Testbed-12 Multi-Tile Retrieval ER

Table of Contents

Publication Date: 2017-06-16

Approval Date: 2017-03-23

Posted Date: 2016-12-26

Reference number of this document: OGC 16-049r1

Reference URL for this document: http://www.opengis.net/doc/PER/t12-A077

Category: Public Engineering Report

Editor: Joan Masó

Title: Testbed-12 Multi-Tile Retrieval ER

OGC Engineering Report

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

With the consolidation of tiling services and the increasing number of instances implementing the WMTS standard, there is a need for having a way to transfer a collection of tiles from one service to another. This might also be useful to transfer all necessary tiles from a WMTS service to a GeoPackage. Currently the only available solution is a client that is able to resolve the identifiers of the tiles needed and that builds a WMTS independent request for each tile. This ER explores different solutions that are more appropriate depending on how many tiles we need to move and the final application of them. Some of the proposed solutions involve changes in the WMTS standard and the use of a WPS. The WPS standard also shows some limitations and extensions that should be addressed.

In essence all solutions should describe two things: A request that contains a filter to a collection of tiles filling regions of the space and a multipart response that contains the tiles preferably in a single package. Depending on the proposed architecture, these tasks are done directly in the client, in the WMTS server or in an intermediate WPS.

Business Value

This ER has the objective of simplifying the transference of collections of tiles from one service to another. It also proposes alternatives that can be useful to download all the necessary tiles for ingesting them in a GeoPackage. Currently this only can be done by requesting one tile at a time. This results in developers having to program routines to download the tiles in the client side. It also requires opening an HTTP connection for every tile, spending a lot of time in setting the connections instead of transferring the necessary data. A more efficient way of doing this is required.

Sometimes it is also necessary to be able to refer to all tiles that are included in a BBOX (e.g. in a viewport) as a single URL. This gap was found in OWS Context and GeoPackage and it is still not possible.

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

This ER is important for the GeoPackage SWG and the OGC in general because it provides a solution for building a GeoPackage that contains tiles easily from a WMTS in a single request.

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

Many times, the data inside a GeoPackage is obtained from web services. It could be good to have a manifest that contains the original service requests. This is particularly useful in a intermittently connected environment. This way the client using a static GeoPackage can request the data form the services to check for updated information and even update the GeoPackage.

Keywords

ogcdocs, testbed-12, WMTS, Tile, GeoPackage

Proposed OGC Working Group for Review and Approval

GeoPackage SWG

Chapter 1. Introduction

1.1. Scope

This OGC® document proposes ways to get more that one tile at the same time for services.

This OGC® document describes an extension for WMTS that proposes the inclusion of a operation called GetTiles. This OGC® document also describes an extension for WPS that proposes a way to be able to get access to partial results while the WPS process is still running (this is complemented by an additional extension to allow additional inputs while the process is still running).

This ER does not discusses security issues derived from having a WMTS secured that receives chained/cascaded requests for other servers.

This OGC® document is applicable to WMTS and WPS services and to GeoPackages based on raster tiles.

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**
Joan Masó	UAB-CREAF

1.3. Future Work

Experiment with a WMTS GetTiles request and submit the results of this experimentation to the WMTS standards group for consideration.

In case that a WPS is acting as a client of a WTMS is secure protected, consider a solution with front-end WPS that is able to deal with security information that is needed for invoking a secured WMTS in the back-end.

Experiment with a WPS that can deliver preliminary results and accept new input parameters while still running. In a future Testbed, experiment with two WPS instances that can exchange information while both running complementary processes. For example, a WPS was running, but now is waiting for a new input that another WPS will deliver as a preliminary result. A classic example for this requirement is the interaction between a hydrological model and an atmospheric model running a water cycle simulation. Both models need information coming from the preliminary results of the other model. If they are implemented as WPS processes, adding the capability to exchange intermediate results will make it possible to have the two models in two independent WPS servers and still be able to exchange information. In this ER, we aim at a WPS process that is able to retrieve tiles from a WMTS one by one and is able to deliver a partial GeoPackage while still retrieving more 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 14-065, OGC WPS 2.0 Interface Standard
- OGC 07-057r7, OGC Web Map Tile Service Standard
- OGC 06-121r9, OGC® Web Services Common Standard

NOTE: This OWS Common Standard contains a list of normative references that are also applicable to this Implementation Standard.

.

Chapter 3. Conventions

3.1. 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].

3.2. Data dictionary tables

The UML model data dictionary is specified herein in a series of tables. The contents of the columns in these tables are described in Table 1.

Table 2. Contents of data dictionary tables

Column title	Column contents
Names (left column)	Two names for each included parameter or association (or data structure). The first name is the UML model attribute or association role name. The second name uses the XML encoding capitalization specified in Subclause 11.6.2 of [OGC 06-121r3]. The name capitalization rules used are specified in Subclause 11.6.2 of [OGC 06-121r3]. Some names in the tables may appear to contain spaces, but no names contain spaces.
Definition (second column)	Specifies the definition of this parameter (omitting unnecessary words such as “a”, “the”, and “is”). If the parameter value is the identifier of something, not a description or definition, the definition of this parameter should read something like “Identifier of TBD”.
Data type and value (third column) or Data type (if are no second items are included in rows of table)	Normally contains two items: The mandatory first item is often the data type used for this parameter, using data types appropriate in a UML model, in which this parameter is a named attribute of a UML class. Alternately, the first item can identify the data structure (or class) referenced by this association, and references a separate table used to specify the contents of that class (or data structure). The optional second item in the third column of each table should indicate the source of values for this parameter, the alternative values, or other value information, unless the values are quite clear from other listed information.

Column title	Column contents
Multiplicity and use (right or fourth column) or Multiplicity (if are no second items are included in rows of table)	Normally contains two items: The mandatory first item specifies the multiplicity and optionality of this parameter in this data structure, either “One (mandatory)”, “One or more (mandatory)”, “Zero or one (optional)”, or “Zero or more (optional)”. The second item in the right column of each table should specify how any multiplicity other than “One (mandatory)” shall be used. If that parameter is optional, under what condition(s) shall that parameter be included or not included? If that parameter can be repeated, for what is that parameter repeated?

When the data type used for this parameter, in the third column of such a table, is an enumeration or code list, all the values specified shall be listed, together with the meaning of each value. When this information is extensive, these values and meanings should be specified in a separate table that is referenced in the third column of this table row.

The data type of many parameters, in the third table column, is specified as “Character String type, not empty”. In the XML Schema Documents specified herein, these parameters are encoded with the xsd:string type, which does NOT require that these strings not be empty.

(These conditions may seem obvious to you, but they are rarely obvious to most readers.)

The contents of these data dictionary tables are normative, including any table footnotes.

(This means that these table footnotes should use the normative verbs "shall", "should", "may", and "can", as defined in Subclause 5.3 "Document terms and definitions" of OWS Common [OGC 06-121r9].)

Chapter 4. Overview

When thinking about use cases where a multitile retrieval is necessary, two *big data* ones (mostly requiring a big number of tiles) come to our mind:

- Use case 1: A client application wants to store a subset of a tiled layer coming from a WMTS for its later use offline
- Use case 2: A GeoPackage needs to be created containing a layer with tiles that will be provided by a WMTS.

In addition, there are other use cases where multitile retrieval is useful even if the number of tiles involved is limited in number:

- Use case 3: A client application wants to reduce the number of server requests by requesting all the tiles needed to present a new view to a user at once;
- Use case 4: A OWS Context document wants to use a single link to a set of tiles that are useful to present a common operational picture.

It is important to note that the time required to prepare a set of tiles can be long, so asynchronous requests need to be taken into consideration when designing multitile approaches.

This ER examines some possibilities depending on the use case:

- Provide a transversal way to encapsulate more than one request to a service in a multi-request approach: KVP encodings are not designed to do this but a POST operation can send a document with a list of requests inside that can be responded by the server in a single MIME multipart file (RFC1341).
- A WPS associated with the dataset: A WMTS service and a WPS are associated to the same dataset. A WPS profile can offer a solution in the form of a process that is able to generate tiles and encapsulate them in a MIME multipart file or in a GeoPackage. This solution offers the asynchronous advantage.

4.1. Use cases

In this subclause we describe the previously mentioned use cases but in a more technical language. The textual description of the cases is accompanied by UML sequence diagrams. The use cases are classified by the volume of tiles involved into *big-data* use cases (making use of the *big-data* expression that is so popular this days), a more modest ones in the volume of tiles that are called *small-volume-of-data* (inventing an expression that wants to refer to a more modest volume of data), and a fully scalable *very-big-data* use case.

4.1.1. Small-volume-of-data use cases

These use cases do not consider the *big-data* approach and are only addressing the multitile request need for a modest number of tiles.

GetTiles new operation in WMTS:

We introduce a new operation into WMTS for retrieving more that one tile at the same time. This operation works in a synchronous mode like the GetTile operation: This solution was initially addressed by the subsection 7 of the OWS-9 Map Tiling Methods Harmonization IP Engineering Report (OGC 12-157r1). This has been reproduced in the WMTS extension section for convenience (with some corrections from the original text).

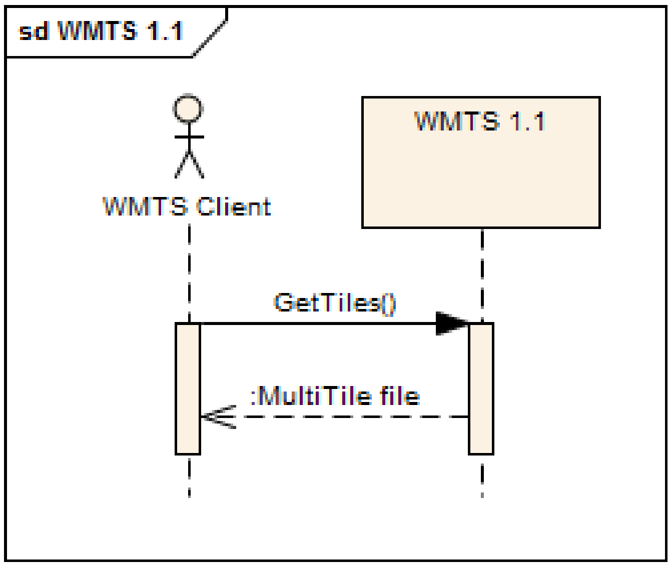

Figure 1. WMTS GetTiles UML sequence diagram

A single GetTiles request will NOT work in a big-data use case if it is synchronous because it can take too much to serve the multitile response. Other alternatives are proposed later.

A synchronous WPS executing a sequence of WMTS GetTile requests

In this approach, a common WMTS 1.0 is requested several times by a WPS that internally decomposes an area at different scale denominators in a sequence of several tiles requests and accumulates the result in a multipart file format that is delivered at the end of the execution. The WPS is responding synchronously so the client will wait to the end of the WPS execution to the the package of tiles.

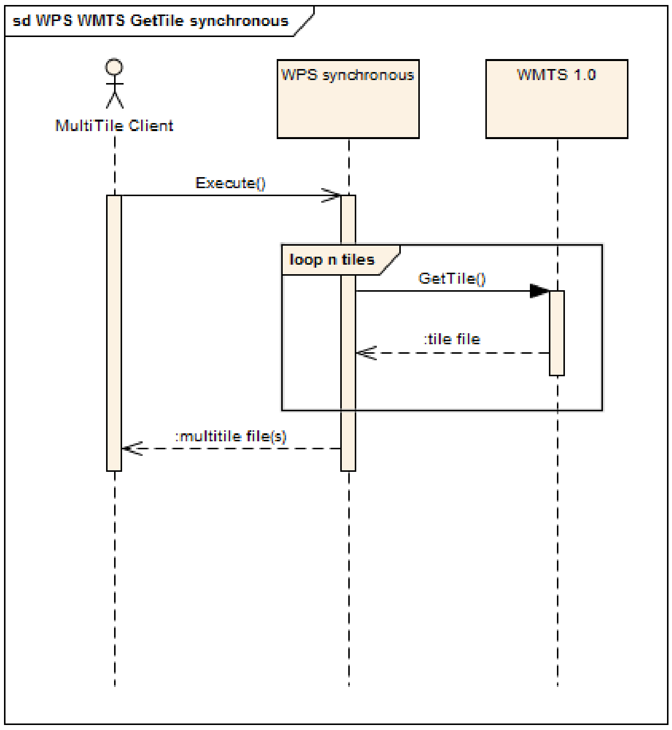

Figure 2. WMTS GetTile with a synchronous WPS UML sequence diagram

This use case does not require any modifications or extensions neither to the WMTS nor the WPS standards. This approach was implemented in practice in the Testbed 12.

4.1.2. Big-data use cases

An asynchronous WPS executing a sequence of WMTS GetTile requests:

In this approach, a common WMTS 1.0 is requested several times by a WPS that internally decomposes an area at different scale denominators in a sequence of several tiles requests and accumulates the result in a multipart file format that is delivered at the end of the execution. The WPS is responding asynchronously so the client can do other tasks while checking from time to time the status of the WPS execution and informing the user until the execution actually ends and the user can get access to the result as a single package.

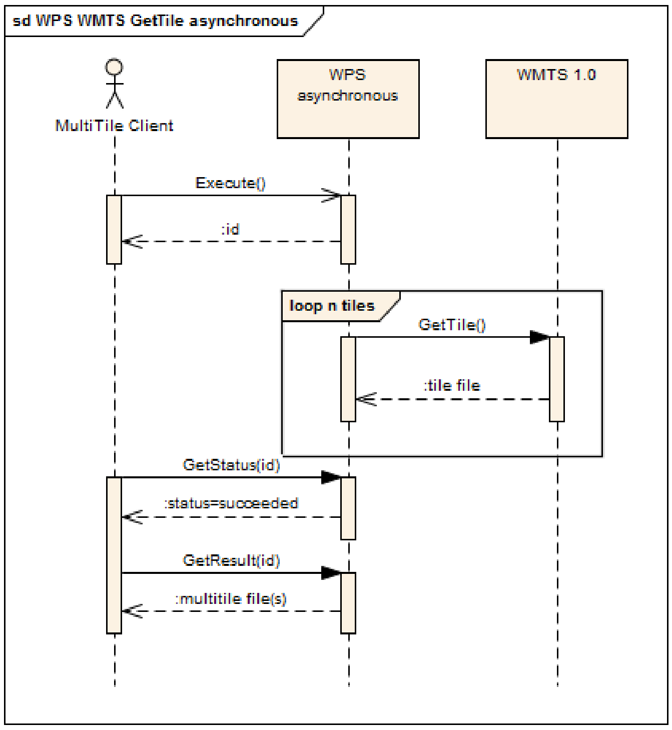

Figure 3. WMTS GetTile with an asyncronous WPS UML sequence diagram

This use case does not require any modifications or extensions neither to the WMTS nor the WPS. This approach was implemented in practice in the Testbed 12.

An asynchronous WPS that requests all tiles one by one using GetTile will require too many individual requests in the back-end and is not considered scalable indefinitely. This is one of the reasons for proposing a GetTiles operation to be included in the WMTS standard.

Add asynchronous behaviour to a WMTS GetTiles requests:

We assume that a synchronous response is not acceptable (too long wait) so we propose to use a WMTS asynchronous that uses the same principle that as described by Peter Vretanos (Cubewex) proposing for WFS and reported in the Testbed 12 - Web Feature Service Synchronization Engineering Report.

There are two proposed approaches:

- A notification that transports the result directly.
- A notification that provides a URL that can be requested later

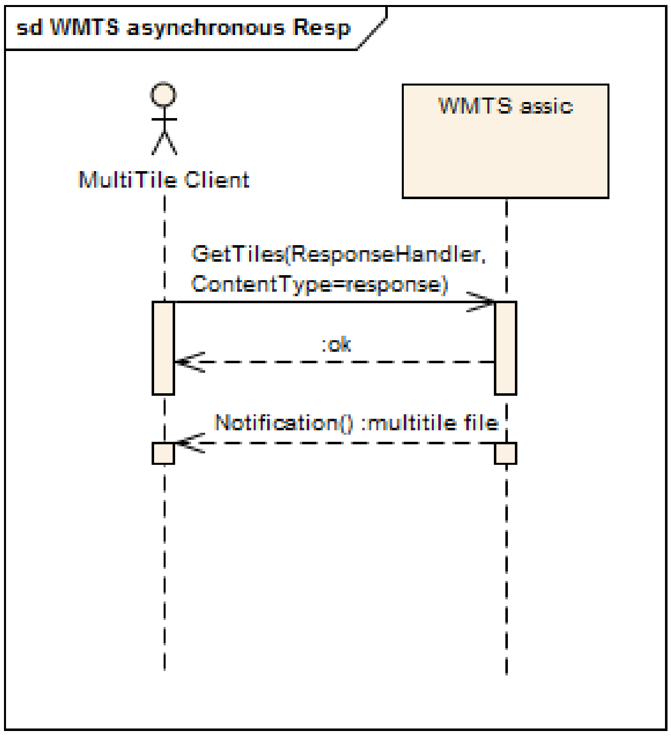
sd WMTS asynchronous Resp
MultiTile Client
WMTS assic
GetTiles(ResponseHandler, ContentType=response)
:ok
Notification() :multitile file

Figure 4. WMTS GetTiles UML sequence diagram for an asynchronous request returning the multitile file directly

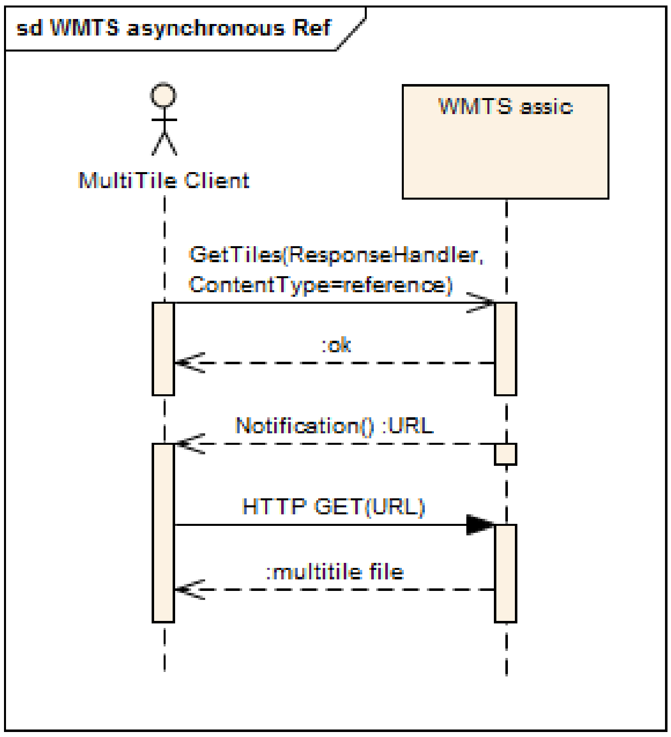

Figure 5. WMTS GetTiles UML sequence diagram for an asynchronous request returning a reference

4.1.3. Very big-data use cases:

Combining a modified WMTS with an asynchronous WPS

A combination of a WPS process that calls a WMTS extended with the GetTiles request.

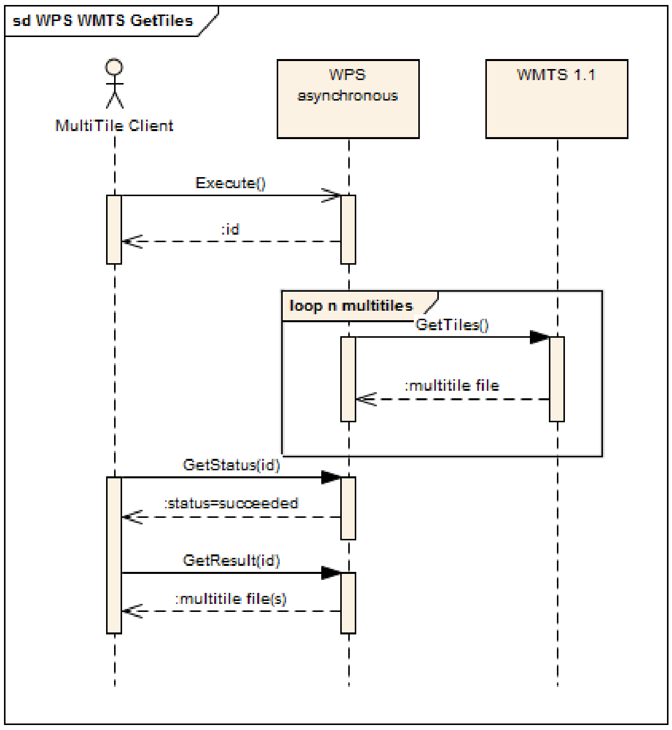

Figure 6. WPS WMTS GetTiles UML sequence diagram combination

Combining a modified WMTS with an asynchronous WPS with intermediate results.

With the proposed solution above, the client needs to wait until the end of the WPS execution to get all tiles as a single package and this could be hours or even days for a big area. Indeed, the current WPS standard does not allow for delivering incremental results (where the client could start exploring/showing the results before the process reaches completion, making use of the available bandwidth). We can consider this behavior a limitation of the WPS. For that reason, in a following section we will present a modification of the WPS to support delivery of intermediate results (or even interacting with other processes in the same way that Open Modelling Interface standard [OpenMI] enables the runtime exchange of data between process simulation models and also between models and other modelling tools such as databases and analytical and visualization applications).

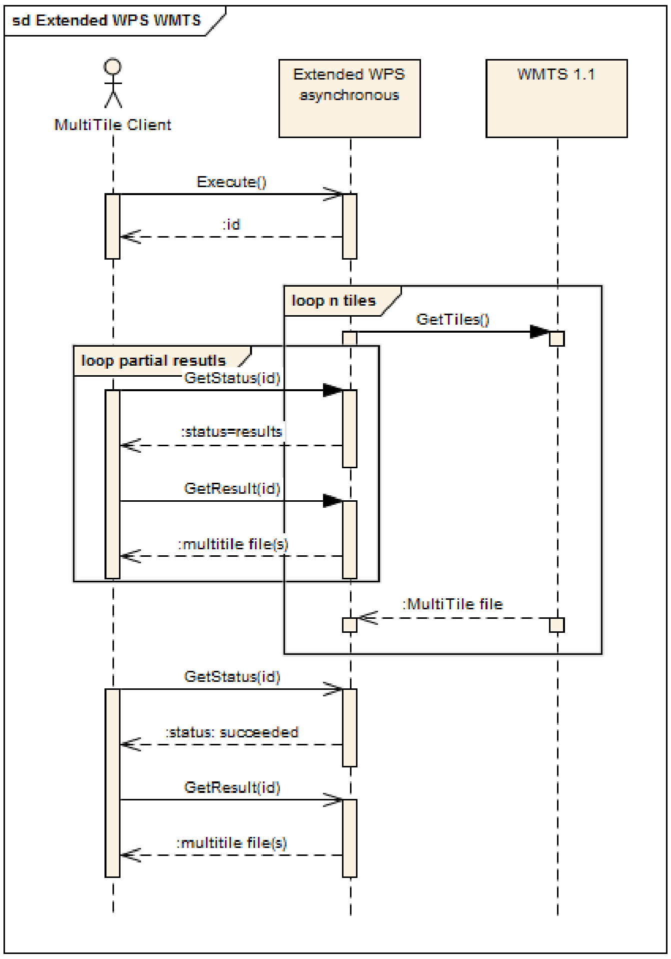

Figure 7. Extended WPS and WMTS combination UML sequence diagram

This proposal is further elaborated in the WPS partial results extension

Chapter 5. WMTS GetTiles extension

This solution was initially addressed by subsection 7 of the OWS-9 Map Tiling Methods Harmonization IP Engineering Report (OGC 12-157r1). This is reproduced in this section for convenience (with some corrections and amendments from the original text).

This section describes an extension of WMTS that defines a GetTiles operation. The objective of this new operation is for the client to be able to request and get back all tiles that cover a particular region (BBOX) at the needed scales (TileMatrices) in a single operation. This is what a KVP request example will look like:
http://www.opengis.uab.es/cgi-bin/ICCTiled/MiraMon.cgi?
SERVICE=WMTS&VERSION=1.0.0&REQUEST=GetTiles&Layer=Topo250k_Vers5_ICC&
Style=&TileMatrixSet=Cat_topo250k_v5_EPSG23031&TileMatrices=200m&
BBox=355000,4539000,475000,4619000&Format=image/jpeg&
CollectionFormat=application/xml&inclusion=linked

The return shall be a multi-part file.

Examples of use cases supporting the need for this approach are:

- A client that is designed for WMS that wants a simple way to receive the tiles that fit in a WMS BBOX.
- A client so simple that is not able to calculate the tile positions in the screen, and needs the server to tell how to arrange the tiles.
- Get an XML document that can be transformed into an XHTML visualization of the required tiles (applying XSLT).
- A client that wants to harvest some/all tiles of a server layer and store them locally for viewing them offline latter in a faster and efficient way.
- A GeoPackage Manifest needs to link tiles with the corresponding WMTS service using a single URL and mimic what is currently possible with WCS or WFS.

5.1. GetTiles Operation

5.1.1. GetTiles Operation Request

A request to perform the GetTiles operation SHALL use the data structure specified in the following table. This table also specifies the data type of values, and multiplicity of each listed parameter, plus the default server behavior when an optional parameter is not included in the operation request.

Table 3. Parameters in GetTiles request operation

Names	Definition	Data type and values	Multiplicity and use
service Service	Service type identifier	Character String type, not empty, SHALL be "WMTS"	One (mandatory)

Names	Definition	Data type and values	Multiplicity and use
request Request	Operation name	Character String type, not empty SHALL be "GetTiles"	One (mandatory)
version Version	Standard version for operation	Character String type, not empty SHALL contain "1.1.0"	One (mandatory)
layer Layer	Layer identifier	Character String type, not empty. identifier that is defined in the ServiceMetadata document	One (mandatory)
style Style	Style identifier	Character String type, not empty. identifier that is defined in the ServiceMetadata document	One (mandatory)
format Format	Output format of the tiles	Character String type, not empty. value that is defined in the ServiceMetadata document	One (mandatory)
Other sample dimensions [a]	Value allowed for this dimension	Character String type, not empty. a single value from a list or a range defined in the ServiceMetadata document	Zero or one (optional)
tileMatrixSet TileMatrixSet	TileMatrixSet identifier	Character String type, not empty. identifier that is defined in the ServiceMetadata document	One (mandatory)
tileMatrices TileMatrices	TileMatrix identifier list b	Character String type, not empty. a list of coma separated values for tile matrices identifiers that are defined in the ServiceMetadata document	Zero or many (optional)[c]
bBox BBox	Bounding box	Four Doubles type. a Bounding Box that defines the area to be recovered. [d]	Zero or one (optional)[e]

Names	Definition	Data type and values	Multiplicity and use
width Width	Display width	Positive Integer type. The horizontal size of the visual display in pixels	Zero or one (optional)[g]
height Height	Display height	Positive Integer type. The vertical size of the visual display in pixels	Zero or one (optional)[g]
collectionFormat CollectionFormat	Output format of the tile collection	Character String type, not empty. value that is defined in the ServiceMetadata document	One (mandatory)
inclusion Inclusion	Tile inclusion	Enumeration type "embedded" or "linked" [g]	Zero or one (optional). Default value is “embedded"
[a] The name of this parameter is, in fact, the identifier of the dimension specified in the ServiceMetadata. See WMS 1.3.0 Annex C for reference. This parameter appears once for each dimension specified for this Layer in the ServiceMetadata document. [b] This will be the identifier for the TileMatrix of the desired scale denominator for the TileMatrixSet requested. [c] If the parameter is not provided, all TileMatrices of the TileMatrixSet will be returned. [d] All tiles that are entirely or partially contained in this bounding box will be returned. In other words, the bounding box does not have to coincide with tile border positions. E.g. it could be the screen veiwport in world coordinates that can be converted to screen coordinates if the TileMatrices has a single identifier. [e] If not provided, all tiles of the TileMatrices will be returned. [f] If included is “embedded”, MIME multi-part document or GeoPackage are recommended as an output format. If Included is “linked”, application/text” is recommended. [g] “linked” means the client request tiles URLs (in KVP/GetTile or RESTful/tile notation) that will be requested later. “embedded” means that tiles will be embedded in the document format.			

5.1.2. GetTiles operation response

The response of a GetTile operation is a text document describing the returned tiles and binary tiles linked to it or embedded in a multi-part format.

The text format enumerates a TileCollection or a TileCollection data type elements (see the following table).

Table 4. Parts of Tile data structure in a TileCollection

Names	Definition	Data type and values	Multiplicity and use
identifier Identifier	tile identifier within the document	ows:CodeType, as adaptation of MD_Identifier class ISO 19115	Zero or one (optional) [a]

Names	Definition	Data type and values	Multiplicity and use
tileURL TileURL	A URL of the tile	AnyURI, not empty.	One (mandatory) [b]
tileMatrix TileMatrix	TileMatrix identifier	Character String type, not empty. a value for tile matrices identifiers that are defined in the ServiceMetadata document	One (mandatory)
tileRow TileRow	Row index in the tile matrix	Non negative integer type. value between 0 and MatrixHeight -1 of this tile matrix defined in the ServiceMetadata document	One (mandatory)
tileCol TileCol	Column index of tile matrix	Non negative integer type. a value between 0 and MatrixWidth -1 of this tile matrix defined in the ServiceMetadata document	One (mandatory)
width Width	Width of the tile	Positive Integer type. width of the tile in pixels	Zero or one (optional)
height Height	Height of the tile	Positive Integer type. height of the tile in pixels	Zero or one (optional)
top Top	Vertical position	Positive Integer type. it is the vertical position from the visual display in pixels. A negative value means that the left side of the tile is outside the top-left corner of the display and should be cut when displayed	Zero or one (optional) [c]
left Left	Horizontal position	Positive Integer type. it is the horizontal position from the visual display in pixels. A negative value means that the left side of the tile is outside the top-left corner of the display and should be cut	Zero or one (optional) [c]

Names	Definition	Data type and values	Multiplicity and use
[a] If the format does not support file identification or name, the order in the file will be considered. [b] If tiles are linked, it will be an external link. If the tiles are embedded, it will be a link to the internal data (e.g. in a MIME multi-part file). [c] Populate only if the number of TileMatrices requested is one.			

This format can be encoded in an XML file, a JSON file or even an HTML page. The file includes URL links or links to parts of a multi-part file. Examples of a multi-part file are MIME multi-part, Open Package Convention file, a ZIP file, or a GeoPackage file.

5.1.3. GetTiles exceptions

When a WMTS Server encounters an error while performing a GetTiles operation, it SHALL return an exception report message as specified in subclause 7.4.1 of OWS Common [OGC 06-121r3]. The allowed standard exception codes SHALL include those listed in following table. For each listed exceptionCode, the contents of the “locator” parameter value SHALL be as specified in the right column.

NOTE To reduce the need for readers to refer to other documents, four values listed below are copied from Table 8 in subclause 7.4.1 of OWS Common [OGC 06-121r3].

Table 5. Exception codes for GetTile operation

exceptionCode value	Meaning of code	“locator” value
OperationNotSupported	Request for an operation that is not supported by this server.	Name of operation not supported
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter.	Name of missing parameter
InvalidParameterValue	Operation request contains an invalid parameter value.	Name of parameter with invalid value
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception.	None, omit “locator” parameter

If the client sends a GetTiles request using unknown parameters (for example time, elevation or any other dimension that are not advertised in the ServiceMetadata document) these unknown parameters SHALL be ignored by the server and will not cause an exception to be generated.

A Bounding Box completely out of the area covered by the tiles will not be considered an exception, thus it will return an empty tile collection.

When a WMTS server responds with an ExceptionReport and the report is transmitted via HTTP, the WMTS server should set the status code of the HTTP response to the corresponding value for the given exceptionCode values, as shown in the following Table. When the ExceptionReport contains more than one Exception, then the HTTP status code value should be based upon the exceptionCode of the first Exception in the ExceptionReport.

```
 -- HTTP exception codes and
meanings on GetTile operation
```

Table 6. HTTP exception codes and meanings on GetTile operation

exceptionCode value	HTTP Status Code	Message
OperationNotSupported	501	Not implemented
MissingParameterValue	400	Bad request
InvalidParameterValue	400	Bad request
NoApplicableCode	500	Internal server error

5.1.4. GetTiles KVP linked request example

This example covers a Bounding Box (355000,4619000) x (475000,4539000) (EPSG:23031) and a width of 600 and a 480 height at 200m of pixel size.

A Landsat ortophotographic map of the region can be obtained from a WMS server using this request:
http://www.opengis.uab.cat/cgi-bin/SatCat/MiraMon.cgi?
SERVICE=WMS&VERSION=1.1.1&REQUEST=GetMap&
SRS=EPSG:23031&BBOX=355000,4539000,475000,4619000&
WIDTH=600&HEIGHT=400&LAYERS=TotCatalunyaED50&
FORMAT=image/jpeg&STYLES=opti_fals&TIME=2011-03

A WMTS GetTile request for a single 640x480 tile of a Catalan topographic map looks like this:
http://www.opengis.uab.es/cgi-bin/ICCTiled/MiraMon.cgi?
SERVICE=WMTS&VERSION=1.0.0&REQUEST=GetTile&
Layer=Topo250k_Vers5_ICC&Style=&
TileMatrixSet=Cat_topo250k_v5_EPSG23031&TileMatrix=200m
&TileRow=1&TileCol=1&Format=image/jpeg

An example of the proposed GetTiles operation request, encoded in KVP, resembles a mixture of the previous two ones. Instead of specifying the *TileRow* and *TileCol* (from GetTile), we have a *BBOX* (from GetMap) and *TileMatrices* as multiplicity more than one. In addition a *CollectionFormat* and *Inclusion* parameter could ba added. The example encoded for HTTP GET is:
http://www.opengis.uab.es/cgi-bin/ICCTiled/MiraMon.cgi?
SERVICE=WMTS&VERSION=1.0.0&REQUEST=GetTiles&
Layer=Topo250k_Vers5_ICCStyle=&
TileMatrixSet=Cat_topo250k_v5_EPSG23031&
TileMatrices=200m&BBox=355000,4539000,475000,4619000&
Format=image/jpeg&CollectionFormat=application/xml&
inclusion=linked

5.1.5. GetTiles linked response example

A GetTiles operation response for the GetTiles request example is exemplified in the following XML document.

GetTiles response example with URL links

```
<?xml version="1.0"
encoding="UTF-8"?>

<?xml version="1.0" encoding="UTF-8"?>
<TileCollection
        xmlns="http://www.opengis.net/wmts/1.0/get_tiles"
        xmlns:wmts_gt="http://www.opengis.net/wmts/1.0/get_tiles"
        xmlns:ows="http://www.opengis.net/ows/1.1"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/wmts/1.0/get_tiles
 ..\wmtsGetTiles_response.xsd
    http://www.opengis.net/ows/1.1
 ../../../ows/1.1.0/owsAll.xsd">">
    <tile>
        <ows:Identifier>200m_1_0</ows:Identifier>
        <fileURL xlink:href=
"http://www.opengis.uab.es/SITiled/ICC/Topo250k_Vers5_ICC/default/Cat_topo250k_v5_EPSG
23031/200m/1/0.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>1</tileRow>
        <tileCol>0</tileCol>
        <width>640</width>
        <height>480</height>
        <top>-185</top>
        <left>-485</left>
    </tile>
    <tile>
        <ows:Identifier>200m_1_1</ows:Identifier>
        <fileURL xlink:href=
"http://www.opengis.uab.es/SITiled/ICC/Topo250k_Vers5_ICC/default/Cat_topo250k_v5_EPSG
23031/200m/1/1.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>1</tileRow>
        <tileCol>1</tileCol>
        <width>640</width>
        <height>480</height>
        <top>-185</top>
        <left>155</left>
    </tile>
    <tile>
        <ows:Identifier>200m_2_0</ows:Identifier>
        <fileURL xlink:href=
"http://www.opengis.uab.es/SITiled/ICC/Topo250k_Vers5_ICC/default/Cat_topo250k_v5_EPSG
```

```
23031/200m/2/0.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>2</tileRow>
        <tileCol>0</tileCol>
        <width>640</width>
        <height>480</height>
        <top>295</top>
        <left>-485</left>
    </tile>
    <tile>
        <ows:Identifier>200m_2_1</ows:Identifier>
        <fileURL xlink:href=
"http://www.opengis.uab.es/SITiled/ICC/Topo250k_Vers5_ICC/default/Cat_topo250k_v5_EPSG
23031/200m/2/1.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>2</tileRow>
        <tileCol>1</tileCol>
        <width>640</width>
        <height>480</height>
        <top>-295</top>
        <left>-155</left>
    </tile>
</TileCollection>
```

This response is linking to 4 WMTS tiles that are deliberately not square tiling size (640x480), with the origin at (258007, 4751992) and 200m of pixel size, which covers the requested Bounding Box (355000,4619000) x (475000,4539000)

5.1.6. GetTiles KVP embedded request example

http://www.opengis.uab.es/cgi-bin/ICCTiled/MiraMon.cgi?
SERVICE=WMTS&VERSION=1.0.0&REQUEST=GetTiles&
Layer=Topo250k_Vers5_ICC&Style=&
TileMatrixSet=Cat_topo250k_v5_EPSG23031&TileMatrix=200m&
BBox=355000,4539000,475000,4619000&Width=600&Height=400&
Format=image/jpeg&CollectionFormat=Multipart/Related&
inclusion=linked

5.1.7. GetTiles embedded response example

A GetTiles operation response for the GetTiles request example will be the following MIME Multi-part document fragment (contents of the tile parts are omitted):

GetTiles response example as a MIME multi-part file

```
Content-Type: Multipart/Related; boundary=wmts;
        start="TileCollection-Part"
        type="application/xml"

--wmts
```

```
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<TileCollection
        xmlns="http://www.opengis.net/wmts/1.0/get_tiles"
        xmlns:wmts_gt="http://www.opengis.net/wmts/1.0/get_tiles"
        xmlns:ows="http://www.opengis.net/ows/1.1"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/wmts/1.0/get_tiles
 ..\wmtsGetTiles_response.xsd
    http://www.opengis.net/ows/1.1
 http://www.opengis.net/ows/1.1.0/owsAll.xsd">">
    <tile>
        <ows:Identifier>200m_1_0</ows:Identifier>
        <fileURL xlink:href="cid:200m_1_0.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>1</tileRow>
        <tileCol>0</tileCol>
        <width>640</width>
        <height>480</height>
        <top>-185</top>
        <left>-485</left>
    </tile>
    <tile>
        <ows:Identifier>200m_1_1</ows:Identifier>
        <fileURL xlink:href="cid:200m_1_1.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>1</tileRow>
        <tileCol>1</tileCol>
        <width>640</width>
        <height>480</height>
        <top>-185</top>
        <left>155</left>
    </tile>
    <tile>
        <ows:Identifier>200m_2_0</ows:Identifier>
        <fileURL xlink:href="cid:200m_2_0.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>2</tileRow>
        <tileCol>0</tileCol>
        <width>640</width>
        <height>480</height>
        <top>295</top>
        <left>-485</left>
    </tile>
    <tile>
        <ows:Identifier>200m_2_1</ows:Identifier>
        <fileURL xlink:href="cid:200m_2_1.jpg"/>
        <TileMatrix>200m</TileMatrix>
        <tileRow>2</tileRow>
        <tileCol>1</tileCol>
        <width>640</width>
        <height>480</height>
        <top>-295</top>
        <left>-155</left>
    </tile>
</TileCollection>
--wmts
Content-Type: image/jpeg
Content-Description: tile imatge
Content-Transfer-Encoding: base64
Content-ID: 200m_1_0.jpg
Content-Disposition: inline

...
--wmts
Content-Type: image/jpeg
Content-Description: tile imatge
Content-Transfer-Encoding: base64
Content-ID: 200m_1_1.jpg
Content-Disposition: inline

...
--wmts
Content-Type: image/jpeg
Content-Description: tile imatge
Content-Transfer-Encoding: base64
Content-ID: 200m_2_0.jpg
Content-Disposition: inline

...
--wmts
Content-Type: image/jpeg
Content-Description: tile imatge
Content-Transfer-Encoding: base64
Content-ID: 200m_2_1.jpg
Content-Disposition: inline

...
--wmts--
```

Other formats can be more efficient in distributing tiles such as the Open Package Convention, a ZIP file, or a GeoPackage due to they deliver compressed parts instead of expanded encodings such as base64 needed in the MIME multi-part.

Chapter 6. WPS partial results extension

6.1. Introduction

In Testbed 12 we want to be able to retrieve big amounts of tiles from a WMTS. Moving tiles one by one is impractical. Requesting a WMTS tile collection has been proposed (a GetTiles operation that will return a “multitile” format such as SQLite). Requesting a WMTS to prepare millions of tiles in a single file to be sent in the end on the preparation file as a huge file also seems inefficient (or even impossible due to limitations on file size of some formats, such us the 2Gb limitation).

A solution based on a WPS that gets an asynchronous request and requests a sequence of GetTiles is proposed. It has the advantage that a collection of files can be served as results of a WPS that acts as a coordinator and that the client can request the tiles do other things meanwhile it wait for them to able to download later.

In asynchronous WPS, there is a way to execute a WPS process and monitor the progress of the process regularly. This is useful to inform the user about the status of the process and extrapolate when it is going to be completed. In the last version of WPS, this mechanism has evolved a bit adding the possibility to cancel the process.

6.1.1. The problem

If number of tiles involved in the tile collection initial request is huge, the WPS instance will intensively communicate with the WMTS and several files will be created by the WMTS and will become available as intermediate results of the WPS. Unfortunately, the client will not be able to download any of them until the whole process has been completed and the results are returned to the client.

6.1.2. Technical limitation

The WPS 2.0 OGC 14-065 defines the new approach for asynchronous operations based on the existence of a Job identifier. This job identifier (and other time estimations and percentage of competition) is returned by the service immediately after an asynchronous execution has been requested in a StatusInfo document described on section 9.5. A StatusInfo document is also the result of a GetStatus operation defined in 9.10. A WPS 2.0 client can request GetStatus updates each time it needs them until the process is completed. After that, a GetResult operation will be requested and the result will be a “Processing Result document”. There is no requirement saying that a GetResult shall not be issued until the process is completed but there is enough evidence in the document that GetResults does not return a valid result until the process has been completed completely.

- Section 9.11 says: “The GetResult operation allows WPS clients to query the result of a finished processing job. It is used in conjunction with asynchronous execution.”
- Section 9.11.3 says: “If, for some reason, GetResult was invoked too early and the results have not been computed yet, the server shall respond with the exception defined in Table 50 (ResultNotReady).“

For that reason, it seems clear that the aim of the standards' authors was to discourage exchange of input and outputs between the start and completion of a process.

6.2. WPS extension. Part 1

6.2.1. The proposal

The proposal here is to extend GetStatus and GetResult to be able to provide partial outputs during a process execution while allowing for the process to continue. In addition, it could also be useful in other use cases to use a similar mechanism to request new input.

Changes in GetStatus request:

None

Changes in GetStatus response

The “Table 3 – Basic status set for jobs” of the WPS 2.0 standard needs to be extended to incorporate a new value:

Table 7. Extensions for status of a process returned by GetStatus (Part 1).

Value	Description
Results	There are partial results available. The process is running

Changes in GetStatus exceptions

None

Changes in GetResult request

The GetResult operation allows WPS clients to query the result of a finished processing job or a partial result of a process still running. It is used in conjunction with asynchronous execution.

Changes in GetResult response:

The document response does not change. Nevertheless in a partial result case, it is not required that all results are present in the results file and only the available ones should be present.

6.2.2. The WMTS WPS combination use case

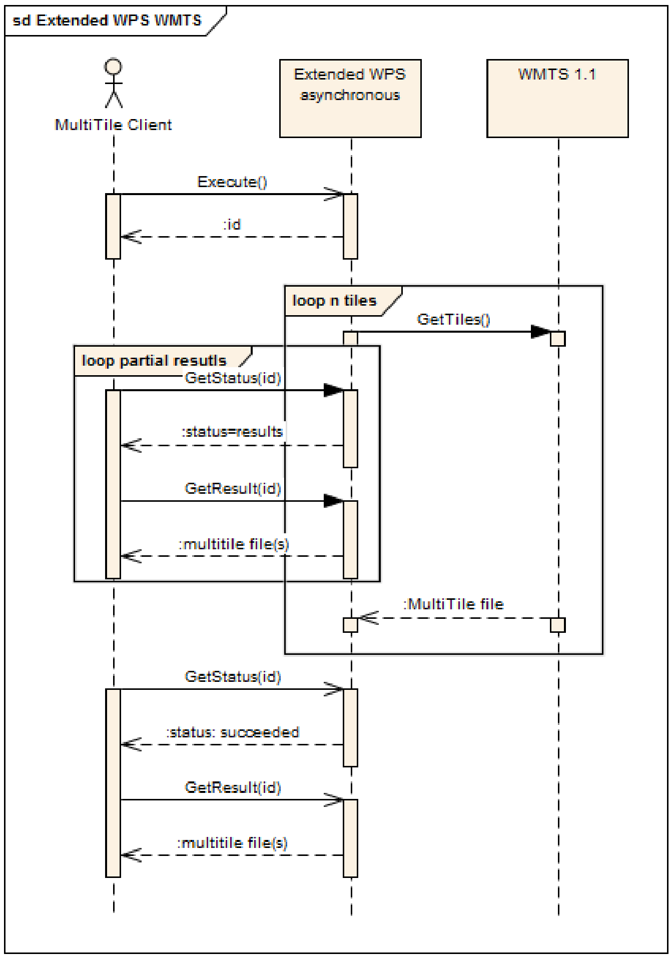

Figure 8. Extended WPS and WMTS combination UML sequence diagram

6.3. WPS extension. Part 2

This is an extension to request and send new inputs. There is no need for the multi-tile download use case described in this document, but it is defined here for completeness.

6.3.1. The proposal

Changes in GetStatus response

The “Table 3 – Basic status set for jobs” of the WPS 2.0 standard needs to be extended to incorporate a new value:

Table 8. Extensions for status of a process returned by GetStatus (Part 2).

Value	Description
Inputs	The process requires new inputs. The process is running but it might be idle

The GetStatus response of WPS 2.0 will include an extra attribute:

Table 9. Extensions for the GetStatus response extra attribute.

Names	Definition	Data type and values	Multiplicity and use
Id	Unambiguous identifier or name of an input item	URI	Zero or more (optional). Included if Status is “Inputs”.

New operation

A new operation needs to be defined: SendInput

Table 10. SendInput new request extension.

Names	Definition	Data type and values	Multiplicity and use
JobID	Job identifier	Character String	This shall be a JobID the client has received during process execution
One (mandatory)	Input	Data inputs provided to this process execution	DataInputType structure, see Table 43 of the WPS 2.0
Zero or one (optional)	Output	Specification of additional outputs expected from the process execution, including the desired format and transmission mode for each output.	OutputDefinitionType structure, see Table 44 of the WPS 2.0

Chapter 7. WPS instance implementation

7.1. Introduction

In Testbed 12, two of the proposed alternatives in the Overview Clause were implemented and tested. The two alternatives are the ones that do not require modifications in the involved standards (WPS and WMTS).

These are the two UML sequence diagrams that show how the WPS client, the WPS server and the WMTS server choose to interact:

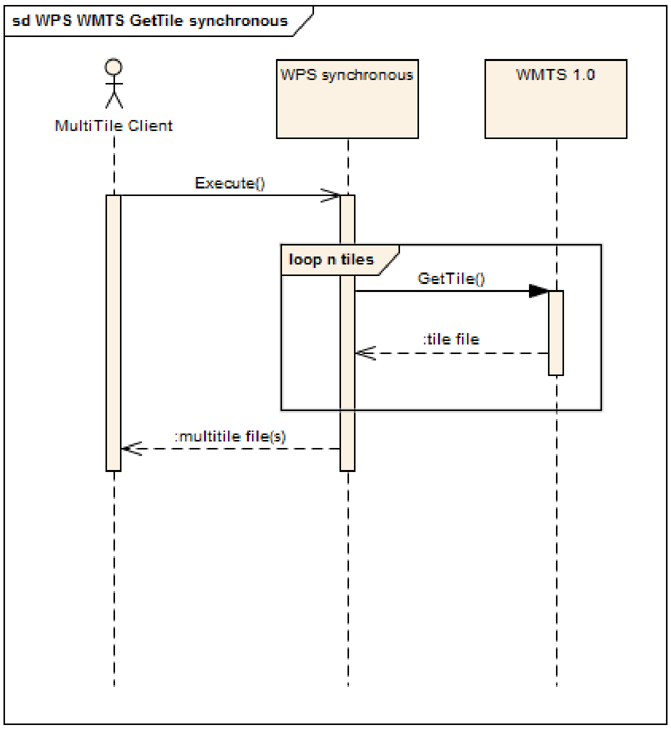

Figure 9. WMTS GetTile with a synchronous WPS UML sequence diagram

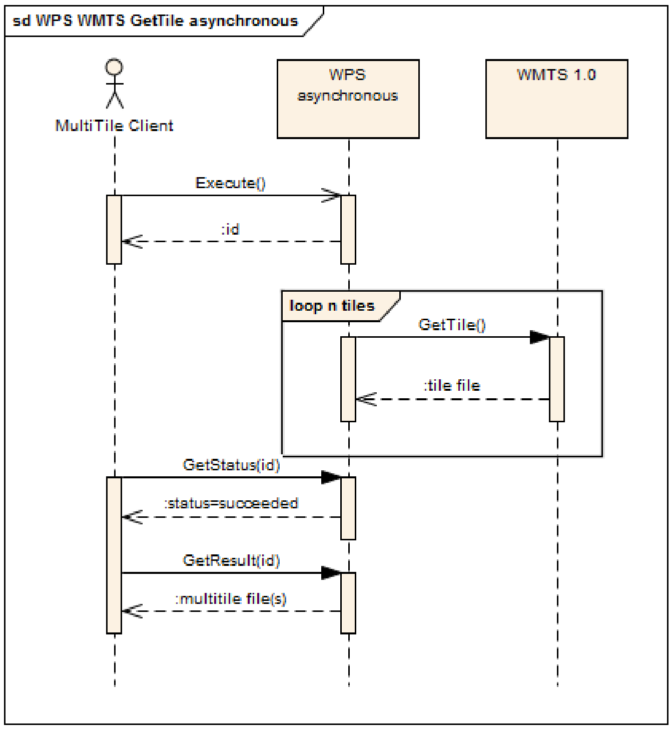

Figure 10. WMTS GetTile with an asyncronous WPS UML sequence diagram

The second approach is preferable because the client does not have to wait to the end of the process and can do other things meanwhile.

7.2. The implemented WPS instance

Another testbed 12 participant (GIS Research Center, Feng Chia University, Taichung, Taiwan [GIS.FCU]) developed a modification of the GeoServer implementation (supports both WPS 1.0 and 2.0) that has an additional process name called "gs:GeoPackage". The name of the process derives from the fact that the return of the process execution will be a GeoPackage with multiple tiles inside.

The following GetCapabilities fragment shows the GeoPackage process name available for WPS Execute requests.

GetCapabilities fragment showing the new process available

```xml
<wps:Capabilities xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ows=
"http://www.opengis.net/ows/1.1" xmlns:wps="http://www.opengis.net/wps/2.0.0"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xml:lang="en" service="WPS" version="
2.0.0" xsi:schemaLocation="http://www.opengis.net/wps/2.0.0
http://schemas.opengis.net/wps/2.0.0/wpsAll.xsd">
    <ows:ServiceIdentification>
        <ows:Title>Prototype GeoServer WPS</ows:Title>
        <ows:Abstract/>
        <ows:ServiceType>WPS</ows:ServiceType>
        <ows:ServiceTypeVersion>1.0.0</ows:ServiceTypeVersion>
        <ows:ServiceTypeVersion>2.0.0</ows:ServiceTypeVersion>
    </ows:ServiceIdentification>
    <ows:ServiceProvider>
        <ows:ProviderName>GISFCU</ows:ProviderName>
        ...
    </ows:ServiceProvider>
    <wps:ProcessOfferings>
        ...
        <wps:Process wps:processVersion="2.0.0">
            <ows:Identifier>gs:GeoPackage</ows:Identifier>
            <ows:Title>GeoPackage</ows:Title>
            <ows:Abstract>generate geopackage from wmts.</ows:Abstract>
        </wps:Process>
        ...
    </wps:ProcessOfferings>
    ...
</wps:Capabilities>
```

Full capabilities can be obtained from this URL: http://59.120.223.164:443/geoserver/ows?service=wps&version=2.0.0&request=GetCapabilities

7.3. Description of the gs:GeoPackage process

Currently, these are the accessible inputs that the WPS gs:GeoPackage process require in the form of table.:

Name	Description and Type	Multiplicity	Observations
minZoom	define the min zoom in from 1~23, default is 1.	one, mandatory	It is related with the TileMatrix name with maximum scale denominator. The use of TileMatrixMin should be better. This parameter could be defined optional

Name	Description and Type	Multiplicity	Observations
maxZoom	define the max zoom in from 1~23, default is 1	one, mandatory	It is related with the TileMatrix name with minimum scale denominator. The use of TileMatrixMin should be better. This parameter could be defined optional
tileFormat	define the tile format, default is jpg	one, mandatory	This parameter could be defined optional
bbox	define the boundary box of the geopackge, format: LeftLng,LowerLat,RightLng,UpperLat	one, mandatory	
layerName	define the layer name wants to be tiled	one, mandatory	
outputName	define the output gpkg file name, default output name is output.gpkg	one, mandatory	This parameter seems unnecessary since the client can find out the name of the output file in the success status message and change the name when retrieving the data into the client computer.

The objective of this process is to generate a sequence of WMTS requests that covers the area defined by *bbox* and the scale range covered by *maxZoom* and *minZoom*. In this ER, we would like to suggest that in a future version, *maxZoom* and *minZoom* be replaced by a TileMatrix parameter with multiplicity *one or more*.

These are other parameters that are implicitly defined in the WPS and cannot be controlled in the current version but could be included in future versions of the service:

Name	Description and Type	Multiplicity	Internal value used
wmtsServerURL	defines the URL of the capabilites document of the server.	one, mandatory	https://tb12.cubewerx.com/a042/cubeserv
tileMatrixSet	defines the name of the tile matrix set.	zero or one, optional	3395
style	Style of the layer retrieved	zero or one, optional	default
other dimensions	Other dimensions and their values in a string	zero or one, optional	
outputType	Type of the output	zero or one, optional	other values could be accepted a part of gpkg such as ZIP or MIME multi-part etc.

The following DescribeProcess shows the GeoPackage process current inputs and outputs definition.

DescribeProcess fragment showing the new process inputs and outputs

```
<wps:ProcessDescriptions>
    <ProcessDescription wps:processVersion="2.0.0" statusSupported="true"
storeSupported="true">
        <ows:Identifier>gs:GeoPackage</ows:Identifier>
        <ows:Title>GeoPackage</ows:Title>
        <ows:Abstract>generate geopackage from wmts.</ows:Abstract>
        <DataInputs>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>minZoom</ows:Identifier>
                <ows:Title>minZoom</ows:Title>
                <ows:Abstract>define the min zoom in from 1~23, default is
1.</ows:Abstract>
                <LiteralData>
                    <ows:DataType>xs:int</ows:DataType>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>maxZoom</ows:Identifier>
                <ows:Title>maxZoom</ows:Title>
                <ows:Abstract>define the min zoom in from 1~23, default is
23.</ows:Abstract>
                <LiteralData>
                    <ows:DataType>xs:int</ows:DataType>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>tileFormat</ows:Identifier>
                <ows:Title>tileFormat</ows:Title>
                <ows:Abstract>define the tile format, default is jpg.</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>bbox</ows:Identifier>
                <ows:Title>bbox</ows:Title>
                <ows:Abstract>define the boundary box of the geopackge, format:
LeftLng,LowerLat,RightLng,UpperLat</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>layerName</ows:Identifier>
                <ows:Title>layerName</ows:Title>
                <ows:Abstract>define the layer name wants to be tiled.</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>outputName</ows:Identifier>
                <ows:Title>outputName</ows:Title>
                <ows:Abstract>define the output gpkg file name, default output name is
output.gpkg.</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
        </DataInputs>
        <ProcessOutputs>
            <Output>
                <ows:Identifier>GeoPackage</ows:Identifier>
                <ows:Title>GeoPackage</ows:Title>
                <LiteralOutput/>
            </Output>
        </ProcessOutputs>
    </ProcessDescription>
</wps:ProcessDescriptions>
```

7.3.1. Example of an execute POST message

This is an example of a POST message that needs to be submitted to the this URL: http://59.120.223.164:443/geoserver/wps to execute the GeoPackage process.

Execute example

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute version="2.0.0" service="WPS" >
    <ows:Identifier>gs:GeoPackage</ows:Identifier>
    <wps:DataInputs>
        <wps:Input>
            <ows:Identifier>minZoom</ows:Identifier>
            <wps:Data>
                <wps:LiteralData>1</wps:LiteralData>
            </wps:Data>
        </wps:Input>
        <wps:Input>
            <ows:Identifier>maxZoom</ows:Identifier>
            <wps:Data>
                <wps:LiteralData>1</wps:LiteralData>
            </wps:Data>
        </wps:Input>
        <wps:Input>
            <ows:Identifier>tileFormat</ows:Identifier>
            <wps:Data>
                <wps:LiteralData>jpg</wps:LiteralData>
            </wps:Data>
        </wps:Input>
        <wps:Input>
            <ows:Identifier>outputName</ows:Identifier>
            <wps:Data>
                <wps:LiteralData>output.gpkg</wps:LiteralData>
            </wps:Data>
        </wps:Input>
        <wps:Input>
            <ows:Identifier>bbox</ows:Identifier>
            <wps:Data>minLon,minLat,maxLon,maxLat
                <wps:LiteralData>
-113.3506260963461,30.5714609234194,127.9521005274938,45.01357346325526
            </wps:LiteralData>
            </wps:Data>
        </wps:Input>
        <wps:Input>
            <ows:Identifier>layerName</ows:Identifier>
            <wps:Data>
                <wps:LiteralData>National_Land_Cover</wps:LiteralData>
            </wps:Data>
        </wps:Input>
    </wps:DataInputs>
</wps:Execute>
```

The WPS process needs to internally transform the BBOX into a list of TileCol and TileRow pairs. Annex H of the WMTS 1.0 standard document OGC 07-057r7 provides a pseudocode to translate a bbox to tilecol and tilerow given a TileMatrix name. Then, a sequence of WMTS requests is generated and the resulting tiles are retrieved. The result of process executions is a GeoPackage with all retrieved tiles embedded in the file.

Appendix A: Revision History

Table 11. Revision History

Date	**Release**	**Editor**	**Primary clauses modified**	**Descriptions**
Sep 20, 2016	Joan Maso	.1	all	first full version
Nov 2, 2016	Joan Maso	1.0	all	comments integrated; text finalized and ready for TC review.
Dec 26, 2016	Joan Maso	r1	all	comments received and integrated.

Appendix B: Bibliography

- OGC 12-128r12, OGC Geopackage Encoding Standard
- OGC 11-014r3, OGC® Open Modelling Interface (OpenMI) Interface Standard
- OGC 12-157r1, OWS-9 Map Tiling Methods Harmonization IP Engineering Report
- OGC 16-044, Testbed 12 - Web Feature Service Synchronization Engineering Report
- John T. Sample and Elias Ioup, 2010, Tile-Based Geospatial Information Systems. Springer. ISBN: 978-1-4419-7630-7 (Print) or 978-1-4419-7631-4 (Online).
- N. Borenstein and N. Freed, Innosoft, 1992 RFC 1341, MIME (Multipurpose Internet Mail Extensions). Chapter 7.2 The Multipart Content-Type.
- X. Pons, J. Masó, 2016, A comprehensive open package format for preservation and distribution of geospatial data and metadata. Computers & Geosciences 97 (2016) 89–97.