Testbed-12 WMS/WMTS Enhanced Engineering Report

Table of Contents

Publication Date: 2017-06-14

Approval Date: 2017-03-23

Posted Date: 2016-11-14

Reference number of this document: OGC 16-042r1

Reference URL for this document: http://www.opengis.net/doc/PER/t12-NA005

Category: Public Engineering Report

Editor: Lingjun Kang, Liping Di, Eugene Yu

Title: Testbed-12 WMS/WMTS Enhanced Engineering Report

Testbed-12 WMS/WMTS Enhanced Engineering Report (16-042r1)

COPYRIGHT

WARNING

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER’S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by

destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Abstract

This Engineering Report (ER) describes requirements, challenges and solutions regarding improving multidimensional Earth Observation (EO) data access, discovery and visualization through Web Map Service (WMS), Web Map Tile Service (WMTS), and corresponding extensions. The ER will highlight solutions and recommendations of following main topics.

1) WMTS enhancements for time-varying layer access/discovery

In this topic, the ER will cover the following proposed work items enhancing time-varying layer access and discovery. The ER will also discuss the discussions and proposals of extending current WMTS/WMS specifications for the enhancements.

a. New time-query semantic/encoding for time-varying layer query;
b. WMTS Capabilities extension for time-varying layer discovery; and
c. Improvements on avoiding empty tile request and better server-client communication (e.g., empty tiles).

2) WMS enhancements for NetCDF

In this topic, the ER will cover the following proposed work items to better support NetCDF-CF data visualization and exploration.

a. Extensions on WMS GetMap request parameters to enrich map render options (with non-standard ncWMS interfaces); and
b. Extensions on WMS GetCapabilities request parameter to improve discovery of WMS GetCapabilities document (with large number of layers being advertised).

3) WMTS enhancements for multidimensional domain discovery

In this topic, the ER will cover the following proposed work items to improve inspecting layer domain and discovery of the relationship between multidimensional layers.

a. Extensions on WMS GetMap request parameters to enrich map render options (with non-standard ncWMS interfaces); and
b. Extensions on WMS GetCapabilities request parameter to improve discovering WMS GetCapabilities document (with large number of layers being advertised).

Business Value

This engineering report reviews the time-varying layer/NetCDF supports and limitations in current WMS/WMTS specifications. The engineering report also covers proposed enhancements and extensions on current WMS/WMTS specifications. The contents of this report will therefore improve the geospatial map service interoperability through:

- Improved understandings of time-varying layer access supports in current standards and the possible space for extensions to achieve better time-varying access/discovery;
- Improved time-varying layer access with proposed advanced time-query semantics;
- Efficient multidimensional layer (including time layer) discovery through extensions on Capabilities document; and
- Improved NetCDF access capabilities in extended WMS.

What does this ER mean for the Working Group and OGC in general

The importances of this ER to the WMS 1.4 Standards Working Group (SWG) are that the solutions and proposals present in this ER will serve as the references to extend existing WMS/WMTS standards for enhanced time-varying layer access/discovery and better NetCDF access support.

How does this ER relate to the work of the Working Group

The WMS 1.4 SWG is dedicated in improving both web map service standards and the EO data access through the internet. The proposed extensions of WMS/WMTS in this ER will serve as the reference for WMS 1.4 SWG's future endeavor in:

- Improving the efficient access to time-varying layers;
- Improving the web map rendering options and GetCapabilities filtering; and
- Improving the map layer domain inspection/layer relationship discovery.

Keywords

ogcdocs, testbed-12, engineering report, WMS, WMTS, NetCDF, time-varying layer, multidimensional layers, etc.

Chapter 1. Introduction

This engineering report describes the requirements, challenges, solutions and recommendations regarding the access, discovery, and visualization of time-varying/multidimensional Earth Observations through WMS and WMTS with the proposed enhancements.

1.1. Scope

This OGC® Engineering Report reviews and evaluates the map service interoperability (i.e. time-varying layer access and NetCDF support) in the following specifications:

[WMS 1.3.0] *OpenGIS Web Map Service (WMS) Implementation Specification*, version 1.3.0, OGC document 06-042, 2006-03-15, http://portal.opengeospatial.org/files/?artifact_id=14416

[WMTS 1.0.0] *OpenGIS Web Map Tile Service Implementation Standard*, version 1.0.0, OGC document 07-057r7, 2010-04-06, http://portal.opengeospatial.org/files/?artifact_id=35326

[WFS 2.0.0] *OGC® Web Feature Service 2.0 Interface Standard*, version 2.0.2, OGC document 09-025r2, 20104-07-10, http://docs.opengeospatial.org/is/09-025r2/09-025r2.html

[ISO 8601] *Data elements and interchange formats — Information interchange — Representation of dates and times*, https://en.wikipedia.org/wiki/ISO_8601

1.2. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors.

Table 1. Contacts

Name	Organization
Lingjun Kang (editor)	CSISS/GMU
Liping Di (editor)	CSISS/GMU
Eugene Yu (editor)	CSISS/GMU
Maso Pau Joan	Universitat Autònoma de Barcelona (CREAF)
Satish Sankaran	Esri
Matthew Cechini	Columbus Technologies & Services, Inc.
Andrea Aime	GeoSolutions S.A.S.
Craig Bruce	CubeWrex for Polar View
Stephan Meißl	EOX IT Services GmbH
Perry Peterson	PYXIS
Lei Hu	CSISS/GMU

1.3. Future work

- GML geometry representation for the spatial footprint of available WMTS tiles. Currently the spatial domain is represented as a bounding box, a polygon or a multi-polygon could be used to better represent scattered datasets.
- As an alternative for clients unable to perform polygon intersections, a direct binary representation with regard to the availability of WMTS tile is expected. The direct representation of WMTS tile availability will free the client from complex math calculation in deriving the column/row number of available WMTS tiles.

1.4. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

Chapter 2. References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

- OGC: OGC 06-121r9, OGC® Web Services Common Standard (2010)
- OGC: OGC 06-042, OpenGIS Web Map Service (WMS) Implementation Specification, version 1.3.0 (2006)
- OGC: OGC 07-057r7, OpenGIS Web Map Tile Service Implementation Standard 1.0.0 (2010)
- OGC: OGC 09-025r2, OGC® Web Feature Service 2.0 Interface Standard, version 2.0.2 (2014)
- ISO: ISO 8601, Data elements and interchange formats — Information interchange — Representation of dates and times (2004)
- The University of Reading: ncWMS User Guide, https://reading-escience-centre.gitbooks.io/ncwms-user-guide/content/ (2010)

Chapter 3. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r9], OpenGIS Web Map Service (WMS) Implementation Specification, version 1.3.0 [OGC 06-042], and in OpenGIS Web Map Tile Service Implementation Standard 1.0.0 [OGC 07-057r7] shall apply.

Chapter 4. Conventions

4.1. Abbreviated terms

Most of the abbreviated terms listed in subclause 5.1 of the OWS Common Implementation Specification [OGC 06-121r9], OpenGIS Web Map Service (WMS) Implementation Specification, version 1.3.0 [OGC 06-042], and OpenGIS Web Map Tile Service Implementation Standard 1.0.0 [OGC 07-057r7] apply to this document.

4.2. UML notation

Most diagrams that appear in this standard are presented using the Unified Modeling Language (UML) static structure diagram, as described in Subclause 5.2 of [OGC 06-121r9].

4.3. Used parts of other documents

This document uses significant parts of document [OGC 06-121r9]. To reduce the need to refer to that document, this document copies some of those parts with small modifications. To indicate those parts to readers of this document, the largely copied parts are shown with a light grey background (15%).

Chapter 5. Overview

This engineering report describes the requirements, challenges, solutions and recommendations regarding the access, discovery and visualization of time-varying/multidimensional Earth Observations through WMS and WMTS with proposed enhancements.

Chapter 6. Status Quo & New Requirements Statement

6.1. Status Quo

6.1.1. Earth observation data access through map service

With the advancements of earth observation and interoperability technologies, this era has witnessed the increasing availability of EO data with high temporal resolution and multiple dimensions. This leads to the need of enhancing current map services (i.e., WMS and WMTS) for better access and visualization of the EO data. On the other hand, the large volumes of EO data lead to the need of improving current map services' support on the standard EO data structure (i.e. NetCDF).

6.1.2. Time-varying/multi-dimension EO data access challenges

The EO data access has been greatly facilitated with the support of WMS and WMTS. However, the time-varying EO data with high temporal resolution and dimensions brings the following new challenges:

1. Support time-varying layer access through WMTS/WMS with proper and less ambiguous time-query semantic;
2. Support metadata content filter through the WMS/WMTS Capabilities document with the layers of large number;
3. Support rich map render options in WMS/WMTS;
4. Improve WMS/WMTS server-client communication to prevent empty tile requesting; and
5. Extend current WMS to better support NetCDF data access.

6.2. Requirements Statement

1. WMTS for time-varying layer
    a. Extend WMTS time-query semantic/encoding to achieve accurate/efficient time-varying layer query.
    b. Extend WMTS Capabilities to achieve better time-varying layer discovery in GetCapabilities document.
2. WMS for NetCDF
    a. Extend WMS to better support NetCDF.
3. NetCDF-CF data visualization and exploration
    a. Extend WMS with selected non-standard WMS operations from ncWMS.
    b. Extend WMS Capabilities operation to achieve efficient capabilities content filtering/retrieval.

4. Multidimensional WMTS domain discovery
    a. Extend WMTS interfaces to achieve efficient inspection on layer domain and discovery on multidimensional layer relationship.
5. WMS/WMTS client
    a. Develop enhanced client-side communication to achieve efficient WMTS request (e.g., by avoiding unnecessary empty tile requests).

Chapter 7. Solutions

7.1. Targeted Solutions

WMS/WMTS Enhancements aim at improving the following aspects related with WMS/WMTS map access/rendering.

1. WMTS enhancements for time-varying layer query/discovery.
2. WMS extensions for NetCDF.
3. Multidimensional WMTS domain discovery.

7.1.1. WMTS enhancements for time-varying layers query/discovery

WMTS time-query enhancements

A collection of time-query semantics/encoding were proposed as the enhancements on WMTS supports of temporal query. Additionally, WMTS Capabilities document was extended as a way to advertise proposed time-query semantics.

7.1.2. WMS extensions for NetCDF

Improvements on the WMS support of map rendering

A collection of extended GetMap parameters was introduced from ncWMS to improve WMS support of map rendering.

Improvements on the mechanisms of WMS metadata filtering

The extension was proposed for WMS GetCapabilities operation as a way to achieve better capabilities content filter/discovery.

Improvements on the WMS support for map querying

GetTimeSeries operation was introduced as the extension to retrieve the evolution of a phenomenon on a specific point over time or elevation, or at a fixed them and elevation, but along a path.

7.1.3. Multidimensional WMTS domain discovery

Extensions on WMTS interface to support efficient layer domain inspection/relationship discovery

The following three operations were proposed to support efficient inspecting layer domain and discovering the relationships among multidimensional layers:

1. DescribeDomains;
2. GetHistogram; and
3. GetFeature.

7.2. Recommendations

7.2.1. Recommendations for the time-varying layer access through WMTS

The following time-query semantics are proposed as the extension of WMTS to better support time-varying layer query.

Table 2. WMTS time-query semantics/encoding extension

Proposed time-query semantics	Description
at	Selects a server-defined semantic for a single time instant
asof	Selects all times from negative infinity up to and including the given time instant.
series	Selects one frame from a fixed time series identified by the first time instant of the frame.
interval	Selects all times in a given arbitrary inclusive time interval.
anim.at	Selects an animation of “at” image frames.
anim.asof	Selects an animation “as-of” image frames.
anim.series	Selects an animation of fixed-time-series image frames.
anim.interval	Selects an animation of arbitrary-interval image frames with an arbitrary time resolution.

WMTS Capabilities document extension

The proposed time-query semantics/encoding was recommended to be advertised through Capabilities document with the following extensions:

1. Extend Capabilities <Dimension> element for time-query declaration; and
2. Extend Capabilities <ResourceURL> element for time-query declaration in the RESTFul API.

7.2.2. Recommendations for extending map render options on WMS GetMap operation

The following extended ncWMS GetMap parameters were proposed and introduced to support additional map render options in WMS.

Table 3. Extended render options of WMS GetMap operation

Extended WMS parameter	Description
COLORSCALERANGE	Of the form min,max this is the scale range used for plotting the data (mapped to the COLORSCALERANGE_MIN and COLORSCALERANGE_MAX env vars) <<<<<<< HEAD
NUMCOLORBANDS	The number of discrete colors to plot the data. Must be between 2 and 250 (mapped to the NUMCOLORBANDS env variable)
ABOVEMAXCOLOR	The color to plot values which are above the maximum end of the scale range. Colors are of the form 0xRRGGBB or 0xAARRGGBB, and it also accepts “transparent” and “extend”
BELOWMINCOLOR	The color to plot values which are below the minimum end of the scale range. Colors are of d the form 0xRRGGBB or 0xAARRGGBB, and it also accepts “transparent” and “extend”
LOGSCALE	“true” or “false” - whether to plot data with a logarithmic scale
OPACITY	The percentage opacity of the final output image as a number between 0 and 100 (maps to OPACITY env var by translating it to a number between 0 and 1)
ANIMATION	“true” or “false” - whether to generate an animation. The ncWMS documentation states that TIME has to be of the form <tt class="docutils literal">starttime/endtime</tt>, but C:\work\ogc\ows12\work\LSA_NA005_ER\er_wms_wmts_OGC16-042\includes\images\redblue-default.png TIME needs to be a list of discrete times instead. Animation requires using the “image/gif” as the response format (as the only format supporting animation)

7.2.3. Recommendations for extending filtering options on WMS GetCapabilities operation

Table 4. Extended GetCapabilities parameter

Extended GetCapabilities parameter	Description
DATASET	The parameter will be set with layer/dataset name. Once the parameter is provided in the request, only content regarding the layer/dataset will be returned.

7.2.4. Recommendations for extending WMTS interface support efficient layer domain inspection/relationship discovery

Table 5. Proposed WMTS interfaces for layer domain inspection/relationship discovery

Extended operation	**Description**
DescribeDomains	The extended operation is used for compact domain inspection.
GetHistogram	The extended operation is used for inspecting domain distribution histogram.
GetFeature	The extended operation is used for detailed value combination inspection and description.

Chapter 8. WMTS for time-varying layers

8.1. Introduction

This proposal extends the existing WMTS 1.0.0 and WMS 1.3.0 time-query mechanisms to provide general time filtering for general “scattershot” source data in a tiled environment. Scattershot source data is not pre-composited or mosaicked but is available in discrete, randomly-overlapping scenes that are time-stamped as originally captured (though are normally orthorectified). General time filtering needs to support client-selected time intervals, not just time instants. Enabling fully-arbitrary time intervals fulfills the needs of general time filtering, but is impractical to use with WMTS services that are based on pre-generated tiles because of the combinatorial explosion of possible min and max times for an interval, so simpler semantics are also needed. Tiles can be used for efficient WMS-Map rendering as well.

Eight useful semantics for time-based access are defined: ‘at’ time instant (as already in existing WMS/WMTS), ‘as-of’ time instant, fixed time-series, arbitrary time interval, and animated versions of the previous four. The arbitrary-interval semantic includes the capabilities of the simpler semantics, but it is important to allow simple server instances to provide limited, optimized service.

The WMTS Capabilities document reports precisely what times are available by giving a list of time semantics and start/end time intervals and resolutions with continuousness, time-granularity, and interval-bound-inclusiveness taken care of. ISO 8601 time-value encoding is used and it is important that time values be character-for-character-precise for accessing static, pre-generated tiles through the WMTS REST interface.

8.2. Existing OWS time support

Time support exists in both the WMTS and WMS interfaces.

8.2.1. Existing WMTS time support

The latest official version of WMTS (Web Map Tile Service) is 1.0.0 [WMTS 1.0.0]. It includes a limited mechanism for dimensional queries including the dimension of time. An example XML Capabilities document might include:

```
<Dimension>
  <ows:Identifier>Time</ows:Identifier>
  <ows:UOM>ISO8601</ows:UOM>
  <ows:Title>Time</ows:Title>
  <ows:Abstract>Monthly datasets</ows:Abstract>
  <Default>null</Default>
  <Value>2007-05</Value>
  <Value>2007-06</Value>
  <Value>2007-07</Value>
</Dimension>
```

Specific values must be explicitly enumerated, which is not a scalable approach. If data were available on a daily basis for 2000 through 2015, there would need to be 5,844 enumerated **Value** tags. Alternatively, if a time-interval notation with a resolution of one day were available, only a single **Value** would need to be given. Time values are opaque in that the client does not parse them but simply parrots them back to the server. This effectively makes the values format-independent, but also means that clients cannot reinterpret them into into specialized calendar-based GUI components, for instance.

With Key-value-pair requests, the named **Dimension** identifier and selected value are included along with the standard request parameters. REST tile requests are indicated in the Capabilities document using templates with substitution variables in curly braces, such as:

```
<ResourceURL format="image/png" resourceType="tile"
template="https://website.org/wmts/mylayer/{Style}/{Time}/{TileMatrixSet}/{TileMatrix}
/{TileRow}/{TileCol}.png"/>
```

8.2.2. Existing WMS time support

The latest official version of the WMS (Web Map Service) is 1.3.0 [WMS 1.3.0]. It has more-sophisticated time support than WMTS. An example declaration section might be:

```
<Dimension name="Time" units="ISO8601" default="null">1996-01-01/2003-10-
17/P1D</Dimension>
```

Unlike with WMTS, the WMS declaration is stuffed into a single XML tag with attributes and uses time intervals with declared resolutions, making them much more scalable but also requiring viewer clients to parse the values and perform date calculations. For multiple intervals, WMS adds a syntax of using a comma separator (“,”) between allowed time ranges as opposed to the WMTS method of using multiple **Value** fields. Time-instant values are represented using ISO 8601 [ISO 8601] notation.

The WMS data model for time is clearly for pre-composited/mosaicked full-coverage source data that is neatly stacked by discrete if not regular time instants. The ‘nearestValue’ mechanism assumes that there is a *single* nearest time instant, as confirmed by its warning message in the HTTP header.

The ‘multipleValues’ attribute distinguishes whether a list of values or only a single value may be requested. It is not clear whether a time interval qualifies as a single value or as multiple values, but Example 2 in section C.3.6 implies that an interval is considered to be multiple values suitable for an animation response.

WMS appears to offer only one base time-filtering semantic, which can be called “at,” or a variant from using the ‘nearestValue’ attribute that can be called “nearest-to,” plus animated versions of the previous two using the ‘multipleValues’ attribute. Only one of “at” or “nearest-to” can be in force in a declaration and the selection of animation depends on at least one factor outside of the scope of the time declaration (the data format for the result map).

It is not clear what happens if an interval request is made when 'multipleValues' is not supported by a server. Also, an interval request requires a resolution according to Table C.2, implying a "multiple-at" semantic rather than a true interval. It is not clear if the resolution in a request is allowed to be "0" as described immediately after the table (though "0" is not a valid ISO-8601 period syntax; "PT0S" would be valid). Section C.3.2 gives the template "TIME=start_time/current," but this is not a valid syntax according to Table C.2 which requires a resolution.

In summary, neither WMS nor WMTS provides adequate support for general time filtering of general source data. Also, the exact WMS time semantics are not explained clearly enough.

8.3. Time-query semantics and encodings

8.3.1. Data environment

The source-data environment envisioned is a fully general "scattershot" one where the source data is contained in "scenes" such as satellite or aerial images that can have various resolutions, locations, coordinate systems, acquisition times, and "color" types (e.g., RGB, Color Palette, RGBI, numeric), sample sizes, and data formats and can overlap each other in arbitrary ways and can have large gaps between coverage areas. Interfaces like WMTS hide the source-data semantics, though these semantics have implications for the exact tile content retrieved.

8.3.2. Timeless queries and overlapping-content ordering

A time-capable web service should respond intelligently to queries that do not filter by time. One method would be to return content with the data with the latest acquisition time on top, though other ordering methods might override time in cases where some source data has a significantly finer resolution that other data or has a higher user-assigned priority. In principle, time-based queries could also be rendered in some order other than latest-time-on-top where source scenes overlap, if suitable for a service.

There should be an obvious way to request data without special time filtering, such as requesting the default value for a time dimension, using a timeless REST template, or having the interface define special semantics for the time parameters.

8.3.3. Time-value encoding basis

The basis for representing time values and semantics here is ISO 8601 notation, as in WMS 1.3.0. ISO provides several different encodings for the same time instant, but for an interface like WMTS REST, it is important to specify exact, character-for-character encodings. One approach would be to allow servers to specify the encoding of all of the components of a time value using a formatting template in order to accommodate prepared static data that has a pre-determined format, but this would probably not be a worthwhile exercise. Clients would need to be significantly more complicated to recognize such formatting and compose proper time values. Prescribing a time format can complicate a server in that it may need to provide a mapping between internal and external time-value representations but allows the client to be statically programmed.

The prickly issues of time zones and time granularity must also be dealt with for both source data and open interfaces. If a source time value is merely "2015-03-17", precisely what time is meant by

that? Midnight? Noon? The whole day? In what time zone or zones? To avoid ambiguity and simplify coding, all time values passed in the enhanced WMTS interface shall be in the UTC time zone and the encoding shall include the explicit ISO designation character of "**Z**" (Zulu) where applicable according to the ISO-encoding rules to make this specific semantic extremely visible. Using arbitrary time zones would be an unnecessary complication and using an implicit time zone relative to the server is very awkward because of local Daylight Savings / Summer Time rules.

8.3.4. Time instant

A time instant shall have the ISO general form "*YYYY*-*MM*-*DD***T***hh*:*mm*:*ss*.*ddddddddd***Z**", where the time components are Year, Month, Day, Hour, Minute, Second, and sub-second Decimals, using as many components, decimals, and punctuation characters as needed to encode times to a selected granularity. The "**-**", "**T**", "**:**", "**.**", and "**Z**" characters are literal. The time value shall end with the ISO "**Z**" character whenever the Hour (or finer) time component is included. The ISO comma (",") character is disallowed for the decimal point since exact coding will be needed for the WMTS REST interface and there is already an incompatible use for the comma character in the WMS Capabilities document. ISO also provides a mechanism to use years outside the range of 0001 to 9999 that is compatible with the approach used here.

In principle, sub-second precision could extend indefinitely, but the finest practical accuracy even for synchronizing atomic clocks is around one nanosecond. Choosing a granularity limit of less than a nanosecond is generally a mistake, since systems far too often go through the evolutionary cycle of selecting seconds, then upgrading to milliseconds out of necessity, then microseconds, and finally nanoseconds (ns), because common computers can produce that amount of resolution (though not necessarily that amount of accuracy), though GPS synchronization can produce around 14 ns of accuracy in principle and 100 ns in practice. Just look at the evolution of the POSIX or SQL interfaces to see compatibility problems caused by choosing successively finer time granularities. The smartest move is just to start at ns resolution and be done with it. A ns within a second can be conveniently represented inside a system using a 32-bit integer.

8.3.5. Time granularity

The server shall identify the granularity of time values it will accept for character-for-character REST encodings with each specification of allowed time values using an integer to represent the number of time components and sub-second decimals. The following table gives the legal granularity values:

Table 6. Time granularity

Granularity Integer	Time Granularity
0	Any legal granularity 1–15 (special wildcard)
1	Year
2	Month
3	Day
4	Hour
5	Minute

Granularity Integer	Time Granularity
6	Second
7	0.1 Second / Decisecond
8	0.01 Second / Centisecond
9	0.001 Second / Millisecond
10	0.000 1 Second / Hectomicrosecond
11	0.000 01 Second / Decamicrosecond
12	0.000 001 Second / Microsecond
13	0.000 000 1 Second / Hectonanosecond
14	0.000 000 01 Second / Decananosecond
15	0.000 000 001 Second / Nanosecond

The server and client are free to produce time values encoded to any legal granularity **except** where exact encoding to the indicated granularity is required by a REST interface. If the server specifies the wildcard granularity (code 0), then the client is free to use any legal granularity in all requests **including** with all REST interfaces.

It is useful to have a specific limit to granularity precision so that inclusive and exclusive time boundaries can be expressed unambiguously. This limit shall be the **nanosecond**. For example, the last legal time instant before “2015-03-17T00:00:00.000000000Z” is “2015-03-16T23:59:59.999999999Z”, so this value can be used explicitly as an inclusive upper limit to express a time “less than” or excluding “2015-03-17T00:00:00.000000000Z”.

All time instants expressed to any granularity in the interface shall be interpreted internally by both client and server as representing the given value expanded to its first nanosecond. For example, “2015-03-17” would be interpreted as “2015-03-17T00:00:00.000000000Z”. Time instants re-expressed to a lesser granularity are always truncated instead of rounded. For example, “2015-03-16T23:59:59.999999999Z” re-expressed to Second granularity is “2015-03-17T23:59:59Z”. (One would not round up 10:45 to say that it is 11 o’clock.)

8.3.6. Time interval

A time interval is specified using the inclusive first time instant and the inclusive last time instant, later than or equal to the first time instant, separated by the slash (“/”) character. Both time instants are interpreted as being expanded to their first nanosecond. For example, “2013-01-01T17Z/2015-03-17T17Z” would be interpreted as the inclusive interval from “2013-01-01T17:00:00.000000000Z” to “2015-03-17T17:00:00.000000000Z”. The two given time instants can be expressed to different granularities except where precise encoding is required by a REST interface. Be careful to give the appropriate ending instant recognizing that it is inclusive. Also beware of unintentional overlap in a list of time intervals that are not meant to overlap.

ISO also allows either or both time instants to be empty strings corresponding to positive and negative infinity, but this notation is not needed here as explicit interval starting and ending times

can always be given unless a layer has no time information available at all (perhaps because it is completely empty of content). In such a case, the server has various options, such as reporting a special null-value as an acceptable old-style time value that it knows how to interpret.

For comparisons, using the nanosecond granularity limit makes it possible to effectively indicate whether time boundaries are inclusive or exclusive. For example, if a time interval is to be specified from noon 2015-03-17 EDT to 2pm but excluding both bounds, a proper expression would be "2015-03-17T16:00:00.000000001Z/2015-03-17T17:59:59.999999999Z". If the two bounds were inclusive, it could be expressed more compactly as "2015-03-17T16Z/2015-03-17T18Z".

The slash ("/") interval-separator character is very problematic for the WMTS REST interface because it is the folder separator used in most URI schemes and maps to the web-server file system, so using it would cause awkward interference with parsing pathnames. For this reason, in the WMTS REST-request URI only, two dash characters ("**--**", U+002D U+002D) shall be used in place of the slash character when encoding interval time values and other time parameters like resolutions.

8.3.7. Time resolution

ISO provides an encoding for time repetitions, but it is needlessly awkward. WMS defines a simpler format for repeated time instants/resolutions/intervals: "*first_instant*/*last_instant*/*period*", using an ISO period as the last component, which is used here.

The period component has the general format "**P**[*y*__Y__][*m*__M__][*d*__D__]T[*h*__H__][*m*__M__][*s*__S__]", where the square brackets indicate each optional component. The letter "**P**" is always present and means "Period". The other components give the number of years, months, days, hours, minutes, and/or seconds, where each value can be any floating-point number. Note that different implementations are likely to interpret non-round results slightly differently. The letter "**T**" must always be used as a separator in the appropriate position if any time components to its right are used, as the letter "**M**" would otherwise be ambiguous. Examples are "P1D" for one day, "PT1H" for one hour, and "P1.5D" or "P1DT12H" for one-and-a-half days.

A time interval with a resolution identifies a series of discrete time instants defined by the starting time instant plus all whole-number multiples (0, 1, …) of the given resolution (period) earlier than or equal to the last time instant of the interval. (The "last" instant is a limit that is not necessarily included in the set of specified instants.) For example, "2010-01-05T17Z/2016-03-25T17Z/P14D" defines the first nanosecond of noon EST (disregarding Daylight Savings) of every second Tuesday in the time interval. The first time instant of the series is "2010-01-05T17Z", the second, "2010-01-19T17Z", and the last, "2016-03-22T17Z". Note that precise time values in UTC do not necessarily align with time granularities, nor do resolutions.

If a time interval is given without a resolution to describe the times for which data is available, it means that any time instant (or interval) within the interval bounds can be requested (subject to any granularity restrictions).

It can also be useful to provide a resolution for a single time instant in order to distinguish it from data prepared for different time resolutions available at the same instant (e.g., daily, monthly, yearly relative to instant "2015-01-01"). The special notation "*instant*/*resolution*" is used where appropriate for this purpose.

8.3.8. Frames per second

For animations, an additional parameter can be appended to a time specification to request the number of frames per second to use for the animation. The format is "/**F***fps*". The "*fps*" is a floating-point number. The default value is server-specific.

8.4. WMTS interface extensions

Time is added to the WMTS 1.0.0 interface using the existing **Dimension** metadata structure. The Capabilities document indicates precisely what times are available with a granularity indicator and a list of time semantics with start/end time intervals sometimes with time resolutions. ISO 8601 time-value encoding is used and it is important that time values be character-for-character-precise when used with static, pre-generated REST tiles. This allows a time-based WMTS to be implemented using only a plain web server with no special runtime component or by using a large-scale service like Amazon CloudFront to enable potentially millions of concurrent users.

8.4.1. Capabilities Dimension declaration

The structure of the WMTS 1.0.0 **Dimension** declaration is retained but with new values in the **UOM** and **Value** fields. This creates conflict between these field values which may necessitate a new version number for the WMTS to distinguish the semantics.

The **UOM** field (Unit Of Measure) shall identify and parameterize the granularity of a new time dimension and has the following format: "**ISO8601**/*granularity*". The UOM "*granularity*" value is the numeric granularity level from 0 to 15 shown in the table in the "Time granularity" clause. It is important to explicitly express the time granularity to use with REST requests up front exactly one time in order to be clear and unambiguous. It is also useful to have a different UOM value from the WMTS 1.0.0 literal "ISO8601" to indicate for old clients that actually check that the enumerated values are to be handled differently from 1.0.0. Also, unless and until this specification is incorporated into the official WMTS standard, a different dimension name such as "**QTime**" (query time) should also be used to avoid compatibility problems with clients that are hard-wired to recognize the dimension name "**Time**" with the old encoding.

The time instants and time semantics available for querying are given in the **Value** fields of the **Dimension**. An example **Dimension** declaration is:

```
<Dimension>
  <ows:Identifier>QTime</ows:Identifier>
  <ows:UOM>ISO8601/3</ows:UOM>
  <ows:Title>Time</ows:Title>
  <ows:Abstract>blah blah blah</ows:Abstract>
  <Default>alltime</Default>
  <Value>at:2007-05-01/2007-07-31/P1D</Value>
  <Value>asof:2007-05-01/2007-07-31/P1D</Value>
  <Value>series:2007-05-01/2007-07-31/P1D</Value>
  <Value>interval:2007-05-01/2007-07-31</Value>
  <Value>anim.asof:2007-05-01/2007-07-31/P1D/F12</Value>
  <Value>anim.interval:2007-05-01/2007-07-31</Value>
</Dimension>
```

Time values with the new semantics shall be identified by special short prefix strings. The WMTS 1.0.0 opaque-value semantics are maintained for all values that do not start with a new-time-semantic prefix. The new prefix shall be followed by an interval with a usually optional resolution which identifies precisely what time instants are legal for querying. If a resolution is given, it restricts legal query instants to only those that are integer multiples of the resolution from the starting time; otherwise, any time instant within the interval is legal. With time semantics that use a query interval in the client request, only legal time instants can be used for the starting and ending times of the interval. Animation semantics can also include an optional frames-per-second parameter if it is fixed by the server. Specifically, the Capabilities-declaration format is "*prefix*:*first_instant*/*last_instant*[/*resolution*].

The new time-semantic Value-prefix strings for single-frame still images are:

at:

Selects a server-defined semantic for a single time instant. The WMTS 1.0.0 interface provides an interface to query tiles "at" specific server-defined time instants. The true meaning of the time instant is not clear, as it could be truly "at" a specific time instant, a time-series interval starting at or ending at the indicated time, data "as-of" a certain time, etc. Often, the precise semantic is unimportant or already known by the user. The server gives an accessible-time interval with an optional resolution; the client requests the sought time instant. The "at" client-query format is "**at:***instant*", for example, "at:2007-05-20".

asof:

Selects all times from the first time instant up to and including the given time instant. The "as-of" time-value semantic is probably the most useful for casual viewing. It can be used as a "time machine" to examine how the coverage area looked according to the most recently available data as of any time in the past. A GUI control might be a slider bar covering the time range of the available data. The server gives an accessible-time interval with an optional resolution; the client requests the ending time instant. The "as-of" client-query format is "**asof:***instant*", for example, "asof:2007-05-20".

series:

Selects one frame from a fixed time series identified by the first time instant of the frame. The resolution breaks the time interval into equal slices/frames (except perhaps for the last slice)

and each slice includes its starting time and excludes the starting time of the next slice. The exact semantic of whether the time series defines instants or sub-intervals of time is server-dependent. Beware of the potential "fencepost problem" caused by the ending instant being inclusive. The server gives an accessible-time interval with a **required** resolution; the client requests a time instant plus the server-given resolution value (needed to disambiguate if multiple resolutions are available). In the case of a REST request, the resolution component shall be character-for-character exactly what the server gave. The time-series client-query format is "**series:***instant*/*resolution*", for example, "series:2007-05-20/P1D". The implied query interval is from *instant* to *instant* + *resolution* − 1 nanosecond, modulo the data-availability interval.

interval:

Selects all times in a given arbitrary inclusive time interval. Normally, all source data that overlaps the time interval is selected for processing. The server gives an accessible-time interval with an optional resolution; the client requests a time interval. The time-series client-query format is "**interval:***first_instant*/*last_instant*", for example, "interval:2007-05-20/2007-06-08".

Each still-image/single-frame method has an animated counterpart with a prefix-prefix of "**anim.**". An animation is an orthogonal concept that selects multiple sequential frames of the base semantic with a time interval and at a specified resolution. The resulting content will normally be generated into a format that can hold multiple frames of animation, such as an MPEG4 video file, and with an optional given frame rate. Specifically, the Capabilities-declaration format is "*prefix***:***first_instant* /*last_instant*/*resolution*[/*frames_per_second*]". All of the different semantics have the same query format, with only the meaning of the content being different (at, as-of, interval).

anim.at:

Selects an animation of "at" image frames. The server gives an accessible-time interval with a required resolution; the client requests a time interval with the same resolution. With a REST request, the resolution must character-for-character the same.

anim.asof:

Selects an animation "as-of" image frames. The server gives an accessible-time interval with a required resolution; the client requests a time interval with the same resolution. With a REST request, the resolution must character-for-character the same. An example client query time might be "anim.asof:2007-05-11/2007-07-08/P1D/F12".

anim.series:

Selects an animation of fixed-time-series image frames. The server gives an accessible-time interval with a required resolution; the client requests a time interval with the same resolution. With a REST request, the resolution must character-for-character the same.

anim.interval:

Selects an animation of arbitrary-interval image frames with an arbitrary time resolution. The server gives an accessible-time interval with an **optional** resolution; the client requests a time interval with an arbitrary client-chosen time resolution that need not have any relation to the server-advertised resolution if there was one. An example client query time might be "anim.interval:2007-05-06/2007-07-28/P7D/F2".

The client can request any legal time defined by any **Value** field of a **Dimension** or by the **Default**

field. The client and server are free to generate time values to any legal granularity except in the case of a WMTS REST request, where the client must use the time granularity indicated by the server unless the wildcard granularity (code 0) is indicated. The REST granularity, the granularities in Capabilities, the granularities in non-REST requests, and the time resolutions in different **Value** fields may all be different.

As an example, a WMTS may offer a static-REST "as-of" time Dimension Value that overlaps new imagery scenes every second Tuesday but maintains a history of previous data versions. It might advertise its valid times as "asof:2010-01-05T12Z/2016-03-23T12Z/P14D" with a granularity of Hours (code 4). The first valid request time instant would be "2010-01-05T12Z", the second "2010-01-19T12Z", and the most recent, "2016-03-22T12Z".

To help with the forward-compatibility of old clients, the **Default** field could specify a special value to request "timeless" or all-time tiles. Servers could also accept the raw REST template-variable encoding as selecting the default value. E.g., a REST request might be for "https://…/mylayer/mystyle/**{QTime}**/smerc/…" or "…/**%7BQtime%7D**/…". The server could recognize that a client does not understand **Dimension** usage and select the default time value. It could also recognize the client parroting back the literal time values specified in the **Dimension** declaration if invalid as meaning that an old client does not know how to parse new time values, and react accordingly. Faulty JavaScript clients may generate the value "**undefined**" when substituting the variable and the WMS specification seems to also allow the value "**default**".

8.4.2. Key-value request interface

A key-value client shall request times for tiles by using the **Dimension Identifier** as the key, selecting an appropriate **Dimension Value** field, and computing a legal time value according to the time semantic. The request value shall include the same time-semantic prefix as the **Dimension Value**.

An example request might be:

https://website.org?SERVICE=WMTS&REQUEST=GetTile&VERSION=1.0&LAYER=my_layer&STYLE=my_style&TILEMATRIXSET=smerc&TILEMATRIX=4&TILEROW=3&TILECOL=6&FORMAT=image/PNG&QTime=series:2007-06-04/P1D

Note that key-value parameter names are case-insensitive in OGC requests.

8.4.3. REST request interface

The REST method is specified in the Capabilities document using the **ResourceURL** template structure. The time key and value are computed similarly to key-value requests, and the value is substituted into the URI template variable with the same name as the key. For example, variable substring "**{QTime}**".

There are three differences in generating the time values, however. The first difference is that the time instants shall be generated to the granularity indicated by the server in the **UOM** field. The second difference is that because of incompatibilities between REST URIs and the slash character used in ISO time intervals, etc., the slash character ("/") in computed time values shall be substituted with two dash characters ("**--**"). The third difference is that with "**series:**", "**anim.at:**",

“**anim.asof:**” and “**anim.series:**” time values, the time-resolution component shall be character-for-character exactly what the server indicated in the **Dimension Value**.

An example **ResourceURL** and REST-request URI are:

```
<ResourceURL format="image/png" resourceType="tile"
template="https://website.org/wmts/my_layer/{Style}/{QTime}/{TileMatrixSet}/{TileMatri
x}/{TileRow}/{TileCol}.png">
```

URI example:

https://website.org/wmts/my_layer/my_style/series:2007-06-04-P1D/smerc/4/3/6.png

The special characters in the REST URI may be escaped with the “%*xx*” hexadecimal notation. This is compatible with character-for-character encoding.

8.5. WMTS server implementation with enhanced time-varying layer support

For the Testbed-12 project, CubeWerx on behalf of Polar View, deployed a WMTS that serves tiles for arbitrary time intervals (trivially including the simpler semantics). As the tiles are necessarily generated on-the-fly, the “secret sauce” for such a server is its internal optimizations for dealing with a large volume of source data. Several such optimizations are planned. The time-based WMTS is accessed by a third-party viewer client to demonstrate interoperability for the project. CubeWerx also implemented a parallel WMS with time-query and animation capabilities.

We wanted to directly tackle the problem of handling arbitrarily selected time intervals so that we only need to go through the implementation effort once and it can be used for all of the time filtering needed. Specialized and limited time semantics like monthly time-series or monthly “as-of” tiles could have been pre-generated which would operate more efficiently, but we decided to focus on optimizing the general solution to perform adequate performance.

8.5.1. OWS Interfaces

The CubeWerx WMTS server implements the proposed time-query interface and uses wildcard time granularity (code 0) and supply “**at:**”, “**asof:**”, “**series:**”, and “**interval:**” time semantics through the REST and key-value-pair WMTS interfaces. The corresponding WMS interface offers the same single-frame semantics through a key-value-pair interface and also offers the animated versions of the semantics. Offering animations through the WMTS interface would make little sense since a viewer client would not be able to synchronize the multiple animated tiles on its display, even if it could process and position video files for playback.

8.5.2. Source data

The data used for the project is 16-bit single-band Advanced Synthetic Aperture Radar raster data that exists in small scenes at disconnected times near the north pole. The original source data has 37,939 TIFF images that occupy 6.7 TB of storage and cover two years of EnviSAT data from 2010-03-

01 to 2012-02-29.

The original source data had a several problems: the timestamp values were encoded into the filenames instead of the TIFF headers; the no-data value was not explicit; the data contained a great deal of empty pixels and hence was needlessly bulky; the data was internally organized into scanlines rather than tiles, making access to zoomed-in portions of the images slower; it needed a great deal of contrast stretching to be visible; and the data was not corrected for sensor-intensity variations.

A script program was developed to repackage the source data so it could be used conveniently and efficiently. The timestamps were parsed out of the filenames and put into the Date/Time TIFF header; the no-data value was set explicitly using the GDAL No-Data tag; the pixel data was compressed within the TIFF format using the lossless Deflate compression method which reduced the data to about one-third of its original size; the new TIFF files were internally organized into 256×256 tiles for efficient access; and a suitable contrast-stretching method was developed. The source data now occupies 2.1 TB of storage. Corrected versions of data source scenes are still being sought.

8.5.3. Contrast enhancement

Contrast enhancement was an “unscheduled problem” in the project. The source data has a 16-bit pixel-sample range, or 0 to 65,535, but most of the actual data values are in the range of 400 to 2300, though a few samples in each image can go up to the 20,000 range or beyond. An ad-hoc method was needed to determine what range to use, how to record it in the images, and how to apply it at runtime.

Initially, a contrast enhancement specific to each individual source image was calculated by scanning the source samples and computing a histogram of the distribution of values and a percentile range was extracted. Samples from the 3rd to the 97th percentile produced a good contrast. A special value was written into the TIFF Description tag to record the contrast stretching that is needed on rendering the image and code was implemented to apply this stretching on-the-fly when reading the image, so that the original sample values remain intact in the source data.

After some experimentation it was decided to use a constant contrast stretch for all of the images in the collection. This helps to make the images have a more consistent appearance when selecting time same area at different times and when generating animations.

The source data itself, however, is ‘uncorrected’ in that the sensitivity of the sensor varies over the swath and between images in the same area, and this cannot be normalized by any simple means. The data provider is looking into providing corrected source data for us to use.

8.5.4. Dateline wrap-arounds

Handling dateline wrap-arounds was another “unscheduled problem” in the project. Since the source data is in the Polar Stereographic projection, EPSG code 3413, some scenes cross over the the antimeridian at ±180° longitude, which will call the “dateline”. This was not dealt with properly, so we needed improve our handling of time complication in many levels of our system. This took some time and effort. On the other hand, the source data does not touch the North Pole since the EnviSAT satellite does not pass close enough to the North Pole; there is an almost perfect empty circle

around the North Pole when all of the data is plotted.

8.5.5. Performance and functionality enhancement

Numerous changes were needed to our existing system to handle time queries efficiently. The low-level spatial-database-access methods needed to be reorganized and then augmented with time functionality. Low-level methods were implemented to parse, generate, and compute time values for the new external representation.

A special optimization was added to the map/tile-rendering methods to make them use far less memory when source-data scenes are deeply stacked on top of each other, since the source data for this project can be stacked thousands of levels deep, and a related optimization was added to avoid retrieving and rendering earlier-timed source data that is completely hidden behind later source data that is plotted over top. The processing of reprojection distortions was improved to better handle strongly diverging resolutions, as between the Polar Stereographic and Spherical Mercator projections near the North Pole.

The WMTS and WMS Capabilities-document reporting and request parsing was updated for time queries, including time-series for Day, Week, Month, and Year recurrence periods. The WMS interface was also extended to handle animation requests that return an MPEG-4 video stream.

The general approach used is to render all of the source-data scenes that are selected by the spatial and time-interval constraints of a query, but this can result in too much work when zoomed way out to an area that covers too many source files. So, an optimization was added to partially pre-generate tile content in this case to drastically reduce the amount of work that needs to be done on-the-fly.

The result is that time-filtering WMTS queries normally take under one second with some taking about two seconds to execute. This is considered to be “good enough” for this project. Several additional optimizations were considered, all of which reduce the amount of processing that needs to be done on-the-fly to satisfy a request, but they are not needed for this project. Tile queries for all of time use fully pre-generated tiles when zoomed out so that all queries for the native tile-image format can be satisfied directly by the web server within a millisecond (excluding network-transfer time) and queries for non-native formats can be satisfied in under 50 milliseconds. The scalability of the server for concurrent users is around 10 users for time-filtering queries and hundreds of users for all-time queries.

8.6. AWST WMTS client implementation

For the Testbed-12 project, AWST on behalf of Polar View deployed a WMTS viewer client (http://80.120.183.78:3001/) that allows the user to select time constraints for viewing the data.

The client is based on the Cesium – WebGL Virtual Globe and Map Engine, the TerriaJS software developed by NICTA, Australia and the NationalMap software which was originally established to provide access to spatial data from Australian government agencies. All three are open source software projects. The client software is organized such that these individual open source components can independently be updated when new versions become available.

The source data on the server side is accessed via WMTS where the client automatically generates

data queries when the user selects a data source in the client, navigates on the Virtual Globe 2D or 3D map or changes the zoom level. Queries are generated for all tiles covered by the currently visible area in the map.

The WMTS access is enhanced by time queries, where the user selects the desired query type for each data source. While all allowable query types are in principle supported, the **asof:** and **series:** time semantics are explicitly configured for user selection where series options are preconfigured for daily, weekly, monthly, and yearly series. The query time is selected in an interactive time slider in the client GUI with the selectable time range defined by the server providing the data. The following example shows the GUI design of AWST WMTS client. The supported time-query semantics are selectable under the dropdown box on the left side panel (1). The temporal value is selectable on the time slider bar on the bottom of the GUI. With the selected time-query semantic and temporal value, the AWST WMTS client generates the collection of WMTS requests with enhanced time-query semantics and requests the corresponding time-varying layers on the back-end WMTS server.

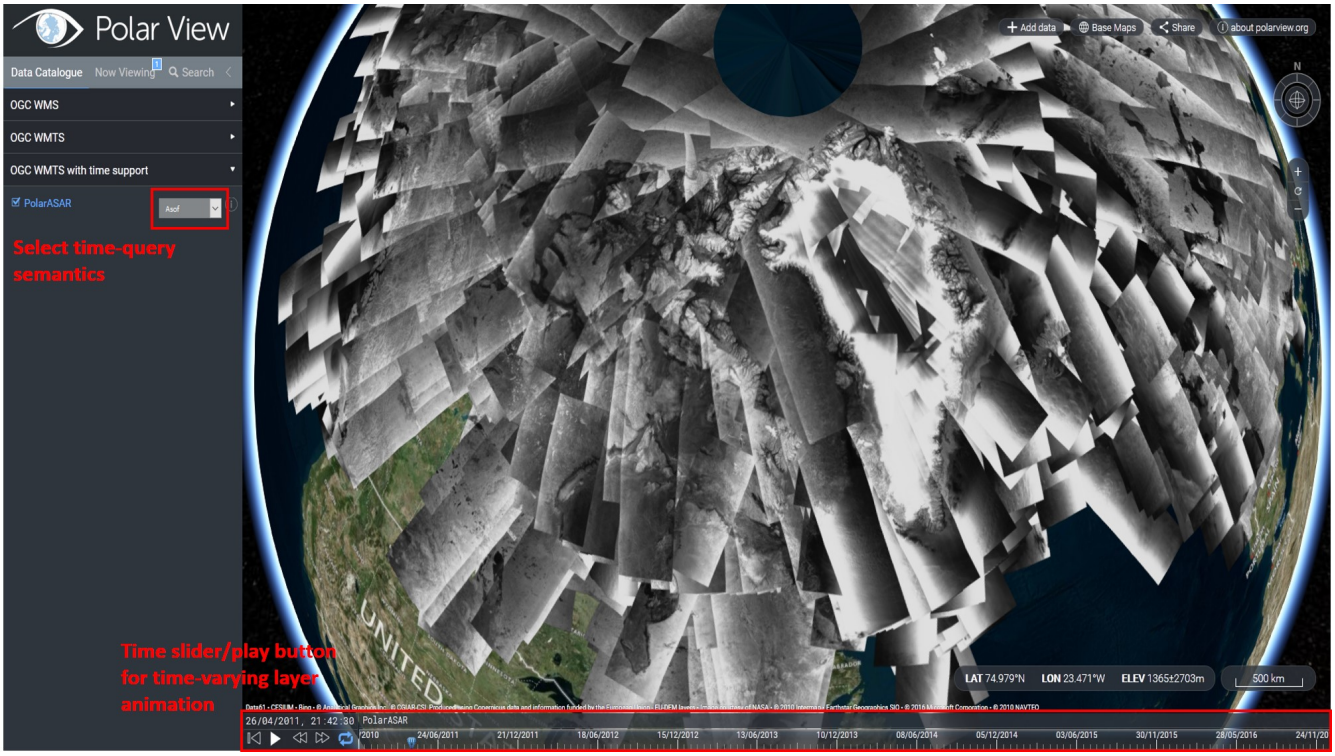

Figure 1. AWST WMTS/WMS client with new time-query semantic support

Data-animation options are implemented and tested for two different approaches. In the first animation approach, the time slider is enhanced by a “play button” feature to scan thought the available time period at different speeds. As the time advances the client automatically generates queries with the new time for the same geographic area in daily intervals. Faster scans through time quickly lead to performance issues if the server response arrives only after the next query has already been submitted. The user does not see a smooth animation in this case but rather gets incomplete images with only a few tiles, which seem to randomly change.

A number of improvements were implemented in order to achieve a better image-update performance. The server side performance improvements are described in the previous sections. The client was modified to pre-fetch the data in the sense that data for a new requested time slice is loaded while the previous image is still displayed. This has the effect that the user sees complete images in the client as long as data loading completes before the tile requests for the next time slice

are submitted. Further, better parallelization of tile requests has been investigated. This aspect was found to depend on the specific web browser, which is used to launch the visualization client. While the number of parallel http queries in generically limited to a maximum of 6 in Google Chrome, this limit is configurable in the Firefox browser. Setting this limit to 32 essentially allows requesting all tiles at the same time, at which point the server performance becomes the next limiting factor. In summary this type of time scanning is useful for automatically updating images at limited speed but does not yield a movie-like animation experience for the user. Animation performance is more limited in 3D mode where the 3D re-projection is performed in the client.

In the second animation approach, the data source is accessed via WMS to use the MPEG-4 video stream provided by the server implementation. The user can select the data source via WMS and then request a video for the currently displayed area where start and end time of the period to be covered are selected in the client's time slider element. The video query is submitted to the server upon user request, and the user can continue with other activities in the client while waiting for the server response. Once the video stream is received it is automatically opened in the default video player. As opposed to the first approach this is an asynchronous interaction where the user has to wait a while for the server to respond and the response time depends on the amount of data which is covered both geographically and by the requested time period. However, the client is not blocked while waiting for the response, hence the user can continue working with the client, and the received video stream yields a smooth animation experience, which does not depend on external performance limitations.

Following is a sample WMS request for a small video of ice moving in the Nares Strait (between Greenland and Canada) that is generated on-the-fly with a WMS request. The generated video is 2.9 MB in size. It takes about 31 seconds to generate, and returns an MPEG4 video file that should be playable in most browsers. The "PT12H" in the QTime parameter means that each frames covers 12 hours of data time and the "F4" means to generate four frames per second in the output video file.

https://tb12b.cubewerx.com/cubewerx/cubeserv?SERVICE=WMS&VERSION=1.3.2&REQUEST=GetMap&FORMAT=video%2Fmp4&TRANSPARENT=false&BGCOLOR=0xB2C0DC&LAYERS=PolarASAR_fe86e.PolarASAR&STYLES=&CRS=EPSG%3A3857&WIDTH=960&HEIGHT=540&BBOX=-9336731.666666667,15569482,-4196902.333333333,18470636&SHOW_FRAME_TIMESTAMPS=true&QTime=anim.asof:2010-03-02/2010-04-01/PT12H/F4

In parallel to work on the data interface and performance improvements, the user interface of the client was enhanced with a focus on users who are primarily interested in the polar regions.

The client development work in this project provided insights into both technical performance and usability issues and yielded feasible approaches to tackle these issues. Future work may include further optimization of data pre-loading, improvements to performance of the viewer in 3D mode and further usability enhancements to the user interface focusing on specific use cases within larger scenarios.

Chapter 9. WMS for NetCDF

9.1. Improvements on the WMS support of map rendering

With the rising demand from the diverse end-user communities for geospatial tools to handle multidimensional products, many geospatial applications now have new functionalities that enable the end user to store, access, analyze, and visualize these EO data sets. One such visualization approach of NetCDF files uses the ncWMS. In this section, the non-standard WMS parameters from ncWMS are introduced as the extension to the WMS to improve the map render and GetCapabilities content filtering.

9.1.1. Extensions on WMS GetMap operation to support additional map render options

Besides standard GetMap parameters, the following additional parameters from ncWMS are introduced as the extension parameters of WMS GetMap operation.

Table 7. Extended WMS GetMap parameters

Extended WMS parameter	Description
COLORSCALERANGE	Of the form min,max this is the scale range used for plotting the data (mapped to the COLORSCALERANGE_MIN and COLORSCALERANGE_MAX env vars)]
NUMCOLORBANDS	The number of discrete colors to plot the data. Must be]between 2 and 250 (mapped to the NUMCOLORBANDS env variable)
ABOVEMAXCOLOR	The color to plot values which are above the maximum end of the scale range. Colors are of the form 0xRRGGBB or 0xAARRGGBB, and it also accepts “transparent” and “extend”
BELOWMINCOLOR	The color to plot values which are below the minimum end of the scale range. Colors are of d the form 0xRRGGBB or 0xAARRGGBB, and it also accepts “transparent” and “extend”
LOGSCALE	“true” or “false” - whether to plot data with a logarithmic scale
OPACITY	The percentage opacity of the final output image as a number between 0 and 100 (maps to OPACITY env var by translating it to a number between 0 and 1)

Extended WMS parameter	Description
ANIMATION	“true” or “false” - whether to generate an animation. The ncWMS documentation states that TIME has to be of the form <tt class="docutils literal">starttime/endtime</tt>, but C:\work\ogc\ows12\work\LSA_NA005_ER\er_wms_wmts_OGC16-042\includes\images\redblue-default.png TIME needs to be a list of discrete times instead. Animation requires using the “image/gif” as the response format (as the only format supporting animation)

9.1.2. Examples

The supported render parameters improve the flexibilities and effects of map rendering through WMS GetMap. The improvements are illustrated through the following examples.

- **Render map with dynamic palette through WMS GetMap**
  - Launch a viewer (GetMap with format=application/openlayer):

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
test:rain&styles=&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=application/openlayers
```

  - Retrieve an image (GetMap):

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
test:rain&styles=&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=image/jpeg
```

  - Retrieve the legend (GetLegend):

```
http://cloudsdi.geo-
solutions.it/geoserver/ows?service=WMS&request=GetLegendGraphic&format=image/png
&width=20&height=20&layer=test:rain
```

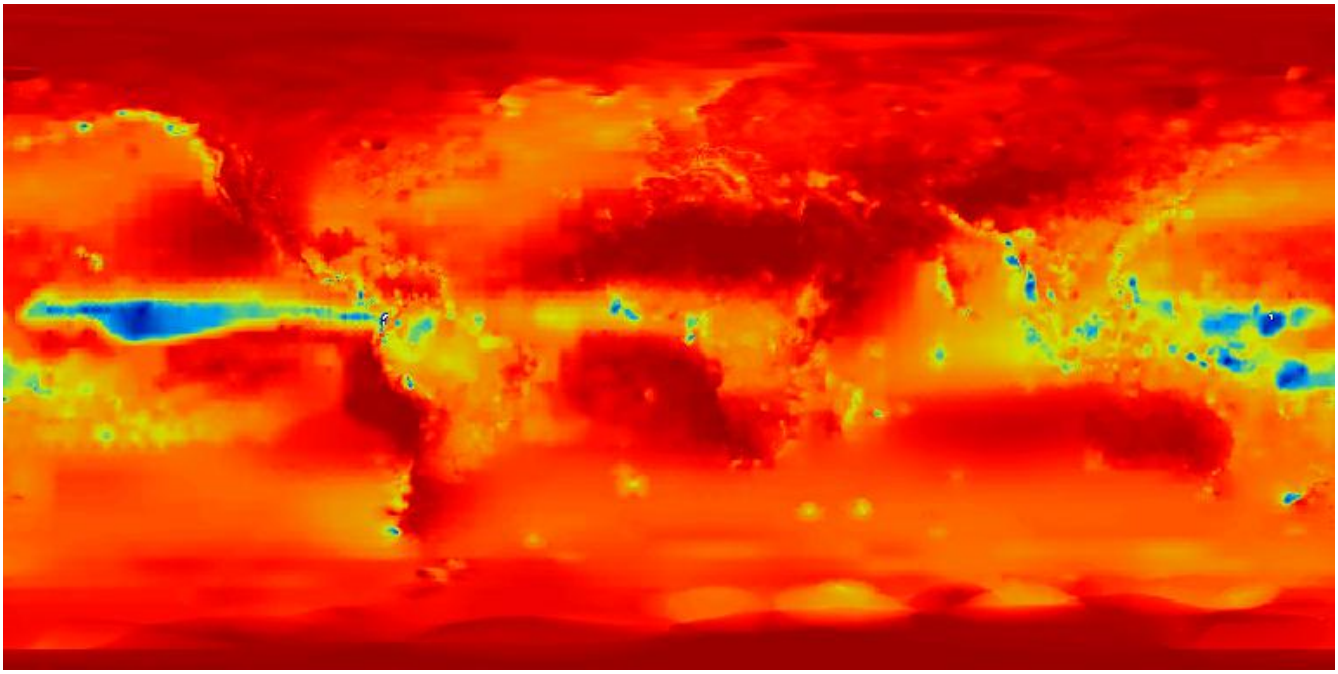

Figure 2. WMS GetMap result rendered with dynamic palette

- **Render map with custom range/custom color number/above and below range colors through WMS GetMap**
  - Launch a viewer (GetMap with format=application/openlayer):

```
http://cloudsdi.geo-solutions.it/geoserver/test/wms?http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
test:rain&styles=&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=application/openlayers
&colorscalerange=100,5000&numcolorbands=10&abovemaxcolor=0x000000&belowmincolor=
0xFFFF00
```

  - Retrieve an image (GetMap):

```
http://cloudsdi.geo-solutions.it/geoserver/test/wms?http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
test:rain&styles=&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=image/jpeg&colorscaler
ange=100,5000&numcolorbands=10&abovemaxcolor=0x000000&belowmincolor=0xFFFF00
```

  - Retrieve the legend (GetLegend):

```
http://cloudsdi.geo-
solutions.it/geoserver/ows?service=WMS&request=GetLegendGraphic&format=image%2Fp
ng&width=20&height=20&layer=test%3Arain&colorscalerange=100,5000&numcolorbands=1
0&abovemaxcolor=0x000000&belowmincolor=0xFFFF00
```

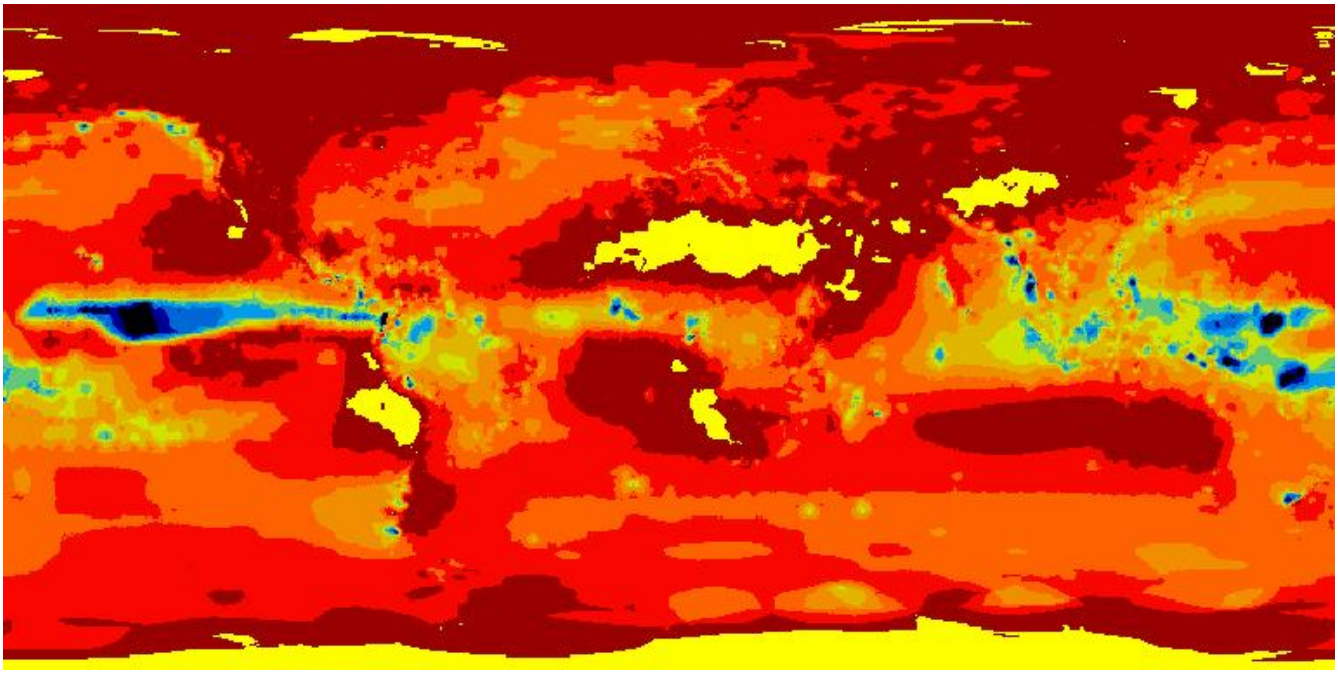

Figure 3. WMS GetMap result rendered with customized color range/color number

- **Render map with log-scaling color schema through WMS GetMap**
  - Launch a viewer (GetMap with format=application/openlayer):

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
test:rain&styles=&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=application/openlayers
&colorscalerange=1,6000&logscale=true
```

  - Retrieve an image (GetMap):

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
test:rain&styles=&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=image/jpeg&colorscaler
ange=1,6000&logscale=true
```

  - Retrieve the legend (GetLegend):

```
http://cloudsdi.geo-
solutions.it/geoserver/ows?service=WMS&request=GetLegendGraphic&format=image%2Fp
ng&width=20&height=20&layer=test%3Arain&colorscalerange=1,6000&logscale=true
```

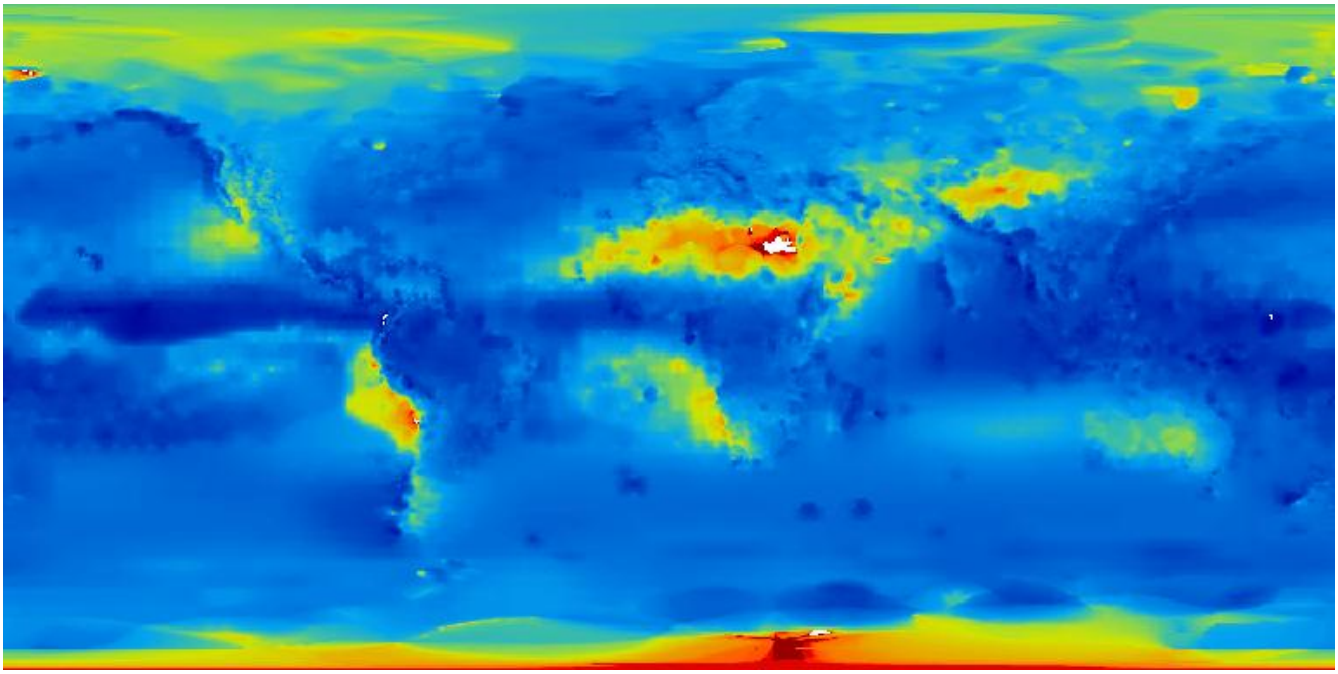

Figure 4. WMS GetMap result rendered with log-scaling color schema

- **Customize layer opacity through WMS GetMap**
  - Launch a viewer (GetMap with format=application/openlayer):

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
countries,test:rain&styles=,&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=application/openlayers
&opacity=50
```

  - Retrieve an image (GetMap):

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?service=WMS&version=1.1.0&request=GetMap&layers=
countries,test:rain&styles=,&bbox=-180.0,-
90.0,180.0,90.0&width=768&height=384&srs=EPSG:4326&format=image/jpeg&opacity=50
```

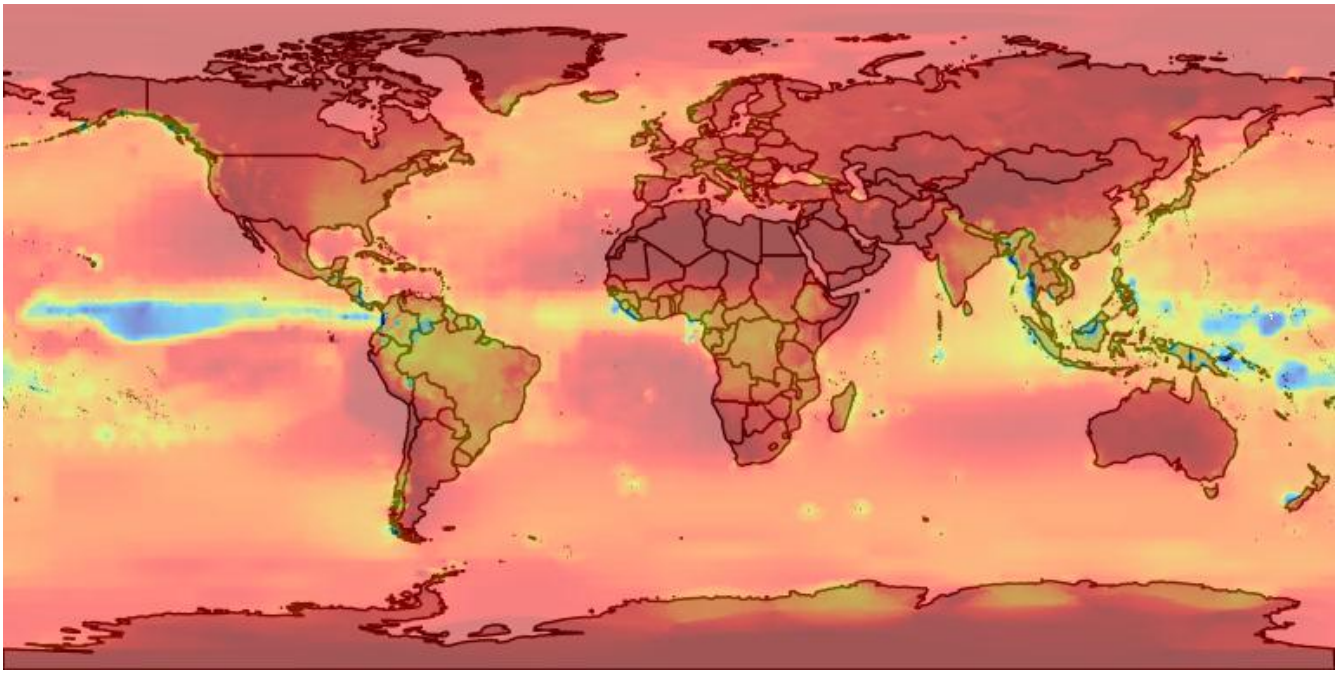

Figure 5. WMS GetMap result rendered with customized opacity

- **Generate layer animation through WMS GetMap**
  - Layer animation through time dimension (layer composite through the time interval between 2016-02-01 and 2016-03-01):

```
http://cloudsdi.geo-solutions.it/geoserver/wms?STYLES=,x-
Sst&LAYERS=countries,flexpart&FORMAT=image%2Fgif&SERVICE=WMS&VERSION=1.1.1&REQUE
ST=GetMap&SRS=EPSG%3A4326&BBOX=-137,15,-110,42&WIDTH=600&HEIGHT=600&time=2016-
02-01/2016-03-01&animation=true&bgcolor=0x0066CC
```

  - Layer animation on elevation dimension

```
http://cloudsdi.geo-solutions.it/geoserver/wms?STYLES=,x-
Sst&LAYERS=countries,flexpart&FORMAT=image%2Fgif&SERVICE=WMS&VERSION=1.1.1&REQUE
ST=GetMap&SRS=EPSG%3A4326&BBOX=-137,15,-
110,42&WIDTH=600&HEIGHT=600&elevation=0/2000&animation=true&bgcolor=0x0066CC
```

9.2. Improvements on the mechanisms of WMS metadata filtering

9.2.1. Limitation on current WMS GetCapabilities filtering

The WMS layer metadata is advertised in the GetCapabilities document. With the increasing number of layers being advertised through the GetCapabilities document, requesting the GetCapabilities document tends to consume significant bandwidth and internet resource. In this section, an additional GetCapabilities parameter (i.e., DATASET) is proposed as a way for the client to efficiently subset relevant GetCapabilities content based on the dataset/layer of interest.

The parameter still makes the server return a valid capabilities document that can be used by any application, it's thus suited to be used from existing clients and linked from dedicated search pages (e.g., a CSW/OpenSearch search result).

This is unlike XPATH based approaches, that do not require an external search engine, allow actual text based searches into the capabilities document, returning all the matched elements (e.g., to get a full list of all layer names). This approach does however require a client that has been developed to leverage the search and handle its results.

9.2.2. Extensions on WMS GetCapabilities operation

Extension on GetCapabilities request parameter

Table 8. Extended GetCapabilities parameters

Extended GetCapabilities parameter	Description
DATASET	The parameter will be set with layer/dataset name. Once the parameter is provided in the request, only content regarding the layer/dataset will be returned.

GetCapabilities response for specified layer

A schema-valid GetCapabilities response is returned with layer metadata being specified in the DATASET request parameter.

9.2.3. Examples

- **Retrieve the GetCapabilities document with the full set of layer metadata**
  - Full capabilities document:

```
http://cloudsdi.geo-
solutions.it/geoserver/ows?service=wms&request=GetCapabilities
```

- **Retrieve the GetCapabilities document based on a specific layer**
  - The GetCapabilities for the flexpart layer:

```
http://cloudsdi.geo-
solutions.it/geoserver/ows?service=wms&request=GetCapabilities&dataset=flexpart
```

  - The GetCapabilities for the rain layer:

```
http://cloudsdi.geo-
solutions.it/geoserver/ows?service=wms&request=GetCapabilities&dataset=rain
```

9.3. Improvements on the WMS support for map querying

ncWMS extends the GetFeatureInfo with a few variants targeted to retrieve evolution of a phenomenon on a specific point over time or elevation, or at a fixed them and elevation, but along a path. In particular:

- **GetTimeSeries** has the same structure as a GetFeatureInfo call, but requires a TIME parameter expressing a time range, and will generate either a dump of all values at the point of interest in the specified time interval, or generate a chart
- **GetVerticalProfile** has the same structure as GetFeatureInfo, but requires an ELEVATION parameter expressing an elevation range, and will generate either a dump of all values at the point of interest in the specified elevation interval, or generate a chart
- **GetTransect** has the same structure as a GetMap, but requires a LINESTRING parameter with a comma separated list of "x,y" values, and will extract a chart of the phenomenon evolution along the path. If the dataset contains an elevation, a parallel chart will be generated showing the elevation at the same points.

During the testbed a GetTimeSeries implementation has been included.

- **Retrieve the evolution of the "flexpart" phenomenon over time at point clicked on the map, in the 2016-02-23T03:00:00.000Z/2016-02-26T00:00:00.000Z time range:**

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?SERVICE=WMS&VERSION=1.1.1&REQUEST=GetTimeSeries&FOR
MAT=image%2Fgif&TRANSPARENT=false&STYLES&LAYERS=test%3Aflexpart&SRS=EPSG%3A4326&WID
TH=600&HEIGHT=600&BBOX=-137.4169921875%2C20.56640625%2C-
111.0498046875%2C46.93359375&time=2016-02-23T03:00:00.000Z/2016-02-
26T00:00:00.000Z&QUERY_LAYERS=test%3Aflexpart&INFO_FORMAT=image/png&x=335&Y=217
```

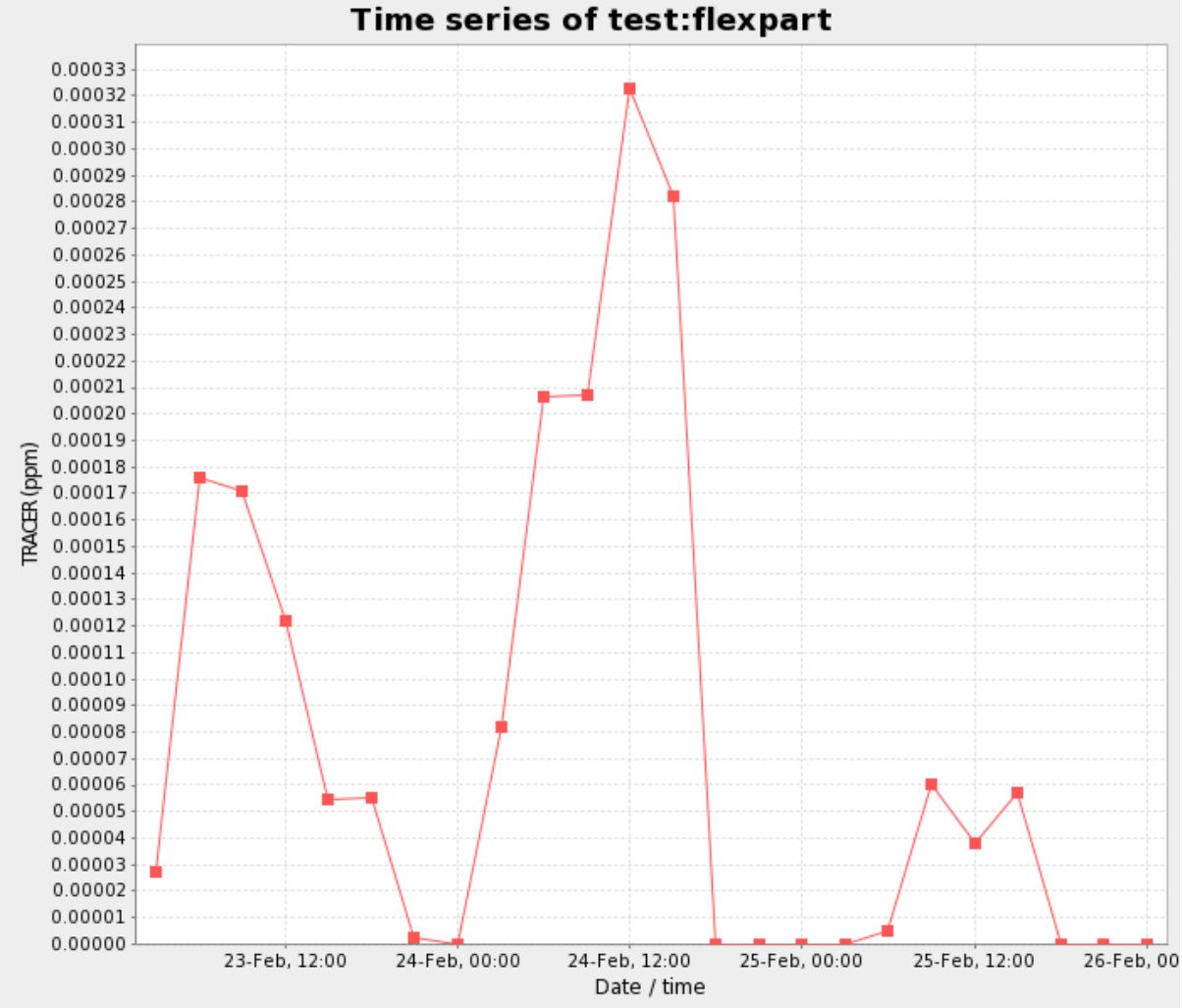

Figure 6. WMS GeTimeSeries result as a chart

- **Changing the output format to text/csv returns a dump of the exact location along with a table of times and values:**

```
http://cloudsdi.geo-
solutions.it/geoserver/test/wms?SERVICE=WMS&VERSION=1.1.1&REQUEST=GetTimeSeries&FOR
MAT=image%2Fgif&TRANSPARENT=false&STYLES&LAYERS=test%3Aflexpart&SRS=EPSG%3A4326&WID
TH=600&HEIGHT=600&BBOX=-137.4169921875%2C20.56640625%2C-
111.0498046875%2C46.93359375&time=2016-02-23T03:00:00.000Z/2016-02-
26T00:00:00.000Z&QUERY_LAYERS=test%3Aflexpart&INFO_FORMAT=text/csv&x=335&Y=217
```

```
# Latitude: 37.3974609375
# Longitude: -122.6953125
Time (UTC),TRACER (ppm)
2016-02-23T03:00:00.000Z,2.7452661015558988E-5
2016-02-23T06:00:00.000Z,1.7581375141162425E-4
2016-02-23T09:00:00.000Z,1.70844592503272E-4
2016-02-23T12:00:00.000Z,1.2185460946056992E-4
2016-02-23T15:00:00.000Z,5.4905322031117976E-5
2016-02-23T18:00:00.000Z,5.5376200180035084E-5
2016-02-23T21:00:00.000Z,2.3092284209269565E-6
2016-02-24T00:00:00.000Z,0.0
2016-02-24T03:00:00.000Z,8.217969298129901E-5
2016-02-24T06:00:00.000Z,2.0651771046686918E-4
2016-02-24T09:00:00.000Z,2.0721326291095465E-4
2016-02-24T12:00:00.000Z,3.229356952942908E-4
2016-02-24T15:00:00.000Z,2.822736569214612E-4
2016-02-24T18:00:00.000Z,0.0
2016-02-24T21:00:00.000Z,0.0
2016-02-25T00:00:00.000Z,0.0
2016-02-25T03:00:00.000Z,0.0
2016-02-25T06:00:00.000Z,4.860657554672798E-6
2016-02-25T09:00:00.000Z,6.0236787248868495E-5
2016-02-25T12:00:00.000Z,3.81594400096219E-5
2016-02-25T15:00:00.000Z,5.742753273807466E-5
2016-02-25T18:00:00.000Z,0.0
2016-02-25T21:00:00.000Z,0.0
2016-02-26T00:00:00.000Z,0.0
```

Chapter 10. Multidimensional WMTS domain discovery

10.1. Objective

This document describes a WMTS 1.0.0 extension adding multidimensional domain discovery operations/resources. The use cases under consideration are:

- Datasets with large and irregular dimension domains which cannot be described by a simple `start/end/interval` syntax;
- Datasets with multiple dimensions, such as time, run-time, elevation, some dimensions being related to each other (e.g. for a given time there is a small set of corresponding run-times); and
- Datasets scattered over both space and time (e.g., typical satellite collections) where not all possible combinations of space and time locate an actual data sample.

While the current WMTS dimension support can report multiple dimensions, there is no way to discover the relationships between dimensions, moreover, large domains cannot be practically reported due to the requirement to enumerate all possible values in the capabilities document.

NOTE	While this document focuses on WMTS, the need to drill down into a N-dimensional domain to locate pockets of data also occurs while using WMS and WMTS, thus this extension tries to have limited WMTS dependencies to allow easy integration with the other two protocols as well.

10.2. Existing OWS dimension support

Explicit multiple dimension support exists in both WMTS and WMS interfaces based on slightly different semantics, while the EO-WCS extension supports a single time dimension as part of the coverage descriptions metadata.

None of the above protocols, however, provide a way to discover the structure of the dimension domain, nor provide good support for large scattered dimensions, but all would benefit from such support.

While this document focuses on WMTS, similar concepts and operations should be applicable to the other two protocols as well, with little conceptual and practical adaptations.

10.2.1. Existing WMTS dimension support

The latest official version of WMTS (Web Map Tile Service) is 1.0.0 [WMTS 1.0.0]. It includes a limited mechanism for dimension based queries.

```
<Dimension>
  <ows:Identifier>dimension1</ows:Identifier>
  <ows:UOM>theUnit</ows:UOM>
  <ows:Title>...</ows:Title>
  <ows:Abstract>...</ows:Abstract>
  <Default>d1Default</Default>
  <Value>d1v1</Value>
  <Value>d1v2</Value>
  <Value>d1v3</Value>
</Dimension>
```

```
<Dimension>
  <ows:Identifier>dimension2</ows:Identifier>
  <ows:UOM>theUnit</ows:UOM>
  <ows:Title>...</ows:Title>
  <ows:Abstract>...</ows:Abstract>
  <Default>d2Default</Default>
  <Value>d2v1</Value>
  <Value>d2v2</Value>
  <Value>d2v3</Value>
</Dimension>
```

Specific values must be explicitly enumerated, implying the practical usage is limited to small domains, and the dimension is opaque to the client, which is going to use the dimension values as-is in requests, without any understanding of them.

10.2.2. Existing WMS dimension support

The latest official version of WMS (Web Map Service) is 1.3.0 [WMS 1.3.0]. It has a more-sophisticated dimension support than WMTS. An example declaration section might describe regular domains in a compact way by providing a start, end, and resolution:

```
<Dimension name="time" units="ISO8601" default="current">2008-01-
15T00:00:00.000Z/2008-11-01T00:00:00.000Z/PT1H</Dimension>
<Dimension name="run-time" units="ISO8601" default="current">2008-01-
12T00:00:00.000Z/2008-10-28T00:00:00.000Z/PT1H</Dimension>
<Dimension name="elevation" units="EPSG:5030" default="0">0/2000/100</Dimension>
```

This allows regular domains to be described in a compact notation, however one that still fails to address discovery of the relationships between the domains. For example, in the above example a 3 days weather forecast is run periodically to generate temperature values at various heights in the atmosphere: it's thus clear to the reader there is a tight relationship between time and run-time, while a single time is related to multiple run-times, the run-time values related to a time `T` are to be found in the interval `[T - 3d,T]`.

A client unaware of such relationship will thus generate requests that match no actual data in the

dataset.

10.3. WMTS interface extensions

The starting point for a client trying to explore the structure of the multidimensional domain is the capabilities document. Services supporting this extension declare dimensions using either:

- The full domain as usual, if the domain is small enough to practically be included in the capabilities document; or
- A single value representing the full domain as "minValue—maxValue", if the domain is too large to be represented as a sequence of values in the capabilities document.

For example:

```
<Dimension>
  <ows:Identifier>dimension1</ows:Identifier>
  <ows:UOM>theUnit</ows:UOM>
  <ows:Title>...</ows:Title>
  <ows:Abstract>...</ows:Abstract>
  <Default>d1Min--d1Max</Default>
  <Value>d1Min--d1Max</Value>
</Dimension>
```

```
<Dimension>
  <ows:Identifier>dimension2</ows:Identifier>
  <ows:UOM>theUnit</ows:UOM>
  <ows:Title>...</ows:Title>
  <ows:Abstract>...</ows:Abstract>
  <Default>d2v1</Default>
  <Value>d2v1</Value>
  <Value>d2v2</Value>
  <Value>d2v3</Value>
</Dimension>
```

This approach will allow clients unaware of the dimension drilling extension to get their usual dimension support. When the range syntax is used in WMTS requests, the server is free to decide on a reasonable behavior, including:

- Stack all data in the range to compose the output;
- Using a single default value contained in the range depending on the nature of the dimension (e.g., min, max, or current);
- Evaluate the specified dimension values, considering those that are explicit and those that are implicit ranges, and locate a suitable list of dimension values actually matching an available dataset.

NOTE

The full range of values has been used, as opposed to a static key as `null` or `all` to reduce the likeliness that HTTP caching headers won't disallow clients to get fresh data when fresher data gets added to the set.

The range separator proposed is not `/` due to its incompatibility with REST oriented architectures. It is to be noted however that the usage of `--` will make ranges using negative values (e.g., elevation) unintuitive to read (e.g., `-100---10` to represent the range between -100 and -10).

The capabilities document of services implementing this extension will also declare two extra operations to discover the details of multiple dimension domain (a detailed description is provided in the following sections):

- **DescribeDomains**, allowing to drill into the dimension structure with a compact output representation; and
- **GetFeature**, allowing to locate all the available dimension values combinations.

A WMTS implementation using the procedure oriented architectural style declares the two new operations in the operations section of the capabilities document, a RESTful one will instead provide two extra url templates at the layer level.

10.3.1. Space as an element of dimension domain discovery

It is to be noted that, while the two dimensional space does not appear as an explicit `Dimension` element in the capabilities document, it is nevertheless an important element of discovery, especially for scattered domains populated by satellite or aerial sources (as opposed to models, which instead tend to always consider the same area).

Each domain resource/operation will then accept a target range of values for each dimension (space included), and will return the restricted domain/values of the others as described below.

Specifically for WMTS, the representation of space is normally provided as a TopLeftCorner in the tile matrix set CRS, a resolution and number of tiles. For the sake of simplicity, the extension will use a bounding box using the same syntax and semantics as the BBOX parameter in WMS 1.3 [WMS 1.3] instead. This won't make WMTS clients and servers more complex, as on both sides there is already a need to convert between tile coordinates and world coordinates.

NOTE

The usage of BBOX also simplifies to integration of the extension with WMS, and to a more limited extent, with WCS (WCS 2.0 uses trims on single axis to limit the area of interest instead of a bounding box).

10.3.2. Compact domain inspection

A new operation/resource is added to the WMTS protocol describing all the dimension domains in a compact document, along with the restricted bounding box of the two dimensional space intercepted by the request.

The domain description is delivered by a new `DescribeDomains` operation in procedure oriented

architectural style, and a `Domains` resource in resource oriented architectural style.

The `DescribeDomains` operation accepts the following parameters:

Table 9. DescribeDomains operation request URL parameters

Name	Optionality and use	Definition and format
"Service=WMTS"	Mandatory	Service type identifier
"Request=DescribeDomains"	Mandatory	Operation name
"Version=1.0.0"	Mandatory	Standard and schema version for this operation
"Layer"	Mandatory	Layer identifier
"TileMatrixSet"	Mandatory	Tile matrix set identifier
"BBOX=minx,miny,maxx,maxy"	Optional	Bounding box corners (lower left, upper right) in CRS units.
"DimensionIdentifier"	Optional, at most one per dimension	A range described as `min/max`, restricting the domain of this dimension
"Format"	Optional, the default being `text/xml`	The desired output format

The following is a sample request assuming a `Dimension` definition with an `ows:Identifier` value of `time`:

```
http://server.com/ows?Service=WMTS&Request=DescribeDomains&Version=1.0.0&Layer=Imagery&TileMatrixSet=CRS84&BBOX=0,0,90,180&time=2008-01-15T00:00:00.000Z/2008-11-01T00:00:00.000Z
```

The `Domains` resource is described by a URL template at the `Layer` level and works in a similar fashion, using `--` as the interval separator in requests

Table 10. URL template variables and possible values for Domains

URL template variable	Meaning	Possible values	Multiplicity
"TileMatrixSet"	Tile matrix set identifier	Identifier found at `./TileMatrixSetLink/TileMatrixSet`	One (mandatory)
"BBOX"	Space restriction	`minx,miny,maxx,maxy` expressed in the TileMatrixSet CRS, or `all` for no restriction	One (Mandatory)
"./Dimension/ows:Identifier"	Restriction for the given domain	`Min—Max` or `all` for no restriction	One for each dimension available (mandatory)

The following are a sample template and a request assuming a `Dimension` definition with an `ows:Identifier` value of `time`:

```
<ResourceURL format="text/xml" resourceType="Domains"
template="http://server.com/ows/wmts/1.0.0/layer/default/{TileMatrixSet}/{BBOX}/{time}
.xml"/>
```

`http://server.com/ows/wmts/1.0.0/layer/default/CRS84/-30,10,20,40/2008-01-15T00:00:00.000Z—2008-11-01T00:00:00.000Z.xml`

The response contains the residual space and domain of various dimensions. To accommodate competing needs of clarity and compactness, the residual domain can be expressed as a comma separate list of:

- Regular distributions expressed as `Min/Max/Interval`, using the same syntax as WMS 1.3;
- Compact scattered domains represented as `Min/Max`, indicating that there is no regular structure between the two extremes; and
- Single values.

A few examples:

- A large scattered elevation domain too large to be represented by a full list can be described as `1.5/2436`;
- An almost regular domain cut on missing data points can be represented as `1/100/1,110/200/1,215/500/1`;
- When restrictions on the various domain axis make the domain small enough a scattered domain can be represented as a simple list, e.g., `1,10,13,25,32`; and
- If the restrictions cause a domain to be completely void of data, an empty list will be returned .

NOTE The document is presented with XML bindings, for consistency with the other WMTS protocol requests. A namespace needs to be chosen, while a potential JSON presentation should be considered as well.

A compact dimension domain description contains the following elements.

Table 11. URL DimensionDomain elements

Element	Meaning	Possible values	Multiplicity
"ows:Identifier"	The dimension name	time,elevation,run-time,...	One (mandatory)
"Size"	Number of elements left in the domain with the current restriction	Non negative number	One or zero (mandatory for Scattered only)

"Domain"	A description of residual dimension domain under the current restrictions	A comma separate list of point values, regular ranges, and scattered ranges, or an empty value if no data is available under the current restrictions	Mandatory

The following document declares the dimension domains for an unrestricted request:

```
http://server.com/ows?Service=WMTS&Request=DescribeDomains&Version=1.0.0&Layer=Imagery
&TileMatrixSet=CRS84
```

```
<Domains>
  <SpaceDomain>
    <BoundingBox CRS="urn:ogc:def:crs:OGC:1.3:CRS84" minx="-180" miny="-90" maxx="180"
maxy="90"/>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain>2008-01-15T00:00:00.000Z/2008-11-01T00:00:00.000Z/PT1H</Domain>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>elevation</ows:Identifier>
    <Domain>0/5000</Domain>
    <Size>238</Sizez>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>band</ows:Identifier>
    <Domain>COAST,B,G,R,NIR,SWIR1,SWIR2,PAN,CIRRUS,TIR1,TIR2</Domain>
  </DimensionDomain>
</Domains>
```

A request restricting on the `-30,10,20,40` bounding box might result in an elevation domain compact enough to be expressed in enumerated mode:

```
http://server.com/ows?Service=WMTS&Request=DescribeDomains&Version=1.0.0&Layer=Imagery
&TileMatrixSet=CRS84&BBOX=-30,10,20,40
```

```
<Domains>
  <SpaceDomain>
    <BoundingBox CRS="urn:ogc:def:crs:OGC:1.3:CRS84" minx="-30" miny="10" maxx="20"
maxy="40"/>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain>2008-05-00T10:30:00.000Z/2008-07-20T23:00:00.000Z/PT1H</Domain>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>elevation</ows:Identifier>
    <Domain>1200,1300,1400,1500,1600,1700</Domain>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>band</ows:Identifier>
    <Domain>COAST,B,G,R,NIR,SWIR1,SWIR2,PAN,CIRRUS,TIR1,TIR2</Domain>
  </DimensionDomain>
</Domains>
```

A further restriction on the same bbox and an elevation value outside of the previous list would result in:

```
http://server.com/ows?Service=WMTS&Request=DescribeDomains&Version=1.0.0&Layer=Imagery
&TileMatrixSet=CRS84&BBOX=-30,10,20,40&elevation=100
```

```
<Domains>
  <SpaceDomain>
    <BoundingBox CRS="urn:ogc:def:crs:OGC:1.3:CRS84" minx="-30" miny="10" maxx="20"
maxy="40"/>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain></Domain>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>elevation</ows:Identifier>
    <Domain></Domain>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>band</ows:Identifier>
    <Domain></Domain>
  </DimensionDomain>
</Domains>
```

NOTE

As an alternative the space domain portion could be represented as a GML polygon in the tilematrixset CRS, allowing smart clients to avoid requesting tiles in empty areas. The extension may require the server to provide the bounding box and allow it to also provide a more precise polygon, clients can decide to use it or ignore it based on their sophistication. e.g.:

```
<Domains>
  <SpaceDomain>
    <BoundingBox CRS="urn:ogc:def:crs:OGC:1.3:CRS84" minx="-30"
miny="10" maxx="20" maxy="40"/>
    <DomainGeometry>
      <gml:MultiSurface srsName="urn:ogc:def:crs:OGC:1.3:CRS84"
srsDimension="2">
         <gml:surfaceMember>
            <gml:Polygon>
               <gml:exterior>
                  <gml:LinearRing>
                     <gml:posList>-30 20 10 40 20 10 -30
-20</gml:posList>
                  </gml:LinearRing>
               </gml:exterior>
            </gml:Polygon>
         </gml:surfaceMember>
      </gml:MultiSurface>
    </DomainGeometry>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain></Domain>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>elevation</ows:Identifier>
    <Domain></Domain>f
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>band</ows:Identifier>
    <Domain>COAST,B,G,R,NIR,SWIR1,SWIR2,PAN,CIRRUS,TIR1,TIR2</Domain>
  </DimensionDomain>
</Domains>
```

Given the `Domains` document is a summary one, server providers are advised to return a suitably simplified representation of the domain, to avoid bloating the document with a large GML geometry.

10.3.3. Histogram generation

Given a scattered domain description, a client might want to provide users a notion of the data distribution between the minimum and maximum value. The histogram divides the interval in

regular buckets, and provides an item count for each bucket.

The distribution histogram is delivered by a new `GetHistogram` operation in procedure oriented architectural style, and a `Histogram` resource in resource oriented architectural style. The client is allowed to suggest a desired resolution for the buckets, the server can accept and use the suggested value, or return a histogram with a lower resolution in the suggested value would result in an exceedingly large response.

The `GetHistogram` operation builds on top of the `DescribeDomain` structure, adding parameters to identify the histogram dimension, and the desired histogram resolution:

Table 12. Histogram operation request URL parameters

Name	Optionality and use	Definition and format
"Service=WMTS"	Mandatory	Service type identifier
"Request=GetHistogram"	Mandatory	Operation name
"Version=1.0.0"	Mandatory	Standard and schema version for this operation
"Layer"	Mandatory	Layer identifier
"TileMatrixSet"	Mandatory	Tile matrix set identifier
"BBOX=minx,miny,maxx,maxy"	Optional	Bounding box corners (lower left, upper right) in CRS units.
"DimensionIdentifier"	Optional, at most one per dimension	A range described as `min/max`, restricting the domain of this dimension
"Histogram"	Required, one value	Name of the dimension for which the histogram will be computed
"Resolution"	Optional	Suggested size of the histogram bucket. Cannot be provided for enumerated dimensions, will use the period syntax for time (e.g. `PT1H`), a number for numeric dimensions, or `auto` to leave the decision to the server.
"Format"	Optional, the default being `text/xml`	The desired output format

The following is a sample request assuming the Capabilities document contains a `Dimension` definition with an `ows:Identifier` value of `time`:

```
http://server.com/ows?Service=WMTS&Request=GetHistogram&Version=1.0.0&Layer=Imagery&TileMatrixSet=CRS84&BBOX=0,0,90,180&time=2008-01-15T00:00:00.000Z/2008-11-01T00:00:00.000Z&histogram=time&resolution=P1D
```

The `Histogram` resource is described by a URL template at the `Layer` level and works in a similar fashion, using `--` as the interval separator in requests.

Table 13. URL template variables and possible values for Histogram

URL template variable	Meaning	Possible values	Multiplicity
"TileMatrixSet"	Tile matrix set identifier	Identifier found at `./TileMatrixSetLink/TileMatrixSet`	One (mandatory)
"BBOX"	Space restriction	`minx,miny,maxx,maxy` expressed in the TileMatrixSet CRS, or `all` for no restriction	One (Mandatory)
"./Dimension/ows:Identifier"	Restriction for the given dowmain	`Min—Max` or `all` for no restriction	One for each dimension available (mandatory)
"Histogram"	Name of the dimension for which the histogram will be computed	A dimension name as found in the capabilities document	One (mandatory)
"Resolution"	Suggested size for the histogram bucket	A period for time dimensions (e.g. `PT1H`), a number for numeric dimensions, or `auto` to leave the decision to the server.	One (mandatory)

The following are a sample template and request assuming a `Dimension` definition with an `ows:Identifier` value of `time`:

```
<ResourceURL format="text/xml" resourceType="Histogram"
template="http://server.com/ows/wmts/1.0.0/layer/default/{TileMatrixSet}/{BBOX}/{time}
/{dimension}/{resolution}/histogram.xml"/>
```

`http://server.com/ows/wmts/1.0.0/layer/default/CRS84/-30,10,20,40/2008-01-15T00:00:00.000Z—2008-11-01T00:00:00.000Z/time/auto/histogram.xml`

The response contains the distribution histogram for the specified dimension, under the specified restrictions.

NOTE The document is presented with XML bindings, for consistency with the other WMTS protocol requests. A namespace needs to be chosen, while a potential JSON presentation should be considered as well.

A histogram description contains the following elements.

Table 14. URL DimensionDomain elements

Element	Meaning	Possible values	Multiplicity
"ows:Identifier"	The dimension name	time,elevation,run-time,...	One (mandatory)

"Domain"	Describes the histogram domain as start, end and resolution.	Start/End/Resolution	Mandatory
"Values"	Lists the number of values found in each bucket	A comma separate list of integer numbers, one count per bucket	Mandatory

In the following request the client requires a hourly histogram over a space of 11 months, the server considers the request excessive and returns a monthly histogram instead:

```
http://server.com/ows?Service=WMTS&Request=GetHistogram&Version=1.0.0&Layer=Imagery&Ti
leMatrixSet=CRS84&histogram=time&resolution=PT1H
```

```
<Histogram>
   <ows:Identifier>time</ows:Identifier>
   <Domain>2008-01-01T00:00:00.000Z/2008-12-01T00:00:00.000Z/P1M</Domain>
   <Values>1,15,0,2,5,7,1,0,3,9,23</Value>
</Histogram>
```

10.3.4. Detailed value combination inspection and description

Once the range of possible values has been suitably restricted, the client might need to enumerate the actual possible value combinations using a `GetFeature` operation, which in a similar way to WFS [WFS 2.0] `GetFeature`, returns a list of features along with dimension values using the same formats as the feature info operation. For this profile GML 3.1 shall be provided as a feature info format ("application/gml+xml; version=3.1") a similar GeoJSON output should be considered as an option.

NOTE Supporting this operation could be made optional, but is strongly recommended for domains that are scattered and/or having inter-dimension dependencies (e.g., even having enumerated values for all dimensions, not all possible combinations will select existing data).

Each dimension shall be represented as an attribute of the feature, while the space component is encoded as a polygon/multipolygon, which can be as simple as a bounding box, or a full fledged footprint of an EO feature.

The `GetFeature` operation accepts the following parameters.

Table 15. GetFeature operation request URL parameters

Name	Optionality and use	Definition and format
"Service=WMTS"	Mandatory	Service type identifier
"Request=GetFeature"	Mandatory	Operation name
"Layer"	Mandatory	Layer identifier

"BBOX=minx,miny,maxx,maxy"	Optional	Bounding box corners (lower left, upper right) in CRS units.
"TileMatrixSet=tileMatrixSetId"	Mandatory	Tile matrix set identifier
"DimensionIdentifier=dimension"	Optional, at most one per dimension	A range described as `min/max`, restricting the domain of this dimension
"Format=application/gml+xml; version=3.1"	Mandatory	One of the formats declared as supported in the capabilities document

The `Features` resource is described by a URL template at the `Layer` level and works in a similar fashion, using `--` as the interval separator in requests.

Table 16. URL template variables and possible values for Features

URL template variable	Meaning	Possible values	Multiplicity
"TileMatrixSet"	Tile matrix set identifier	Identifier found at `./TileMatrixSetLink/TileMatrixSet`	One (mandatory)
"BBOX"	Space restriction	`Bottom,Left,Top,Right` expressed in the TileMatrixSet identifier, or `all` for no restriction	One (Mandatory)
"./Dimension/ows:Identifier"	Restriction for the given dimension	`Min—Max` or `all` for no restriction	One for each dimension available (mandatory)

The encoded GML result contains an indication of the footprints and other dimension values.

```
<?xml version="1.0" encoding="UTF-8"?>
<wmts:FeatureCollection xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:wmts="http://www.opengis.net/wmts/1.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" numberOfFeatures="1">
    <gml:featureMembers>
        <wmts:feature gml:id="sample.1">
            <wmts:footprint>
                <gml:MultiSurface srsName="urn:ogc:def:crs:OGC:1.3:CRS84"
srsDimension="2">
                    <gml:surfaceMember>
                        <gml:Polygon>
                            <gml:exterior>
                                <gml:LinearRing>
                                    <gml:posList>-10 0 -10 10 10 10 10 0 -10
0</gml:posList>
                                </gml:LinearRing>
                            </gml:exterior>
                        </gml:Polygon>
                    </gml:surfaceMember>
                </gml:MultiSurface>
            </wmts:footprint>
            <wmts:dimension name="time">2008-05-00T10:30:00.000Z</wmts:dimension>
            <wmts:dimension name="run-time">2008-02-00T10:30:00.000Z</wmts:dimension>
            <wmts:dimension name="elevation">1200</wmts:dimension>
        </wmts:feature>
    </gml:featureMembers>
    ...
</wmts:FeatureCollection>
```

NOTE	The XML above places dimension names as attribute in order to allow usage of a simple, static XML schema to describe the GML collection, while at the same time avoiding the extra requirement of dimension names being valid QName.

10.4. Use cases

Use case description

In this use case, the WMTS client interacts with WMTS server which implements the following three new WMTS operations:

1. DescribeDomains: DescribeDomains is well suited to evaluate dimension dependencies;
2. GetHistogram: GetHistogram is used to evaluate data concentration in an interval of interest; and
3. GetFeature: GetFeature is used for inspecting the spatial footprint of available map tiles.

The use case will demonstrate:

1. How to leverage the new DescribeDomains operation for inspecting layer domain info (e.g. domain values) and discovering the relationship among multidimensional layers;
2. How to leverage the new GetHistogram operation for inspecting domain distribution histogram of layer;
3. How to leverage the new GetFeature operation for inspecting detailed domain info (e.g. spatial footprint) of each layer; and
4. How to leverage the proposed new WMTS operations to achieve efficient map tile access (e.g. preventing empty tile request).

WMTS client/server interaction overview

The WMTS client/server interactions are illustrated with the following sequence diagram. The detail workflow is also discussed in this sub section.

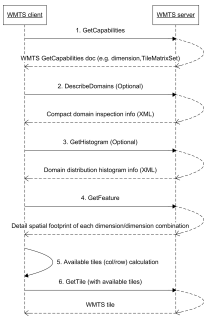

Figure 7. Sequence Diagram

Use case for inspecting multiple dependent domains

This example shows interaction for a forecast model having elevation, time and REFERENCE_TIME (the forecast execution time). The elevation domain is compact and independent of the other two dimensions. Time and REFERENCE_TIME depend on each other instead.

1. The WMTS client requests GetCapabilities document

The WMTS client retrieves the WMTS GetCapabilities document for the following information.

Available layers (XPath:/Capabilities/Contents/Layer/ows:Title)

Dimension info under each available layer (XPath:/Capabilities/Contents/Layer/Dimension)

Tile matrix configuration (XPath:/Capabilities/Contents/TileMatrixSet)

2. The WMTS client requests DescribeDomains for compact layer domain inspection

- Describe the full domain without any restriction:

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=DescribeDomains&Version=1.0.0&Layer=te
st:flexpart_time_series&TileMatrixSet=EPSG:900913
```

In the following response, the enumeration of domain values are returned for each dimension under the layer of test:flexpart_time_series.

```
<?xml version="1.0" encoding="UTF-8"?>
<Domains>
    <SpaceDomain>
        <BoundingBox CRS="EPSG:4326" minx="-180.125" miny="-90.125" maxx="179.875"
maxy="89.9375"/>
    </SpaceDomain>
    <DimensionDomain>
        <ows:Identifier>elevation</ows:Identifier>

<Domain>0.0,200.0,400.0,600.0,800.0,1000.0,1200.0,1400.0,1600.0,1800.0,2000.0,3000.0,4
000.0,5000.0,6000.0,7000.0,8000.0,9000.0</Domain>
        <Size>18</Size>
    </DimensionDomain>
    <DimensionDomain>
        <ows:Identifier>REFERENCE_TIME</ows:Identifier>
        <Domain>2016-02-23T00:00:00.000Z,2016-02-24T00:00:00.000Z,2016-02-
25T00:00:00.000Z</Domain>
        <Size>3</Size>
    </DimensionDomain>
    <DimensionDomain>
        <ows:Identifier>time</ows:Identifier>
        <Domain>2016-02-23T03:00:00.000Z,2016-02-23T06:00:00.000Z,2016-02-
23T09:00:00.000Z,2016-02-23T12:00:00.000Z,2016-02-23T15:00:00.000Z,2016-02-
23T18:00:00.000Z,2016-02-23T21:00:00.000Z,2016-02-24T00:00:00.000Z,2016-02-
24T03:00:00.000Z,2016-02-24T06:00:00.000Z,2016-02-24T09:00:00.000Z,2016-02-
24T12:00:00.000Z,2016-02-24T15:00:00.000Z,2016-02-24T18:00:00.000Z,2016-02-
24T21:00:00.000Z,2016-02-25T00:00:00.000Z,2016-02-25T03:00:00.000Z,2016-02-
25T06:00:00.000Z,2016-02-25T09:00:00.000Z,2016-02-25T12:00:00.000Z,2016-02-
25T15:00:00.000Z,2016-02-25T18:00:00.000Z,2016-02-25T21:00:00.000Z,2016-02-
26T00:00:00.000Z,2016-02-26T03:00:00.000Z,2016-02-26T06:00:00.000Z,2016-02-
26T09:00:00.000Z,2016-02-26T12:00:00.000Z,2016-02-26T15:00:00.000Z,2016-02-
26T18:00:00.000Z,2016-02-26T21:00:00.000Z,2016-02-27T00:00:00.000Z,2016-02-
27T03:00:00.000Z,2016-02-27T06:00:00.000Z,2016-02-27T09:00:00.000Z,2016-02-
27T12:00:00.000Z,2016-02-27T15:00:00.000Z,2016-02-27T18:00:00.000Z,2016-02-
27T21:00:00.000Z,2016-02-28T00:00:00.000Z</Domain>
        <Size>40</Size>
    </DimensionDomain>
</Domains>
```

- The two time dimensions, time and REFERENCE_TIME are linked with each other (this being a forecast model), assuming the user is interested only in the last forecast, and elevations between 0 and 1000, the client will run a new DescribeDomain to extract the available times.

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=DescribeDomains&Version=1.0.0&Layer=te
st:flexpart_time_series&TileMatrixSet=EPSG:900913&elevation=0.0/1000.0&REFERENCE_TIME=
2016-02-25T00:00:00.000
```

In the following response, the domain values are only returned for the dimensions (i.e. elevation, time, REFERENCE_TIME) overlapping the elevation dimension between 0.0 and 1000.0 and REFERENCE_TIME `2016-02-25T00:00:00.000`

```
<?xml version="1.0" encoding="UTF-8"?><Domains xmlns="http://demo.geo-
solutions.it/share/wmts-multidim/wmts_multi_dimensional.xsd"
xmlns:ows="http://www.opengis.net/ows/1.1">
  <SpaceDomain>
    <BoundingBox CRS="EPSG:4326" minx="-180.0625" miny="-0.0625" maxx="-0.0625"
maxy="89.9375"/>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>elevation</ows:Identifier>
    <Domain>0.0,200.0,400.0,600.0,800.0,1000.0</Domain>
    <Size>6</Size>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>REFERENCE_TIME</ows:Identifier>
    <Domain>2016-02-25T00:00:00.000Z</Domain>
    <Size>1</Size>
  </DimensionDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain>2016-02-25T03:00:00.000Z,2016-02-25T06:00:00.000Z,2016-02-
25T09:00:00.000Z,2016-02-25T12:00:00.000Z,2016-02-25T15:00:00.000Z,2016-02-
25T18:00:00.000Z,2016-02-25T21:00:00.000Z,2016-02-26T00:00:00.000Z,2016-02-
26T03:00:00.000Z,2016-02-26T06:00:00.000Z,2016-02-26T09:00:00.000Z,2016-02-
26T12:00:00.000Z,2016-02-26T15:00:00.000Z,2016-02-26T18:00:00.000Z,2016-02-
26T21:00:00.000Z,2016-02-27T00:00:00.000Z,2016-02-27T03:00:00.000Z,2016-02-
27T06:00:00.000Z,2016-02-27T09:00:00.000Z,2016-02-27T12:00:00.000Z,2016-02-
27T15:00:00.000Z,2016-02-27T18:00:00.000Z,2016-02-27T21:00:00.000Z,2016-02-
28T00:00:00.000Z</Domain>
    <Size>24</Size>
  </DimensionDomain>
</Domains>
```

- Likewise, the user might be interested in elevations between 0.0 and 500.0 and times between 2016-02-23T03:00:00.000Z and 2016-02-23T12:00:00.000Z

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=DescribeDomains&Version=1.0.0&Layer=te
st:flexpart_time_series&TileMatrixSet=EPSG:900913&elevation=0.0/1000.0&time=2016-02-
23T03:00:00.000Z/2016-02-23T12:00:00.000Z
```

In the following response, the domain values are only returned for the dimensions (i.e. elevation, time, REFERENCE_TIME) overlapping the elevation dimension between 0.0 and 1000.0 and the time dimension between 2016-02-23T03:00:00.000Z and 2016-02-23T12:00:00.000Z.

```
<?xml version="1.0" encoding="UTF-8"?>
<Domains>
    <SpaceDomain>
        <BoundingBox CRS="EPSG:4326" minx="-180.125" miny="-90.125" maxx="179.875"
maxy="89.875"/>
    </SpaceDomain>
    <DimensionDomain>
        <ows:Identifier>elevation</ows:Identifier>
        <Domain>0.0,1000.0</Domain>
        <Size>2</Size>
    </DimensionDomain>
    <DimensionDomain>
        <ows:Identifier>REFERENCE_TIME</ows:Identifier>
        <Domain>2016-02-23T00:00:00.000Z</Domain>
        <Size>1</Size>
    </DimensionDomain>
    <DimensionDomain>
        <ows:Identifier>time</ows:Identifier>
        <Domain>2016-02-23T03:00:00.000Z,2016-02-23T06:00:00.000Z,2016-02-
23T09:00:00.000Z,2016-02-23T12:00:00.000Z</Domain>
        <Size>4</Size>
    </DimensionDomain>
</Domains>
```

The client can use the **SpaceDomain** output to avoid requesting empty tiles. In this example the SpaceDomain does not change by the DescribeDomain request as the forecast model is operating over a fixed area.

3. The WMTS client requests GetHistogram for inspecting domain distribution histogram

The client is supposed to have two histogram bars in the UI, showing an elevation and time histogram:

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=GetHistogram&Version=1.0.0&Layer=test:
flexpart_time_series&TileMatrixSet=EPSG:4326&histogram=elevation&time=2016-02-
23T00:00:00.000Z/2016-02-26T00:00:00.000Z
```

In the following response, the domain value histogram is returned for the elevation dimension, which overlaps the time dimension between 2016-02-23T00:00:00.000Z and 2016-02-26T00:00:00.000Z. The histogram shows a growing number of samples over time as the overlap of predicted times between model runs increases.

```
<?xml version="1.0" encoding="UTF-8"?>
<Histogram xmlns="http://demo.geo-solutions.it/share/wmts-
multidim/wmts_multi_dimensional.xsd" xmlns:ows="http://www.opengis.net/ows/1.1">
  <ows:Identifier>time</ows:Identifier>
  <Domain>2016-02-23T03:00:00.000Z/2016-02-26T00:00:00.000Z/PT1H</Domain>

<Values>10,0,10,0,0,10,0,0,10,0,0,10,0,0,10,0,0,10,0,0,10,0,0,20,0,0,20,0,0,20,0,0,20,
0,0,20,0,0,20,0,0,20,0,0,20,0,0,30,0,0,30,0,0,30,0,0,30,0,0,30,0,0,30,0,0,30,0,0,30,0,0,30</V
alues>
</Histogram>
```

This information, along with the DescribeDomain results, helps drive the user navigation in the dataset until a small enough dataset can be located.

4. The WMTS client requests GetFeature for detailed spatial footprint of each layer

Once the user drives the client in an area with few enough datasets, the client can issue a GetFeature and show the footprint of each dataset.

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=GetFeature&Version=1.0.0&Layer=test:fl
expart_time_series&TileMatrixSet=EPSG:900913&reference_time=2016-02-
23T00:00:00.000Z/2016-02-24T00:00:00.000Z&bbox=-180,-90,180,90&elevation=0/200
```

The follow snippet of GetFeature response advertises one available WMTS layer with the dimensions of 'elevation', 'time' and 'REFERENCE_TIME'. The response also advertises the spatial footprint of the WMTS layer in the element of wmts:footprint.

```
<wmts:feature gml:id="TRACER.1701">
    <wmts:footprint>
        <gml:Polygon srsDimension="2"
srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
            <gml:exterior>
                <gml:LinearRing srsDimension="2">
                    <gml:posList>-180.125 -90.125 -180.125 89.875 179.875 89.875
179.875 -90.125 -180.125 -90.125</gml:posList>
                </gml:LinearRing>
            </gml:exterior>
        </gml:Polygon>
    </wmts:footprint>
    <wmts:dimension name="elevation">0.0</wmts:dimension>
    <wmts:dimension name="time">2016-02-23T09:00:00.000Z</wmts:dimension>
    <wmts:dimension name="REFERENCE_TIME">2016-02-23T00:00:00.000Z</wmts:dimension>
</wmts:feature>
```

5. The WMTS client calculates the available map tiles (i.e., tile column/row)

The WMTS client calculates the tile indices (i.e., tile column and row index) from the bounding box (i.e., the spatial footprint of the available layer) by following the algorithm in [OGC 07-057r7] (Annex H, section H.1).

6. The WMTS client requests map tiles

With the calculated tile indices, the WMTS client requests the WMTS tiles with GetTile operation. Since the spatial footprint in the GetFeature response indicates the spatial area where map tiles are available, the calculated tile indices only include non-empty map tiles. This benefits the WMTS clients by preventing the client from accessing empty map tiles.

Use case for inspecting the single domain

In this example the layer `landsat8:B3` contains a dataset with a single time dimension, but scattered in space: given a certain area, the times available will be limited, and then given a certain time, the spatial domain will be restricted.

1. The client looks at the whole dataset

The client starts the interactions by issuing a DescribeDomains and a GetHistogram.

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=DescribeDomains&Version=1.0.0&Layer=la
ndsat8:B3&TileMatrixSet=EPSG:900913
```

The client learns in which area data is actually available by inspecting the spatial domain, and will issue requests only in that area.

```
<?xml version="1.0" encoding="UTF-8"?>
<Domains xmlns="http://demo.geo-solutions.it/share/wmts-
multidim/wmts_multi_dimensional.xsd" xmlns:ows="http://www.opengis.net/ows/1.1">
  <SpaceDomain>
    <BoundingBox CRS="EPSG:4326" minx="-124.579445520273" miny="32.1019898714178"
maxx="-113.928000658897" maxy="42.8255063857218"/>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain>2016-09-01T00:00:00.000Z,2016-09-03T00:00:00.000Z,2016-09-
06T00:00:00.000Z,2016-09-08T00:00:00.000Z,2016-09-10T00:00:00.000Z,2016-09-
12T00:00:00.000Z,2016-09-15T00:00:00.000Z,2016-09-17T00:00:00.000Z,2016-09-
19T00:00:00.000Z,2016-09-22T00:00:00.000Z,2016-09-24T00:00:00.000Z,2016-09-
26T00:00:00.000Z,2016-09-28T00:00:00.000Z,2016-10-01T00:00:00.000Z,2016-10-
14T00:00:00.000Z,2016-10-19T00:00:00.000Z</Domain>
    <Size>16</Size>
  </DimensionDomain>
</Domains>
```

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=GetHistogram&Version=1.0.0&Layer=lands
at8:B3&TileMatrixSet=EPSG:900913&histogram=time&resolution=P1D
```

The client also issues a GetHistogram, setting the time resolution to one day given the long time span returned by DescribeDomain, and displays the satellite scene distribution over time in a histogram.

```
<?xml version="1.0" encoding="UTF-8"?><Histogram xmlns="http://demo.geo-
solutions.it/share/wmts-multidim/wmts_multi_dimensional.xsd"
xmlns:ows="http://www.opengis.net/ows/1.1">
  <ows:Identifier>time</ows:Identifier>
  <Domain>2016-09-01T00:00:00.000Z/2016-10-19T00:00:00.000Z/P1D</Domain>

<Values>2,2,0,0,2,0,3,0,3,0,2,0,0,3,0,2,0,1,0,0,1,0,4,0,3,0,1,0,0,4,0,0,0,0,0,0,0,0,0,
0,0,0,2,0,0,0,0,3</Values>
</Histogram>
```

2. The user focuses on a specific time range

The user decides to focus on a specific day, e.g. "2016-10-19T00:00:00.000Z", the client thus repeats the DescribeDomain and GetHistogram for that specific day.

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=DescribeDomains&Version=1.0.0&Layer=la
ndsat8:B3&TileMatrixSet=EPSG:900913&time=2016-10-19T00:00:00.000Z
```

The client learns the area with available data is significantly smaller and adjusts the tile requests accordingly.

```
<?xml version="1.0" encoding="UTF-8"?>
<Domains xmlns="http://demo.geo-solutions.it/share/wmts-
multidim/wmts_multi_dimensional.xsd" xmlns:ows="http://www.opengis.net/ows/1.1">
  <SpaceDomain>
    <BoundingBox CRS="EPSG:4326" minx="-121.193217040604" miny="33.4928073850479"
maxx="-117.69125563178" maxy="38.5620181693094"/>
  </SpaceDomain>
  <DimensionDomain>
    <ows:Identifier>time</ows:Identifier>
    <Domain>2016-10-19T00:00:00.000Z</Domain>
    <Size>1</Size>
  </DimensionDomain>
</Domains>
```

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=GetHistogram&Version=1.0.0&Layer=lands
at8:B3&TileMatrixSet=EPSG:900913&histogram=time&resolution=PT1H&time=2016-10-
19T00:00:00.000Z
```

The client also issues a GetHistogram, discovering the are 3 images available for that time value (the time dimension actually being an acquisition date).

```
<?xml version="1.0" encoding="UTF-8"?><Histogram xmlns="http://demo.geo-
solutions.it/share/wmts-multidim/wmts_multi_dimensional.xsd"
xmlns:ows="http://www.opengis.net/ows/1.1">
  <ows:Identifier>time</ows:Identifier>
  <Domain>2016-10-19T00:00:00.000Z/2016-10-19T00:00:00.000Z/PT1H</Domain>
  <Values>3</Values>
</Histogram>
```

The user sees only 3 features are available, and asks the client do display the satellite scene footprints, the client issues a GetFeature request.

Request:

```
http://cloudsdi.geo-
solutions.it/geoserver/gwc/service/wmts?REQUEST=GetFeature&Version=1.0.0&Layer=landsat
8:B3&TileMatrixSet=EPSG:900913&time=2016-10-19T00:00:00.000Z
```

The client issues the request and display the footprints, along with their dimension values, on screen.

```
<?xml version="1.0" encoding="UTF-8"?>
<wmts:FeatureCollection xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" xmlns:wmts="http://www.opengis.net/wmts/1.0">
  <wmts:feature gml:id="B3.36">
    <wmts:footprint>
      <gml:Polygon xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:xlink="http://www.w3.org/1999/xlink" srsDimension="2"
srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
        <gml:exterior>
          <gml:LinearRing srsDimension="2">
            <gml:posList>-120.40187133757576 36.36560140384616 -120.40187133757576
38.562018169309354 -117.6912556317801 38.562018169309354 -117.6912556317801
36.36560140384616 -120.40187133757576 36.36560140384616</gml:posList>
          </gml:LinearRing>
        </gml:exterior>
      </gml:Polygon>

    </wmts:footprint>
    <wmts:dimension name="time">2016-10-19T00:00:00.000Z</wmts:dimension>
  </wmts:feature>
  <wmts:feature gml:id="B3.37">
    <wmts:footprint>
      <gml:Polygon xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:xlink="http://www.w3.org/1999/xlink" srsDimension="2"
srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
        <gml:exterior>
          <gml:LinearRing srsDimension="2">
            <gml:posList>-120.80220778578197 34.92951790502201 -120.80220778578197
37.13622680705192 -118.12945559206337 37.13622680705192 -118.12945559206337
34.92951790502201 -120.80220778578197 34.92951790502201</gml:posList>
          </gml:LinearRing>
        </gml:exterior>
      </gml:Polygon>

    </wmts:footprint>
    <wmts:dimension name="time">2016-10-19T00:00:00.000Z</wmts:dimension>
  </wmts:feature>
  <wmts:feature gml:id="B3.38">
    <wmts:footprint>
      <gml:Polygon xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```
xmlns:gml="http://www.opengis.net/gml" xmlns:sch="http://www.ascc.net/xml/schematron"
xmlns:xlink="http://www.w3.org/1999/xlink" srsDimension="2"
srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
        <gml:exterior>
          <gml:LinearRing srsDimension="2">
            <gml:posList>-121.19321704060368 33.49280738504794 -121.19321704060368
35.708652302764484 -118.55954944311665 35.708652302764484 -118.55954944311665
33.49280738504794 -121.19321704060368 33.49280738504794</gml:posList>
          </gml:LinearRing>
        </gml:exterior>
      </gml:Polygon>

    </wmts:footprint>
    <wmts:dimension name="time">2016-10-19T00:00:00.000Z</wmts:dimension>
  </wmts:feature>
</wmts:FeatureCollection>
```

Appendix A: Revision History

Table 17. Revision History

Date	Release	Editor	Primary clauses modified	Descriptions
April 10, 2016	Lingjun Kang Liping Di Eugene Yu	.1	all	initial ER version
June 25, 2016	Lingjun Kang Liping Di Eugene Yu	.2	all	First Complete ER Draft
September 30, 2016	Lingjun Kang Liping Di Eugene Yu	.3	all	Initial ER submission
October 31, 2016	Lingjun Kang Liping Di Eugene Yu	.4	all	ER submission

Appendix B: Bibliography

[1] OGC: OGC 06-121r9, OGC® Web Services Common Standard. (2010).

[2] OGC: OGC 06-042, OpenGIS Web Map Service (WMS) Implementation Specification, version 1.3.0. (2006).

[3] OGC: OGC 07-057r7, OpenGIS Web Map Tile Service Implementation Standard 1.0.0. (2010).

[4] OGC: OGC 09-025r2, OGC® Web Feature Service 2.0 Interface Standard, version 2.0.2. (2014).

[5] ISO: ISO 8601, Data elements and interchange formats — Information interchange — Representation of dates and times. (2004).

[6] Reading e-Science Centre at the University of Reading, UK: ncWMS User Guide