DigitalGlobe™

Web Coverage Service Developer Guide

Cloud Services | August 2013

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 About This Document

This document covers the concepts of Web Coverage Service (WCS), Open Geospatial Consortium (OGC) standards for WCS, capabilities of WCS and ways to integrate DigitalGlobe Cloud Services (DGCS)-WCS in GIS-based custom application development.

1.2 Targeted Audience

This document is designed to help developers of GIS-based custom application development. Developers new to WCS can read about the DGCS-WCS framework, capabilities, integration procedures and development best-practices to design methods for creating innovative world-class GIS applications.

1.3 What is WCS?

DigitalGlobe's WCS provides an interface for users to access WCS via web browsers. The current implementation includes a license file that is coded into the imagery. OGC-compliant WCS-clients can strip the license file away and display the imagery correctly. However, images downloaded directly out of WCS via a web browser will not be displayed in GIS tools.

The basic WCS allows querying and retrieval of coverage. The WCS describes discovery, query, or data transformation operations. The client generates the request and posts it to a web coverage server using HTTP. The web coverage server then executes the request. The WCS specification uses HTTP as the distributed computing platform, although this is not a hard requirement.

1.4 References

- http://www.opengeospatial.org/standards
- http://en.wikipedia.org/wiki/GIS#OGC_standards
- http://www.opengeospatial.org/standards/wcs
- http://en.wikipedia.org/wiki/Web_Coverage_Service
- http://en.wikipedia.org/wiki/Geography_Markup_Language

2 Open Geospatial Consortium (OGC)

2.1 About OGC

The Open Geospatial Consortium (OGC®) is an international voluntary consensus standards organization, which originated in 1994. In the OGC, more than 400 commercial, governmental, nonprofit and research organizations worldwide collaborate in a consensus process. This process encourages development and implementation of open standards for geospatial content and services, GIS data processing and data sharing.

A predecessor organization, the Open GRASS Foundation (OGF), started in 1992. From 1994 to 2004 the organization also used the name “Open GIS Consortium”.

2.2 The OGC Process

The OGC exists to enable a fast, effective, inclusive, user-driven process to develop, test, demonstrate, and promote the use of geospatial information and services by using OpenGIS® Standards.

The OGC has defined the standards around different GIS Cloud Services by following the process of identifying and addressing existing problems in the GIS world. The steps described below are the process followed by OGC:

- Identifying Problem
- Crafting Solution
- Evaluating Proposed Solution, and
- Implementing Standards

One of the major problems identified and addressed is interoperability. The following were discussed and prioritized as part of identifying and addressing the interoperability problem.

- Sharing maps on the Web.
- Delivering data to different systems easily.
- Common language to speak about geospatial data or services.
- Finding and pulling together data from our automated sensors.

2.3 OGC Standards and Specification

OGC Standards and Specifications are technical documents that detail interfaces or encodings. Software developers use these documents to build support for the interfaces or encodings into their products and services. These specifications are the main “products” of the OGC and have been developed by the membership to address specific interoperability challenges.
The OGC technical documents have been developed by members to address specific interoperability challenges. The OGC documents are available at no cost to everyone. Refer to Table 2.1 for a list of documents currently available on the OGC website.

TABLE 2.1 OGC DOCUMENT TYPES

OGC DOCUMENT TYPE	DESCRIPTION
OpenGIS Implementation Standard	A document containing an OGC consensus, technology-dependent standard for application programming interfaces and related standards based on the Abstract Specification or domain-specific extensions to the Abstract Specification. There are five subtypes: Interface, Encoding, Profile, Application Profile, and Application Schema.
Abstract Specification	A document (or set of documents) containing an OGC consensus, technology-independent standard for application programming interfaces and related standards based on object-oriented or other IT-accepted concepts. It describes and/or models an application environment for interoperable geoprocessing and geospatial data and services products.
Best Practices	A document containing discussion related to the use and/or implementation of an adopted OGC document. Best Practices documents are an official position of the OGC and thus represent an endorsement of the content of the paper.

Discussion Papers	A document containing discussion of some technology or standard area for release to the public. Discussion Papers are not the official position of the OGC and contain a statement to that effect.
White Papers	A publication released by the OGC to the public that states a position on a social, political, technical or other subject, often including a high-level explanation of an architecture or framework of a solution.

2.4 OGC Standards

OGC Standards are written for a more technical audience and detail the interface structure between software components. An interface specification is considered to be at the implementation level of detail if, when implemented by two different software engineers in ignorance of each other, the resulting components plug and play with each other at that interface.

2.5 Abstract Specification

The OGC Technical Committee (TC) has developed architecture in support of its vision of geospatial technology and data interoperability called the OGC Abstract Specification. The Abstract Specification provides the conceptual foundation for most OGC specification development activities. Open interfaces and protocols are built and referenced against the Abstract Specification, thus enabling interoperability between different brands and different kinds of spatial processing systems. The Abstract Specification provides a reference model for the development of OGC Implementation Specifications.

2.6 OGC Reference Model (ORM)

The OGC Reference Model (ORM) provides a framework for the ongoing work of the OGC. The ORM describes the OGC Standards Baseline (SB) focusing on the relationships between the OpenGIS Specification documents. The OGC SB consists of the approved OGC Abstract and Implementation Specifications as well as OGC Best Practices documents. Best Practices documents are official positions of the OGC members and quite often are provided as supporting technical information for the adopted Specifications.

Advantages or the purpose of ORM are:

- Provides an overview of OGC Standards Baseline
- Provides insight into the current state of the work of the OGC
- Serves as a basis for coordination and understanding of the documents in OGC SB
- Provides a useful resource for defining architectures for specific applications.

➔ NOTE: Visit the following link for detailed information on OGC standards and specifications. http://www.opengeospatial.org/standards

3 Geography Markup Language (GML)

3.1 Introduction to GML

The Geography Markup Language (GML) is the XML code used by the OGC to express geographical features. GML serves as a modeling language for geographic systems as well as an open interchange format for geographic transactions on the Internet. Note that the concept of feature in GML is a very general one and includes not only conventional "vector" or discrete objects, but also coverages and sensor data. The ability to integrate all forms of geographic information is key to the utility of GML.

GML was conceived and evolved for a variety of reasons, the most important reasons being:

- To provide a language for expressing geographic entities – to create application specific geographic vocabularies.
- To enable the encoding of geographic information consistent with these vocabularies.
- To support geospatial queries and transactions across the Internet.

GML is feature-centric. Features are entities – things that describe aspects of the real world from the perspective of a particular application community – whether circumscribed by geography or function or both. GML vocabularies are created by communities of interest. These vocabularies are called GML Application Schemas. If you look at such an Application Schema you will find real world objects like Buildings, Roads, Buoys, Navigation Aids, Airline Flight Paths, Vehicles and Railway Switches. Each such object is defined in the schema by listing its properties. For example, a Building might be described by:

```
<abc:Building gml:id='b143'>
      <abc:height>40</abc:height>
      <abc:footprint>
            <gml:polygon></gml:polygon>
      </abc:footprint>
</abc:Building>
```

Note that the Building (feature) has two properties, namely height and footprint. The height property in this case has an integer value (number of stories), while the footprint property has a Polygon (shape) for a value.

GML application schemas can be the basis of standards themselves – such as S57GML, cityGML, geoRSS GML and AIXM, or they can be informal creations for only a very small community. Which is the case is up to the community.

GML application schemas should NOT be confused with GML profiles. A GML profile is a subset of GML, defined usually by the subset tool (part of the GML specification), consisting of selected element, attribute and type declarations and all dependent components from the GML core schemas (the schemas defined by the GML specification). Application schemas can be built on GML profiles. Some GML profiles are also specifications and this includes the GML Simple Features Profile, the Point Profile, the GML Profile for GMLJP2 and the GML Profile for GeoRSS.

GML was developed to support geographic requests and transactions and this usage predates the WCS developed for this purpose. When a user sends a request for geographic data – e.g. "find all water wells within this county" – there must be a way to express "water well", "county" and the "geometric extend of the county". In WCS, GML is used for this purpose. When the user wants to send a transaction such as "change the shape of the Holmes River to the following …" they need a way to express the river's geometry; GML provides this mechanism in the WCS.

3.2 Overview of GML Schema

GML specifies XML encodings of a number of the conceptual classes defined in the ISO 19100 series of International Standards and the OpenGIS Abstract Specification in conformance with these standards and specifications.

In many cases, the mapping from the conceptual classes to XML is straightforward, while in some cases the mapping is more complex.

In addition, GML provides XML encodings for additional concepts not yet modeled in the ISO 19100 series of International Standards or the OpenGIS Abstract Specification. Examples include moving objects, simple observations or value objects. Additional conceptual classes corresponding to these extensions are also specified in Annex D.

The GML schema comprises the components (XML elements, attributes, simple types, complex types, attribute groups, groups, etc.) that are described in this International Standard. The XML encoding conforms to ISO 19118.

3.3 GML Schema Features

A GML feature is a feature encoded using GML. Examples include a road, a river, a person, a vehicle, an administrative area, or an event.

The feature schema provides a framework for the creation of GML features and feature collections.

3.3.1 ABSTRACTFEATURETYPE

The basic feature model is given by the `gml:AbstractFeatureType`, defined in the schema as follows:

```
<complexType name="AbstractFeatureType" abstract="true">
      <complexContent>
            <extension base="gml:AbstractGMLType">
                  <sequence>
                        <element ref="gml:boundedBy" minOccurs="0"/>
                        <element ref="gml:location" minOccurs="0"/>
                  </sequence>
            </extension>
      </complexContent>
</complexType>
```

The content model for `gml:AbstractFeatureType` adds two specific properties suitable for geographic features to the content model defined in `gml:AbstractGMLType`.

The value of the `gml:boundedBy` property describes an envelope that encloses the entire feature instance, and is primarily useful for supporting rapid searching for features that occur in a particular location.

The value of the `gml:location` property describes the extent, position or relative location of the feature. `gml:location` is deprecated as part of the standard content model of `gml:AbstractFeatureType`.

3.3.2 ABSTRACTFEATURE

The element `gml:AbstractFeature` is declared as follows:

```
<element name="AbstractFeature"
           type="gml:AbstractFeatureType"
           abstract="true"
           substitutionGroup="gml:AbstractGML"/>
```

This abstract element serves as the head of a substitution group which may contain any elements whose content model is derived from `gml:AbstractFeatureType`. This may be used as a variable in the construction of content models.

`gml:AbstractFeature` may be thought of as anything that is a GML feature and may be used to define variables or templates in which the value of a GML property is “any feature”. This occurs in particular in a GML feature collection where the feature member properties contain one or multiple copies of `gml:AbstractFeature` respectively.

The Other features which are used are **boundedBy**, **BoundingShapeType**, **EnvelopeWithTimePeriod**, **EnvelopeWithTimePeriodType**, **locationName**, **locationReference**, **FeaturePropertyType**, **FeatureArrayPropertyType**.

3.4 Overview

Many applications require definitions of terms which are used within instance documents as the values of certain properties or as reference information to tie properties to standard information values in some way. Units of measure and descriptions of measurable phenomena are two particular examples.

It will often be convenient to use definitions provided by external authorities. These may already be packaged for delivery in various ways, both online and offline. In order that they may be referred to from GML documents it is generally necessary that a URI be available for each definition. Where this is the case then it is usually preferable to refer to these directly.

Alternatively, it may be convenient or necessary to capture definitions in XML, either as a separate document or embedded within an instance document containing features. The definitions may be transcriptions from an external source, or may be new definitions for a local purpose. In order to support this case, some simple components are provided in GML in the form of:

- A generic `gml:Definition`, which may serve as the basis for more specialized definitions.
- A generic `gml:Dictionary`, which allows a set of definitions or references to definitions to be collected.

These components may be used directly, but also serve as the basis for more specialized definition elements in GML, in particular: coordinate operations (Clause 12), coordinate reference systems (Clause 12), datums (Clause 12), temporal reference systems (Clause 14), and units of measure (Clause 16).

Note that the GML definition and dictionary components implement a simple nested hierarchy of definitions with identifiers. The latter provide handles which may be used in the description of more complex relationships between terms. However, the GML dictionary components are not intended to provide direct support for complex taxonomies, ontologies or thesauri. Specialized XML tools are available to satisfy the more sophisticated requirements.

➔ The dictionary schema document is identified by the following location-independent name (using URN syntax): urn:x-ogc:specification:gml:schema-xsd:dictionary:3.2.1.

3.4.1 DICTIONARY SCHEMA

Definition, DefinitionType, remarks

The basic `gml:Definition` element specifies a definition, which can be included in or referenced by a dictionary. It is declared as follows:

The content model for a generic definition is a derivation from `gml:AbstractGMLType`.

```
<element name="Definition" type="gml:DefinitionType"
substitutionGroup="gml:AbstractGML"/>
 <complexType name="DefinitionBaseType">
    <complexContent>
        <restriction base="gml:AbstractGMLType">
             <sequence>
                <element ref="gml:metaDataProperty" minOccurs="0"
maxOccurs="unbounded"/>
                <element ref="gml:description" minOccurs="0"/>
                <element ref="gml:descriptionReference" minOccurs="0"/>
                <element ref="gml:identifier"/>
                <element ref="gml:name" minOccurs="0" maxOccurs="unbounded"/>
              </sequence>
               <attribute ref="gml:id" use="required"/>
         </restriction>
    </complexContent>
</complexType>
<complexType name="DefinitionType">
    <complexContent>
          <extension base="gml:DefinitionBaseType">
             <sequence>
               <element ref="gml:remarks" minOccurs="0"/>
             </sequence>
          </extension>
    </complexContent>
</complexType>
<element name="remarks" type="string"/>
```

The `gml:description` property element shall hold the definition if this can be captured in a simple text string, or the `gml:descriptionReference` property element may carry a link to a description elsewhere.

The `gml:identifier` element shall provide one identifier identifying this definition. The identifier shall be unique within the dictionaries using this definition.

The `gml:name` elements shall provide zero or more terms and synonyms for which this is the definition.

The `gml:remarks` element shall be used to hold additional textual information that is not conceptually part of the definition but is useful in understanding the definition.

Dictionary, DictionaryType

Sets of definitions may be collected into dictionaries or collections. These are declared in the schema as follows:

```
<element name="Dictionary" type="gml:DictionaryType"
substitutionGroup="gml:Definition"/>
<complexType name="DictionaryType">
      <complexContent>
            <extension base="gml:DefinitionType">
                  <choice minOccurs="0" maxOccurs="unbounded">
                        <element ref="gml:dictionaryEntry"/>
                        <element ref="gml:indirectEntry"/>
                  </choice>
                  <attributeGroup ref="gml:AggregationAttributeGroup"/>
            </extension>
      </complexContent>
</complexType>
```

A `gml:Dictionary` is a non-abstract collection of definitions.

The `gml:Dictionary` content model adds a list of `gml:dictionaryEntry` and *`gml:indirectEntry` (deprecated)* properties that contain *or reference* `gml:Definition` objects. A database handle (`gml:id` attribute) is required, in order that this collection may be referred to. The standard `gml:identifier`, `gml:description`, `gml:descriptionReference` and `gml:name` properties are available to reference or contain more information about this dictionary. The `gml:description` and `gml:descriptionReference` property elements may be used for a description of this dictionary. The derived `gml:name` element may be used for the name(s) of this dictionary.

dictionaryEntry, DictionaryEntryType

These elements contain or refer to the definitions which are members of a dictionary. The element `gml:dictionaryEntry` is declared as follows:

```
<element name="dictionaryEntry" type="gml:DictionaryEntryType"/>
<complexType name="DictionaryEntryType">
      <complexContent>
            <extension base="gml:AbstractMemberType">
                  <sequence minOccurs="0">
                        <element ref="gml:Definition"/>
                  </sequence>
                  <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </extension>
      </complexContent>
</complexType>
```

The content model follows the standard GML property pattern, so a `gml:dictionaryEntry` may either contain or refer to a single `gml:Definition`. Since `gml:Dictionary` is substitutable for `gml:Definition`, the content of an entry may itself be a lower-level dictionary.

Note that if the value is provided by reference, this definition does *not* carry a handle (`gml:id`) in this context, so does *not* allow external references to this specific definition in this context. When used in this way the referenced definition will usually be in a dictionary in the same XML document.

Using definitions and dictionaries

Dictionaries and definitions are GML objects, so may be found in independent GML data instance documents.

In application schemas it might be useful to attach a `gml:Dictionary` or `gml:Definitions` to a feature collection in order to record definitions used in properties of members of the collection.

EXAMPLE: The following example shows two instances of dictionaries:

```
<gml:Dictionary gml:id="rockTypes">
      <gml:description>
            A simple dictionary of rock types using components from gmlBase
      </gml:description>
      <gml:identifier codeSpace="http://www.abc.org/terms">
             Rock Types
      </gml:identifier>
      <gml:dictionaryEntry>
            <gml:Definition gml:id="granite">
                  <gml:description>
                         A igneous rock normally composed of quartz, two feldspars
and optional mica
                  </gml:description>
                  <gml:identifier codeSpace="http://www.abc.org/terms">
                           Granite
                   </gml:identifier>
            </gml:Definition>
      </gml:dictionaryEntry>
      <gml:dictionaryEntry>
            <gml:Definition gml:id="sst">
                  <gml:description>
                            A detrital sedimentary rock normally composed of
siliceous grains
                   </gml:description>
                  <gml:identifier codeSpace="http://www.abc.org/terms">
                             Sandstone
                  </gml:identifier>
            </gml:Definition>
      </gml:dictionaryEntry>
      <gml:dictionaryEntry
xlink:href="http://my.big.org/definitions/geology/limestone"/>
</gml:Dictionary>
<gml:Dictionary gml:id="AbridgedGMLdictionary">
<gml:identifier codeSpace="http://www.opengis.net/gml/3.2">
GML Dictionary
</gml:identifier>
<gml:dictionaryEntry>
      <gml:Definition gml:id="term4.1">
            <gml:description>
conceptual schema for data required by one or more applications
</gml:description>
            <gml:identifier codeSpace="http://www.isotc211.org/19101">
application schema
</gml:identifier>
      </gml:Definition>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
      <gml:Definition gml:id="term4.2">
            <gml:description>
```

Continued…

```
…Continued

application schema written in XML Schema in accordance with the rules specified in
ISO 19136
</gml:description>
<gml:identifier codeSpace="http://www.opengis.net/gml/3.2">
GML application schema
</gml:identifier>
          </gml:Definition>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
      <gml:Definition gml:id="term4.3">
            <gml:description>
semantic relationship between two or more classifiers that specifies connections
among their instances

</gml:description>
            <gml:identifer
codeSpace="http://www.uml.org/1.3">association</gml:identifier>
      </gml:Definition>
</gml:dictionaryEntry>
<gml:dictionaryEntry>
      <gml:Definition gml:id="term4.4">
            <gml:description>name-value pair contained in an
element</gml:description>
            <gml:identifer
codeSpace="http://www.w3.org/XML/1998/namespace">attribute</gml:identifier>
      </gml:Definition>
</gml:dictionaryEntry>
<!--.. -->
</gml:Dictionary>
```

3.5 GML-Based Request

The request to the WCS for any of the operations can be sent as a GML-Based request instead of the plain Get URL. The Key and value pair of parameters that we send to the WCS server is now sent through the GML Based request in the following way:

```
<?xml version="1.0" encoding="utf-8"?>
<GetCoverage
xmlns="http://www.opengis.net/WCS"xmlns:DigitalGlobe=http://www.digitalglobe.com
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:gml="http://www.opengis.net/gml" service="WCS" version="1.1.0"
outputFormat="text/xml; subtype=gml/3.1.1" maxFeatures="100" handle="" >
      <Query typeName="DigitalGlobe:FinishedFeature" srsName="urn:x-
ogc:def:crs:EPSG:4326">
            <ogc:Filter>
                  <ogc:Intersects>
                        <ogc:PropertyName>geometry</ogc:PropertyName>
                        <gml:Envelope srsName="urn:x-ogc:def:crs:EPSG:4326"

xmlns:gml="http://www.opengis.net/gml">
                              <gml:lowerCorner>-90 -180</gml:lowerCorner>
                              <gml:upperCorner>90 180</gml:upperCorner>
                        </gml:Envelope>
                  </ogc:Intersects>
            </ogc:Filter>
      </Query>
</GetCoverage>
```

The <GetCoverage> element contains one or more <Query> elements, each of which contain the description of a query. The results of all queries contained in a GetCoverage request are concatenated to produce the result set.

The outputFormat attribute defines the format to use to generate the result set. The default value is GML2.

The optional maxFeatures attribute can be used to limit the number of features that a GetCoverage request retrieves. Once the maxFeatures limit is reached, the result set is truncated at that point.

Each individual query packaged in a GetCoverage request is defined using the <Query> element. The <Query> element defines which feature type to query, what properties to retrieve and what constraints (spatial and non-spatial) to apply to those properties.

The typeName attribute is used to indicate the name of the feature type or class to be queried.

The featureVersion attribute is included in order to accommodate systems that support feature versioning. A value of ALL indicates that all versions of a feature should be fetched. Otherwise, an integer, n, can be specified to return the nth version of a feature. The version numbers start at 1, which is the oldest version. If a version value larger than the largest version number is specified, then the latest version is returned. The default action shall be for the query to return the latest version. Systems that do not support versioning can ignore the parameter and return the only version that they have.

The <PropertyName> element is used to enumerate the feature properties that should be selected during a query and whose values should be included in the response to a GetCoverage request. A client application can determine the properties of a feature by making a DescribeCoverageType request before composing a GetCoverage request. The DescribeCoverageType operation [sec. 8] will generate a GML application schema defining the schema of the feature type. The client can then select the properties to be fetched. In addition, the client can determine which feature properties are mandatory and must be fetched in order for the WCS to be able to generate an instance of the feature type that will validate against the generated GML application schema. In the event that a WCS encounters a query that does not select all mandatory properties of a feature, the WCS will internally augment the property name list to include all necessary property names. A WCS client must thus be prepared to deal with a situation where it receives more property values than it requests.

If no <PropertyName> elements are specified, then all feature properties should be fetched.

The <Filter> element can be used to define constraints on a query. Both spatial and/or non-spatial constraints can be specified as described in the Filter Encoding Specification [3]. If no <Filter> element is contained within the <Query> element, then the query is unconstrained and all feature instances should be retrieved.

4 Web Coverage Service (WCS)

4.1 Introduction

The WCS allows the end user to directly download raster imagery data in either JPEG2000 or GeoTIFF format. The WCS supports the following operations:

GetCapabilities

The GetCapabilities request is used to determine the supported Coverages; each FinishedCatalog product will be listed and described as a Coverage.

DescribeCoverage

The DescribeCoverage request is used to obtain the detailed description of a supported Coverage.

GetCoverage

The GetCoverage request is used to obtain the actual product pixels (imagery) for download.

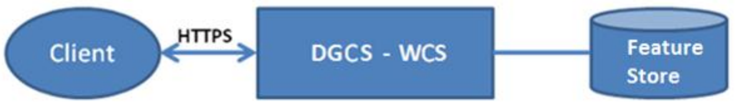

FIGURE 4.1 A TYPICAL STRUCTURE OF A DGCS-WCS APPLICATION

4.2 Service Details

The DigitalGlobe WCS supports KVP (Keyword Value Pair) request encoding only; SOAP (Simple Object Access Protocol) or other protocols are not supported.

The WCS does NOT support the following optional capabilities:

- Range Subsetting
- STORE capability; all coverages are returned synchronously

➔ NOTE: Because full-resolution imagery is being retrieved, sometimes over limited bandwidth connections, the WCS will determine the size of the requested product, and if it exceeds 1GB, an exception will be returned. Since the limit is based on file size, larger areas can be downloaded if the compressed JPEG2000 format is requested compared to GeoTIFF.

➔ NOTE: There will be a set of white-listed domains where a user can download unlimited quantities of Web Coverages. These include domain extensions such as .mil, .gov, etc. Users accessing the service from a non-white-listed domain will be limited to 1GB of downloaded data per day. Once this limit is exceeded, the user's CONNECTID will be disabled from non-white-listed domains, this CONNECTID will still be active from white-listed domains.

The WCS Service treats each available product as a separate coverage; therefore the imagery returned depends on the layer being accessed.

➔ NOTE: The parameter used to request coverage is the Feature Identifier. This makes it convenient to request metadata for the product via the Feature service, and then request the actual imagery product via the WCS, using the same identifier.

➔ NOTE: Since additional Coverages will become available as data is added to the Service, the response to a GetCapabilities request will be updated each time a new product is made available. Therefore, care should be taken when using client-side caching of results.

4.3 WCS Client Server Architecture

The following architecture depicts a sample integration of WCS client and server applications. Client Viewer is a series of HTML pages running inside a web browser that can interact with WCS server via client application through HTTP calls. WCS client manages the interactions with WCS interfaces through HTTP requests and dynamically generates HTML that can run in a Web browser.

WCS server accepts requests from WCS client and viewer client in the form of HTTP URL strings, and returns results encoded as XML, GIF, GML, and so on. The database stores geo-feature data that can be accessed and utilized by the WCS server to generate GML documents or draw maps.

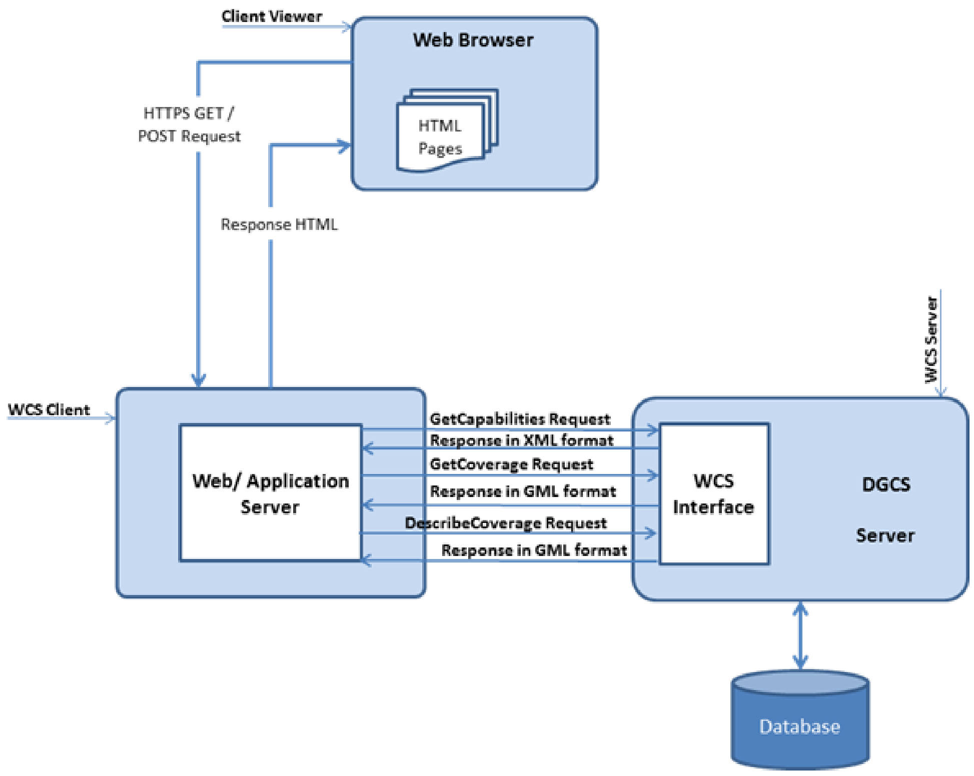

FIGURE 4.2 SAMPLE WCS CLIENT SERVER APPLICATION

4.4 WCS Service Details

The DigitalGlobe Web Coverage Service provides vector metadata, including imagery footprints, in Geographic Markup Language (GML) format. The DigitalGlobe WCS supports the following OGC-defined operations.

4.4.1 GETCAPABILITIES OPERATION

In response to the GetCapabilities request, the Service returns a description of the available Capabilities, including a list of all available Coverages. Since each Finished Product is defined as a Coverage, the GetCapabilities response can be quite large. In addition, as more data is added, the list of Coverages will grow and change over time, so clients must be careful if they are caching the results.

GetCapabilities Request Format

The GetCapabilities Request is formatted as shown; there are no other options available:

```
https://services.digitalglobe.com/deliveryservice/wcsaccess?service=WCS&request=GetCapabilitie
s&version=1.1.1&connectid=<CONNECTID>
```

Replace <ConnectID> with the ConnectID provided by DigitalGlobe. Parameters are not required to be in the same order as shown above.

GetCapabilities Response Format

In response to a GetCapabilities request, the Service returns an XML document that describes each available operation and Coverage.

4.4.2 DESCRIBECOVERAGE OPERATION

DescribeCoverage Request Format

The client provides the following information in a Keyword Value Pair (KVP) format, where the “name” field is the key, and the “value” field is the value; the data is supplied in the format “key=value”; for example, “service=WCS”.

TABLE 4.1 THE PARAMETERS OF A DESCRIBECOVERAGE REQUEST

NAME	VALUE	DESCRIPTION
service*	WCS	Web Coverage Service
version*	1.1.1	Request version
request*	DescribeCoverage	Request name
identifiers*	List of coverages to be described; Example: 55b795c5bd91a1abf7ff0e70ac52127e	The Coverage(s) to be described; each product in the layers are a coverage and can be requested; multiple identifiers are comma-separated; at least one valid identifier must be included. The coverage identifiers can be obtained from the GetCapabilities response.
connectId*	Character String	User’s unique identifier supplied by DigitalGlobe. This is required to access the DGCS.

* mandatory parameter

DescribeCoverage Response Format

In response to a DescribeCoverage request, an XML document is returned that defines the details of each requested Coverage, using a “CoverageDescription” data structure. The elements of this data structure for this WCS are defined in Table 4.11. This data structure is repeated for each Coverage specified via the “identifiers” values in the DescribeCoverage request.

The following example requests a description of the specified coverage:

https://services.digitalglobe.com/deliveryservice/wcsaccess?SERVICE=WCS&REQUEST=DescribeCoverage&version=1.1.1&CONNECTID=<ConnectID>&identifiers=83de5c8e080cb6842e2ee166a4a0ad0d

Replace <ConnectID> with the ConnectID provided by DigitalGlobe. Parameters are not required to be in the same order as shown above.

4.4.3 GETCOVERAGE OPERATION

The GetCoverage request results in an image product being returned to the requesting client. Since the WCS protocol does not support mosaicked images across coverages, separate GetCoverage requests are required for each Online FinishedCatalog product. Each Online FinishedCatalog product equates to a coverage.

GetCoverage Request

The GetCoverage request is used to request a full-resolution product file in one of the supported formats, currently GeoTIFF and JPEG2000.

The client provides the following information in a Keyword Value Pair (KVP) format, where the "name" field is the key, and the "value" field is the value; the data is supplied in the format "key=value"; for example, "service=WCS".

TABLE 4.2 GETCOVERAGE REQUEST PARAMETERS

NAME	VALUE	DESCRIPTION
service*	WCS	Web Coverage Service
version*	1.1.1	Request version
request*	GetCoverage	Request name
identifier*	A coverage identifier; for example: 55b795c5bd91a1abf7ff0e70ac52127e	The Coverage to be described.
boundingBox*	Example:35.6,-117.7,35.7,-117.6	The Bounding Box of the portion of the coverage to be returned (lat-lon).
Output:gridCRS	EPSG:4326 (each valid EPSG value for UTM)	The CRS in which the coverage response will be returned; supported CRSs are WGS84 Lat/Long and WGS84 UTM. If not included, the native CRS of the projection will be returned; for DGCS this will be WGS84 Lat/Long.
Output:gridOffsets*	Examples: UTM: 0.5, 0.5 Lat/Long: .0000045, .0000045	Specifies the pixel size to be returned, in the X and Y dimensions. For gridCRS UTM values this is returned in meters. For gridCRS Lat/Long values this is returned in degrees/pixel. Minimum values for DGCS are 0.5m and .0000045 degrees; these equate to full resolution data for WV01
Output:format*	image/geotiff image/jp2 image/x-mrsid-image image/geopdf	The format in which the coverage response will be returned. Only one value can be supplied.
connectId*	Character String	User's unique identifier supplied by DigitalGlobe. This is required to access the DGCS.

* mandatory parameter

GetCoverage Response

The Response to a valid GetCoverage request is a multi-part MIME message, consisting of an XML file containing coverage metadata, and the coverage file in the requested format. This coverage file will consist of the portion of the coverage contained within the bounding box supplied in the GetCoverage request. Any portion of the bounding box for which no imagery is available in the requested coverage will contain black filled pixels.

The following example requests a TIFF product of the same coverage described in the above example:

```
https://services.digitalglobe.com/deliveryservice/wcsaccess?service=WCS&version=1.1.1&request=
GetCoverage&format=image/tiff&identifier=1be2be7285b20bf2126efd7f6448d844&boundingBox=3
4.959843,-118.013832,36.019478,-
117.814011,urn:ogc:def:crs:EPSG:4326&GridBaseCRS=urn:ogc:def:crs:EPSG:4326&GridOffsets=
0.00010,0.00010&connectid=<ConnectID>
```

Replace <ConnectID> with the ConnectID provided by DigitalGlobe. Parameters are not required to be in the same order as shown above.

4.5 Basic Service Elements

This section specifies aspects of WCS behavior that are independent of particular operations or are common to several operations.

4.5.1 HTTP REQUEST

HTTP functions as a request-response protocol in the client-server computing model. In HTTP, a web browser, for example, acts as a client, while an application running on a computer hosting a web site functions as a server. The client submits an HTTP request message to the server. The server, which stores content, or provides resources, such as HTML files and images, or generates such content as required, or performs other functions on behalf of the client, returns a response message to the client. A response contains completion status information about the request and may contain any content requested by the client in its message body.

An HTTP URL locates the Online Resource of each operation supported by a service instance. The URL may be different for each operation, or the same, at the discretion of the service provider.

HTTP supports two request methods: GET and POST. One or both of these methods may be defined for a particular web service and offered by a service instance. The use of the Online Resource URL differs in each case.

HTTP GET

An Online Resource URL intended for HTTP GET requests, is, in fact, only a URL prefix to which additional parameters must be appended in order to construct a valid Operation request. A URL prefix is defined as an opaque string including the protocol, hostname, optional port number, path, a question mark '?', and, optionally, one or more server-specific parameters ending in an ampersand '&'. The prefix uniquely identifies the particular service instance.

A client can append the necessary request parameters as name/value pairs in the form "name=value&". The resulting URL must be valid according to the HTTP Common Gateway Interface (CGI) standard, which mandates the presence of '?' before the sequence of query parameters and the '&' between each parameter. Table 4.3 summarizes the components of an operation request URL.

The URL prefix must end in either a **'?'** (in the absence of additional server-specific parameters) or a '**&'.** In practice, however, Clients **should** be prepared to add a necessary trailing **'?'** or '**&'** before appending the operation parameters as per DG-WCS specification in order to construct a valid request URL. Please refer to Table 4.4 for a list of reserved characters as per HTTP rules.

TABLE 4.3 A GENERAL GET REQUEST

URL COMPONENT	DESCRIPTION
http://host[:port]/path?{name[=value]&}	URL prefix of service operation. [ ] denotes 0 or 1 occurrence of an optional part; {} denotes 0 or more occurrences. The prefix is entirely at the discretion of the service provider.
name=value&	One or more standard request parameter name/value pairs defined by a Web Coverage Service. The actual list of required and optional parameters mandated for each operation is described in the Table 6.2.

TABLE 4.4 RESERVED CHARACTERS IN HTTP GET QUERY

CHARACTER	RESERVED USAGE
?	Indicates the start of query string.
&	Use between parameters in query string.
=	Use between name and value of parameter.

CHARACTER	RESERVED USAGE
/	Use between MIME type and subtype in format parameter value.
:	Use between Namespace and Identifier in SRS parameter value.
,	Use between individual values in list-oriented parameters.

HTTP POST

An Online Resource URL intended for HTTP POST requests is a complete and valid URL to which clients transmit encoded requests in the *body* of the POST document. DGCS-WCS do not require additional parameters to be appended to the URL in order to construct a valid target for the Operation request.

The following figure shows a sample of a HTTP Post request:

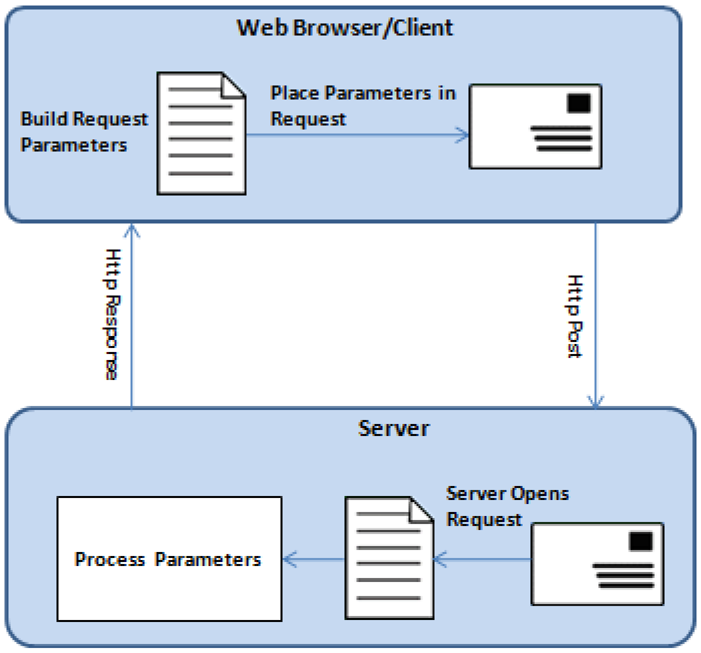

FIGURE 4.3 SAMPLE HTTP POST REQUEST

Advantages of HTTP Post Instead of HTTP Get:

- The parameter’s name and value are visible to the user and to anyone who is looking at the URL in the browser.
- GET requests are passed as the URL string and are therefore limited by the URL length limit specified by the browser.
- HTTP Post method can upload files to the server.

HTTPS

DigitalGlobe offers WCS using HTTPS. HTTPS is HTTP over a secure communication channel that allows encrypted information to be transferred between machines over the Internet.

The use of HTTPS does not affect the description of the requests and responses described in this document, but may require additional actions to be taken on both the client and the service in order to initiate secure communication.

HTTP response

Upon receiving a valid HTTP request, the service sends a response corresponding exactly to the request as detailed based on parameters for specific operations.

Response objects will be accompanied by other HTTP entity headers as appropriate and to the extent possible. In particular, the Expires and Last-Modified headers provide important information for caching; Content-Length may be used by clients to know when data transmission is complete and to efficiently allocate space for results, and Content-Encoding or Content-Transfer-Encoding may be necessary for proper interpretation of the results. If the request is invalid, the service will issue a Service Exception which is explained in detail in Section 4.9 Service Exceptions.

4.6 Output Formats

The optional outputFormat attribute specifies the format of the response to a web service request. The default value is text/xml; subtype=gml/3.1.1 indicating that a valid GML3 document, that validates against a valid GML3 application schema, must be generated. For backward compatibility, the values GML2 or text/xml; subtype=gml/2.1.2 may be specified indicating that a valid GML2 document that validates against a valid GML2 application schema, must be generated. Refer to Table 4.5Table 4.5 for a list of possible values for the outputFormat attribute.

TABLE 4.5 VALUES FOR OUTPUTFORMAT ATTRIBUTE

OUTPUTFORMAT VALUE	DESCRIPTION
GML2	This value is kept for backward compatability and indicates that an XML instance document must be generated that validates against a GML2 application schema.
text/xml; subtype=gml/2.1.2	Same as GML2.
text/xml; subtype=gml/3.1.1	This value indicates that an XML instance document must be generated that validates against a GML3 application schema. This is the default values of the outputFormat attribute if the attribute is not specified in the GetCoverage request.

4.7 Request Parameters

As per the specification standards of WCS, a client application has to form the HTTPS-based URL dynamically, based on requirement or operation it has to perform. The following are the list of important parameters that are part of the WCS URL.

4.7.1 BASE URL

For every request to DigitalGlobe WCS server, the client needs to append parameters to the base URL. DigitalGlobe provides the base URL for WCS, which is used as the common base URL as described below.

Base URL:

https://services.digitalglobe.com/deliveryservice/wcsaccess?&connectid=xxxxxxxxxxxxxxxxxxxxxxxxxx

Username and Password are required only for some accounts. All others require a Connect ID.

CONNECTID

ConnectID is a parameter name which needs to be appended with the appropriate value, along with the base URL mentioned above. The value for these parameters is a unique 32-digit alphanumeric value. It is a mandatory parameter which should be part of every request that the client makes with the server. Please contact DigitalGlobe to get your unique ConnectID.

ConnectID format: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
where **x** is an alpha numeric code

SERVICE

This parameter defines the type of service the client is requesting. As mentioned above, DigitalGlobe provides different services like WMS, WCS, WMTS and WCS. The client needs to provide appropriate values based on the service to be requested. The value for this parameter always is "**WCS**" for WCS clients.

Example: service=WCS

VERSION

The Version parameter specifies the protocol version number. The version number indicates the specification defined by OGC. The format of version number contains three positive integers, separated by decimal points, in the form "**x.y.z"**. The numbers "y" and "z" will never exceed 99. Each Feature Service provided by DigitalGlobe is numbered independently as per respective OGC specification standards. The latest version of DigitalGlobe WCS implemented for the OGC specification is 1.1.0.

The version number appears in following two places:

- In response XML of GetCapabilities request describing WCS service
- In the parameter list of client requests to the WCS service

In response to a GetCapabilities request containing a version number, a WCS server either responds with an output that conforms to that version of the specification, or negotiates a mutually agreeable version if the requested version is not implemented on the server. If no version number is specified in the request, the server responds with the highest version it understands and labels the response accordingly. Refer to OGC specification for WCS on negotiation rules.

Example: version=1.1.0 (recommended until DigitalGlobe implements new version per OGC specification)

REQUEST

The REQUEST parameter indicates the service operation that is being invoked. The value shall be the name of one of the operations offered by DigitalGlobe Web Coverage Service. Refer to Section 4.1 on page 16 for different operations supported by DigitalGlobe WCS.

Example: request=GetCapabilities

FORMAT

The FORMAT parameter specifies the output format of the response to a request operation. Formats are expressed in both Capabilities XML and in operation requests using MIME types. Each Operation has a distinct list of supported formats. Some formats may be offered by several operations, and are then duplicated as needed in each list. If a request contains a Format not offered by the WCS server, the server throws a Service Exception (with code "InvalidFormat"). Refer to Table 4.3 for different format types supported by DigitalGlobe WCS for different response types.

Example: format=image/jpeg

EXCEPTIONS

The EXCEPTIONS parameter indicates the format in which the Client wants to be notified of Service Exceptions. The only value of the EXCEPTIONS parameter that is defined for WCS Web Service is "application/vnd.ogc.se_xml", which means "Service Exception XML". Individual error messages appear as <ServiceException> elements within the <ServiceExceptionReport> in Service Exception XML. Refer to Section 4.9 on page 28 for more details on Service Exceptions.

BBOX (Bounding Box)

The Bounding Box (BBOX) is a set of four comma-separated decimal, scientific notation or integer values that represents the georeferenced bounding parameters of Area Of Interest (AOI). These values specify the minimum X, minimum Y, maximum X, and maximum Y ranges, in that order, expressed in units of the Spatial Reference System (SRS) of the request, such that a rectangular area is defined in those units.

The four bounding box values indicate the outside edges of a rectangle, as in Figure 4.4. In the figure, minimum X is the left edge, maximum X the right, minimum Y the bottom, and maximum Y the top. The relation of the Bounding

Box to the image pixel matrix is shown in the figure by showing that the bounding box goes around the “outside” of the pixels of the image rather than through the centers of the border pixels. In this context, individual pixels have an area.

Rules to follow while defining BBOX:

- A Bounding Box should not have zero area.
- Minimum X should be less than or equal to the Maximum X and Minimum Y should be less than or equal to the Maximum Y.

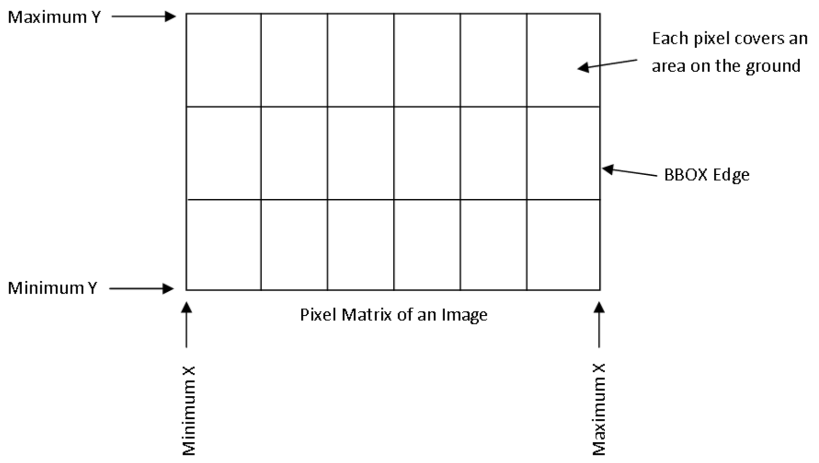

FIGURE 4.4 PICTORIAL REPRESENTATION OF BOUNDING

Example: BBOX=-88.1035780267704,40.4568762655891,-88.0928025063267,40.4638383078358

Request Parameter Rules

While forming a request URL, client applications should follow these rules:

- Parameter names are not case-sensitive, but parameter values are case-sensitive.
- Parameter names are typically shown in uppercase for typographical clarity, not as a requirement.
- Parameters may be specified in any order.
- When request parameters are duplicated with conflicting values, the response from the server may be undefined.
- Parameters consisting of lists (for example, BBOX, LAYERS and STYLES in WCS requests) shall use the comma (“,”) as the separator between items in the list. Additional white space shall not be used to delimit list items.
- Two successive commas indicate an empty item, as does a leading comma or a trailing comma. An empty list (“”) shall be interpreted either as a list containing no items or as a list containing a single empty item, depending on context.

4.8 Integration Procedure

A WCS client application is a program that communicates with the DGCS WCS server using the three functions: GetCapabilities, DescribeCoverage, and GetCoverage. More specifically, in a typical WCS client-server interaction, the following steps can be followed:

STEP-1

The client must first request **GetCapabilities** from the WCS server in order to determine what the WCS server can do and what Coverage the WCS server can provide.

Example Request:

```
https://services.digitalglobe.com/deliveryservice/wcsaccess?service=WCS&request=GetCapabilitie
s&version=1.1.1&connectid=<ConnectID>
```

Username and Password parameters may not be applicable depending on your account type. Replace <ConnectID> with the ConnectID provided by DigitalGlobe. Parameters are not required to be in the same order as shown above.

Understanding URL

The URL shown above contains Base URL and few parameters as explained in Section 4.7 on page 22. The key parameter for this request is "**request=GetCapabilities**" which fetches the capabilities of Web Coverage Service and response in the form of XML data.

TABLE 4.6 UNDERSTANDING URL PARAMETERS FOR GETCAPABILTIES REQUEST

PARAMETER	PARAMETER VALUE	DESCRIPTION
SERVICE*	WCS	Explained under SERVICE in Section 4.7 on page 22.
REQUEST*	GetCapabilities	The value for this parameter should always be "**GetCapabilities**" for step-1.
VERSION	1.1.0	Refer to VERSION in Section 4.7 on page 22.
CONNECTID*	**<CONNECTID>** provided by DigitalGlobe	The value for this parameter is an unique 32-digit alphanumeric value provided by DigitalGlobe (Explained under CONNECTID in Section 4.7 on page 22). A valid CONNECTID is mandatory for every request.

* mandatory parameter

Response

In response to a GetCapabilities request, the DGCS WCS server produces an XML document containing the WCS server's service metadata, describing all the operations it supports, and providing information about the available Coverage. The client application has to parse the XML capabilities document to retrieve the necessary information used to request a Coverage. The Document Object Model (DOM) is a widely-used and efficient XML parser, which can be utilized to parse the XML document and retrieve the information. The DOM represents an XML document as a tree of nodes that can be easily traversed and edited with its standard interfaces.

The response XML to the GetCapabilities request contains the following details:

- WCS Service details, such as Name, Title, URL
- Contact Information, such as Person, Organization, Address, Telephone, Fax and Email
- WCS Capabilities, such as GetCapabilties, GetCoverage and DescribeCoverage along with respective formats and URLs.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wcs:Capabilities version="1.1.1" xmlns:wcs="http://www.opengis.net/wcs/1.1.1"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:ows="http://www.opengis.net/ows/1.1" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://services.digitalglobe.com/deliveryservice/schemas/wcs/1.
1.1/wcsGetCapabilities.xsd" updateSequence="103">
- <ows:ServiceIdentification>
  <ows:Title>DigitalGlobe Web Coverage Service</ows:Title>
  <ows:Abstract />
- <ows:Keywords>
  <ows:Keyword>WCS</ows:Keyword>
  </ows:Keywords>
  <ows:ServiceType>WCS</ows:ServiceType>
  <ows:ServiceTypeVersion>1.1.0</ows:ServiceTypeVersion>
  <ows:ServiceTypeVersion>1.1.1</ows:ServiceTypeVersion>
  <ows:Fees>NONE</ows:Fees>
  <ows:AccessConstraints>NONE</ows:AccessConstraints>
  </ows:ServiceIdentification>
- <ows:ServiceProvider>
  <ows:ProviderName>DigitalGlobe Inc</ows:ProviderName>
  <ows:ProviderSite xlink:href="http://www.digitalglobe.com" />
- <ows:ServiceContact>
<ows:IndividualName>Customer Service Department</ows:IndividualName>
  <ows:PositionName>Customer Service Department</ows:PositionName>
- <ows:ContactInfo>
- <ows:Phone>
  <ows:Voice>800.496.1225</ows:Voice>
  <ows:Facsimile>303.684.4562</ows:Facsimile>
  </ows:Phone>
- <ows:Address>
<ows:ElectronicMailAddress>info@digitalglobe.com</ows:ElectronicMailAddress>
  </ows:Address>
  <ows:OnlineResource xlink:href="http://www.digitalglobe.com" />
  </ows:ContactInfo>
  </ows:ServiceContact>
</ows:ServiceProvider>
- <ows:OperationsMetadata>
- <ows:Operation name="GetCapabilities">
- <ows:DCP>
- <ows:HTTP>
 <ows:Get
xlink:href="https://services.digitalglobe.com/deliveryservice/wcsaccess?CONNECTID=<C
onnectID>&" />
</ows:HTTP>
  </ows:DCP>
- <ows:DCP>
- <ows:HTTP>
 <ows:Post
xlink:href="https://services.digitalglobe.com/deliveryservice/wcsaccess?CONNECTID=<C
onnectID>&" />
  </ows:HTTP>
  </ows:DCP>
      </ows:Operation>
- <ows:Operation name="DescribeCoverage">
- <ows:DCP>
```

Continued…

```
 Continued…
- <ows:HTTP>
 <ows:Get
xlink:href="https://services.digitalglobe.com/deliveryservice/wcsaccess?CONNECTID=<C
onnectID>&" />
  </ows:HTTP>
  </ows:DCP>
- <ows:DCP>
- <ows:HTTP>
<ows:Post
xlink:href="https://services.digitalglobe.com/deliveryservice/wcsaccess?CONNECTID=<C
onnectID>&" />
  </ows:HTTP>
  </ows:DCP>
- <ows:Parameter name="identifiers">
- <ows:AllowedValues>
  <ows:Value>8cdc188a1bbcfbd63d3e3b5dd33fb670</ows:Value>
  <ows:Value>ebf263376c6760208f3b1ae507cba96a</ows:Value>
  <ows:Value>a2d2378e875f6795f91e918260451919</ows:Value>
  <ows:Value>02859051eed8bae0074f69cb80258297</ows:Value>
  <ows:Value>367b09f68f5f2666a5dfe520bc0572f9</ows:Value>
  <ows:Value>3983a2b14fde7536219fafa9a3506eda</ows:Value>
  <ows:Value>20dbcfc2a96408c9aa5ecaf0171c6828</ows:Value>
  <ows:Value>9d4a061b0f6cd4108345cc051d22a840</ows:Value>
  <ows:Value>9d771d78abdaba43bf0fc80e3f37c1f6</ows:Value>
  <ows:Value>f65ab16fcad242fafbcca8ea791bb7ba</ows:Value>
  <ows:Value>9a7363215ded5b331261dc6d388d66c6</ows:Value>
              .
              .
              .
              .
  <ows:Value>03785bff63ec79a681d825ec6477a9ef</ows:Value>
  <ows:Value>8215565cf0a9461c9e735833cfe84da7</ows:Value>
  <ows:Value>235d46b6f9731dea74480a740c1b1ea6</ows:Value>
  <ows:Value>d1acbf77b1c14ce4163cc7d7faed3399</ows:Value>
  <ows:Value>4ca1289e3dcd7fe5ed03f2896a2ed75e</ows:Value>
  <ows:Value>1ef44cf4f707c1d6657d839b84a790e8</ows:Value>
  <ows:Value>d8a158665ac1f20a615b6733642c7d6f</ows:Value>
  <ows:Value>b7a2441fcb59dc9f834ca8d04042a1ae</
```

STEP-2

The client can request DescribeCoverage with the WCS server's capabilities information in order to get the Coverage information. Once a user has obtained a description of the supported Coverage, the GetCoverage request is used to access the metadata associated with one or more identifiers.

Example Request:

https://services.digitalglobe.com/deliveryservice/wcsaccess?service=WCS&version=1.1.1&request=DescribeCoverage&identifiers=32caf848a43a491580707d70f5e5eb71&connectid=<ConnectID>

Replace <ConnectID> with the ConnectID provided by DigitalGlobe. Parameters are not required to be in the same order as shown above.

Understanding URL

The Parameters available in the DescribeCoverage request are shown in Table 4.7. The client provides the following information in a Keyword Value Pair (KVP) format, where the "name" field is the key, and the "value" field is the value; the data is supplied in the format "key=value"; for example, "service=WCS".

TABLE 4.7 UNDERSTANDING URL PARAMETERS FOR DESCRIBECOVERAGE REQUEST

NAME	VALUE	DESCRIPTION
Service*	WCS	Web Coverage Service
Version*	1.1.0	Request version
Request*	DescribeCoverage	Request name
Identifiers*	List of coverages to be described; Example: 32caf848a43a491580707d70f5e5eb71	The Coverage(s) to be described; each product in the layers are coverage and can be requested. Multiple identifiers are comma-separated; at least one valid identifier must be included. The coverage identifiers can be obtained from the GetCapabilities response.
connectId*	Globally Unique Identifier <CONNECTID> supplied to the service user by DigitalGlobe.	This CONNECTID is required to access the DigitalGlobe Cloud Services.

Response

In response to a valid DescribeCoverage request, the WCS returns a GML document containing zero or more Coverages that match the request criteria. Each Coverage is described by a list of properties contained in a GML document.

STEP-3

To get the description of the supported **GetCoverage**, the user can use the following URL:

https://services.digitalglobe.com/deliveryservice/wcsaccess?SERVICE=WCS&REQUEST=GetCoverage&version=1.1.1&CONNECTID=<ConnectID>&identifier=ad9afcf5092274d8df13c894033ca9d1&FORMAT=image/jpeg&BoundingBox=32.56664276123047,36.03737258911133,32.68387985229492,36.175987243652344,urn:ogc:def:crs:EPSG::4326&GridBaseCRS=urn:ogc:def:crs:EPSG::4326&GridCS=urn:ogc:def:cs:OGC:0.0:Grid2dSquareCS&GridType=urn:ogc:def:method:WCS:1.1:2dGridIn2dCrs&GridOrigin=32.56664501123057,36.175989493652445&GridOffsets=0.0000045,0.0000045

Replace <ConnectID> with the ConnectID provided by DigitalGlobe. Parameters are not required to be in the same order as shown above.

Understanding URL

The client provides the following information in a Keyword Value Pair (KVP) format, where the “name” field is the key, and the “value” field is the value; the data is supplied in the format “key=value”; for example, “service=WCS”. Refer to Table 4.10 for the URL Parameters for the GetCoverage request.

Response

In response to a GetCoverage request, a GML document is returned that defines the elements of the requested Coverage. The elements of each supported feature type are defined in Table 4.9, Table 4.10, and Table 4.11. In the cases where returned data is dependent on the value of elements within the Coverage, the dependency is noted in the Description column.

4.9 Service Exceptions

In the event that a Web Coverage Service encounters an error while processing a request or receives an unrecognized request, it will generate an XML document indicating that an error has occurred.

An **<ExceptionReport>** element will contain one or more WCS-processing exceptions specified using the <**Exception**> element. The mandatory version attribute is used to indicate the version of the service exception report schema. For this version of the specification, this value is fixed at 1.1.0.

Individual exception messages are contained within the **<ExceptionText>** element.

The following is an example of an exception report. This exception indicates that no coverage could be found for the code 334034.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <ows:ExceptionReport version="1.1.0"
xsi:schemaLocation="http://www.opengis.net/ows/1.1
http://services.digitalglobe.com/deliveryservice/schemas/ows/1.1.0/owsAll.xsd"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
- <ows:Exception exceptionCode="NoApplicableCode">
  <ows:ExceptionText>DescribeCoverage: Could not find coverage: 334034 null
Translator error DescribeCoverage: Could not find coverage:
334034</ows:ExceptionText>
  </ows:Exception>
  </ows:ExceptionReport>
```

4.10 WCS Layers

TABLE 4.8 WCS LAYERS

OGC LAYER(S)	DESCRIPTION
Identifier <n>	Each feature available to an account is returned in WCS as an identifier with a unique number and basic metadata.

4.11 API Reference

This section provides a list of all possible request parameters for every WCS operation as well as detailed information about corresponding response.

The client should provide the respective information in a Keyword Value Pair (KVP) format for every WCS request, where the “name” field is the key, and the “value” field is the value; the data is supplied in the format “key=value”; for example, “service=WCS”.

In the M/O column, “M” indicates a mandatory parameter, “O” an optional parameter.

GetCapabilities

The following table shows all possible request parameters for GetCapabilities operation of WCS server.

TABLE 4.9 WCS GETCAPABILITIES REQUEST PARAMETERS

PARAMETER	PARAMETER VALUE	DESCRIPTION
SERVICE*	WCS	Explained in SERVICE of Section 4.7.
REQUEST*	GetCapabilities	The value for this parameter should always be “**GetCapabilities**” for step-1.
VERSION	1.1.0	Refer VERSION in Section 4.7.
CONNECTID*	<CONNECTID> provided by Digital Globe	Value for this parameter is an unique 32-digit alphanumeric value given by DigitalGlobe (Explained in CONNECTID of Section 4.7). Valid CONNECTID is mandatory for every request.

* mandatory parameter

GetCoverage

Refer to Table 4.10 for the URL Parameters for the GetCoverage request.

TABLE 4.10 GETCOVERAGE REQUEST PARAMETERS

NAME	VALUE	DESCRIPTION
service*	WCS	Web Coverage Service.
version*	1.1.1	Request version.
request*	GetCoverage	Request name.
identifier*	A coverage identifier; for example: 55b795c5bd91a1abf7ff0e70ac52127e	The Coverage to be described.
boundingBox*	Example: 35.6,-117.7,35.7,-117.6	Bounding Box of the portion of the coverage to be returned (lat-lon).
gridCRS	EPSG:4326 (each valid EPSG value for UTM)	The CRS in which the coverage response will be returned; supported CRSs are WGS84 Lat/Long and WGS84 UTM. If not included, the native CRS of the projection will be returned; for DGCS this will be WGS84 Lat/Long.
GridOffsets	Example: GridOffsets=0.8,08, would return 80 cm/pixel image	Specifies the pixel size to be returned in the X and Y dimensions. For gridCRS UTM values this is returned in meters. For gridCRS Lat/Long values this is returned in degrees/pixel. Minimum values for DGCS are 0.5m and .0000045 degrees; these equate to full resolution data for WV01.
GridBaseCRS	4326	If the value is missing, the native coordinate system will be used instead, i.e. 4326 or UTM.
format*	image/geotiff image/jp2 image/x-mrsid-image/tiff image/jpeg	The format in which the coverage response will be returned; only one value can be supplied.
connectId*	Character String	User's unique identifier supplied by DigitalGlobe; required to access the DGCS.

* mandatory parameter

DescribeCoverage

TABLE 4.11 DESCRIBECOVERAGE REQUEST PARAMETERS

PARAMETER	DATA TYPE	RANGE/EXAMPLE	DESCRIPTION
service*	String	WCS	Web Coverage Service.
version*		1.1.1	Request version.
request*	String	GetCoverage	Request name.
identifier*	String	A coverage identifier; for example: 83de5c8e080cb6842e2ee166 a4a0ad0d	The ID of the coverage being described; each coverage ID is also a FeatureId for the corresponding feature.
boundingBox*	WGS84Bounding Box	Example: LowerCorner>322658.586857 5001 3319942.861114297 UpperCorner>335865.973088	Lower left/upper right corners of coverage bounding box, in WGS:84 coordinates. For coverages this will be the bounding box of a Strip product, or a GeoCell product (lat-lon).

PARAMETER	DATA TYPE	RANGE/EXAMPLE	DESCRIPTION
		2284 3333143.2153002457	Refer to BBOX in Section 4.7.1 on page 22 for more information.
gridCRS	GridCrsType	urn:ogc:def:cs:OGC:0.0:Grid2dSquareCS (each valid EPSG value for UTM)	The CRS in which the coverage response will be returned; supported CRSs are WGS84 Lat/Long and WGS84 UTM. If not included, the native CRS of the projection will be returned; for DGCS this will be WGS84 Lat/Long. Please refer to the following link for the schema definition: http://schemas.opengis.net/wcs/1.1.0/wcsGridCRS.xsd
GridOffsets		Example: 0.5 0 0 -0.5 GridOffsets=0.5,05, would return 50 cm/pixel image	Specifies the pixel size to be returned in the X and Y dimensions. For gridCRS UTM values this is returned in meters. For gridCRS Lat/Long values this is returned in degrees/pixel. Minimum values for DGCS are 0.5m and .0000045 degrees; these equate to full resolution data for WV01.
GridBaseCRS		urn:ogc:def:crs:EPSG::32636	If the value is missing, the native coordinate system will be used instead, i.e. 4326 or UTM.
format*	String	image/tiff image/jp2 image/x-mrsid-image image/geopdf	The format in which the coverage response will be returned; only one value can be supplied. GeoTIFF and JP2 are supported for all coverages.
connectId*	Character String		User’s unique identifier supplied by DigitalGlobe; required to access the DGCS.
Field:Interpolation Methods*	String	Nearest Cubic linear	Comma-separated list of available interpolations for this field.
Axis identifier*	Byte	Available Keys: 1, 2, 3	The Axis data structure (used in conjunction with the Interpolation method) allows the user to select a specific range of values recorded in the coverage. An axis basically represents a single “control variable”. DigitalGlobe supports vector-valued and color (multiband) imagery. Each axis has a number of named keys. For examole, 1 key (named “1”) for panchromatic data and three bands for color (named ”1”, “2” and “3”). These are defined to be of type “Byte”, therefore the expected value at each point in the grid must be between 0 and 255. These key names and their axis identifier (“BAND”) must be used exactly in the RangeSubset request parameter.
supportedCRS*	String	Examples: urn:ogc:def:crs:EPSG::4326, urn:ogc:def:crs:EPSG::32639	Comma-separated list of reference systems in which the coverage can be supplied; WGS-84/Geographic and WGS-84/UTM are supported for all coverages.

* mandatory parameter

Glossary

AOI

Area of Interest. The area on the Earth that you want to view.

Bilinear Interpolation

Bilinear interpolation uses the value of the four nearest cell centers to determine the value on the output raster. The new value is a weighted average of these four values, adjusted to account for their distance from the center of the output cell. The result is a smoother-looking surface than provided by “nearest neighbor”.

Bicubic Interpolation

Bicubic interpolation combines data points on a two-dimensional grid. This method outputs the smoothest surface of all interpolation methods.

GeoTIFF format

A GeoTIFF file is a TIFF file that is embedded with geographic data tags.

GML

Geography Markup Language. GML is XML code used to express geographical features.

Nearest Neighbor Interpolation

Uses the value of the closest point and disregards all other values, yielding a piecewise-constant interpolant.

OGC

Open GIS Consortium. An international standards organization comprised of commercial, governmental, nonprofit and research organizations. They support geospatial content development as well as data processing and sharing.

OWS

OGC Web Service Common.

Partition

The unit of measure based on the tile zoom level grid for tar file creation for imagery tiles. All tiles and associated metadata for a partition will be tar-compressed into a single file.

UTM

Universal Transverse Mercator Geographic Coordinate System. UTM utilizes a two-dimensional Cartesian system to specify locations on the Earth’s surface.

WCS

Web Coverage Service.

WFS

Web Feature Service.

WMS

Web Map Service.

WMTS

Web Map Tile Service.

Index