Testbed-12 Implementing Asynchronous Services Response Engineering Report

Table of Contents

Publication Date: 2017-06-30

Approval Date: 2017-06-29

Posted Date: 2016-12-08

Reference number of this document: OGC 16-023r3

Reference URL for this document: http://www.opengis.net/doc/PER/t12-A067

Category: Public Engineering Report

Editor: Benjamin Pross

Title: Testbed-12 Implementing Asynchronous Services Response Engineering Report

---

Testbed-12 Implementing Asynchronous Services Response Engineering Report (OGC 16-023r3)

COPYRIGHT

WARNING

This document is an OGC Public Engineering Report created as a deliverable of an initiative from the OGC Innovation Program (formerly OGC Interoperability Program). It is not an OGC standard and not an official position of the OGC membership.It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

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

Most of current OGC specifications define synchronous communication patterns, i.e. after sending a request to an OGC service, clients need to wait for the response. But several applications, e.g. delivery of information about events or executing complex environmental models with long runtime, need asynchronous client-server interaction pattern that do not require clients to keep the connection to the server continuously open in order to wait for responses. At the moment, there are several approaches how to add asynchronous communication to existing OGC services: One option is to use a WPS façade, as the WPS specification already defines asynchronous service responses. Another option is to add extensions to the different specifications and the third option is developed by the OGC Publish-Subscribe Working Group. This ER summarizes and compares the results from the different activities for asynchronous service responses and provides recommendations for future activities.

Business Value

This benefit of this ER will be recommendations for handling asynchronous communication for different OGC Web services. Based on the findings, a common baseline for asynchronous service communication could be derived.

Technology Value

The described approach for using WPS to enable synchronous services with asynchronous functionality is a new use case for web-based processing.

Keywords

ogcdocs, testbed-12, SOAP, asynchronous

Proposed OGC Working Group for Review and Approval

WPS 2.0 SWG, WFS/FES SWG, PubSub SWG

Chapter 1. Introduction

1.1. Scope

Most of current OGC specifications define synchronous communication patterns, i.e. after sending a request to an OGC service, clients need to wait for the response. But several applications, e.g. delivery of information about events or executing complex environmental models with long runtime, need asynchronous client-server interaction pattern that do not require clients to keep the connection to the server continuously open in order to wait for responses. At the moment, there are several approaches how to add asynchronous communication to existing OGC services: One option is to use a WPS facade, as the WPS specification already defines asynchronous service responses. Another option is to add extensions to the different specifications. The extension specifically addressed by this ER was for WFS. The third option was developed by the OGC Publish-Subscribe Working Group. An example of using the Pub/Sub concepts with a Catalog Service is described in the Testbed-12 PubSub / Catalog ER [OGC 16-137]. The three methods are discussed in section 8. The goal of this ER is to summarize and compare the results from using a WPS facade and an extension for WFS for asynchronous service responses and to provide recommendations for future activities.

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
Benjamin Pross	52°North GmbH
Peter Vretanos	Cubewerx

1.3. Future Work

See section 9.

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 06-121r9, OGC® Web Services Common Standard
- OGC 09-025r2 OGC® Web Feature Service 2.0 Interface Standard
- OGC 14-065 OGC® WPS 2.0 Interface Standard
- OGC 16-137 OGC® Testbed-12 PubSub / Catalog ER

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] shall apply. In addition, the following terms and definitions apply.

3.1. Facade

Software design pattern. The goal is to simplify access to underlying complex functionality.

Chapter 4. Conventions

4.1. Abbreviated terms

- Async Asynchronous
- OWS OGC Web Service
- SOAP Simple Object Access Protocol
- Sync Synchronous
- WCS Web Coverage Service
- WFS Web Feature Service
- WPS Web Processing Service

Chapter 5. Overview

This ER describes two approaches to enable synchronous OGC service with asynchronous capabilities. First, a WPS facade approach is described and tested against WFS and WCS services. Second, an approach is presented, to enable WFS with asynchronous capabilities using additional parameters in the request.

First, some background information is given, followed by a description of the different asynchronous service implementations. Finally, recommendations for future work are given.

Chapter 6. Background

6.1. Asynchronous Request Pattern

Three basic request-response patterns can be identified. (1) Synchronous requests:

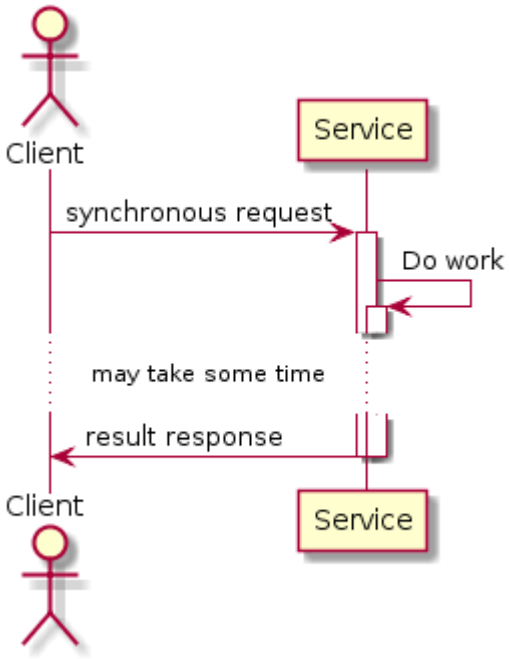

Figure 1. Synchronous request sequence

The client has to keep the connection alive until the service has finished its work and is ready to return the response. If the connection is interrupted (e.g. when using a mobile device), also the result of the request is lost, as the client has no information about where to re-connect to get the specific result that was generated by the previous request.

(2) Asynchronous requests using a pull mechanism:

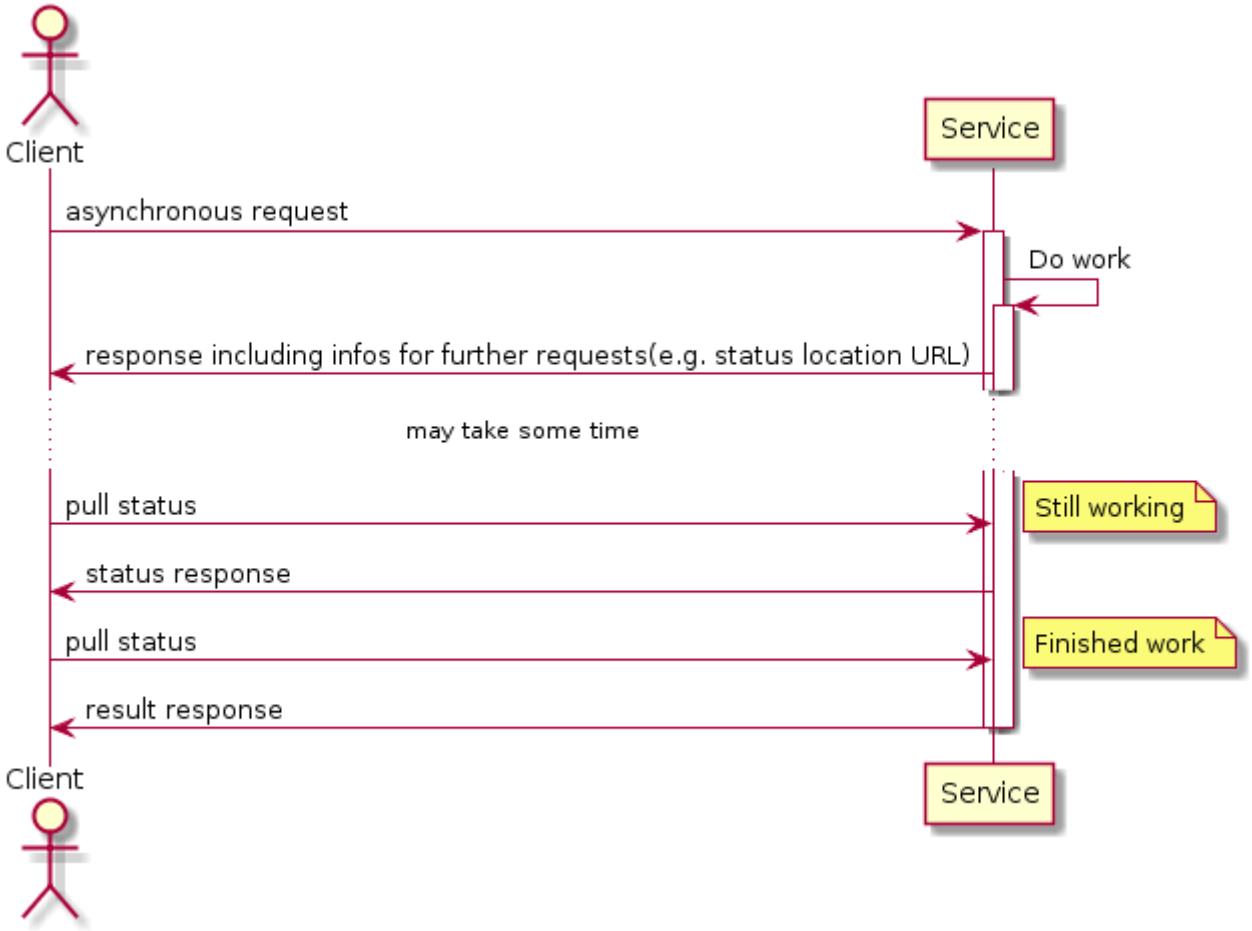

Figure 2. Asynchronous request sequence - pull

After the client sends the asynchronous request, the service immediately responds. The response must contain information about where the client can request status-updates and finally the result. As the response is sent immediately and is small, this pattern is less error-prone. The service uses its own infrastructure to store the status and result (using a distributed infrastructure would be possible, but this would introduce a source for errors). The client knows the location of the result and is able to get it at any time. In a mobile environment, the client could therefore wait for a moment of stable/strong connection.

(3) Asynchronous requests using a push mechanism:

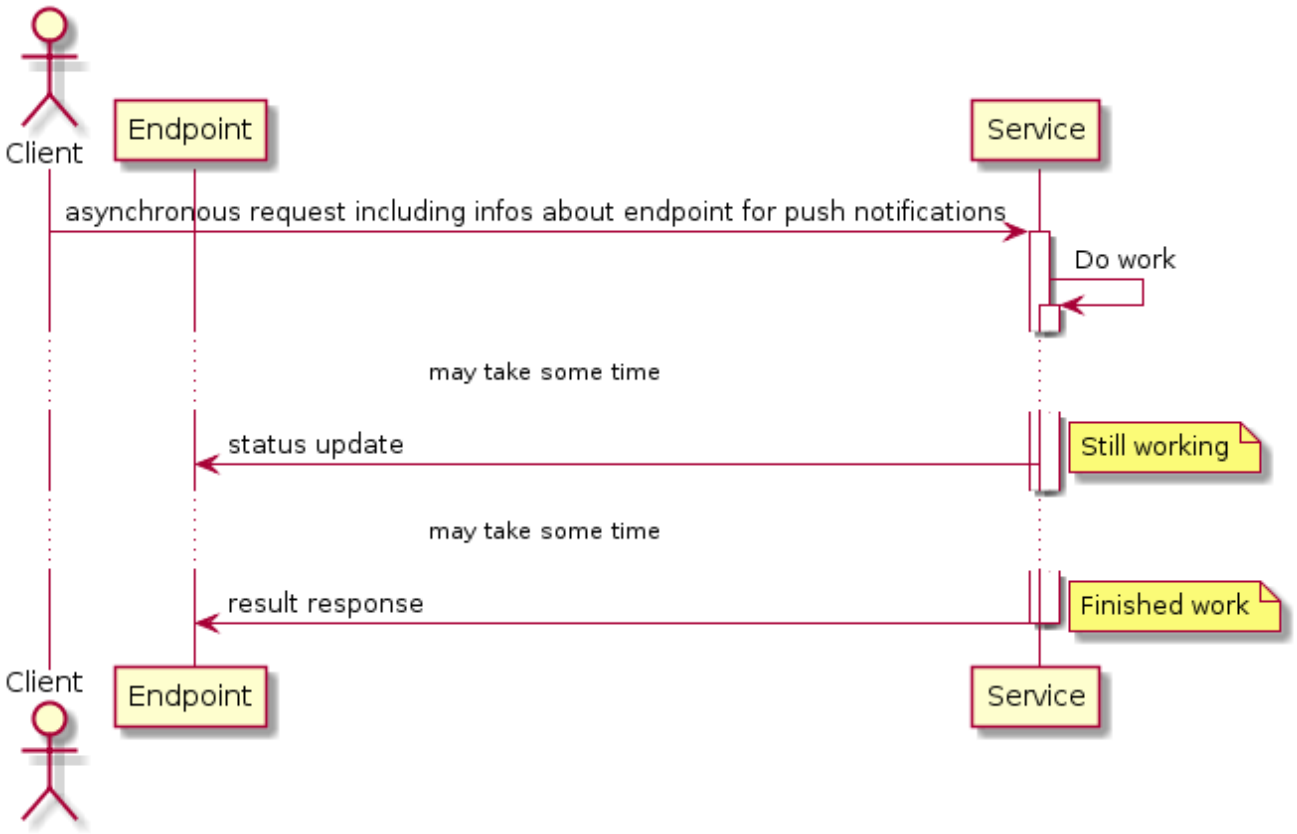

Figure 3. Asynchronous request sequence - push

Following the push-based pattern, the client has to provide the service with information about an endpoint, that the service can connect to and push status-updates and finally the result. The client doesn't need a direct response from the service at all, which could be beneficial in mobile environments. On the downside, the service has to actively connect to an endpoint, which introduces a source for errors. If the endpoint cannot be reached, the service has no means of informing the client. If the endpoint for pushing status/result is not tightly coupled to the client, the client has to pull the endpoint.

6.1.1. WPS facade

As the Web Processing Service interface standard specifies a pull-based mechanism for asynchronous execution, the WPS could be used as a facade in front of a service that offers no asynchronous request mechanism. The following image shows the sequence diagram of a basic facade approach:

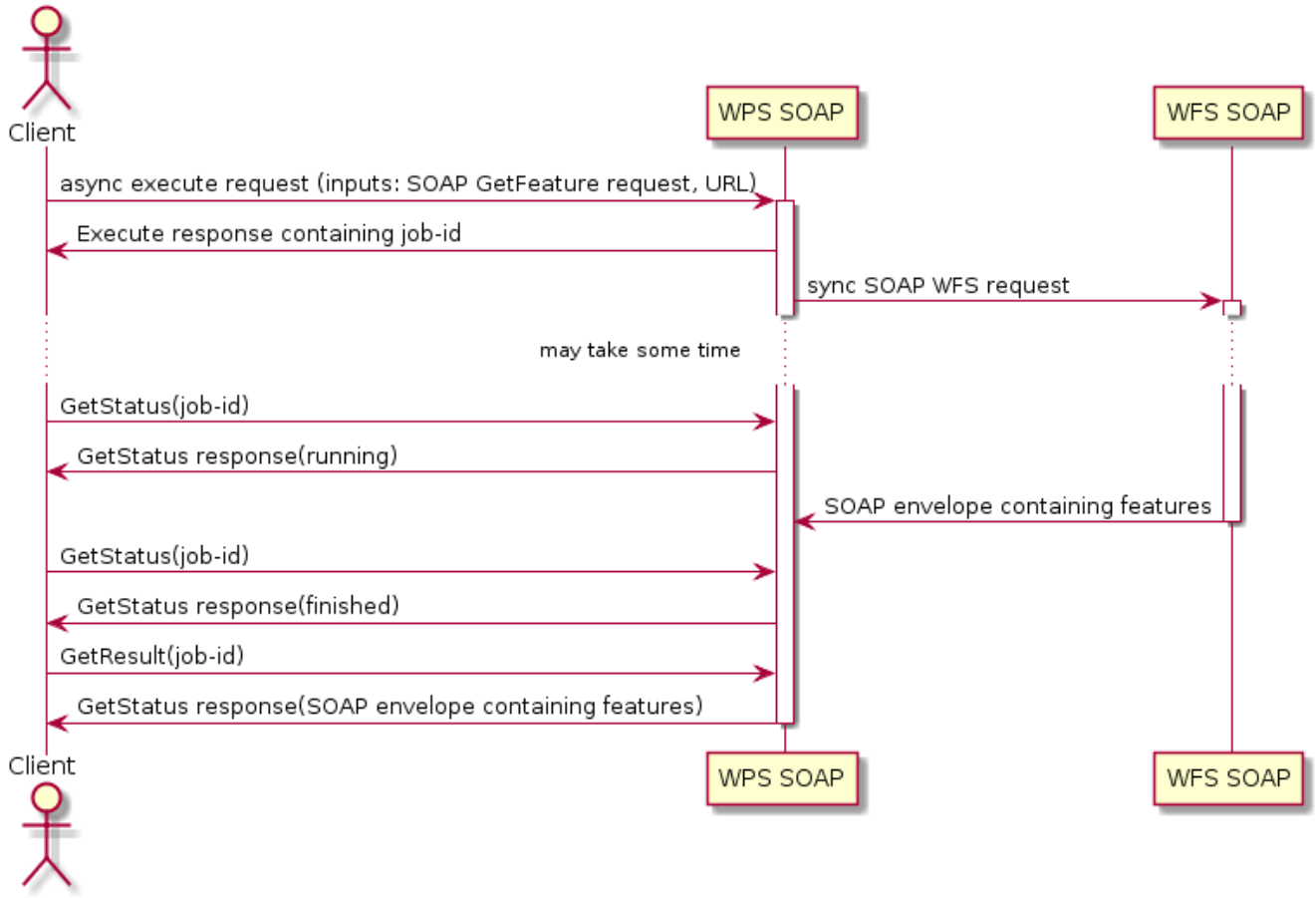

Figure 4. WPS facade sequence

The task of keeping the connection alive is moved from the client to the WPS, which presumably resides in a well connected environment in contrast to e.g. a mobile client. Still, when requesting large amounts of data, the connection between WPS and data service could reach timeout-limits.

Chapter 7. Implementations

7.1. WPS Facade

The WPS facade approach was tested with a SOAP WFS provided by the Arizona State University and a SOAP WCS provided by George Mason University/Jacobs University. Also, a secured SOAP WFS was accessed that was provided by Secure Dimensions/CubeWerx. The process description is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:ProcessOfferings xmlns:wps="http://www.opengis.net/wps/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ows="http://www.opengis.net/ows/2.0"
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:ProcessOffering processVersion="1.0.0" jobControlOptions="sync-execute async-
execute" outputTransmission="value reference">
    <wps:Process>
      <ows:Title>testbed12.cmd.AsyncFacadeProcess</ows:Title>
      <ows:Abstract>Process acting as async facade for a SOAP endpoint</ows:Abstract>
      <ows:Identifier>testbed12.cmd.AsyncFacadeProcess</ows:Identifier>
      <wps:Input minOccurs="1" maxOccurs="1">
        <ows:Title>request</ows:Title>
        <ows:Identifier>request</ows:Identifier>
        <ns:ComplexData xmlns:ns="http://www.opengis.net/wps/2.0">
          <ns:Format default="true" mimeType="application/soap+xml"/>
          <ns:Format mimeType="text/xml"/>
        </ns:ComplexData>
      </wps:Input>
      <wps:Input minOccurs="1" maxOccurs="1">
        <ows:Title>endpoint-url</ows:Title>
        <ows:Identifier>endpoint-url</ows:Identifier>
        <ns:LiteralData xmlns:ns="http://www.opengis.net/wps/2.0">
          <ns:Format default="true" mimeType="text/plain"/>
          <ns:Format mimeType="text/xml"/>
          <LiteralDataDomain>
            <ows:AnyValue/>
            <ows:DataType ows:reference="xs:anyURI"/>
          </LiteralDataDomain>
        </ns:LiteralData>
      </wps:Input>
      <wps:Output>
        <ows:Title>response</ows:Title>
        <ows:Identifier>response</ows:Identifier>
        <ns:ComplexData xmlns:ns="http://www.opengis.net/wps/2.0">
          <ns:Format default="true" mimeType="application/soap+xml"/>
          <ns:Format mimeType="text/xml"/>
        </ns:ComplexData>
      </wps:Output>
    </wps:Process>
  </wps:ProcessOffering>
</wps:ProcessOfferings>
```

The Process was kept generic, so that it is able to work with a wide variety of services. The following table explains the inputs:

Table 2. WPS Async facade process inputs

Input name	Description	Format
request	The request that shall be send to the service	XML/SOAP
endpoint-url	The URL of the service endpoint	anyURI

The following outputs are provided by the process:

Table 3. WPS Async facade process outputs

Output name	Description	Format
response	The response	XML/SOAP

7.1.1. Experiments

In the following section, we describe some experiment we conducted with different SOAP services.

SOAP WFS

The following XML shows an example execute-request encapsulating a GetFeature request to a SOAP WFS provided by the Arizona State University:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute xmlns:wps="http://www.opengis.net/wps/2.0"
  xmlns:ows="http://www.opengis.net/ows/2.0"
xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd"
  service="WPS" version="2.0.0" response="document" mode="async">
  <ows:Identifier>testbed12.cmd.AsyncFacadeProcess</ows:Identifier>
  <wps:Input id="request">
    <wps:Data mimeType="application/soap%2Bxml">
      <soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
        <soap:Head></soap:Head>
        <soap:Body>
          <wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs/2.0"
            xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:gml="http://www.opengis.net/gml/3.2"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation="http://schemas.opengis.net/wfs/2.0/wfs.xsd
http://schemas.opengis.net/gml/3.2.1/gml.xsd
http://schemas.opengis.net/filter/2.0/filterAll.xsd"
            service="WFS" version="2.0" outputFormat="application/gml+xml;
version=3.2" count="3000">
            <wfs:Query typeNames="landuse"></wfs:Query>
          </wfs:GetFeature>
        </soap:Body>
      </soap:Envelope>
    </wps:Data>
  </wps:Input>
  <wps:Input id="endpoint-url">
    <wps:Data>

<wps:LiteralValue>http://polar.geodacenter.org/services/ows/wfs/soap/1.2</wps:LiteralV
alue>
    </wps:Data>
  </wps:Input>
  <wps:Output id="response" transmission="value"
    mimeType="application/soap%2Bxml" />
</wps:Execute>
```

Note that the mode-attribute is set to *async* and the response to *document*. The later means, that the response of the upstream service will be encapsulated in a WPS Result document. The attribute could also be set to *raw* to obtain the upstream service response directly. The WPS directly returns a StatusInfo document containing a jobId that can be used to obtain further status updates and finally the result:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:StatusInfo xmlns:wps="http://www.opengis.net/wps/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:JobID>ca291352-593a-4ddc-bdab-d1f079d7125c</wps:JobID>
  <wps:Status>Accepted</wps:Status>
</wps:StatusInfo>
```

For subsequent status requests the following URL-syntax must be used:

```
http://myhost/myWPSWebapp?Request=GetStatus&Service=WPS&version=2.0.0&jobid=ca291352-
593a-4ddc-bdab-d1f079d7125c
```

After the WPS process has successfully requested the upstream service, the status will be updated:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:StatusInfo xmlns:wps="http://www.opengis.net/wps/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:JobID>ca291352-593a-4ddc-bdab-d1f079d7125c</wps:JobID>
  <wps:Status>Succeeded</wps:Status>
</wps:StatusInfo>
```

To obtain the result from the WPS the following URL-syntax must be used:

```
http://myhost/myWPSWebapp?Request=GetResult&Service=WPS&version=2.0.0&jobid=ca291352-
593a-4ddc-bdab-d1f079d7125c
```

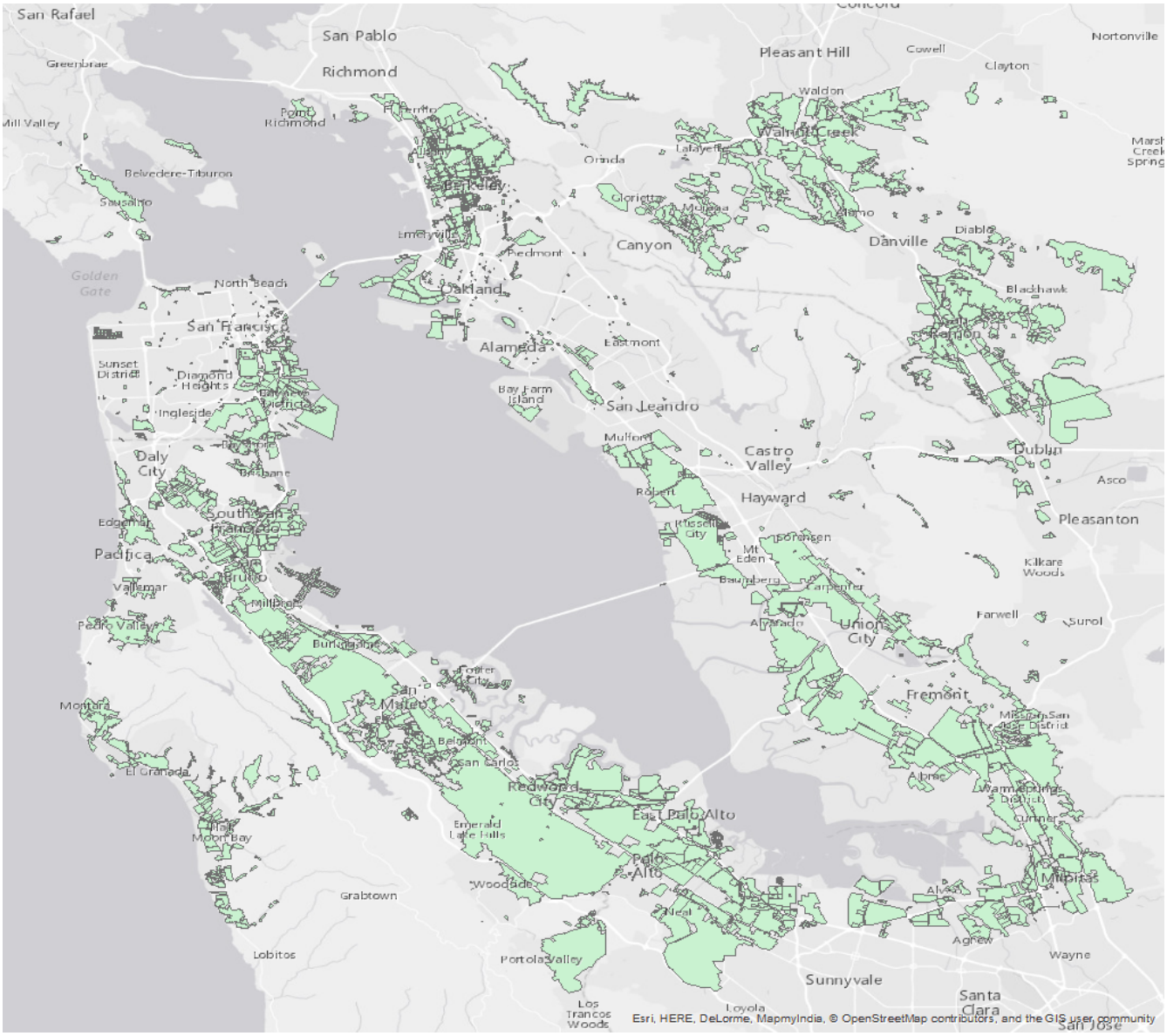

Figure 5. The 3000 landuse features

The following table shows some statistics of the load time for loading 1000, 2000 and 3000 (all) features using the SOAP WFS directly and via the WPS facade (sync execute for testing). 100 requests were send during each test.

Table 4. Statistics for load times comparison requesting the SOAP WFS for 1000 features

Method	Min (ms)	Max (ms)	Mean (ms)	Standard deviation (ms)
Direct	4712	12314	5067,18	982
Via WPS Facade	7522	20897	9463,19	1474,69

Table 5. Statistics for load times comparison requesting the SOAP WFS for 2000 features

Method	Min (ms)	Max (ms)	Mean (ms)	Standard deviation (ms)
Direct	8122	14632	8624,91	1179,64
Via WPS Facade	13771	28492	15430,2	1956,01

Table 6. Statistics for load times comparison requesting the SOAP WFS for 3000 features

Method	Min (ms)	Max (ms)	Mean (ms)	Standard deviation (ms)
Direct	11056	28070	11779,59	1930,31
Via WPS Facade	18536	39433	20892,55	2851,57

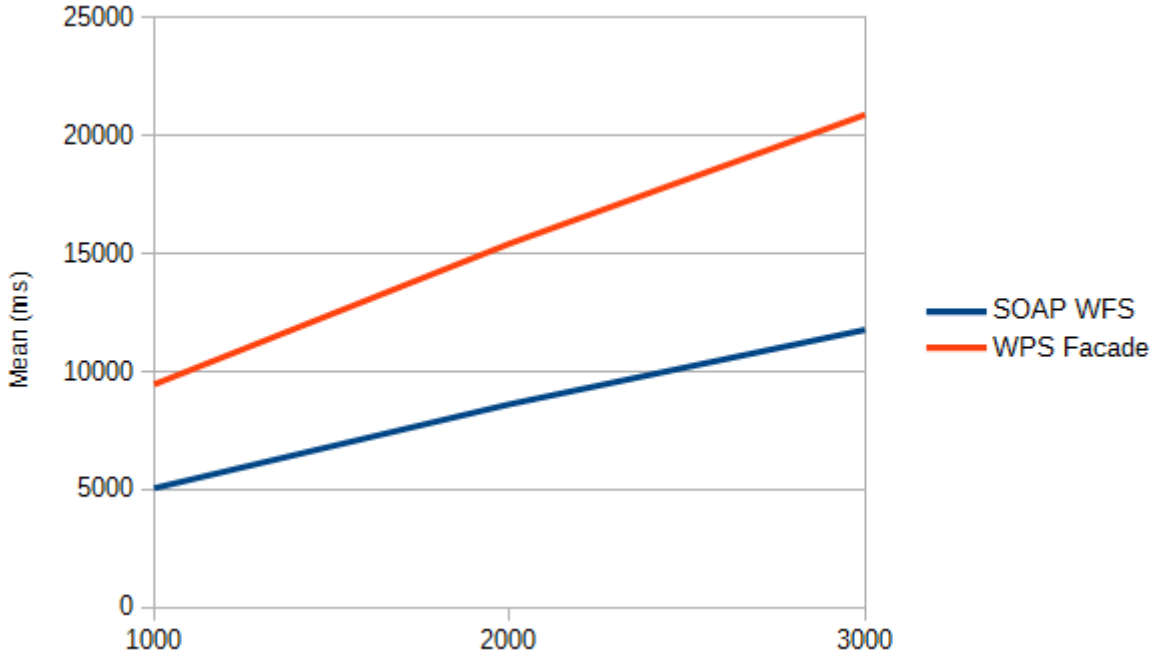

Figure 6. Mean related to the number of requested features

SOAP WCS

The following XML shows an example execute-request encapsulating a GetCoverage request to a SOAP WCS provided by the George Mason University:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute xmlns:wps="http://www.opengis.net/wps/2.0"
  xmlns:ows="http://www.opengis.net/ows/2.0"
xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd"
  service="WPS" version="2.0.0" response="document" mode="async">
  <ows:Identifier>testbed12.cmd.AsyncFacadeProcess</ows:Identifier>
  <wps:Input id="request">
    <wps:Data mimeType="application/soap%2Bxml">
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.opengis.net/wcs/2.0">
   <soap:Header/>
   <soap:Body>
      <ns:GetCoverage service="WCS" version="2.0.0">

<ns:CoverageId>GEOTIFF:"/home/zsun/testfiles/data/2010_305_30H.tif":Band</ns:CoverageI
d>
         <ns:format>image/GEOTIFF</ns:format>
      </ns:GetCoverage>
   </soap:Body>
</soap:Envelope>
    </wps:Data>
  </wps:Input>
  <wps:Input id="endpoint-url">
    <wps:Data>

<wps:LiteralValue>http://cube.csiss.gmu.edu/axis2/services/GMU_SOAP_WCS_Service.GMU_SO
AP_WCS_ServiceHttpSoap12Endpoint</wps:LiteralValue>
    </wps:Data>
  </wps:Input>
  <wps:Output id="response" transmission="value" mimeType="application/soap%2Bxml" />
</wps:Execute>
```

As response for this request, a the coverage (base64 encoded tiff image) is returned wrapped in a SOAP envelope.

The following table shows some statistics of the load time for loading two different coverages using the SOAP WFS directly and via the WPS facade (sync execute for testing).

The size of the first coverage is 416 KB. 100 requests were sent during each test.

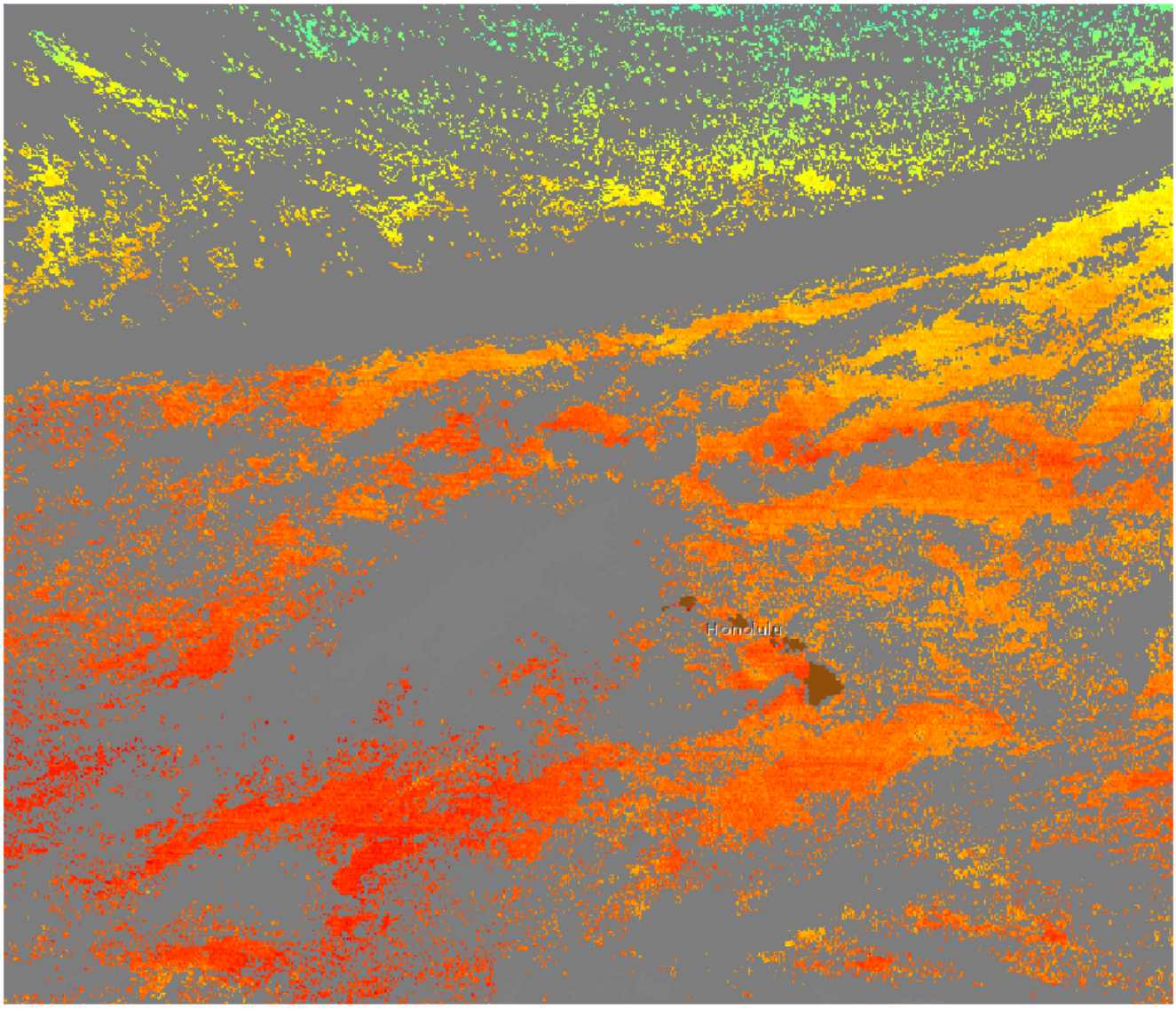

Figure 7. First test coverage

Table 7. Statistics for load times comparison requesting the SOAP WCS for a coverage of 416 KB

Method	Min (ms)	Max (ms)	Mean (ms)	Standard deviation (ms)
Direct	780	9119	1031,68	1930,31
Via WPS Facade	2082	2715,23	20892,55	1250,72

The size of the first coverage is 84,6 MB. 10 requests were sent during each test.

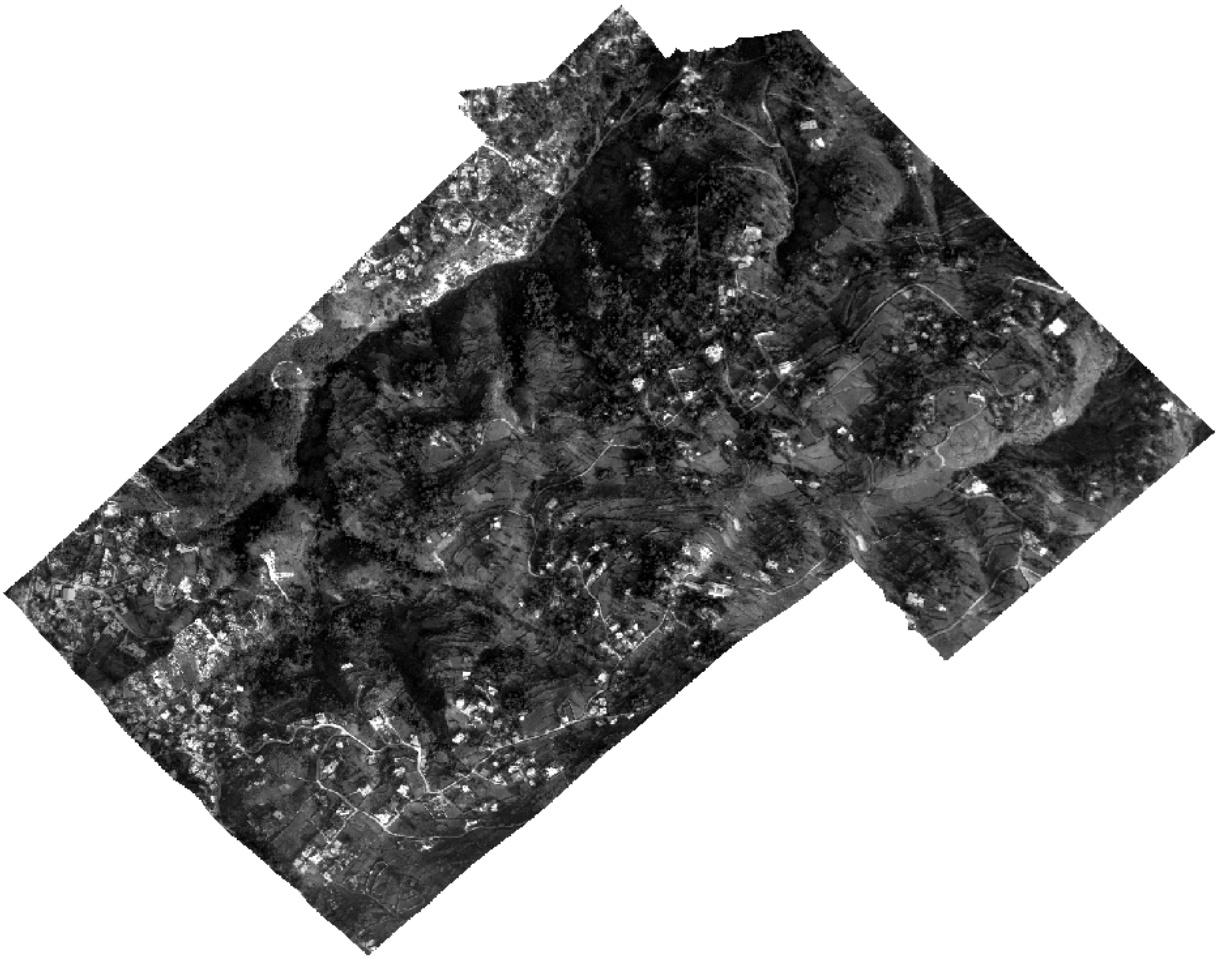

Figure 8. Second test coverage

Table 8. Statistics for load times comparison requesting the SOAP WCS for a coverage of 84,6 MB

Method	**Min (ms)**	**Max (ms)**	**Mean (ms)**	**Standard deviation (ms)**
Direct	129403	194848	144381,5	25606,58
Via WPS Facade	151145	195953	163789,5	13288,59

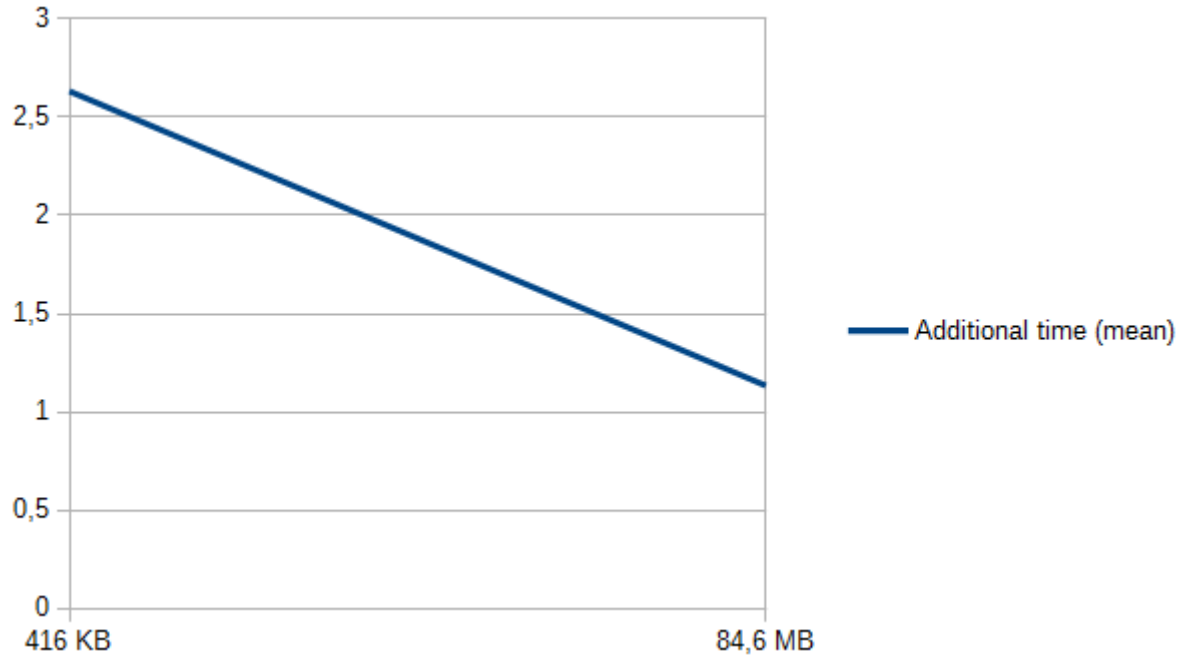

Figure 9. Additional time depending on data size

Conclusions

Some conclusions from the tests:

- The WPS requests seem to take significantly longer, probably due to the read/write overhead
- The load time of the direct SOAP WFS requests seem to be circa half the time the requests take using the WPS facade. This seems to be true for increasing numbers of requested features, i.e. larger amounts of data. Note that 3000 features in the GML 3.2 format have a size of circa 5 MB, which is still relatively small.
- For smaller coverages, the load time of the direct SOAP WCS requests also seem to be circa half the time the requests take using the WPS facade. For larger requests, the additional time reduces to 12%, on average.
- For larger amounts of data, the overhead of using the WPS facade seems to be less significant.

However, already the relatively small number of requested features lead to load times of over ten seconds. For the larger coverage, the direct request takes over two minutes, on average. For sync execute the connection must be kept alive for the whole duration, which could lead to issues in the field.

7.2. Asynchronous request processing for WFS

7.2.1. Introduction

This clause describes a light-weight protocol for processing long-running operations asynchronously. Rather than the standard request-response invocation pattern – where a client submits a request and then waits to get the response from the server – an asynchronous request is executed in the background by the server and either a notification message is sent when operation processing has completed or (depending on the conformance classes implemented) the server may

be polled periodically to determine the execution status of the request.

The target service implementation for this capability for Testbed- 12 is the WFS. However, it is envisioned that the approach described here could apply to a number of OGC services (WCS, CSW, etc.).

For the WFS standard, the following operations may be invoked asynchronously:

1. GetPropertyValue
2. GetFeature
3. GetFeatureWithLock
4. LockFeature
5. Transaction

7.2.2. Conformance classes

Two conformance classes are defined for asynchronous request processing:

1. Asynchronous Processing
2. Asynchronous Polling

For the Asynchronous Processing conformance class, a server implements the ability to accept an asynchronous request, acknowledge that the request was successfully accepted, process the request in the background and finally send a notification — using the specified response handler — when the request's processing has been completed. Servers that implement this class may optionally provide a hypermedia control that may be used to cancel the asynchronous request after it has been invoked.

For the Asynchronous Polling conformance class, a server implements the ability to accept an asynchronous request, acknowledge that the request was successfully accepted, process the request in the background and allow the server to be polled periodically to obtain the execution status and progress of the request's execution. Once request processing has been completed a hypermedia control, provided by the server, will allow the request's response to be retrieved.

7.2.3. Request semantics

Asynchronous request processing shall be triggered by the presence of the ResponseHandler parameter.

When an operation is invoked asynchronously, the server shall immediately respond with an acknowledgement message (see XML encoding) that indicates that the request has been successfully accepted or an exception messages if there was a problem. At this point communication with the server shall terminate.

The server shall then proceed to process the request in the background, taking as much time as necessary to complete its processing.

For servers that implement the Asynchronous Processing class, when request processing has

completed a notification message shall be sent using the scheme(s) specified as the value of the ResponseHandler parameter. The content of the notification message is discussed in clause Notification message content.

For servers that implement the Asynchronous Polling class, the acknowledgement message shall contain a means by which the server may be polled to determine the execution status of the request (see XML encoding). When request processing has been completed, the acknowledgement message shall contain a means by which the request's response may be retrieved.

Parameter discussion

The value of the ResponseHandler parameter shall be a list of one or more values. The value of each element of the list shall either be a URI or the token "poll".

If a list element value of the ResponseHandler parameter is a URI, then its form shall be a valid expression of the one the notification schemes that the server claims to support in its capabilities document (see [CapabilitiesResponse]).

This standard does not define a normative set of notification schemes but possible schemes include:

- Email scheme as per RFC 2368
  - Example — mailto:tb12@pvretano.com
- Sms scheme as per Apple Inc. (or perhaps RFC 5724)
  - Example – sms:1-555-555-5555
- Webhook as per http://www.webhooks.org
  - Example — http://www.myserver.com/posthandler

The specific set of schemes supported for each operation shall be advertised in the server's capabilities document (see [CapabilitiesResponse]).

If a list element value of the ResponseHandler parameter is the token "poll", the server shall, in its acknowledgement message (see XML encoding), provide a hypermedia control (see Response semantics) that may be used to poll the server periodically to determine the execution status of the request.

If more than one instance of the "poll" token appears as a list element value of the ResponseHandler parameter, the extraneous instances of the token shall be ignored. The "poll" token need only appear once to trigger the inclusion of the status, progress and control elements within the acknowledgement messages (see Response semantics).

XML encoding

The following XML-Schema fragment defines the XML-encoding for this parameter:

```
<xsd:element name="ResponseHandler" type="xsd:anyUri"/>
```

KVP encoding

Table 1 defines the KVP-encoding for the ResponseHandler parameter.

Table 9. KVP-encoding for the ResponseHandler parameter

URL Component	O/M	Description
RESPONSEHANDLER	O	An URI indicating the notification method

NOTE O = Optional, M = Mandatory

Examples

The following example fetches NHD flowlines from the TB12 A007 WFS using an asynchronous XML-encoded GetFeature request:

```
<?xml version="1.0" ?>
<GetFeature
   version="2.5.0"
   service="WFS"
   xmlns="http://www.opengis.net/wfs/2.5"
   xmlns:fes="http://www.opengis.net/fes/2.5"
   xmlns:gml="http://www.opengis.net/gml/3.2"
   xmlns:cw="http://schemas.cubewerx.com/namespaces/null"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/wfs/2.5
                       http://schemas.cubewerx.com/schemas/wfs/2.5/wfs.xsd
                       http://www.opengis.net/gml/3.2
                       http://schemas.opengis.net/gml/3.2.1/gml.xsd
                       http://schemas.cubewerx.com/namespaces/null

http://tb12.cubewerx.com/a007/cubeserv?datastore=USGS&amp;service=WFS&amp;version=2.5&
amp;request=DescribeFeatureType&amp;typeNames=NHDFlowline">
   <Query typeNames="cw:NHDFlowline">
      <fes:Filter>
         <fes:BBOX>
            <fes:ValueReference>GEOMETRY</fes:ValueReference>
            <gml:Envelope srsName="http://www.opengis.net/def/crs/epsg/0/4326">
               <gml:lowerCorner>37.709077 -122.513476</gml:lowerCorner>
               <gml:upperCorner>37.839064 -122.351771</gml:upperCorner>
            </gml:Envelope>
         </fes:BBOX>
      </fes:Filter>
   </Query>
   <ResponseHandler>mailto:tb12@pvretano.com</ResponseHandler>
</GetFeature>
```

The following example fetches NHD flowlines from the TB12 A007 WFS using an asynchronous

KVP-encoded GetFeature request:

```
http://tb12.cubewerx.com/a007/cubeserv?datastore=USGS&
   service=WFS&
   version=2.0.2&
   request=GetFeature&
   typeNames=NHDFlowline&
   count=100&
   outputFormat=application%2Fgml%2Bxml&
   responseHandler=mailto:tb12@pvretano.com&
   bbox=37.709077,-122.513476,37.839064,-122.351771,urn:ogc:def:crs:EPSG::4326
```

NOTE	Examples in this clause may be formatted to facilitate readability as is the case in this example which has been wrapped to highlight the request parameters.

7.2.4. Response

Introduction

This clause defines the XML-Schema of the acknowledgement message that is used to signal that an asynchronous request has been successfully accepted. The same message schema is also used in response to a polling request to indicate the execution status of an asynchronous request.

XML encoding

The following XML-Schema fragment defines the ows:Acknowledgment element:

```
<xsd:element name="Acknowledgment"
             type="ows:Acknowledgement" id="Acknowledgement"/>
<xsd:complexType name="Acknowledgement" id="AcknowledgementType">
   <xsd:sequence>
      <xsd:element ref="atom:link" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="Status"
                   type="wfs:ExecutionStatusType" minOccurs="0"/>
      <xsd:element name="PercentCompleted"
                   type="xsd:nonNegativeInteger" minOccurs="0"/>
   </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="ExecutionStatusType">
   <xsd:union>
      <xsd:simpleType>
         <xsd:restriction base="xsd:string">
            <xsd:enumeration value="cancelled"/>
            <xsd:enumeration value="completed"/>
            <xsd:enumeration value="executing"/>
            <xsd:enumeration value="pending"/>
         </xsd:restriction>
      </xsd:simpleType>
      <xsd:simpleType>
         <xsd:restriction base="xsd:string">
            <xsd:pattern value="other:\w{2,}"/>
         </xsd:restriction>
      </xsd:simpleType>
   </xsd:union>
</xsd:simpleType>
```

When an operation is invoked asynchronously, the server shall respond immediately with an ows:Acknowledgment message indicating that the server has successfully accepted the request or an OGC exception message indicating an error. If successfully accepted, the HTTP status code shall be set to "202 Accepted".

Asynchronous Processing class

Response semantics

For servers that implement the Asynchronous Processing conformance class, the ows:Acknowledgment message may contain an atom:link element, with rel="cancel", that may be used to cancel the asynchronously invoked operation.

The response to resolving the rel="cancel" link shall be an ows:Acknowledgment message that shall contains the wfs:Status element with its value set to "cancelled". The HTTP status code in this case shall be set to "200 OK".

NOTE	The "cancel" link may also be included in the response's HTTP header using the Link field (see RFC 5988).

NOTE This document does not define a specific template, form or encoding of any link that appears in an acknowledgement message. Server implementations are free to encode the URI value of the href attribute of an atom:link element in whatever way they deem suitable.

NOTE When resolving links, this should be done with the same credentials as the original asynchronous request.

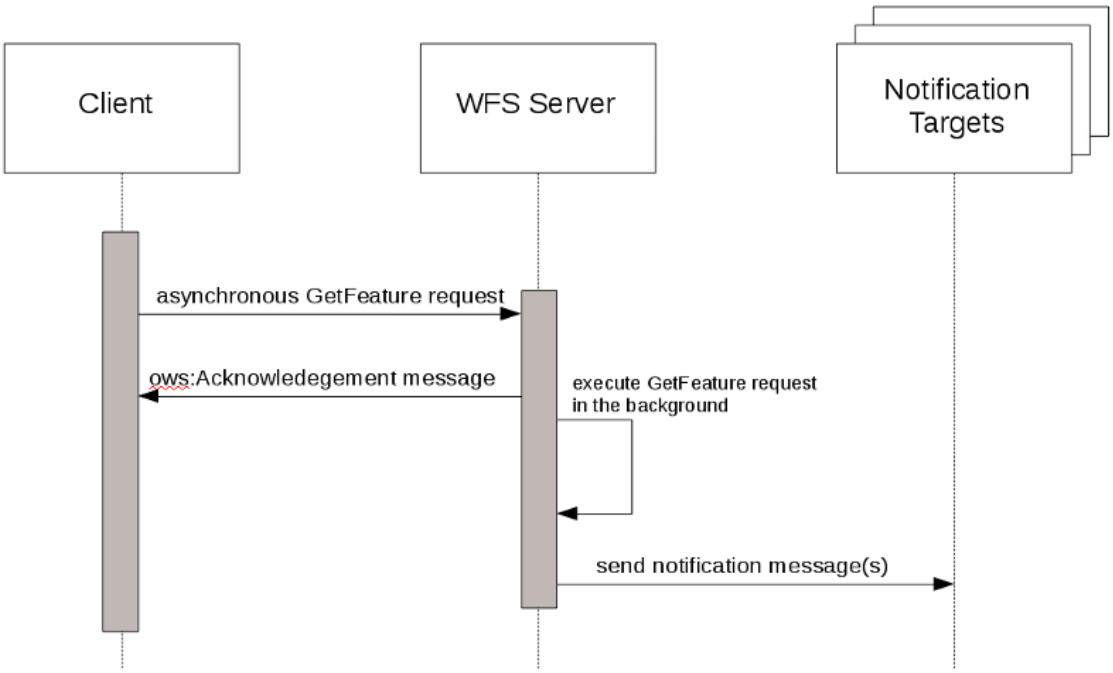

Figure 10. Sequence diagram for the Asynchronous Process Class

Examples

Example 1: Simple acknowledgement.

```
<ows:Acknowledgement/>
```

Example 2: Simple acknowledgement with a hypermedia control to cancel the request.

```
<ows:Acknowledgement>
   <atom:link rel="cancel" href="http://www.someserver.com/jobs/cancel/1013"/>
</ows:Acknowledgement>
```

Asynchronous Polling class

Response semantics

For servers that implement the Asynchronous Polling conformance class, the ows:Acknowledgment message shall include an atom:link element, with rel="monitor", that may be periodically resolved to determine the execution status of an asynchronous request.

The response to resolving the rel="monitor" link shall be a ows:Acknowledgment message that shall contain the wfs:Status element indicating the execution status of the asynchronous request and may include a wfs:PercentCompleted element with a percentage value indicating how much of the request has been completed. The HTTP status code in this case shall be set to "200 OK".

Requesting the execution status of an asynchronous request after its processing has been completed — and the operation's response is still available — shall result in an ows:Acknowledgement message that shall contain a wfs:Status element with its value set to "completed" and shall also include an atom:link element, with rel="http://www.opengis.net/def/rel/ogc/1.0/operationResponse", that provides a URI that may be used to retrieve the response.

Requesting the execution status of an asynchronous request after its processing has been completed — and the operation's response is no longer available (e.g. is has expired) — shall result in an OGC exception message and the HTTP status code shall be set to "404 Not Found".

NOTE The rel "http://www.opengis.net/def/rel/ogc/1.0/operationResponse" is an extension relation type (see RFC 5988, Section 4.2) and shall, in due course, be defined with OGC Naming Authority.

NOTE The "http://www.opengis.net/def/rel/ogc/1.0/operationResponse", "monitor" and "cancel" links may also be included in the response's HTTP header using the Link field (see RFC 5988).

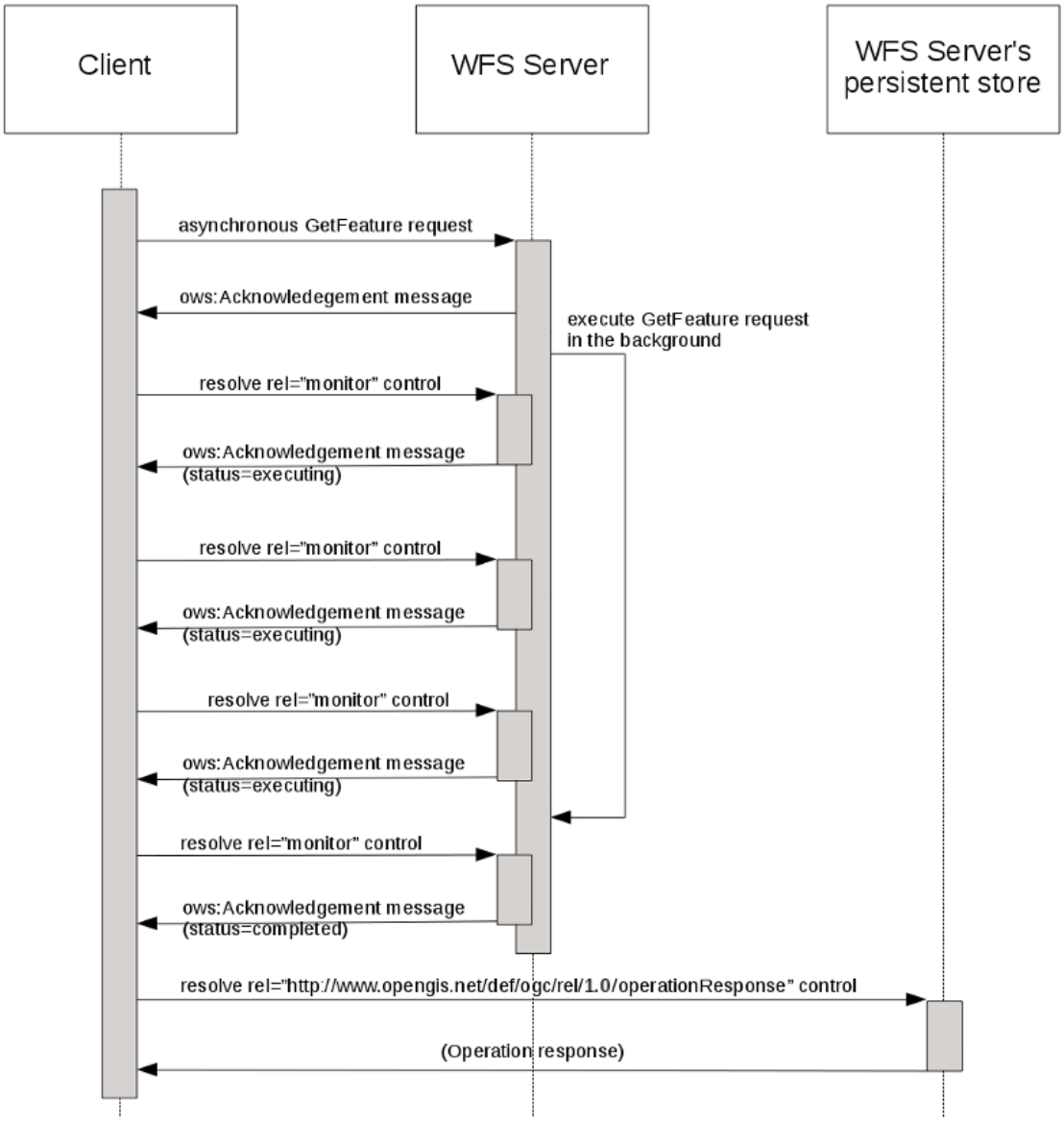

Figure 11. Sequence diagram for the Asynchronous Polling Class

Examples

Example 1: The following example shows the response that a server that implements the Asynchronous Polling conformance class might initially generate in response to an asynchronously invoked operation. The acknowledgement message contains hypermedia controls to monitor the execution status of the request and to cancel the request. The execution status at this time is *pending* indicating that the request has been queued for execution.

```
<ows:Acknowledgement>
   <atom:link rel="monitor" href="http://www.someserver.com/jobs/1013"/>
   <atom:link rel="cancel" href="http://www.someserver.com/jobs/cancel/1013"/>
   <ows:Status>pending</ows:Status>
</ows:Acknowledgement>
```

Example 2: The following example shows a polling response some time after an operation was invoked asynchronously. The acknowledgement message contains hypermedia controls to monitor the execution status of the request and to cancel the request.

```
<ows:Acknowledgement>
   <atom:link rel="monitor" href="http://www.someserver.com/jobs/1013"/>
   <atom:link rel="cancel" href="http://www.someserver.com/jobs/cancel/1013"/>
   <ows:Status>executing</ows:Status>
   <ows:PercentCompleted>47</ows:PercentCompleted>
</ows:Acknowledgement>
```

Example 3: This following examples shows the polling response after request processing has been completed. Resolving the hypermedia control with rel="http://www.opengis.net/def/rel/ogc/1.0/operationResponse" will retrieve the request's response if it is still available; if the response is not available (e.g. it has expired from the cache) resolving the control would result in an OGC exception message and a "404 Not Found".

```
<ows:Acknowledgement>
   <atom:link rel="http://www.opengis.net/def/rel/ogc/1.0/operationResponse"
              href="http://www.someserver.com/jobs/results/1013"/>
   <ows:Status>Completed</ows:Status>
</ows:Acknowledgement>
```

7.2.5. Notification message content

For servers that implement the Asynchronous Processing conformance class, an operation's response shall be accessible via the notification message sent by the server using the specified response handler(s) (see XML encoding) to signal that request processing has been completed.

In general the content of a notification message shall either be the operation's complete response, or a reference to it, or an exception message.

The specific content of a notification message is not defined in this document because it is dependent on the scheme(s) specified as the value of the ResponseHandler parameter (see XML encoding). For size-limited schemes, such as sms, a URL reference to the response would seem to be most appropriate since the entire response content is unlikely to fit into the message space. For other schemes, such as webhooks, the content of the notification message can be the complete response of the operation (e.g. the response to a GetFeature request). The following table contains informative recommendations for the content of notification messages based on the scheme being used:

Table 10. Recommended content for the notification message based on scheme

Notification scheme	Recommended content (good response)	Recommended content (exception)
mailto:	An email message containing a URL for retrieving the operation's response.	An email message containing a narrative that describes the exception; an optional attachment with the server's actual OGC exception message may also be included

Notification scheme	Recommended content (good response)	Recommended content (exception)
sms:	A URL for retrieving the operation's response; tiny URLs may be used if the retrieval URL is particularly long	A URL for retrieving the server's OGC exception message; tiny URLs may be used if the retrieval URL is particularly long
http: (webhook)	The operation's complete response	The complete OGC exception message

7.2.6. Capabilities document

Introduction

A server that implements the Asynchronous Processing conformance class shall advertise this fact in its capabilities document using the ImplementsAsyncProcessing service constraint.

A server that implements the Asynchronous Polling conformance class shall advertise this fact in its capabilities document using the ImplementsAsyncPolling service constraint.

A server that support asynchronous request processing shall, in its capabilities document, use the ResponseHandlerSchemes operation constraint to indicate which notification schemes it supports (e.g. mailto for email, http for webhooks, etc.) for each operation that may be executed asynchronously. The "poll" token shall be included in the list of response handler schemes if the service implements the Asynchronous Polling conformance class.

Examples

The following example shows the capabilities document for a server that implements the Asynchronous Processing conformance class.

```
   <?xml version="1.0" encoding="UTF-8"?>
   <WFS_Capabilities
      version="2.0.0"
      xmlns="http://www.opengis.net/wfs/2.0"
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xmlns:ows="http://www.opengis.net/ows/1.1"
      xmlns:cw="http://schemas.cubewerx.com/namespaces/null"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.opengis.net/wfs/2.0
http://schemas.cubewerx.com/schemas/wfs/2.0/wfs.xsd http://www.opengis.net/ows/1.1
http://schemas.cubewerx.com/schemas/ows/1.1.0/owsAll.xsd">
      <ows:ServiceIdentification xmlns:ows="http://www.opengis.net/ows/1.1">
         <ows:Title>CubeSERV WFS - New CubeSTOR (mysql)</ows:Title>
         <ows:Abstract xml:lang="en">WFS-compliant web feature server by CubeWerx
Inc.</ows:Abstract>
         <ows:ServiceType>WFS</ows:ServiceType>
         <ows:ServiceTypeVersion>2.5.0</ows:ServiceTypeVersion>
         <ows:ServiceTypeVersion>2.0.2</ows:ServiceTypeVersion>
         <ows:ServiceTypeVersion>2.0.0</ows:ServiceTypeVersion>
```

```
         <ows:ServiceTypeVersion>1.1.1</ows:ServiceTypeVersion>
         <ows:ServiceTypeVersion>1.1.0</ows:ServiceTypeVersion>
         <ows:ServiceTypeVersion>1.0.0</ows:ServiceTypeVersion>
      </ows:ServiceIdentification>
      <ows:ServiceProvider xmlns:ows="http://www.opengis.net/ows/1.1"
         xmlns:xlink="http://www.w3.org/1999/xlink">
         <ows:ProviderName>CubeWerx Inc.</ows:ProviderName>
         <ows:ProviderSite xlink:href="http://www.cubewerx.com/"/>
         <ows:ServiceContact>
            <ows:IndividualName>Mike Galluchon</ows:IndividualName>
            <ows:PositionName>Technical Support</ows:PositionName>
            <ows:ContactInfo>
               <ows:Phone>
                  <ows:Voice>(819) 771-8303</ows:Voice>
                  <ows:Facsimile>(819) 771-8388</ows:Facsimile>
               </ows:Phone>
               <ows:Address>
                  <ows:DeliveryPoint>815 boulevard de la Carrière, bureau
202</ows:DeliveryPoint>
                  <ows:City>Gatineau</ows:City>
                  <ows:AdministrativeArea>Québec</ows:AdministrativeArea>
                  <ows:PostalCode>J8Y 6T4</ows:PostalCode>
                  <ows:Country>Canada</ows:Country>

<ows:ElectronicMailAddress>support@cubewerx.com</ows:ElectronicMailAddress>
               </ows:Address>
            </ows:ContactInfo>
         </ows:ServiceContact>
      </ows:ServiceProvider>
      <ows:OperationsMetadata>
         <ows:Operation name="GetCapabilities">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="AcceptVersions">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
                  <ows:Value>2.0.0</ows:Value>
                  <ows:Value>1.1.1</ows:Value>
                  <ows:Value>1.1.0</ows:Value>
                  <ows:Value>1.0.0</ows:Value>
               </ows:AllowedValues>
```

```
            </ows:Parameter>
            <ows:Parameter name="AcceptFormats">
               <ows:AllowedValues>
                  <ows:Value>text/xml</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="Sections">
               <ows:AllowedValues>
                  <ows:Value>ServiceIdentification</ows:Value>
                  <ows:Value>ServiceProvider</ows:Value>
                  <ows:Value>OperationsMetadata</ows:Value>
                  <ows:Value>FeatureTypeList</ows:Value>
                  <ows:Value>SupportsGMLObjectTypeList</ows:Value>
                  <ows:Value>Filter_Capabilities</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <ows:Operation name="GetDescription">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="outputFormat">
               <ows:AllowedValues>
                  <ows:Value>text/html</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <ows:Operation name="DescribeFeatureType">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
```

```
                  <ows:Value>2.0.0</ows:Value>
                  <ows:Value>1.1.1</ows:Value>
                  <ows:Value>1.1.0</ows:Value>
                  <ows:Value>1.0.0</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="TypeName">
               <ows:AllowedValues>
                  <ows:Value>cw:builtupa_1m</ows:Value>
                  <ows:Value>cw:coastl_1m</ows:Value>
                  <ows:Value>cw:depthl_1m</ows:Value>
                  <ows:Value>cw:polbndl_1m</ows:Value>
                  <ows:Value>cw:soundings</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="outputFormat">
               <ows:AllowedValues>
                  <ows:Value>text/xml; subtype="bxfs/0.0.3"</ows:Value>
                  <!-- ... -->
                  <ows:Value>XMLSCHEMA</ows:Value>
                  <!-- ... -->
                  <ows:Value>application/gml+xml; version=3.2</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <ows:Operation name="ListStoredQueries">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
                  <ows:Value>2.0.0</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <ows:Operation name="DescribeStoredQueries">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
```

```
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
                  <ows:Value>2.0.0</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <ows:Operation name="GetPropertyValue">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
                  <ows:Value>2.0.0</ows:Value>
                  <ows:Value>1.1.1</ows:Value>
                  <ows:Value>1.1.0</ows:Value>
                  <ows:Value>1.0.0</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="outputFormat">
               <ows:AllowedValues>
                  <ows:Value>text/xml; subtype="bxfs/0.0.3"</ows:Value>
                  <!-- ... -->
                  <ows:Value>application/gml+xml; version=3.2</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="resolve">
               <ows:AllowedValues>
                  <ows:Value>none</ows:Value>
                  <ows:Value>local</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="srsName">
               <ows:AllowedValues>
```

```
                  <ows:Value>urn:ogc:def:crs:EPSG::42110</ows:Value>
                  <!-- ... -->
                  <ows:Value>urn:ogc:def:crs:EPSG::102002</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <ows:Operation name="GetFeature">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
                  <ows:Value>2.0.0</ows:Value>
                  <ows:Value>1.1.1</ows:Value>
                  <ows:Value>1.1.0</ows:Value>
                  <ows:Value>1.0.0</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="outputFormat">
               <ows:AllowedValues>
                  <ows:Value>text/xml; subtype="bxfs/0.0.3"</ows:Value>
                  <!-- ... -->
                  <ows:Value>application/gml+xml; version=3.2</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="resolve">
               <ows:AllowedValues>
                  <ows:Value>none</ows:Value>
                  <ows:Value>local</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Parameter name="srsName">
               <ows:AllowedValues>
                  <ows:Value>urn:ogc:def:crs:EPSG::42110</ows:Value>
                  <!-- ... -->
                  <ows:Value>urn:ogc:def:crs:EPSG::102002</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
            <ows:Constraint name="ResponseHandlerSchemes">
               <ows:AllowedValues>
                  <ows:Value>mailto:</ows:Value>
```

```
               </ows:AllowedValues>
            </ows:Constraint>
         </ows:Operation>
         <ows:Operation name="GetAccessibility">
            <ows:DCP>
               <ows:HTTP>
                  <ows:Get

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql&amp;
"/>
                  <ows:Post

xlink:href="http://tb12.cubewerx.com/cubewerx/cubeserv?datastore=foundation_mysql"/>
               </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="version">
               <ows:AllowedValues>
                  <ows:Value>2.5.0</ows:Value>
                  <ows:Value>2.0.2</ows:Value>
                  <ows:Value>2.0.0</ows:Value>
                  <ows:Value>1.1.1</ows:Value>
                  <ows:Value>1.1.0</ows:Value>
                  <ows:Value>1.0.0</ows:Value>
               </ows:AllowedValues>
            </ows:Parameter>
         </ows:Operation>
         <!-- ================================================================ -->
         <!--                      CONFORMANCE SECTION                         -->
         <!-- ================================================================ -->
         <ows:Constraint name="ImplementsBasicWFS">
            <ows:NoValues/>
            <ows:DefaultValue>TRUE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsTransactionalWFS">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsLockingWFS">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="KVPEncoding">
            <ows:NoValues/>
            <ows:DefaultValue>TRUE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="XMLEncoding">
            <ows:NoValues/>
            <ows:DefaultValue>TRUE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="SOAPEncoding">
            <ows:NoValues/>
```

```
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsInheritance">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsRemoteResolve">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsResultPaging">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsStandardJoins">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsSpatialJoins">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsTemporalJoins">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsFeatureVersioning">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ManageStoredQueries">
            <ows:NoValues/>
            <ows:DefaultValue>FALSE</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="ImplementsAsyncProcessing">
            <ows:NoValues/>
            <ows:DefaultValue>TRUE</ows:DefaultValue>
         </ows:Constraint>
         <!-- ============================================================== -->
         <ows:Constraint name="CountDefault">
            <ows:NoValues/>
            <ows:DefaultValue>10</ows:DefaultValue>
         </ows:Constraint>
         <ows:Constraint name="QueryExpression">
            <ows:AllowedValues>
               <ows:Value>wfs:Query</ows:Value>
               <ows:Value>wfs:StoredQuery</ows:Value>
            </ows:AllowedValues>
         </ows:Constraint>
      </ows:OperationsMetadata>
      <FeatureTypeList>
```

```
        <FeatureType>
           <Name>cw:builtupa_1m</Name>
           <Title>builtupa_1m</Title>
           <DefaultCRS>urn:ogc:def:crs:EPSG::4269</DefaultCRS>
           <OtherCRS>urn:ogc:def:crs:EPSG::42110</OtherCRS>
           <!-- ... -->
           <OtherCRS>urn:ogc:def:crs:EPSG::102002</OtherCRS>
           <ows:WGS84BoundingBox>
              <ows:LowerCorner>-179.1296 -53.1674</ows:LowerCorner>
              <ows:UpperCorner>178.4432 70.9927</ows:UpperCorner>
           </ows:WGS84BoundingBox>
        </FeatureType>
        <FeatureType>
           <Name>cw:coastl_1m</Name>
           <Title>coastl_1m</Title>
           <DefaultCRS>urn:ogc:def:crs:EPSG::4326</DefaultCRS>
           <OtherCRS>urn:ogc:def:crs:EPSG::42110</OtherCRS>
           <!-- ... -->
           <OtherCRS>urn:ogc:def:crs:EPSG::102002</OtherCRS>
           <ows:WGS84BoundingBox>
              <ows:LowerCorner>-179.999 -85.5827</ows:LowerCorner>
              <ows:UpperCorner>179.9999 83.6274</ows:UpperCorner>
           </ows:WGS84BoundingBox>
        </FeatureType>
        <FeatureType>
           <Name>cw:depthl_1m</Name>
           <Title>depthl_1m</Title>
           <DefaultCRS>urn:ogc:def:crs:EPSG::4269</DefaultCRS>
           <OtherCRS>urn:ogc:def:crs:EPSG::42110</OtherCRS>
           <!-- ... -->
           <OtherCRS>urn:ogc:def:crs:EPSG::102002</OtherCRS>
           <ows:WGS84BoundingBox>
              <ows:LowerCorner>-179.9999 -78.1118</ows:LowerCorner>
              <ows:UpperCorner>179.9999 89.9186</ows:UpperCorner>
           </ows:WGS84BoundingBox>
        </FeatureType>
        <FeatureType>
           <Name>cw:polbndl_1m</Name>
           <Title>polbndl_1m</Title>
           <DefaultCRS>urn:ogc:def:crs:EPSG::4269</DefaultCRS>
           <OtherCRS>urn:ogc:def:crs:EPSG::42110</OtherCRS>
           <!-- ... -->
           <OtherCRS>urn:ogc:def:crs:EPSG::102002</OtherCRS>
           <ows:WGS84BoundingBox>
              <ows:LowerCorner>-179.9999 -89.9999</ows:LowerCorner>
              <ows:UpperCorner>179.9999 89.9979</ows:UpperCorner>
           </ows:WGS84BoundingBox>
        </FeatureType>
        <FeatureType>
           <Name>cw:soundings</Name>
           <Title>soundings</Title>
```

```
            <DefaultCRS>urn:ogc:def:crs:EPSG::4326</DefaultCRS>
            <OtherCRS>urn:ogc:def:crs:EPSG::3857</OtherCRS>
            <!-- ... -->
            <OtherCRS>urn:ogc:def:crs:EPSG::102002</OtherCRS>
            <ows:WGS84BoundingBox>
               <ows:LowerCorner>-138.6394 69.9340</ows:LowerCorner>
               <ows:UpperCorner>-138.6035 69.9549</ows:UpperCorner>
            </ows:WGS84BoundingBox>
         </FeatureType>
      </FeatureTypeList>
      <Filter_Capabilities xmlns="http://www.opengis.net/fes/2.0" ...>
         <!-- ... -->
      </Filter_Capabilities>
   </WFS_Capabilities>
```

7.2.7. Implementation for Testbed-12

An implementation of the Asynchronous Processing class was deployed for Testbed-12 at the following URL on Amazon's Cloud:

```
http://tb12.cubewerx.com/a007/cubeserv?datastore=USGS&
   service=WFS&
   request=GetCapabilities
```

The implementation was unremarkable. As shown in the server's capabilities document, this deployment only implements the "mailto:" response handler. Technology integration experiments were successfully performed with the ESRI (DG002) and GMU (A109) thread clients.

The following is an example of the notification message that the A007 server generates in response to an asynchronous GetFeature request:

```
The feature set download that you requested on 2016-10-27 at 08:09:51 EDT
is now available at:

http://tb12.cubewerx.com/a007/cubeserv/default/cachedDownloads/urn:cubewerx:wfs:4353a3
f8-9c3e-11e6-a272-97ec260e3694:GetFeature:response

It is approximately 8KB in size, and will be available for the
next 72 hours (i.e., until 2016-10-30 at 08:09:51 EDT).
```

Chapter 8. Discussion

Three methods for implementing asynchronous service responses were investigated during Testbed-12:

1. WPS facades;
2. Specific extensions to each OGC Web Service (OWS) with asynchronous request/response capabilities;
3. OGC PubSub (described in [OGC 16-137]).

The three methods are discussed in the following:

8.1. WPS facades

WPS facades were successfully tested with WFS and WCS SOAP implementations, also including SOAP security. Due to its generic nature and role also as supporting service for other OGC Web services, the WPS is well suited to act as asynchronous facade (see WPS 1.0 standard [OGC 05-007r7], Introduction [1: The WPS 1.0 standard speaks of WPS wrapping other geoprocessing services. However, this can be extended to other services, e.g. for data retrieval or cataloging.]). The downside is that client will need to be able to communicate with the WPS interface instead of the original Web service. This could be aided by creating a WPS profile for asynchronous facade processes. For the profile, the inputs and outputs could be fixed, e.g. literal URI input for the endpoint of the original service and soap/xml complex input for the request, as well as soap/xml complex output for the response.

8.2. Specific extensions to each OGC Web Service with asynchronous request/response capabilities

The concept of adding asynchronous capabilities directly to the web service was tested successfully with WFS during Testbed-12. It seems to be a suitable solution to receive responses asynchronously. The downside of this solution is that the service interfaces need to be changed and the new capabilities cannot be used by older client implementations. A change request to the OGC Web Service Common Implementation Specification (OGC 06-121r9) has been posted to start discussion in the TC and finally the standardization process for the asynchronous extension method.

8.3. OGC Pubsub

The PubSub specification was tested with a Catalog Service to implement asynchronous responses. The use case was the notification of new records matching the client's criteria of interest. The PubSub specification is agnostic about what is defined as a change or event that causes a notification to be send. This could also be a finished WPS process or WFS transaction, hence the PubSub solution could use WPS facades or the specific extensions to OWS as delivery method.

Chapter 9. Recommendations

9.1. WPS facade

The experiments show that there is a tendency of less overhead in using the WPS facade for increasing data size. These experiments need to be extended to give the results more meaning. Also, the asynchronous WPS request workflow - initial request, status request, result request - should be incorporated. Above that, pull based mechanisms for WPS should be investigated, as well as methods for requesting partial results or tiling/paging of results.

9.2. Async WFS

The light-weight asynchronous request processing protocol described in the *Asynchronous request processing* clause of this ER is sufficiently generic that it could easily be implemented in other OGC web services as well. To that end, a change request has been posted (http://ogc.standardstracker.org/show_request.cgi?id=416) requesting that the description of the protocol be added to OGC Web Service Common Implementation Specification (OGC 06-121r9).

Appendix A: Revision History

Table 11. Revision History

Date	Release	Editor	Primary clauses modified	Descriptions
April 12, 2016	B. Pross	.1	all	initial version
April 12, 2016	P. Vretanos	.1	7	Added WFS section
April 15, 2016	B. Pross	.1	all	Added outline and relevance section
September 14, 2016	B. Pross	.1	7	Cleaned up, work on WPS implementation chapter
September 29, 2016	B. Pross	.1	all	Updated content
September 30, 2016	B. Pross	.1	all	Updated various sections
Octover 11, 2016	B. Pross	.1	6,7	Update background section, editorial changes, Added further test results
October 12, 2016	B. Pross	.1	7,8	Add more experiments, recommendations
October 17, 2016	B. Pross	.1	all	Incorporated comments from review
October 27, 2016	P. Vretanos	.1	8	Finalize draft ER
October 28, 2016	B. Pross	.1	2,3,9	Finalize draft ER
November 3, 2016	P. Vretanos	.1	9	Incorporated comments from review
November 7, 2016	B. Pross	.1	1	Incorporated comments from review

Date	Release	Editor	Primary clauses modified	Descriptions
November 30, 2016	B. Pross	.1	7.3	Incorporated comments from review

Appendix B: Bibliography

[1] OGC,: OGC Testbed 11 Demonstration. (2015).

[2] Lee, J., Zlatanova, S.: 3D geoinformation science. Springer (2009).