WPS Primer

In this primer we will demonstrate how to access and execute a remote prosessing service using the OGC Web Processing Service (WPS) protocol.

Find a WPS server

Find a data web source that hosts a WPS server, Deltares hosts a test server at http://dtvirt5.deltares.nl/wps/?

Request an overview of the capabilities of a WPS server

Ask for what the server has to offer with a standard WxS GetCapabilities call:

keyword	value	source
service	WPS	Mandatory WPS standard value
version	1.0.0	Optional for GetCapabilities
request	GetCapabilities	Mandatory WPS standard value
language	Usually 'en' (English)	Optional

```
http://dtvirt5.deltares.nl/wps/?
Request=GetCapabilities&
Service=WPS
```

This URL returns an xml file that contains an inventory of the available processes.

Inspect the overview of the content of a WPS server

Look at which WPS processes the server has to offer. For each process there is a tag `ows:identifier` with an identifier and title of the process, as shown in the reduced xml file example below they are `ogrbuffer`, `constituents`, `IDT_simple` and `status_test`:

```
<wps:Capabilities xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:
ows="http://www.opengis.net/ows/1.1"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" service="
WPS" version="1.0.0" xml:lang="en-CA" xsi:schemaLocation="http://www.opengis.net/wps/1.0.0 http://schemas.
opengis.net/wps/1.0.0/wpsGetCapabilities_response.xsd" updateSequence="1">
<ows:ServiceIdentification>
...
</ows:ServiceIdentification>
<ows:ServiceProvider>
...
</ows:ServiceProvider>
...
<ows:HTTP>
<ows:Get xlink:href="http://dtvirt5.deltares.nl/wps?"/>
<ows:Post xlink:href="http://dtvirt5.deltares.nl/wps"/>
</ows:HTTP>
...
<wps:ProcessOfferings>

<wps:Process wps:processVersion="0.1">
<ows:Identifier>ogrbuffer</ows:Identifier>
<ows:Title>Buffer process using OGR</ows:Title>
</wps:Process>
...
<wps:Process wps:processVersion="1">
<ows:Identifier>constituents</ows:Identifier>
<ows:Title>Lookup constituents based on their short name</ows:Title>
</wps:Process>

<wps:Process wps:processVersion="0.1">
<ows:Identifier>IDT_simple</ows:Identifier>
<ows:Title>Interactive Dredge Planning Tool</ows:Title>
</wps:Process>

<wps:Process wps:processVersion="1">
<ows:Identifier>status_test</ows:Identifier>
<ows:Title>echo</ows:Title>
<ows:Abstract>Echo server</ows:Abstract>
</wps:Process>

</wps:ProcessOfferings>
...
</wps:Capabilities>
```

Request a specific process of the WPS server

Where the `GetCapabilities` request returns a full summary of the server's capatbilities, subsequently the `DescribeProcess` request can be used to inquire about specific information on a single process:

keyword	value	source
service	WPS	Mandatory WPS standard value
version	1.0.0	Mandatory for DescribeProcess
request	DescribeProcess	Mandatory WPS standard value
identifer	process name	Mandatory parameter
language	Usually 'en' (English)	Optional

In this example we are going to use the `constituents` process, below the URL for the `DescribeProcess` request.

```
http://dtvirt5.deltares.nl/wps/?
request=DescribeProcess&
service=wps&
version=1.0.0&
identifier=constituents
```

This URL will return an xml file that contains a description of the requested processes.

```
<wps:ProcessDescriptions xmlns:wps="http://www.opengis.net/wps/1.0.0" 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/wps/1.0.0 http://schemas.opengis.net/wps/1.0.0
/wpsDescribeProcess_response.xsd" service="WPS" version="1.0.0" xml:lang="en-CA">
<ProcessDescription wps:processVersion="1" storeSupported="false" statusSupported="false">

<ows:Identifier>constituents</ows:Identifier>

<ows:Title>Lookup constituents based on their short name</ows:Title>

<ows:Abstract>
Lookup the speed of the constiuent based on the name of the constituent (following Doodson).
</ows:Abstract>

<DataInputs>

<Input minOccurs="1" maxOccurs="1">
<ows:Identifier>date</ows:Identifier>
<ows:Title>Date for which to calculate the nodal factors</ows:Title>
<LiteralData>
<ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#string">string</ows:DataType>
<ows:AnyValue/>
<DefaultValue/>
</LiteralData>
</Input>

<Input minOccurs="1" maxOccurs="1">
<ows:Identifier>constituent</ows:Identifier>
<ows:Title>Name of the constituent to look up (M2,...)</ows:Title>
<LiteralData>
<ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#string">string</ows:DataType>
<ows:AnyValue/>
</LiteralData>
</Input>

<Input minOccurs="1" maxOccurs="1">
<ows:Identifier>nodal</ows:Identifier>
<ows:Title>Calculate nodal factors</ows:Title>
<LiteralData>
<ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#boolean">boolean</ows:DataType>
<ows:AnyValue/>
<DefaultValue>False</DefaultValue>
</LiteralData>
</Input>

</DataInputs>

<ProcessOutputs>

<Output>
<ows:Identifier>u</ows:Identifier>
<ows:Title>u taken from Schureman</ows:Title>
<LiteralOutput>
<ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#float">float</ows:DataType>
</LiteralOutput>
</Output>

<Output>
```

```
<ows:Identifier>speed</ows:Identifier>
<ows:Title>Speed of the constituent (radians per hour)</ows:Title>
...
</Output>

<Output>
<ows:Identifier>FF</ows:Identifier>
<ows:Title>FF nodal factor, taken from Schureman</ows:Title>
<LiteralOutput>
<ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#float">float</ows:DataType>
</LiteralOutput>
</Output>

<Output>
<ows:Identifier>VAU</ows:Identifier>
<ows:Title>V+u taken from Schureman</ows:Title>
<LiteralOutput>
<ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#float">float</ows:DataType>
</LiteralOutput>
</Output>

</ProcessOutputs>
</ProcessDescription>
</wps:ProcessDescriptions>
```

In the `DescribeProcess` document all information about the specific process is given. In the document general information is given in a title and abstract. Besides these also different in- and outputs are defined including titles, datatypes, default values, etc. In the document above we find three different inputs and four different outputs.

Execute a specific process of the WPS server

After finding out the required input parameters for the specific process, subsequently the `Execute` request can be send:

keyword	value	source
service	WPS	Mandatory WPS standard value
version	1.0.0	Mandatory for Execute
request	Execute	Mandatory WPS standard value
identifer	process name	Mandatory parameter
DataInputs	inpute parameters	Mandatory for Execute
language	Usually 'en' (English)	Optional

Since processes can contain multiple inputs, the DataInputs parameter looks somewhat different. The shape of this parameter is shown below. The different parameters are all after the equal sign (=) of datainputs and between brackets. The different parameters are separated by semicolons (;). Which input parameters are required and how these can be used is found in the DescribeProcess document, at the <ows:identifier> tag the identifier of the input parameter is given, this identifier is used in the URL.

```
datainputs=[parameter_1=5;parameter_2=9;parameter_3=false]
```

For the purpose of this example only a single input value will be used in the execute request below.

```
http://dtvirt5.deltares.nl/wps/?
request=Execute&
service=wps&
version=1.0.0&
identifier=constituents&
datainputs=[constituent=M2]
```

The result of the Execute request is an XML document containing the different output parameters with corresponding output values.

```
<wps:ExecuteResponse xmlns:wps="http://www.opengis.net/wps/1.0.0" 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/wps/1.0.0 http://schemas.opengis.net/wps/1.0.0/wpsExecute_response.
xsd" service="WPS" version="1.0.0" xml:lang="en-CA" serviceInstance="http://dtvirt5.deltares.nl/wps?
service=WPS&request=GetCapabilities&version=1.0.0" statusLocation="http://dtvirt5.deltares.nl/outputwps/pywps-
137639971224.xml">

<wps:Process wps:processVersion="1">
<ows:Identifier>constituents</ows:Identifier>
<ows:Title>Lookup constituents based on their short name</ows:Title>
<ows:Abstract>...</ows:Abstract>
</wps:Process>

<wps:Status creationTime="2013-08-13T15:15:12Z">
<wps:ProcessSucceeded>PyWPS Process constituents successfully calculated</wps:ProcessSucceeded>
</wps:Status>

<wps:ProcessOutputs>

<wps:Output>
<ows:Identifier>u</ows:Identifier>
<ows:Title>u taken from Schureman</ows:Title>
<wps:Data>
<wps:LiteralData dataType="float" uom="None">12.5910330876</wps:LiteralData>
</wps:Data>
</wps:Output>

<wps:Output>
<ows:Identifier>speed</ows:Identifier>
<ows:Title>Speed of the constituent (radians per hour)</ows:Title>
<wps:Data>
<wps:LiteralData dataType="float" uom="r">0.505868049939</wps:LiteralData>
</wps:Data>
</wps:Output>

<wps:Output>
<ows:Identifier>FF</ows:Identifier>
<ows:Title>FF nodal factor, taken from Schureman</ows:Title>
<wps:Data>
<wps:LiteralData dataType="float" uom="None">1.0281626242</wps:LiteralData>
</wps:Data>
</wps:Output>

<wps:Output>
<ows:Identifier>VAU</ows:Identifier>
<ows:Title>V+u taken from Schureman</ows:Title>
<wps:Data>
<wps:LiteralData dataType="float" uom="None">16.2091031443</wps:LiteralData>
</wps:Data>
</wps:Output>

</wps:ProcessOutputs>
</wps:ExecuteResponse>
```

Asking for a specific output format.

The tidal_predict method supports both `text/csv` and `application/json`. The available attribute values are in the `DescribeProcess` xml file.

```
<wps:ProcessDescriptions ...>
<ProcessDescription ...>
<ProcessOutputs>
<Output><ows:Identifier>tide</ows:Identifier>
<ows:Title>Calculated water level for requested locations and date</ows:Title><ComplexOutput>
<Default>
   <Format><MimeType>text/csv</MimeType></Format>
</Default>
<Supported>
   <Format><MimeType>text/csv</MimeType></Format>
   <Format><MimeType>application/json</MimeType></Format>
</Supported></ComplexOutput>
</Output>
</ProcessOutputs>
</ProcessDescription>
</wps:ProcessDescriptions>
```

To get the non-default json output instead of the default csv output, add a keyword `&responsedocument` (outside `datainputs` brackets), using the WPS standard that you can specify attributes of a something by padding `=@attributename=attributename`. In this case we add an attribute `mimetype=application/json` to the output value `tide` as `&responsedocument=tide=@mimetype=application/json`, yielding the following url:

```
<?xml version="1.0" encoding="utf-8"?>
...
http://dtvirt5.deltares.nl/wps?request=Execute&service=wps&version=1.0.0
&identifier=tidal_predict
&datainputs=[location=LINESTRING%282%2052,3%2053%29;
            startdate=2020-01-01;
              enddate=2020-01-02]
&responsedocument=tide=@mimetype=application/json
```

Which gives:

```
<?xml version="1.0" encoding="utf-8"?>
<wps:ExecuteResponse xmlns:wps="http://www.opengis.net/wps/1.0.0" 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/wps/1.0.0 http://schemas.opengis.net/wps/1.0.0/wpsExecute_response.xsd"
service="WPS" version="1.0.0" xml:lang="en-CA" serviceInstance="http://dtvirt5.deltares.nl/wps?service=WPS&amp;
request=GetCapabilities&amp;version=1.0.0">
    <wps:Process wps:processVersion="0.1">
        <ows:Identifier>tidal_predict</ows:Identifier>
        <ows:Title>Tidal prediction tool</ows:Title>
        <ows:Abstract>
Tidal prediction tool can be used for different tidal prediction requests.
Prediction is calculated from Topex/Poseidon dataset.
Constitutuents provided by OSU TPXO.
</ows:Abstract>
    </wps:Process>
    <wps:Status creationTime="2013-08-18T22:22:11Z">
        <wps:ProcessSucceeded>PyWPS Process tidal_predict successfully calculated</wps:ProcessSucceeded>
    </wps:Status>
    <wps:ProcessOutputs>
        <wps:Output>
            <ows:Identifier>tide</ows:Identifier>
            <ows:Title>Calculated water level for requested locations and date</ows:Title>
            <wps:Data>
                <wps:ComplexData mimeType="application/json">
[{"date":1577836800000000000,"h":0.9055217703,"lat":52.0,"lon":2.0},{"date":1577840400000000000,"h":
0.5269456712,"lat":52.0,"lon":2.0},{"date":1577844000000000000,"h":-0.0264890836,"lat":52.0,"lon":2.0},{"date":
1577847600000000000,"h":-0.5541336177,"lat":52.0,"lon":2.0},{"date":1577851200000000000,"h":-0.9109530928,"lat":
52.0,"lon":2.0},{"date":1577854800000000000,"h":-1.0493201774,"lat":52.0,"lon":2.0},{"date":
1577858400000000000,"h":-0.9827940939,"lat":52.0,"lon":2.0},{"date":1577862000000000000,"h":-0.7274707407,"lat":
52.0,"lon":2.0},{"date":1577865600000000000,"h":-0.2892092144,"lat":52.0,"lon":2.0},{"date":
1577869200000000000,"h":0.2907139097,"lat":52.0,"lon":2.0},{"date":1577872800000000000,"h":0.8872141613,"lat":
52.0,"lon":2.0},{"date":1577876400000000000,"h":1.3083130215,"lat":52.0,"lon":2.0},{"date":1577880000000000000,"
h":1.3872365816,"lat":52.0,"lon":2.0},{"date":1577883600000000000,"h":1.085052368,"lat":52.0,"lon":2.0},{"date":
1577887200000000000,"h":0.5201416491,"lat":52.0,"lon":2.0},{"date":1577890800000000000,"h":-0.1020110267,"lat":
52.0,"lon":2.0},{"date":1577894400000000000,"h":-0.6028626472,"lat":52.0,"lon":2.0},{"date":
1577898000000000000,"h":-0.9006199205,"lat":52.0,"lon":2.0},{"date":1577901600000000000,"h":-0.9942235126,"lat":
52.0,"lon":2.0},{"date":1577905200000000000,"h":-0.8999725194,"lat":52.0,"lon":2.0},{"date":
1577908800000000000,"h":-0.6170361442,"lat":52.0,"lon":2.0},{"date":1577912400000000000,"h":-0.1592478845,"lat":
52.0,"lon":2.0},{"date":1577916000000000000,"h":0.3852466596,"lat":52.0,"lon":2.0},{"date":1577919600000000000,"
h":0.8422715702,"lat":52.0,"lon":2.0},{"date":1577923200000000000,"h":1.0239596024,"lat":52.0,"lon":2.0},
{"date":1577836800000000000,"h":0.3589546692,"lat":53.0,"lon":3.0},{"date":1577840400000000000,"h":
0.1607531366,"lat":53.0,"lon":3.0},{"date":1577844000000000000,"h":-0.1128007448,"lat":53.0,"lon":3.0},{"date":
1577847600000000000,"h":-0.360735561,"lat":53.0,"lon":3.0},{"date":1577851200000000000,"h":-0.5135657472,"lat":
53.0,"lon":3.0},{"date":1577854800000000000,"h":-0.5523997776,"lat":53.0,"lon":3.0},{"date":
1577858400000000000,"h":-0.488455418,"lat":53.0,"lon":3.0},{"date":1577862000000000000,"h":-0.332647575,"lat":
53.0,"lon":3.0},{"date":1577865600000000000,"h":-0.0899128505,"lat":53.0,"lon":3.0},{"date":
1577869200000000000,"h":0.2167441457,"lat":53.0,"lon":3.0},{"date":1577872800000000000,"h":0.5221087736,"lat":
53.0,"lon":3.0},{"date":1577876400000000000,"h":0.72895627,"lat":53.0,"lon":3.0},{"date":1577880000000000000,"
h":0.7560079569,"lat":53.0,"lon":3.0},{"date":1577883600000000000,"h":0.5892076622,"lat":53.0,"lon":3.0},
{"date":1577887200000000000,"h":0.2938556551,"lat":53.0,"lon":3.0},{"date":1577890800000000000,"h":
-0.0239550777,"lat":53.0,"lon":3.0},{"date":1577894400000000000,"h":-0.276009934,"lat":53.0,"lon":3.0},{"date":
1577898000000000000,"h":-0.425450966,"lat":53.0,"lon":3.0},{"date":1577901600000000000,"h":-0.475583489,"lat":
53.0,"lon":3.0},{"date":1577905200000000000,"h":-0.4359643097,"lat":53.0,"lon":3.0},{"date":
1577908800000000000,"h":-0.3055336229,"lat":53.0,"lon":3.0},{"date":1577912400000000000,"h":-0.0905682739,"lat":
53.0,"lon":3.0},{"date":1577916000000000000,"h":0.1642704437,"lat":53.0,"lon":3.0},{"date":1577919600000000000,"
h":0.3712641261,"lat":53.0,"lon":3.0},{"date":1577923200000000000,"h":0.4383804482,"lat":53.0,"lon":
3.0}]
</wps:ComplexData>
            </wps:Data>
        </wps:Output>
    </wps:ProcessOutputs>
</wps:ExecuteResponse>
```


The service is validated with the timeseries from live.getij.nl (Dutch tidal information)