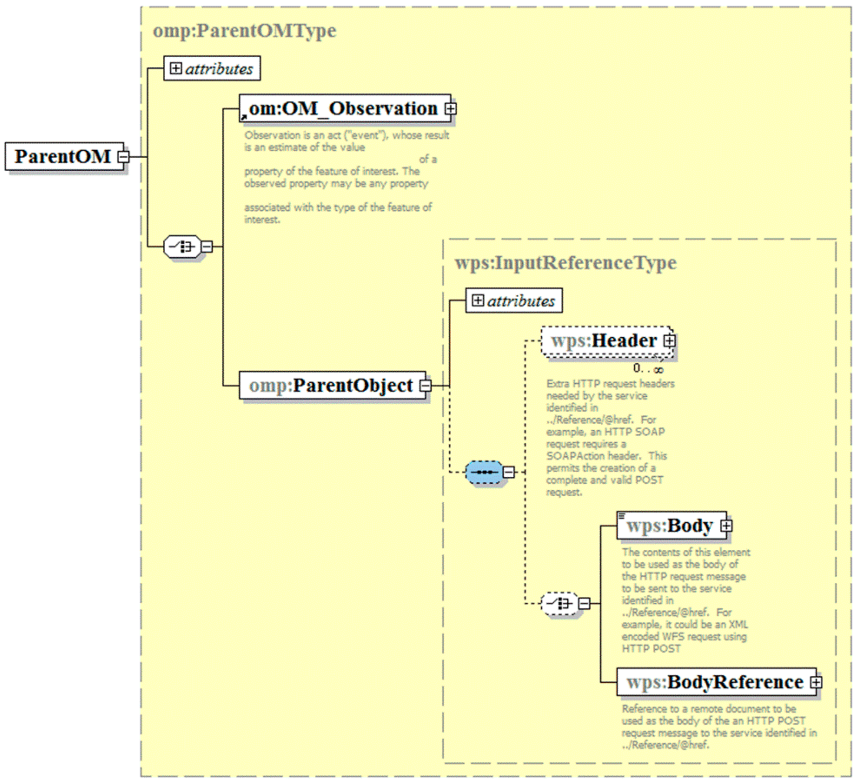

omp:ParentOMType
attributes
om:OM_Observation
Observation is an act ("event"), whose result is an estimate of the value of a property of the feature of interest. The observed property may be any property associated with the type of the feature of interest.
ParentOM
omp:ParentObject
wps:InputReferenceType
attributes
wps:Header
0..∞
Extra HTTP request headers needed by the service identified in ../Reference/@href. For example, an HTTP SOAP request requires a SOAPAction header. This permits the creation of a complete and valid POST request.
wps:Body
The contents of this element to be used as the body of the HTTP request message to be sent to the service identified in ../Reference/@href. For example, it could be an XML encoded WFS request using HTTP POST
wps:BodyReference
Reference to a remote document to be used as the body of the an HTTP POST request message to the service identified in ../Reference/@href.

Figure 4. ParentOM XML Schema.

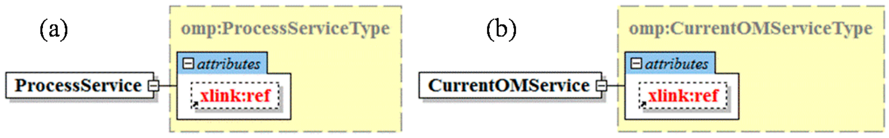


Figure 5. (**a**) ProcessService and (**b**) CurrentOMService Schemas.

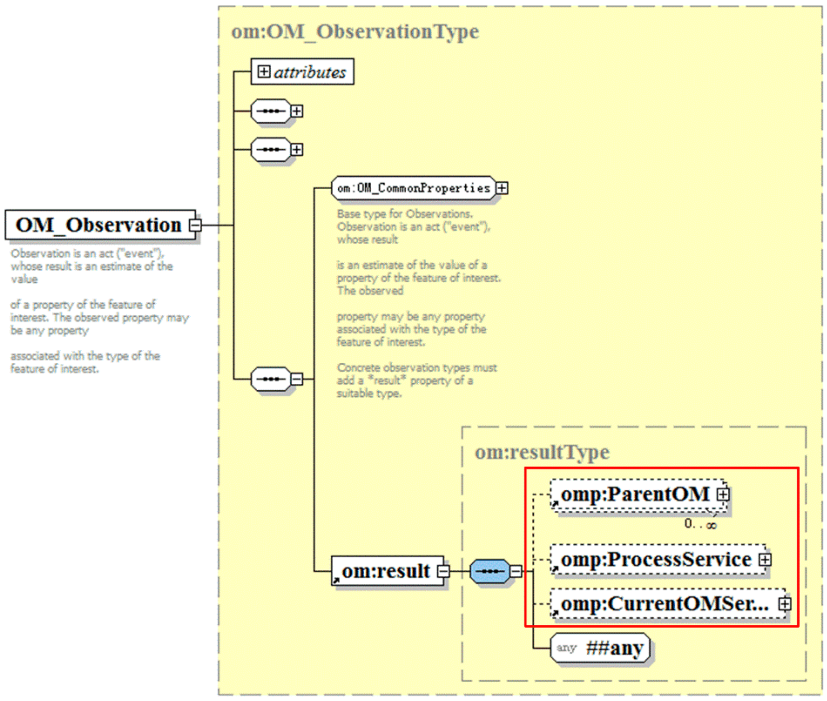

om:OM_ObservationType
attributes
OM_Observation
Observation is an act ("event"), whose result is an estimate of the value of a property of the feature of interest. The observed property may be any property associated with the type of the feature of interest.
om:OM_CommonProperties
Base type for Observations. Observation is an act ("event"), whose result is an estimate of the value of a property of the feature of interest. The observed property may be any property associated with the type of the feature of interest. Concrete observation types must add a "result" property of a suitable type.
om:resultType
om:result
omp:ParentOM
0..∞
omp:ProcessService
omp:CurrentOMSer...
any ##any

Figure 6. Extended elements in O&M.

2.3. Service Implementation

The implementation of the remote sensing observation provenance model in Sensor Web services is important to applications. The proposed provenance model is embedded in O&M and adds some child elements to the O&M result element. All Sensor Web services should respond to this new information when implemented. This response requires an understanding of the relationships between services and the O&M, and the exact effect that the O&M has on the services. The interactions and internal implementations of the O&M and Sensor Web services (SOS, Web Processing Service (WPS) [29] and Sensor Planning Service (SPS) [30]) are shown in Table 1.

Table 1. Relationships between Sensor Web services and O&M.

Service Name	Operation Name	Input Parameters	Output Parameters
SOS	GetObservation	any parameters that can be O&M and also cannot be O&M	O&M
SOS	InsertObservation	O&M	SOS response
SPS	Submit	any parameters that can be O&M and also cannot be O&M	Can generate an O&M Output for DescribeResultAccess operation exposing
WPS	Execute	May include O&M	Can be O&M

Table 1 shows that, the services and their operations, input, and output parameters have relationships with O&M.

SOS is a standard service interface in SWE. SOS provides access to observations from sensors and sensor systems in a standard manner that is consistent for all sensor systems, including remote, *in-situ*, fixed and mobile sensors. GetObservation and InsertOservation operate the O&M in SOS. GetObservation is invoked to obtain O&M. InsertOservation is invoked to insert O&M into SOS. SOS stores and manages O&M. The three elements (omp:ParentOM, omp:ProcessService, and omp:CurrentOMService), in the om:result should be carefully considered in the storing strategy, when storing the O&M in a SOS system.

SPS is also a standard service interface in SWE. SPS was designed and developed to allow clients to determine the feasibility of a desired set of collection requests for one or more sensors/platforms in an interoperable service. A client directly submits collection requests to these sensors/platforms. GetCapabilities, DescribeTasking, Submit, and DescribeResultAccess are the main operations in SPS. Submit is the core operation in SPS that allows planning sensor or sensor systems to submit a task. It contains two processes: submitting a task, and encapsulating the processing into SimpleProcess or AggregateProcess, or the SPS service into the ProcessService when planning the O&M result. The result of the Submit operation is exposed by the DescribeResultAccess operation.

WPS is an OGC specification. WPS defines a standardized interface that facilitates the publishing of geospatial processes (including any algorithm, calculation, or model that operates on spatially referenced data), the discovery of these processes and the binding of these processes by clients. GetCapabilities, DescribeProcess, and Execute are the three operations in WPS. After the Execute operation, the O&M contains the provenance information if the output is O&M and processes provenance information similar to SPS.

The above analysis highlights factors of particular importance to O&M provenance, but methods for developing the services are unrestricted.

2.4. Tracking Algorithm

This section introduces the algorithms that track provenance information based on the described model and encoding method. A *CSO* may be dealt with by complex processes. To obtain the provenance information, we need to know: what are the *HSOs* for a *CSO*; what are the *HSOs* for a *CSO* at a specified time point or time interval; to which sensors does a *CSO* refer; and what processing and services were used to result in a certain *CSO*? To answer these questions, we designed an abstract algorithm and four implementations for tracking P_{data}, $P_{processing}$, P_{sensor}, and $P_{service}$. The abstract processes of **Algorithm 1** are: input the provenance data and the required provenance object, track the provenance, obtain the nearest historical provenance information, validate the provenance information, and return the provenance result, as shown in Figure 6a. To better explain **Algorithm 1**, we used a pseudofunction to describe the abstract process, as shown in Figure 7b. **Algorithms 2–5** are implementations of **Algorithm 1**, so they have the same steps. The specific implementations of **Algorithms 2–5** are as follows.

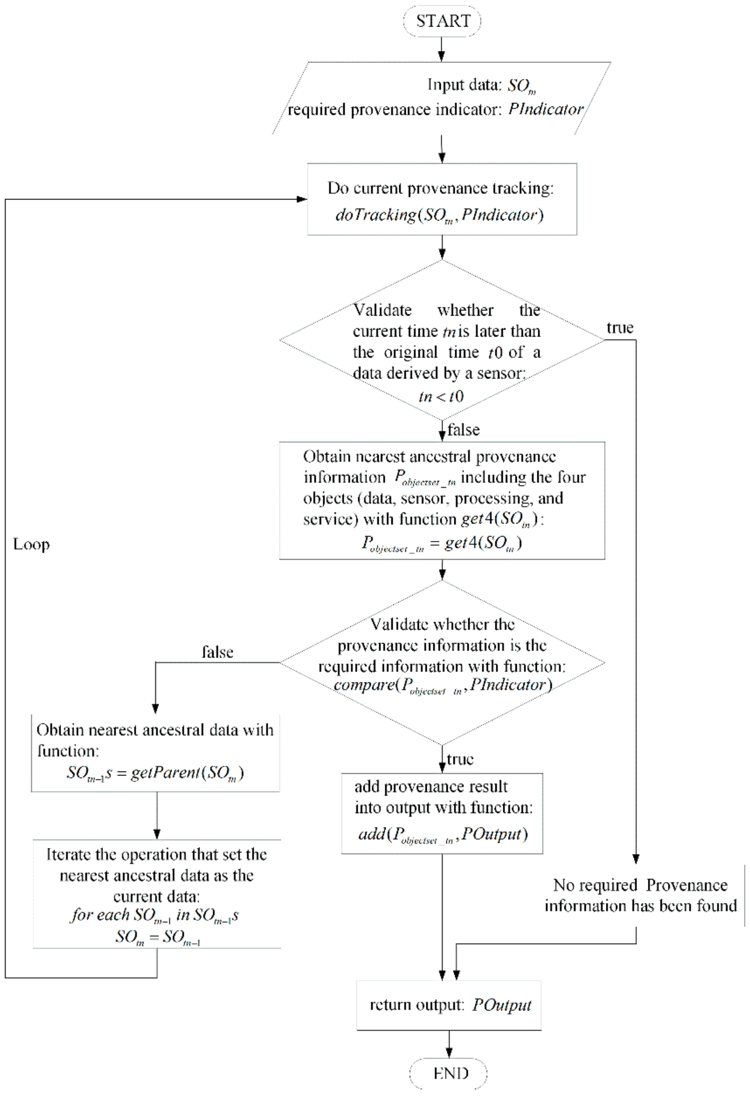

START
Input data: SO_m
required provenance indicator: *PIndicator*
Do current provenance tracking: $doTracking(SO_m, PIndicator)$
Validate whether the current time tn is later than the original time $t0$ of a data derived by a sensor: $tn < t0$
true
false
Obtain nearest ancestral provenance information $P_{objectset\_tn}$ including the four objects (data, sensor, processing, and service) with function $get4(SO_m)$: $P_{objectset\_tn} = get4(SO_m)$
Loop
Validate whether the provenance information is the required information with function: $compare(P_{objectset\_tn}, PIndicator)$
false
true
Obtain nearest ancestral data with function: $SO_{m-1}s = getParent(SO_m)$
Iterate the operation that set the nearest ancestral data as the current data: $for\ each\ SO_{m-1}\ in\ SO_{m-1}s$ $SO_m = SO_{m-1}$
add provenance result into output with function: $add(P_{objectset\_tn}, POutput)$
No required Provenance information has been found
return output: *POutput*
END

Figure 7. Flow of **Algorithm 1**.

```
Algorithm 1: Abstract tracking algorithm
Input: current status observation $SO_{tn}$
       required provenance information indicator $PIndicator$
Output: provenance result $POutput$ indicated by $PIndicator$
  Use: $doTracking(SO_{tn}, PIndicator)$ tracks the provenance information of $SO_{tn}$ at $tn$
       $get4(SO_{tn})$ obtains the set of four objects, $P_{objectset\ tn}$, from $SO_{tn}$
       $compare(P_{objectset\_tn}, PIndicator)$ determines whether $PIndicator$ is in $P_{objectset\_tn}$
       $add(P_{objectset\_tn}, POutput)$ adds the provenance information into the output $POutput$
       $getParent(SO_{tn})$ obtains the latest historical observations
STEP 1: Start tracking using the function $doTracking(SO_{tn}, PIndicator)$.
STEP 2: Validate that the current observation time ($tn$) occurs before the initial time of the sensor
        $t0(tn < t0)$. If this is true, the correct result cannot be found, even at the initial observation, and we return a
        "not found" exception. If it is false, we proceed to the next step.
STEP 3: Obtain the set of four objects ($P_{objectset\_tn}$) from $SO_{tn}$ using the function $get4(SO_{tn})$. The implementation of
        $get4(SO_{tn})$ is based on the encoding method of the provenance model in $SO_{tn}$.
STEP 4: Compare the provenance objects from $PIndicator$ with the objects in
        $P_{objectset\_tn}(P_{objectset\_tn}=\{P_{data}, P_{sensor\_tn}, P_{process\_tn}, P_{service\_tn}\})$ using $compare(P_{objectset\_tn}, PIndicator)$. The function will
        return true if the objects in $PIndicator$ are in $P_{objectset\_tn}$, and then the result is added to the output ($POutput$) using
        $add(P_{objectset\_tn}, POutput)$. Otherwise, we obtain the latest historical observations (parent observations), $SO_{tn-1}s$,
        using $getParent(SO_{tn})$ and proceed to the next step. The observations for each $SO_{tn-1}$ in $SO_{tn-1}s$ comprise the
        current observation set.
STEP 5: Reset each $SO_{tn-1}$ in $SO_{tn-1}s$ as $SO_{tn}$ and invoke $doTracking(SO_{tn}, PIndicator)$. If $doTracking(SO_{tn}, PIndicator)$ and its
recursive functions are finished, return the provenance result, $POutput$.
```

```
Algorithm 2: Tracking historical data
Input: current status observation $SO_{tn}$
       required provenance information indicator $PIndicator = P_{data\ tm}$
Output: historical data $P_{data\ tm}$
Annotation: Algorithm 2 is an implementation of Algorithm 1. The steps are the same as in Algorithm 1, and we
            define the necessary functions as follows.
              For $get4(SO_{tn})$, $SO_{tn}$ is an O&M. The O&M embeds the four objects of the provenance model and thus
              returns $SO_{tn}$.
            $compare(P_{objectset\_tn}, PIndicator)$ returns true if the times match. Otherwise, it returns false.
            $add(P_{objectset\_tn}, POutput)$ directly adds $P_{data\ tn}$ to $POutput$, where $P_{data\ tn}$ is obtained from the child element of
            //result/om:OM_Observation/result.
            $SO_{tn-1}s = getParent(SO_{tn})$ executes the following steps.
                1: If the child element of //result/om:OM_Observation (which shows the element ParemtOM in
                O&M with an XML Xpath (http://www.w3.org/TR/xpath/)) is om:OM_Observation, we return
                om:OM_Observation. If not, we proceed to the next substep.
                2: If the child element of //result/omp:ParemtOM is not om:ParentObject, we return an error
                message. Otherwise, we obtain the om:ParentObject: xlink:href, method, mineType, encoding, and
                schema values. If the method is HTTP GET or POST, we return an error message. If the method is
                a HTTP GET, we send a GET request to xlink:href to obtain the result (O&M). If the method is
                HTTP POST, we proceed to the next substep.
                3: If the Header is required, we obtain the Header message and POST the message from the body or
                bodyReference. Then, we send a POST request to xlink:href by integrating the POST with the Header
                message to obtain the result. If the Header is not needed, we directly obtain the POST message from
                the body or bodyReference. We then send a POST request to xlink:href to obtain the result.
```

```
Algorithm 3: Tracking processing objects
Input: current status observation $SO_{tn}$
        required provenance information indicator $PIndicator=P_{processing\_tm}$
Output: processing object $P_{processing\_tm}$
Annotation: Algorithm 3 is an implementation of Algorithm 1. The steps are the same as Algorithm 1, and we define
        two specific functions as follows.
    $compare(P_{objectset\_tn}, PIndicator)$ returns true if the times match. Otherwise, it returns false.
    $add(P_{objectset\_tn}, POutput)$ executes the following steps.
        1: If the child element of om:OM_Observation/om:procedure is SimpleProcess, $P_{processing\_tm}$ is
           assigned the SimpleProcess information. If the child element of om:OM_Observation/om:procedure
           is AggregateProcess, $P_{processing\_tm}$ is assigned the AggregateProcess information. Otherwise, we
           return an error message.
        2: The omp:ProcessService information of $SO_{tm}$ is obtained using
           Xpathom:OM_Observation/om:result/omp:ProcessService. Then, the omp:ProcessService
           information is added to $P_{processing\_tm}$.
        3: Return $P_{processing\_tm}$.
```

```
Algorithm 4: Tracking service objects
Input: current status observation $SO_{tn}$
        required provenance information indicator $PIndicator=P_{service\_tm}$
Output: service object $P_{service\_tm}$
Annotation: Algorithm 4 is an implementation of Algorithm 1. The steps of Algorithm 4 are the same as Algorithm
        1, and we define two specific functions.
    $compare(P_{objectset\_tn}, PIndicator)$ returns true if the times match. Otherwise, it returns false.
    $add(P_{objectset\_tn}, POutput)$ executes the following steps.
        1: The om:result information of $SO_{tm}$ is obtained using Xpathom:OM_Observation/
           om:result. If the children elements have omp:ParentOM, $P_{service\_tm}$ includes omp:ParentOM
           information. If the children elements have omp:ProcessService, $P_{service\_tm}$ includes
           omp:ProcessService information. If the child elements have omp:CurrentOMService, $P_{service\_tm}$
           includes omp:CurrentOMService information.
        2: Return $P_{service\_tm}$.
```

```
Algorithm 5: Tracking sensor objects
Input: current status observation $SO_{tn}$
        required provenance information indicator $PIndicator=P_{sensor\_t0}$
Output: sensor object $P_{sensor\_t0}$
Annotation: Algorithm 5 is an implementation of Algorithm 1. The steps of Algorithm 5 are the same as Algorithm
        1, and we define two specific functions.
    $compare(P_{objectset\_tn}, PIndicator)$ returns true if the times match. Otherwise, it returns false.
    $add(P_{objectset\_tn}, POutput)$ executes the following steps.
        1: The om:procedure information of $SO_{t0}$ is obtained using
           Xpathom:OM_Observation/om:procedure. If the child element of om:procedure is
           PhysicalComponent, $P_{sensor\_t0}$ includes PhysicalComponent information. If the child element of
           om:procedure is PhysicalSystem, $P_{sensor\_t0}$ includes PhysicalSystem information. Otherwise, it
           returns an error message.
        SUBSTEP 2: Return $P_{sensor\_t0}$.
```

3. Experimental Section

To test the proposed provenance method, we ran an experiment that generated and tracked vegetation conditions in a Sensor Web enabled environment. Vegetation conditions are critical to decisions in agriculture and ecology, in the public and private sectors. Numerous methods and observations can be used to evaluate the vegetation condition [31,32]. It can be derived using statistical indices such as the normalized difference vegetation index (NDVI) and the vegetation condition index (VCI) [32]. These are defined as

$$NDVI = \frac{NIR - IR}{NIR + IR} \tag{7}$$

and

$$VCI = \frac{NDVI - NDVI_{\min}}{NDVI_{\max} - NDVI_{\min}} \tag{8}$$

In Equation (7), NIR is the near-infrared band data for a pixel, and IR is the infrared band data. In Equation (8), $NDVI_{min}$ and $NDVI_{max}$ are the minimum and maximum NDVIs for a pixel during a specific time period. We used Moderate Resolution Imaging Spectroradiometer (MODIS) observations to calculate the NDVI and VCI. The MODIS observations were obtained from the Land Processes Distributed Active Archive Centre (https://lpdaac.usgs.gov/). We used the "MODIS/Terra Surface Reflectance Daily L2G Global 250 m SIN Grid v005" dataset. The MODIS provides 250-m resolution daily images for the red and NIR bands (0.6 μm–0.9 μm). The experimental observation period was May of each year from 2000–2012, and the area under investigation covered the 48 contiguous states of the United States.

In this experiment, the MODIS data were planned using SPS, encoded with O&M, published with SOS, and processed with WPS. After this, we set up the provenance instance of the MODIS data in a Sensor Web enabled environment. We tracked the provenance of four objects using this provenance instance to evaluate the performance of our method.

3.1. Experiment Design

The six main processing of this experiment are shown in Figure 8.

Processing 1: SPS used the raw MODIS observations rather than real MODIS sensor terra data, because it cannot control the sensor [5]. The MODIS observations were planned based on spatio-temporal information by invoking the SPS submit operation from the MODIS data center [5].

Processing 2: We assumed that MODIS scans the earth's surface after the SPS. All the required MODIS observations were collected. The MODIS observations encoded in the O&M format were inserted into SOS by invoking the InsertObservation operation from the FTP server. At this point, the MODIS sensor information has already been registered into SOS by the client using the RegisterSensor operation. An SPS instance simulated the planning of the satellite that carries the MODIS sensor. The parameters of the planning task include information on the time and location. The MODIS FTP server provided 250-m resolution daily images for the red and NIR bands (0.6–0.9 μm). The time parameter was the date. The location should was transformed into tiles [33] that cover the 48 contiguous states of the United States. There were 25 MODIS tile Hierarchical Data File (HDF) files for each item in the dataset.

Processing 3: The MODIS files were pre-processed by the WPS to generate a daily MODIS file. This processing adjusted for noise, cloud, and snow cover, corrected the angles, and implemented mosaic and clip procedures. The files were then inserted into SOS in the O&M format using the standard method.

Processing 4: The daily NDVI was calculated be the WPS. The inputs for this WPS processing were the daily MODIS O&M products from SOS. The output was the daily NDVI product with O&M formatting. The result was inserted into SOS by the InsertObservation operation.

Processing 5: The weekly NDVI was calculated be the WPS. The input for this WPS contained seven daily NDVI. The output was a weekly NDVI. After processing, the result was inserted into SOS.

Processing 6: The weekly VCI was calculated. The inputs were the current weekly NDVI product from the current year and the 7-day daily NDVI products from the previous year, which have the same date order. The output was a weekly VCI product, which was managed by SOS.

All the O&Ms included the provenance information created by the method in Section 2.2.

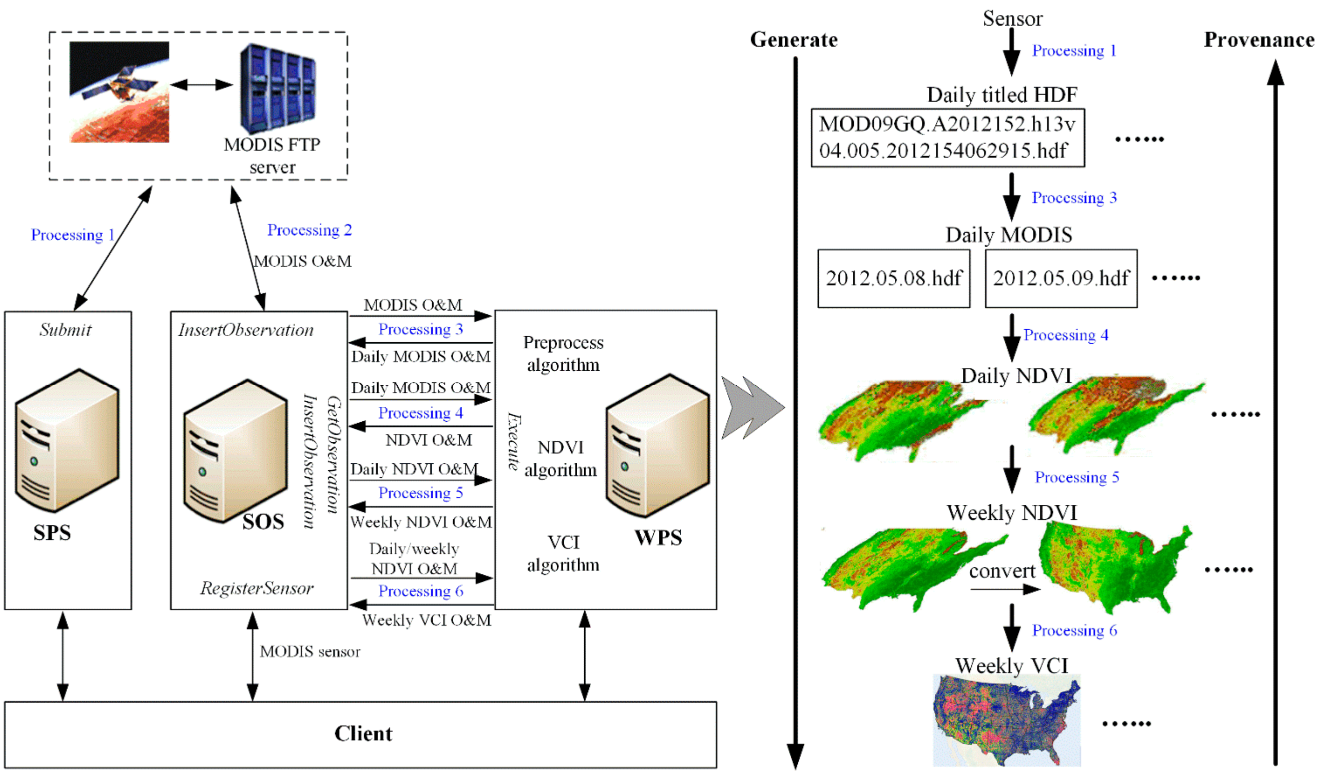
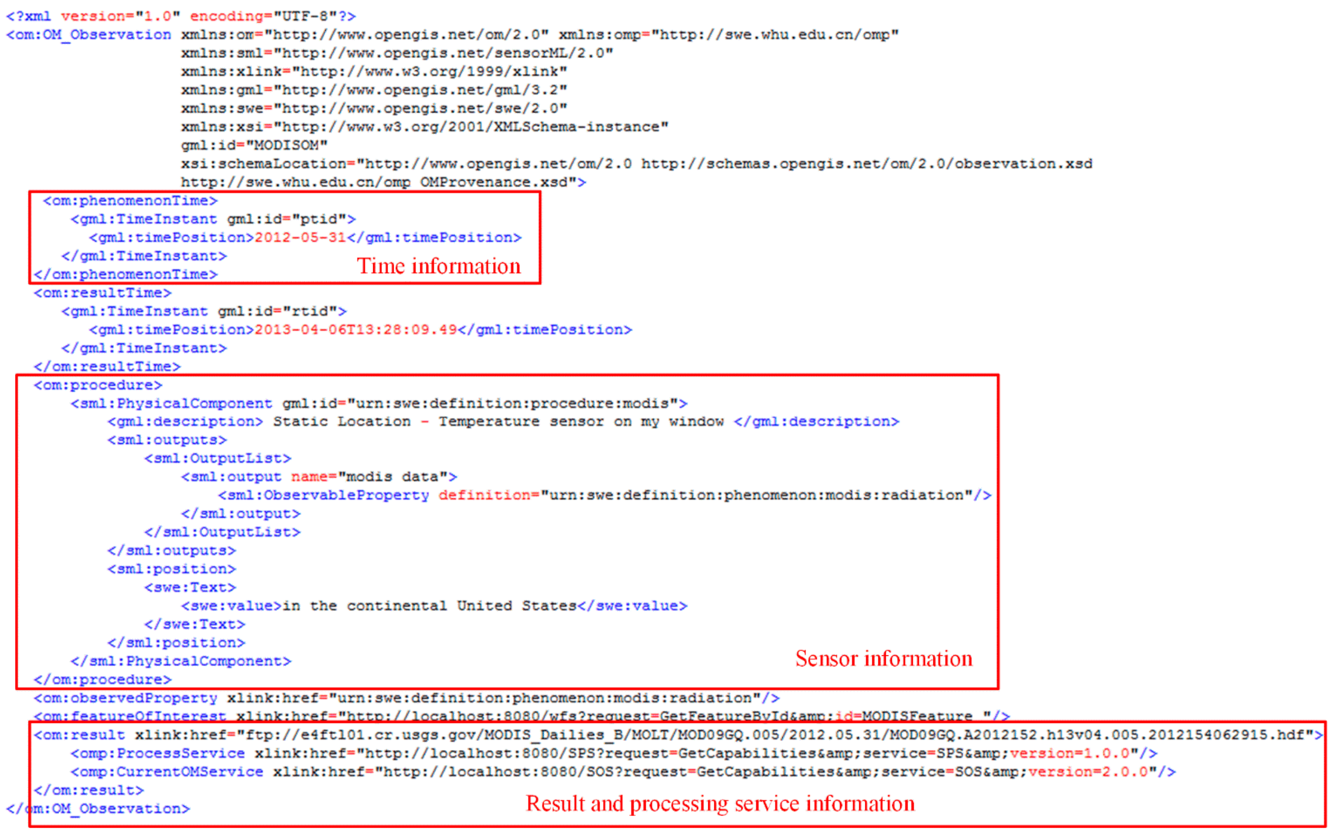


Figure 8. Six processing of the experiment.

3.2. Experiment Results and Discussion

3.2.1. Representation of the Provenance Model in O&M

We ran the six processing in Figure 8. The observations from a single month (May) over 13 years (from 2000 to 2012) were downloaded with SPS using spatio-temporal information. This resulted in 10,051 HDF files that were over 700 gigabytes in size, which covered the 48 contiguous states of the United States. An HDF file with an O&M format is shown in Figure 9.

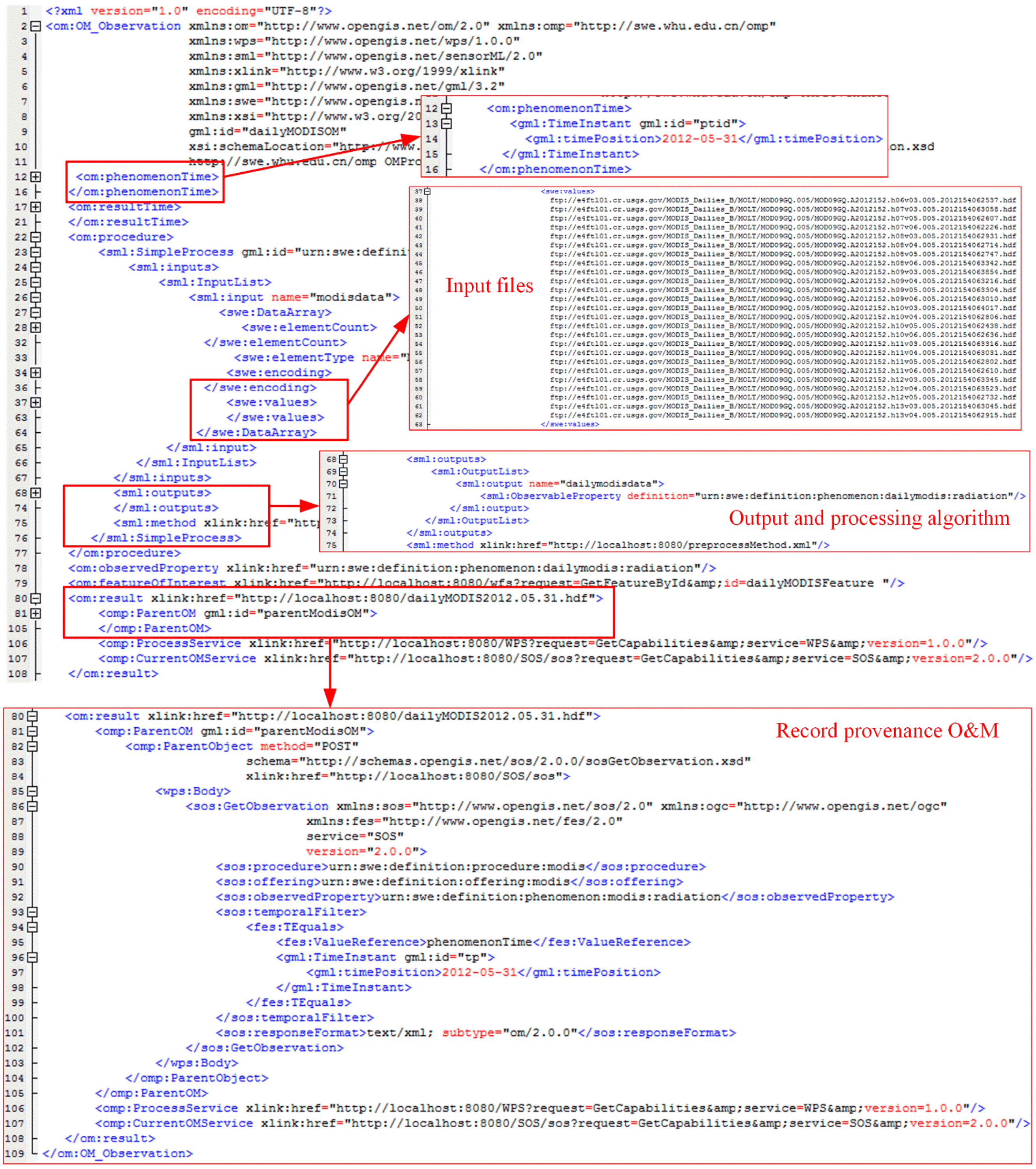


```
<?xml version="1.0" encoding="UTF-8"?>
<om:OM_Observation xmlns:om="http://www.opengis.net/om/2.0" xmlns:omp="http://swe.whu.edu.cn/omp"
                   xmlns:sml="http://www.opengis.net/sensorML/2.0"
                   xmlns:xlink="http://www.w3.org/1999/xlink"
                   xmlns:gml="http://www.opengis.net/gml/3.2"
                   xmlns:swe="http://www.opengis.net/swe/2.0"
                   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   gml:id="MODISOM"
                   xsi:schemaLocation="http://www.opengis.net/om/2.0 http://schemas.opengis.net/om/2.0/observation.xsd
                   http://swe.whu.edu.cn/omp OMProvenance.xsd">
   <om:phenomenonTime>
      <gml:TimeInstant gml:id="ptid">
        <gml:timePosition>2012-05-31</gml:timePosition>
     </gml:TimeInstant>
  </om:phenomenonTime>
  <om:resultTime>
     <gml:TimeInstant gml:id="rtid">
        <gml:timePosition>2013-04-06T13:28:09.49</gml:timePosition>
     </gml:TimeInstant>
  </om:resultTime>
  <om:procedure>
      <sml:PhysicalComponent gml:id="urn:swe:definition:procedure:modis">
          <gml:description> Static Location - Temperature sensor on my window </gml:description>
          <sml:outputs>
              <sml:OutputList>
                  <sml:output name="modis data">
                      <sml:ObservableProperty definition="urn:swe:definition:phenomenon:modis:radiation"/>
                  </sml:output>
              </sml:OutputList>
          </sml:outputs>
          <sml:position>
              <swe:Text>
                  <swe:value>in the continental United States</swe:value>
              </swe:Text>
          </sml:position>
      </sml:PhysicalComponent>
  </om:procedure>
  <om:observedProperty xlink:href="urn:swe:definition:phenomenon:modis:radiation"/>
  <om:featureOfInterest xlink:href="http://localhost:8080/wfs?request=GetFeatureById&amp;id=MODISFeature "/>
  <om:result xlink:href="ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/2012.05.31/MOD09GQ.A2012152.h13v04.005.2012154062915.hdf">
      <omp:ProcessService xlink:href="http://localhost:8080/SPS?request=GetCapabilities&amp;service=SPS&amp;version=1.0.0"/>
      <omp:CurrentOMService xlink:href="http://localhost:8080/SOS?request=GetCapabilities&amp;service=SOS&amp;version=2.0.0"/>
  </om:result>
</om:OM_Observation>
```

Figure 9. Original tile HDF file.

The entire O&M document was composed of the current observations. The sensor information was encoded using sml:PhysicalComponent. The observation entity was linked with the xlink:href attribute in the om:result element. All observation entity links have the string pattern, "ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/" + date + "/" + tile observation name, similar to the observation link in Figure 9 "ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/2012.05.31/MOD09GQ.A2012152.h13v04.005.2012154062915.hdf". This linked the observation entity in the O&M document without encoding the observation entity, which was between tens to hundreds of megabytes in size. A large XML document is difficult to process and deliver to the Web through general programming methods [34]. This hinders XML [35]. The processing service and current O&M service were linked with the omp:ProcessService and omp:CurrentOMService elements, respectively.

We generated daily MODIS files that covered the 48 contiguous states of the United States. In the WPS, the pre-processing algorithm simply performs mosaic and clip operations, and does not implement the other pre-processing steps discussed in Section 3.1. However, we assumed that the pre-processing algorithm eliminated noise, cloud, and snow cover, corrected the angles, and generated a usable daily MODIS file. This daily MODIS file recorded all the O&M processing. The lack of noise, cloud, and snow cover, and the angle correction may be why it was hard to generate a perfect daily NDVI product, and why there was no effect on the provenance model. The model simply records the algorithm and ignores the precision. The daily NDVI O&M is shown in Figure 10.



```
<?xml version="1.0" encoding="UTF-8"?>
<om:OM_Observation xmlns:om="http://www.opengis.net/om/2.0" xmlns:omp="http://swe.whu.edu.cn/omp"
                   xmlns:wps="http://www.opengis.net/wps/1.0.0"
                   xmlns:sml="http://www.opengis.net/sensorML/2.0"
                   xmlns:xlink="http://www.w3.org/1999/xlink"
                   xmlns:gml="http://www.opengis.net/gml/3.2"
                   xmlns:swe="http://www.opengis.n
                   xmlns:xsi="http://www.w3.org/20
                   gml:id="dailyMODISOM"
                   xsi:schemaLocation="http://www.                                                              on.xsd
                   http://swe.whu.edu.cn/omp OMPro
   <om:phenomenonTime>
  </om:phenomenonTime>
  <om:resultTime>
  </om:resultTime>
  <om:procedure>
      <sml:SimpleProcess gml:id="urn:swe:defini
          <sml:inputs>
              <sml:InputList>
                  <sml:input name="modisdata">
                      <swe:DataArray>
                         <swe:elementCount>
                    </swe:elementCount>
                        <swe:elementType name="
                       <swe:encoding>
                    </swe:encoding>
                       <swe:values>
                       </swe:values>
                  </swe:DataArray>
              </sml:input>
          </sml:InputList>
       </sml:inputs>
       <sml:outputs>
       </sml:outputs>
       <sml:method xlink:href="http
    </sml:SimpleProcess>
  </om:procedure>
  <om:observedProperty xlink:href="urn:swe:definition:phenomenon:dailymodis:radiation"/>
  <om:featureOfInterest xlink:href="http://localhost:8080/wfs?request=GetFeatureById&amp;id=dailyMODISFeature "/>
  <om:result xlink:href="http://localhost:8080/dailyMODIS2012.05.31.hdf">
      <omp:ParentOM gml:id="parentModisOM">
      </omp:ParentOM>
      <omp:ProcessService xlink:href="http://localhost:8080/WPS?request=GetCapabilities&amp;service=WPS&amp;version=1.0.0"/>
      <omp:CurrentOMService xlink:href="http://localhost:8080/SOS/sos?request=GetCapabilities&amp;service=SOS&amp;version=2.0.0"/>
  </om:result>
```

```
    <om:phenomenonTime>
       <gml:TimeInstant gml:id="ptid">
         <gml:timePosition>2012-05-31</gml:timePosition>
      </gml:TimeInstant>
    </om:phenomenonTime>
```

```
<swe:values>
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h06v03.005.2012154062537.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h07v03.005.2012154063058.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h07v05.005.2012154062607.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h07v06.005.2012154062226.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h08v03.005.2012154062931.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h08v04.005.2012154062714.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h08v05.005.2012154062747.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h08v06.005.2012154063342.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h09v03.005.2012154063854.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h09v04.005.2012154063216.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h09v05.005.2012154063304.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h09v06.005.2012154063010.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h10v03.005.2012154064017.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h10v04.005.2012154062806.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h10v05.005.2012154062438.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h10v06.005.2012154062636.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h11v03.005.2012154063316.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h11v04.005.2012154063031.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h11v05.005.2012154062802.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h11v06.005.2012154062610.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h12v03.005.2012154063345.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h12v04.005.2012154063523.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h12v05.005.2012154062732.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h13v03.005.2012154063045.hdf
  ftp://e4ftl01.cr.usgs.gov/MODIS_Dailies_B/MOLT/MOD09GQ.005/MOD09GQ.A2012152.h13v04.005.2012154062915.hdf
</swe:values>
```

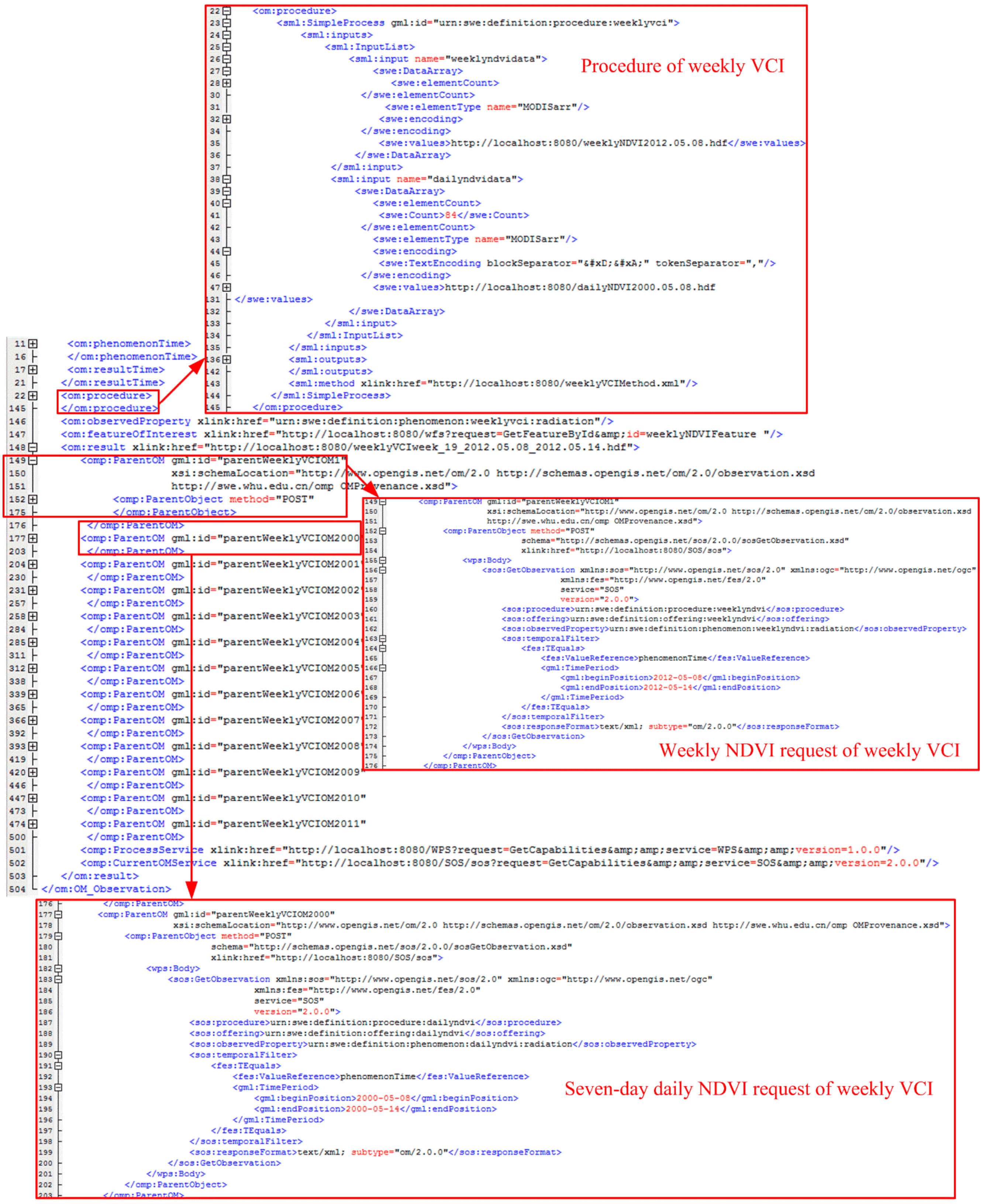
```
      <sml:outputs>
         <sml:OutputList>
            <sml:output name="dailymodisdata">
               <sml:ObservableProperty definition="urn:swe:definition:phenomenon:dailymodis:radiation"/>
            </sml:output>
         </sml:OutputList>
      </sml:outputs>
      <sml:method xlink:href="http://localhost:8080/preprocessMethod.xml"/>
```

```
  <om:result xlink:href="http://localhost:8080/dailyMODIS2012.05.31.hdf">
      <omp:ParentOM gml:id="parentModisOM">
          <omp:ParentObject method="POST"
                            schema="http://schemas.opengis.net/sos/2.0.0/sosGetObservation.xsd"
                            xlink:href="http://localhost:8080/SOS/sos">
              <wps:Body>
                  <sos:GetObservation xmlns:sos="http://www.opengis.net/sos/2.0" xmlns:ogc="http://www.opengis.net/ogc"
                                      xmlns:fes="http://www.opengis.net/fes/2.0"
                                      service="SOS"
                                      version="2.0.0">
                      <sos:procedure>urn:swe:definition:procedure:modis</sos:procedure>
                      <sos:offering>urn:swe:definition:offering:modis</sos:offering>
                      <sos:observedProperty>urn:swe:definition:phenomenon:modis:radiation</sos:observedProperty>
                      <sos:temporalFilter>
                          <fes:TEquals>
                              <fes:ValueReference>phenomenonTime</fes:ValueReference>
                              <gml:TimeInstant gml:id="tp">
                                  <gml:timePosition>2012-05-31</gml:timePosition>
                              </gml:TimeInstant>
                          </fes:TEquals>
                      </sos:temporalFilter>
                      <sos:responseFormat>text/xml; subtype="om/2.0.0"</sos:responseFormat>
                  </sos:GetObservation>
              </wps:Body>
          </omp:ParentObject>
      </omp:ParentOM>
      <omp:ProcessService xlink:href="http://localhost:8080/WPS?request=GetCapabilities&amp;service=WPS&amp;version=1.0.0"/>
      <omp:CurrentOMService xlink:href="http://localhost:8080/SOS/sos?request=GetCapabilities&amp;service=SOS&amp;version=2.0.0"/>
  </om:result>
</om:OM_Observation>
```

Figure 10. Daily MODIS file.

The inputs of this pre-processing algorithm were the daily MODIS tile files. The method was linked using a pre-processing method.xml file that describes the calculations in detail. The omp:ParentOM was used to record the tracking request of the parent's provenance information. In this document, we only used one omp:ParentObject to request the 25 (except 23 May 2001 and 11 May 2012) input "parent" provenance data. The request is an O&M GetObservation request, and the procedure, offering, observedProperty, and time information point to the 25 "parent" observations.

The generated daily and weekly NDVI (a week starts on a Tuesday) have a provenance information structure similar to that of the daily NDVI.


Procedure of weekly VCI
Weekly NDVI request of weekly VCI
Seven-day daily NDVI request of weekly VCI

```
        <om:procedure>
            <sml:SimpleProcess gml:id="urn:swe:definition:procedure:weeklyvci">
                <sml:inputs>
                    <sml:InputList>
                        <sml:input name="weeklyndvidata">
                            <swe:DataArray>
                               <swe:elementCount>
                          </swe:elementCount>
                              <swe:elementType name="MODISarr"/>
                             <swe:encoding>
                          </swe:encoding>
                             <swe:values>http://localhost:8080/weeklyNDVI2012.05.08.hdf</swe:values>
                         </swe:DataArray>
                     </sml:input>
                     <sml:input name="dailyndvidata">
                         <swe:DataArray>
                            <swe:elementCount>
                             <swe:Count>84</swe:Count>
                          </swe:elementCount>
                            <swe:elementType name="MODISarr"/>
                            <swe:encoding>
                             <swe:TextEncoding blockSeparator="&#xD;&#xA;" tokenSeparator=","/>
                          </swe:encoding>
                            <swe:values>http://localhost:8080/dailyNDVI2000.05.08.hdf
 </swe:values>
                        </swe:DataArray>
                    </sml:input>
                 </sml:InputList>
              </sml:inputs>
              <sml:outputs>
              </sml:outputs>
              <sml:method xlink:href="http://localhost:8080/weeklyVCIMethod.xml"/>
           </sml:SimpleProcess>
        </om:procedure>
```

```
    <om:phenomenonTime>
    </om:phenomenonTime>
    <om:resultTime>
  </om:resultTime>
  <om:procedure>
  </om:procedure>
  <om:observedProperty xlink:href="urn:swe:definition:phenomenon:weeklyvci:radiation"/>
  <om:featureOfInterest xlink:href="http://localhost:8080/wfs?request=GetFeatureById&amp;id=weeklyNDVIFeature "/>
  <om:result xlink:href="http://localhost:8080/weeklyVCIweek_19_2012.05.08_2012.05.14.hdf">
     <omp:ParentOM gml:id="parentWeeklyVCIOM1"
                    xsi:schemaLocation="http://www.opengis.net/om/2.0 http://schemas.opengis.net/om/2.0/observation.xsd
                    http://swe.whu.edu.cn/omp OMProvenance.xsd">
          <omp:ParentObject method="POST"
          </omp:ParentObject>
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2000"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2001"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2002"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2003"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2004"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2005"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2006"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2007"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2008"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2009"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2010"
      </omp:ParentOM>
     <omp:ParentOM gml:id="parentWeeklyVCIOM2011"
      </omp:ParentOM>
     <omp:ProcessService xlink:href="http://localhost:8080/WPS?request=GetCapabilities&amp;amp;service=WPS&amp;amp;version=1.0.0"/>
     <omp:CurrentOMService xlink:href="http://localhost:8080/SOS/sos?request=GetCapabilities&amp;amp;service=SOS&amp;amp;version=2.0.0"/>
  </om:result>
</om:OM_Observation>
```

```
        <omp:ParentOM gml:id="parentWeeklyVCIOM1"
                      xsi:schemaLocation="http://www.opengis.net/om/2.0 http://schemas.opengis.net/om/2.0/observation.xsd
                      http://swe.whu.edu.cn/omp OMProvenance.xsd">
            <omp:ParentObject method="POST"
                          schema="http://schemas.opengis.net/sos/2.0.0/sosGetObservation.xsd"
                          xlink:href="http://localhost:8080/SOS/sos">
                <wps:Body>
                    <sos:GetObservation xmlns:sos="http://www.opengis.net/sos/2.0" xmlns:ogc="http://www.opengis.net/ogc"
                                xmlns:fes="http://www.opengis.net/fes/2.0"
                                service="SOS"
                                version="2.0.0">
                        <sos:procedure>urn:swe:definition:procedure:weeklyndvi</sos:procedure>
                        <sos:offering>urn:swe:definition:offering:weeklyndvi</sos:offering>
                        <sos:observedProperty>urn:swe:definition:phenomenon:weeklyndvi:radiation</sos:observedProperty>
                        <sos:temporalFilter>
                            <fes:TEquals>
                                <fes:ValueReference>phenomenonTime</fes:ValueReference>
                                <gml:TimePeriod>
                                    <gml:beginPosition>2012-05-08</gml:beginPosition>
                                    <gml:endPosition>2012-05-14</gml:endPosition>
                                </gml:TimePeriod>
                            </fes:TEquals>
                        </sos:temporalFilter>
                        <sos:responseFormat>text/xml; subtype="om/2.0.0"</sos:responseFormat>
                    </sos:GetObservation>
                </wps:Body>
            </omp:ParentObject>
        </omp:ParentOM>
```

```
        </omp:ParentOM>
       <omp:ParentOM gml:id="parentWeeklyVCIOM2000"
                     xsi:schemaLocation="http://www.opengis.net/om/2.0 http://schemas.opengis.net/om/2.0/observation.xsd http://swe.whu.edu.cn/omp OMProvenance.xsd">
           <omp:ParentObject method="POST"
                          schema="http://schemas.opengis.net/sos/2.0.0/sosGetObservation.xsd"
                          xlink:href="http://localhost:8080/SOS/sos">
               <wps:Body>
                   <sos:GetObservation xmlns:sos="http://www.opengis.net/sos/2.0" xmlns:ogc="http://www.opengis.net/ogc"
                               xmlns:fes="http://www.opengis.net/fes/2.0"
                               service="SOS"
                               version="2.0.0">
                       <sos:procedure>urn:swe:definition:procedure:dailyndvi</sos:procedure>
                       <sos:offering>urn:swe:definition:offering:dailyndvi</sos:offering>
                       <sos:observedProperty>urn:swe:definition:phenomenon:dailyndvi:radiation</sos:observedProperty>
                       <sos:temporalFilter>
                           <fes:TEquals>
                               <fes:ValueReference>phenomenonTime</fes:ValueReference>
                               <gml:TimePeriod>
                                   <gml:beginPosition>2000-05-08</gml:beginPosition>
                                   <gml:endPosition>2000-05-14</gml:endPosition>
                               </gml:TimePeriod>
                           </fes:TEquals>
                       </sos:temporalFilter>
                       <sos:responseFormat>text/xml; subtype="om/2.0.0"</sos:responseFormat>
                   </sos:GetObservation>
               </wps:Body>
           </omp:ParentObject>
        </omp:ParentOM>
```

Figure 11. Part of the O&M document for weekly VCI.

We then calculated the weekly VCI using Equation (8). The NDVI is the current week's NDVI. For example, for 8 May 2012 to 14 May 2012, the NDVI is the weekly NDVI for the 19th week in 2012. NDVImin and NDVImax are the minimum and maximum values for the 7-day NDVI of each previous year (2000 to 2011). The 7-day NDVI of each year has the same start and end times (same ordered day in a year) as 2012, because 8 May 2012 is the 129th day and 14 May 2012 is the 135th day in 2012. The 7-day NDVI of each year represents the 129th to 135th days. The 7-day NDVI for each year may not be the weekly NDVI. Thus, the weekly NDVI should be calculated with the daily NDVI. The inputs for the 19th weekly VCI in 2012 are the 19th weekly NDVI from 2012 and the 7-day daily NDVI for each year from 2000–2012. The result is shown in Figure 11 (one omp:ParentOM element recorded the weekly NDVI and 12 omp:ParentOM elements recorded the 7-day daily NDVI from 2000–2011).

These results show that the provenance object information is represented in the O&M documents, from the raw MODIS observations to the daily MODIS, daily NDVI, weekly NDVI, and weekly VCI.

3.2.2. Tracking Objects with the Tracking Algorithm from O&M

Consider the 19th (8 May 2012 to 14 May 2012) weekly VCI for 2012, as an example. We will use it to demonstrate how **Algorithms 2–5** track the provenance objects. The statistics of the provenance information are shown in Figure 12, and the three provenance objects are as follows (the data is described in Section 3.2.1).

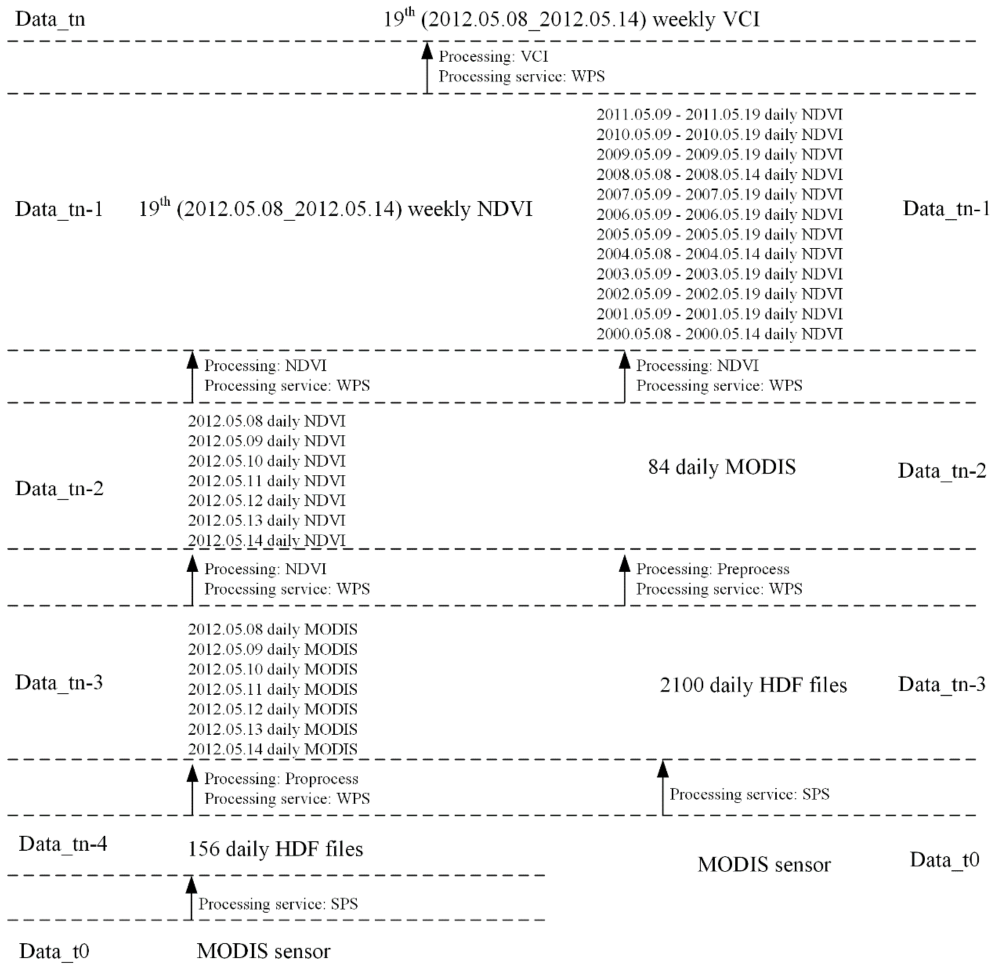


Figure 12. Statistics of the provenance information.

(1) Tracking the sensor. The current observed data (SO_{tn}) is the weekly VCI. We determined the raw MODIS observation ($P_{observation\_t0}$) using **Algorithm 5**. The sensor information was then retrieved as shown in Figure 12. The sensor information was described by PhysicalComponent and its Xpathom:OM_Observation/om:procedure/sml:PhysicalComponent. The sensor information includes the position and output information for the sensor. In this example, we found 2256 raw HDF observations observed by the same MODIS sensor. We tracked a single sensor 2256 times. Therefore, from a theoretical perspective, the provenance processes of the single sensor reflect the provenance processes of multiple sensors.

(2) Tracking the algorithms. An algorithm is represented by the SimpleProcessl element. The SimpleProcessl can contain qualitative and quantitative descriptions of the algorithm. The quantitative description is an executable code and a mathematical expression that defines an algorithm using the mathematical markup language, and an XML application that describes mathematical notations and captures the structure and content of the equation. In this experiment, the processing algorithms were VCI, NDVI, and the pre-processing algorithms, as shown in Figure 12.

(3) Tracking the service. **Algorithm 4** identified the processing service. The algorithms used to calculate the daily MODIS, daily NDVI, weekly NDVI, and weekly VCI were encapsulated in the WPS, as shown in Figure 12.

These results show that the sensor, processing, and service information were appropriately tracked by the algorithms.

3.2.3. Performance Analysis

Time complexity is an important aspect of the performance of this provenance model. The encoding method is similar to an inverted 'tree'; the current observation is the root, and the tracking depth of an object is the minimum number of times that the tracking of the object forms the root. Denote the maximum depth of the current observation as *n*, and the time cost of the provenance at depth i as T_i. Then, the total time is $T_{total} = \sum_{i=0}^{n} T_i$. T_{total} does not exceed $\mathrm{n} \times \max(T_i)(T_{total} \leq \mathrm{n} \times \max(T_i))$, which shows that the maximum time cost at a certain depth provides a reasonable representation of the time complexity of the model. The traversing provenance time of the current observation was not shorter than the queries to some sensors, data, processing, or services. The traversing times of weekly VCI, weekly NDVI, daily NDVI, and daily MODIS were tested as much as possible, instead of simply querying some objects at certain depths to obtain the maximum time cost. The results are shown in Figure 13a–d. Figure 13a shows the traversing time of the weekly VCI. The *x*-axis represents the 19th week of each year. The *y*-axis is the time cost (in milliseconds, ms). Figure 13a shows that the time increased linearly from 2000 to 2012. The maximum did not exceed 140 s. In 2012, there were 2367 tracking documents.

Figure 13b shows the 19th weeks of 2000–2012. This figure reveals that the traversing time for weekly NDVI was approximately 10 s (an average of 9948.615 ms). There were 170 tracking documents. Figure 13c,d show the time cost for the daily NDVI and MODIS for 12 May between 2000 and 2012. No more than 2 s and 1 s were needed to traverse the daily NDVI and MODIS, respectively. There were 27 and 26 tracking documents. Comparing Figure 13a,d, the time cost of each tracking document was 57.186 (135359/2367), 58.521 (9948.615/170), 50.915 (1374.692/27), and 36.980 (961.4615/26). The time costs were very similar, except for the daily MODIS, which may be because of a random error (there are inherently unpredictable fluctuations when performing POST requests to Web services). If there were more tests, we would find fewer random errors, given that testing the time cost of each of the 2367 documents 170 times yielded very similar results.

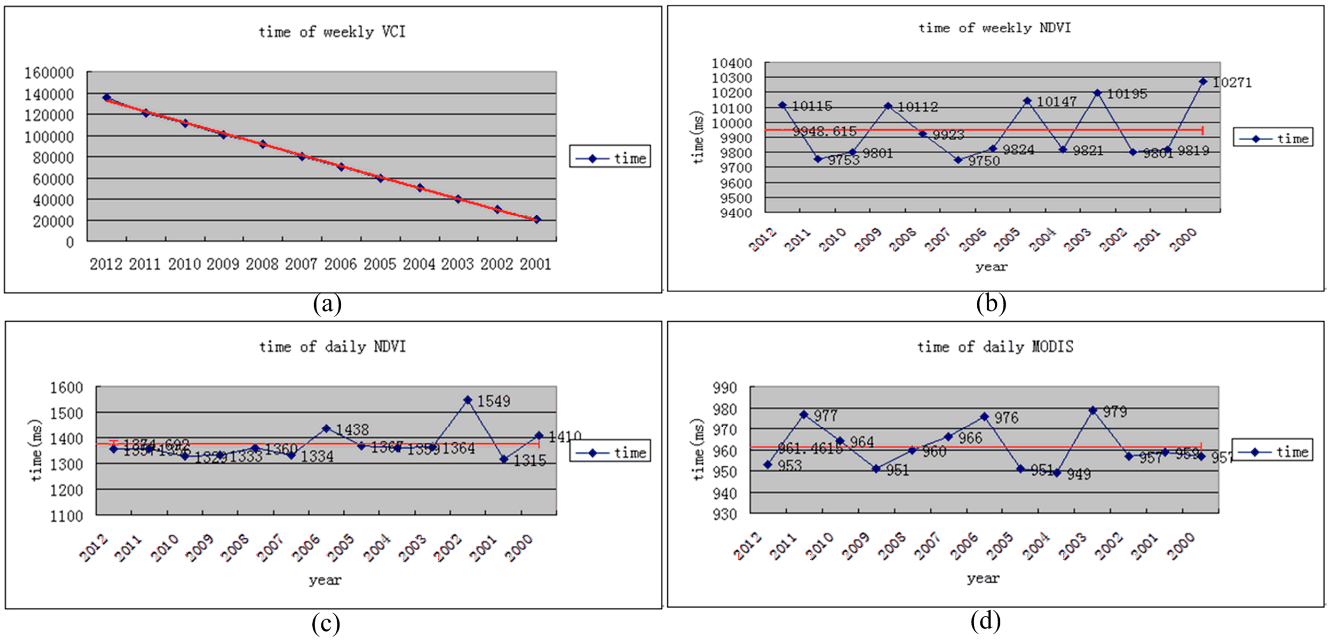


Figure 13. Traversing time of weekly VCI (**a**); weekly NDVI (**b**); daily NDVI (**c**); and daily MODIS (**d**).

The above time analysis reveals the following. In this experiment, when the maximum provenance depth was large (six) and there were hundreds to thousands of tracking documents (2367), the tracking time was short (120 s), and the time cost per document became stable as the tracking documents increased (50–60 ms). Additionally, the execution time grew linearly with the number of tracked documents, as shown in Figure 13a.

Space complexity is another aspect of the performance of this provenance model. The memory or storage required by the provenance information is affected by the size of the O&M document. The data entities are linked by a URI to reduce the size of the O&M document, as mentioned in Section 3.1. Other information is recorded in the O&M document. The O&M document sizes for tile MODIS, daily MODIS, daily NDVI, and weekly NDVI were approximately 3, 7, 5, and 5 KB, respectively. The maximum size for the weekly VCI was approximately 29 KB. Therefore, the O&M was not large. The results demonstrate the space complexity of this experiment was in the order of KBs.

4. Discussion and Conclusions

Tracking provenance information is an important and challenging issue for remote sensing observations in the Sensor Web. This paper proposed a method for representing and tracking provenance in the Sensor Web, focusing on four objects that are generally considered in remote sensing applications. We conducted an experiment to test the representation and tracking of the provenance objects (sensors, processing, data, and services) for vegetation conditions represented by NDVI and VCI. Our conclusions can be summarized as follows.

- The proposed provenance representation model is a Sensor Web, domain-specific model. Compared with provenance studies in terms of database, workflow, Web, international specification, and distributed system methods, the work of this paper mainly focused on a provenance representation that can be integrated with Sensor Web specifications. The provenance model was integrated into the Sensor Web specifications without affecting the structures, semantic relationships, and framework. We proposed a provenance model and a tracking approach, but did not consider the implementation, which is left to a developer.
- The designed provenance method can represent and track provenance information for remote sensing observations in a Sensor Web enabled environment. We conducted an experiment to test the representation and tracking in terms of the sensor, processing, data, and service objects, considering vegetation conditions represented by the NDVI and VCI for May from 2000 to 2012.
- Although the performance of the provenance method is associated with its implementation, we can consider the time and space complexities in this experiment. The time cost of tracking several depths and hundreds or thousands of documents was in the order of tens to hundreds of seconds. We analyzed the performance for our experiment based on six provenance depths. The average tracking time per document ranged from 50 to 60 ms. The size of the O&M documents was in the order of 10 KBs. Numerous remotely sensed observations may be processed up to a dozen times in this application. The execution time grew linearly with the tracked documents Figure 13a. The average time cost per tracking document was not significantly affected, as shown in Figure 13a–d. Therefore, we can deduce that as the execution time and required storage increased, the cost increased linearly.
- The proposed framework can be applied in other environments. Although the provenance model is based on the Sensor Web framework, it may be extended to record provenance information for Web services. If the processed observations are described with O&M, and the parent O&M is tracked with a xlink:href in omp:ParentOM without sending a GET/POST request, the provenance information can also be tracked. If an O&M document is used to describe the observation's metadata, the provenance information can also be recorded. The service information should be set to null. However, the procedure information should be described in more detail to explain how the observations were handled, which may increase the scope of this model.

Future work will focus on optimizing the performance of this framework and testing it using hundreds of tracking depths and thousands of tracking documents, attempting to apply it on a larger scale, and considering a security strategy.

Acknowledgments

This work was supported by grants from the National Basic Research Program of China (973 Program) (no. 2011CB707101), the National Nature Science Foundation of China program (nos. 41301441, 41171315), the Fundamental Research Funds for the Central Universities (no.2042014kf0200), and China Postdoctoral Science Foundation funded project (no. 2014M562050).

Author Contributions

Zeqiang Chen and Nengcheng Chen conceived and designed the project. Zeqiang Chen performed the experiments. Zeqiang Chen and Nengcheng Chen wrote the paper. Nengcheng Chen reviewed and edited the manuscript. All authors read and approved the manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

References and Notes

1. Bröring, A.; Echterhoff, J.; Jirka, S.; Simonis, I.; Everding, T.; Stasch, C.; Liang, S.; Lemmens, R. New generation sensor web enablement. *Sensors* **2011**, *11*, 2652–2699.
2. Chen, Z.; Chen, N.; Yang, C.; Di, L. Cloud computing enabled Web Processing Service for Earth Observation data processing. *IEEE J. Sel. Topics Appl. Earth Obs. Remote Sens.* **2012**, *5*, 1637–1649.
3. Kridskron, A.; Chen, N.; Peng, C.; Yang, C.; Gong, J. Flood detection and mapping of the Thailand Central plain using RADARSAT and MODIS under a Sensor Web enabled environment. *Int. J. Appl. Earth Obs. GeoInform.* **2012**, *13*, 245–255.
4. Chen, N.; Di, L.; Chen, Z.; Gong, J. An efficient method for near-real-time on-demand retrieval of remote sensing observations. *IEEE J. Sel. Topics Appl. Earth Obs. Remote Sens.* **2011**, *4*, 615–625.
5. Chen, Z.; Chen, N.; Di, L.; Gong, J. A flexible Data and Sensor Planning Service for virtual sensors based on Web Service. *IEEE Sens. J.* **2011**, *11*, 1429–1439.
6. Chen, N.; Li, D.; Di, L.; Gong, J. An automatic SWILC classification and extraction for the AntSDI under a Sensor Web enabled environment. *Can. J. Remote Sens.* **2010**, *36(Supplement 1)*, 1–12.
7. Chen, N.; Di, L.; Yu, G.; Gong, J. Geo-processing workflow driven wildfire hot pixel detection under sensor web enabled environment. *Comput. Geosci.* **2010**, *36*, 362–372.
8. W3C Workshop RDF Next Steps document, Provenance requirements for the next version of rdf; 2010; p. 5. Available online: http://www.w3.org/2005/Incubator/prov/wiki/images/3/3f/RDFNextStep_ProvXG-submitted.pdf (accessed on 11 January 2014).
9. A White Paper for NSF EarthCube, Provenance in earth science cyberinfrastructure; 2011; p. 7. Available online: http://semanticommunity.info/@api/deki/files/13808/020_Di.pdf (accessed on 11 January 2014).
10. Groth, P.; Miles, S.; Moreau, L. PReServ: Provenance Recording for Services. In Proceedings of the UK OST e-Science second All Hands Meeting 2005, Nottingham, UK, 19–22 September 2005.
11. Michlmayr, A.; Rosenberg, F.; Leitner, P.; Dustdar, S. Service provenance in qos-aware web service runtimes. In Proceedings of the IEEE International Conference on Web Services 2009, Los Angeles, CA, USA, 6–10 July 2009; pp. 115–122.
12. Buneman, P.; Khanna, S.; Tan, W. Why and where: A characterization of data provenance. In Proceedings of the 8th International Conference on Database Theory, London, UK, 4–6 January 2001.
13. Cheney, J.; Chiticariu, L.; Tan, W. Provenance in databases: Why, how, and where. *Found. Trends Databases.* **2009**, *1*, 379–474.

14. Ram, S.; Liu, J. A new perspective on semantics of data provenance. In Proceedings of the First International Workshop on the role of Semantic Web in Provenance Management (SWPM 2009), Washington, DC, USA, 25–26 October 2009.
15. Simmhan, Y.; Plale, B.; Gannon, D. A framework for collecting provenance in data-centric scientific workflows. In Proceedings of the International Conference on Web Services, Chicago, IL, USA, 18–22 September 2006; pp. 427–436.
16. Zhao, Y.; Wilde, M.; Foster, I. Applying the virtual data provenance model. In Proceedings of the International Provenance and Annotation Workshop 2006 (IPAW2006), Chicago, IL, USA, 3–5 May 2006.
17. Technical Report, Tracking RDF Graph Provenance Using RDF Molecules; 2005; p. 13. Available online: http://ebiquity.umbc.edu/get/a/publication/178.pdf (accessed on 11 January 2014).
18. Hausenblas, M.; Slany, W.; Ayers, D. A performance and scalability metric for virtual RDF graphs. In Proceedings of the 3rd Workshop on Scripting for the Semantic Web (SFSW) at ESWC, Innsbruck, Austria, 30 May 2007.
19. Hartig, O. Provenance information in the web of data. In Proceedings of the Linked Data on the Web Workshop 2009, Madrid, Spain, 20 April 2009.
20. Moreaua, L.; Clifford, B.; Freireb, J.; Futrellec, J.; Gild, Y.; Grothe, P.; Kwasnikowskaf, N.; Milesg, S.; Missierh, P.;Myersc, J.; *et al.* The open provenance model core specication (v1.1). *Future Gener. Comput. Syst.* **2011**, *27*, 743–756.
21. Missier, P.; Belhajjame, K.; Cheney, J. The W3C PROV family of specifications for modelling provenance metadata. In Proceedings of the 16th International Conference on Extending Database Technology, Genoa, Italy, 18–22 March 2013.
22. Di, L.; Shao, Y.; Kang, L. Implementation of geospatial data provenance in a web service workflow environment with ISO 19115 and ISO 19115-2 lineage model. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 5082–5089.
23. Baumann, P. *OGC Implementation Standard 09-110r4: OGC® WCS 2.0 Interface Standard—Core: Corrigendum, Version 2.0.1*; Open Geospatial Consortium: Wayland, MA, USA, 2012.
24. Ledlie, J.; NG, C.; Holland, D.; Muniswamy-Reddy, K.; Braun, U.; Seltzer, M. Provenance-aware sensor data storage. In Proceedings of the 21st International Conference on Data Engineering Workshops 2005, Tokyo, Japan, 5 April 2005.
25. Botts, M.; Robin, A. *OGC Implementation Standard 12-000: OpenGIS SensorML: Model and XML Encoding Standard, Version 2.0.0*; Open Geospatial Consortium: Wayland, MA, USA, 2012.
26. Cox, S. *OGC Abstract Specification 10-004r3: Geographic Information: Observations and Measurements OGC Abstract Specification Topic 20*; Open Geospatial Consortium: Wayland, MA, USA, 2010.
27. Cox, S. *OGC Implementation Standard 10-025r1: Observations and Measurements—XML Implementation, Version 2.0*; Open Geospatial Consortium: Wayland, MA, USA, 2011.
28. Bröring, A.; Stasch, C.; Echterhoff, J. *OGC Implementation Specification 12-006: OGC® Sensor Observation Service Interface Standard, Version 2.0*; Open Geospatial Consortium: Wayland, MA, USA, 2012.
29. Schut, P. *OGC Implementation Specification 05-007r7: OGC® Web Processing Service, Version 1.0.0*; Open Geospatial Consortium: Wayland, MA, USA, 2007.

30. Simonis, I.; Echterhoff, J. *OGC Implementation Standard 09-000: OGC® Sensor Planning Service Implementation Standard*; Open Geospatial Consortium: Wayland, MA, USA, 2011.
31. Singh, R.; Roy, S.; Kogan, F. Vegetation and temperature condition indices from NOAA AVHRR data for drought monitoring over India. *Int. J. Remote Sens.* **2003**, *24*, 4393–4402.
32. Yang, Z.; Di, L.; Yu, G.; Chen, Z. Vegetation condition indices for crop vegetation condition monitoring. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Vancouver, Canada, 24–29 July 2011; pp. 3534–3537.
33. *MODIS Surface Reflectance User's Guide*; 2011; p. 40. Available online: http://www.modis-sr.ltdri.org/products/MOD09_UserGuide_v1_3.pdf (accessed on 11 January 2014).
34. Kurita, H.; Hatano, K.; Miyazaki, J.; Uemura, S. Efficient query processing for large XML data in distributed environments. In Proceeding of the 21st International Conference on Advanced Information Networking and Applications 2007, Niagara Falls, Canada, 21–23 May 2007; pp. 317–322.
35. Ng, W.; Lam, W.Y.; Cheng, J. Comparative analysis of XML compression technologies. *World Wide Web.* **2006**, *9*, 5–33.