WaterML2Import

WaterML2 import

This is an import that makes use of the custom import mechanism described in Custom time series import. The WaterML2 import consists of a xml parser, WaterMLTimeSeriesParser.java, a server parser, WaterMlServerParser.java and a bin directory containing all dependant library files (WaterMlService-stable-201402-bin.zip).

The WaterML standard has been adopted as an OGC standard. The specifications of the WaterML standard can be found at OGC-WaterML2. This import make use of a Request - Response mechanism. First the WaterMlServerParser makes an XML request file and sends this to the WaterMl2 server. Depending on the request a XML Response file is returned containing the timeseries information. The WaterMlServerParser then hands the returned content over to the WaterMlTimeSeriesParser which extracts the timeseries data from the response.

It is possible to setup the WaterMl2 import to read from a WaterMl2 webserver using the WaterMlServerParser or it is possible to read directly form an import directory using the WaterMLTimeSeriesParser.

TimeSeries XML Request format

Here is an example of an XML request file:

```
<?xml version="1.0" ?>
<sos:GetObservation version="2.0.0" service="SOS"
    maxFeatures="3"
    xmlns:sos="http://schemas.opengis.net/sos/2.0.0/"
    xmlns:wfs="http://www.opengis.net/wfs"
    xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:gml="http://www.opengis.net/gml/3.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:om="http://www.opengis.net/om/2.0"
    xmlns:fes="http://www.opengis.net/fes/2.0"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xsi:schemaLocation="http://www.opengis.net/sos/2.0 http://schemas.opengis.net/sos/2.0.0/sos.xsd">
    <sos:featureOfInterest>01446500</sos:featureOfInterest>
    <sos:observedProperty>Discharge</sos:observedProperty>
    <!-- OPTIONAL -->    <om:procedure>Continuous/Instantaneous</om:procedure>
    <!-- OPTIONAL -->
    <om:offering>UNIT</om:offering>
    <sos:temporalFilter>
        <fes:ValueReference>phenomenonTime</fes:ValueReference>
        <gml:TimeInstant gml:id='beginPosition'>
            <gml:timePosition>2012-06-29</gml:timePosition>
        </gml:TimeInstant>
        <gml:TimeInstant gml:id='endPosition'>
            <gml:timePosition>2012-07-06</gml:timePosition>
        </gml:TimeInstant>
    </sos:temporalFilter>
</sos:GetObservation>
```

TimeSeries XML Response format

Here is an example of an XML response file:

```
<wml2:Collection xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:om="http://www.opengis.net/om/2.0"
    xmlns:sa="http://www.opengis.net/sampling/2.0" xmlns:swe="http://www.opengis.net/swe/2.0"
    xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:wml2="http://www.opengis.net/waterml/2.0"
    xmlns:x-wml2="http://www.opengis.net/waterml/2.0" xmlns:gmd="http://www.isotc211.org/2005/gmd"
    xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:sf="http://www.opengis.net/sampling/2.0"
    xmlns:sams="http://www.opengis.net/samplingSpatial/2.0"
    xmlns:gts="http://www.isotc211.org/2005/gts" xmlns:gss="http://www.isotc211.org/2005/gss"
    xmlns:gsr="http://www.isotc211.org/2005/gsr"
    xsi:schemaLocation="http://www.opengis.net/waterml/2.0 http://nwisvaws02.er.usgs.gov/ogc-swie/schemas
/waterml2.xsd"
    gml:id="TO_BE_DETERMINED">
    <gml:identifier codeSpace="http://nwis.waterdata.usgs.gov/NJ/nwis"
        >USGS.01446500</gml:identifier>
    <gml:name codeSpace="http://nwis.waterdata.usgs.gov/NJ/nwis">Delaware River at Belvidere
        NJ</gml:name>
    <wml2:metadata>
        <wml2:DocumentMetadata gml:id="doc.USGS.MP.USGS.01446500">
            <gml:metaDataProperty xlink:href="contact">
                <gml:GenericMetaData>http://cida.usgs.gov</gml:GenericMetaData>
            </gml:metaDataProperty>
            <wml2:generationDate>2012-07-06T05:58:44</wml2:generationDate>
            <wml2:version xlink:href="http://www.opengis.net/waterml/2.0" xlink:title="WaterML 2.0"
            />
        </wml2:DocumentMetadata>
    </wml2:metadata>
    <wml2:observationMember>
        <om:OM_Observation gml:id="obs.USGS.01446500">
            <om:metadata>
                <wml2:ObservationMetadata>
                    <gmd:contact>
                        <gmd:CI_ResponsibleParty>
                            <gmd:organisationName>
                                <gco:CharacterString>New Jersey Water Science
                                    Center</gco:CharacterString>
                            </gmd:organisationName>
                            <gmd:role>
                                <gmd:CI_RoleCode
                                    codeList="http://www.isotc211.org/2005/resources/CodeList/gmxCodelists.
xml#CI_RoleCode"
                                    codeListValue="owner"/>
                            </gmd:role>
                        </gmd:CI_ResponsibleParty>
                    </gmd:contact>
                    <gmd:dateStamp>
                        <gco:DateTime>2012-07-06T05:58:44</gco:DateTime>
                    </gmd:dateStamp>
                    <gmd:identificationInfo/>
                </wml2:ObservationMetadata>
            </om:metadata>
            <om:phenomenonTime>
                <gml:TimePeriod gml:id="pt.USGS.01446500">
                    <gml:beginPosition>2012-06-29</gml:beginPosition>
                    <gml:endPosition>2012-07-06</gml:endPosition>
                </gml:TimePeriod>
            </om:phenomenonTime>
            <om:resultTime>
                <gml:TimeInstant gml:id="forecast.2012-07-06">
                    <gml:timePosition>2012-07-06T05:58:44</gml:timePosition>
                </gml:TimeInstant>
            </om:resultTime>
            <om:validTime>
                <gml:TimePeriod gml:id="vt.USGS.01446500">
                    <gml:beginPosition>2012-06-29</gml:beginPosition>
                    <gml:endPosition>2012-07-06</gml:endPosition>
                </gml:TimePeriod>
            </om:validTime>
            <om:procedure xlink:href="http://www.nemi.gov/Continuous/Instantaneous"
                xlink:title="Continuous/Instantaneous"/>
            <om:observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:Discharge"
```

```
                xlink:title="Discharge"/>
            <om:featureOfInterest>
                <wml2:MonitoringPoint gml:id="USGS.MP.01446500">
                    <sf:sampledFeature
                        xlink:href="http://nwisvaws02.er.usgs.gov/ogc-swie/wfs?request=GetFeature&amp;
featureId=01446500"/>
                    <sf:parameter>
                        <om:NamedValue>
                            <om:name xlink:title="Watershed"/>
                            <om:value>Middle Delaware-Musconetcong</om:value>
                        </om:NamedValue>
                    </sf:parameter>
                    <sams:shape>
                        <gml:Point gml:id="USGS.P.01446500">
                            <gml:pos srsName="urn:ogc:def:crs:EPSG:4269">40.82638889
                                -75.08250000</gml:pos>
                        </gml:Point>
                    </sams:shape>
                    <wml2:descriptionReference
                        xlink:href="http://external.opengis.org/twiki_public/bin/view/HydrologyDWG
/SurfacewaterInteroperabilityExperiment#Use_Case_2"
                        xlink:title="This wiki page describes the IE"/>
                    <wml2:timeZone>
                        <wml2:TimeZone>
                            <wml2:zoneOffset>+00:00</wml2:zoneOffset>
                            <wml2:zoneAbbreviation>EDT</wml2:zoneAbbreviation>
                        </wml2:TimeZone>
                    </wml2:timeZone>
                </wml2:MonitoringPoint>
            </om:featureOfInterest>
            <om:result>
                <wml2:MeasurementTimeseries gml:id="ts_01446500_00060_00000">
                    <wml2:metadata>
                        <wml2:TimeseriesMetadata>
                            <wml2:temporalExtent>
                                <gml:TimePeriod gml:id="USGS.TP.00060_00000">
                                    <gml:beginPosition>2012-06-29</gml:beginPosition>
                                    <gml:endPosition>2012-07-06</gml:endPosition>
                                </gml:TimePeriod>
                            </wml2:temporalExtent>
                        </wml2:TimeseriesMetadata>
                    </wml2:metadata>
                    <wml2:defaultPointMetadata>
                        <wml2:DefaultTVPMeasurementMetadata>
                            <wml2:qualifier xlink:href="http://waterdata.usgs.gov/NJ/nwis/help"
                                xlink:title="Provisional data subject to revision."/>
                            <wml2:uom uom="cfs"/>
                            <wml2:interpolationType
                                xlink:href="http://www.opengis.net/def/interpolationType/WaterML/2.0/Continuous"
                                xlink:title="Continuous/Instantaneous"/>
                        </wml2:DefaultTVPMeasurementMetadata>
                    </wml2:defaultPointMetadata>
                    <wml2:point>
                        <wml2:MeasurementTVP>
                            <wml2:time>2012-06-29T00:00:00-04:00</wml2:time>
                            <wml2:value>2860</wml2:value>
                        </wml2:MeasurementTVP>
                    </wml2:point>
                    <wml2:point>
                        <wml2:MeasurementTVP>
                            <wml2:time>2012-06-29T00:15:00-04:00</wml2:time>
                            <wml2:value>2860</wml2:value>
                        </wml2:MeasurementTVP>
                    </wml2:point>
                    <wml2:point>
                        <wml2:MeasurementTVP>
                            <wml2:time>2012-06-29T00:30:00-04:00</wml2:time>
                            <wml2:value>2860</wml2:value>
                        </wml2:MeasurementTVP>
                    </wml2:point>
                    <wml2:point>
```

```
                        <wml2:MeasurementTVP>
                            <wml2:time>2012-06-29T00:45:00-04:00</wml2:time>
                            <wml2:value>2860</wml2:value>
                        </wml2:MeasurementTVP>
                    </wml2:point>
                    <wml2:point>
                        <wml2:MeasurementTVP>
                            <wml2:time>2012-06-29T01:00:00-04:00</wml2:time>
                            <wml2:value>2860</wml2:value>
                        </wml2:MeasurementTVP>
                    </wml2:point>
<!-- A lot of points inbetween -->
                    <wml2:point>
                        <wml2:MeasurementTVP>
                            <wml2:time>2012-07-06T05:30:00-04:00</wml2:time>
                            <wml2:value>2690</wml2:value>
                        </wml2:MeasurementTVP>
                    </wml2:point>
                </wml2:MeasurementTimeseries>
            </om:result>
        </om:OM_Observation>
    </wml2:observationMember>
</wml2:Collection>
```

Fews configuration

In order to activate the WaterML import as a FEWS import, it is required to setup a **TimeSeriesImportRun** module configuration file and an accompanying **IdMap** file. Also the bin directory containing all WaterML resources must be placed in a location that can be accessed by the FEWS system.

WaterML2 Server import

As of Stable 2017.02 the WaterMlServerParser and WaterMlTimeSeriesParser have been moved to the module lib-parsers-serializer. This implies that they can be configured using the importTypeStandard option and that there is no longer a need to provide the parser classes is a separate binary directory.

Version 2017.02 and later

Here is a 2017.02 and later example import module configuration file that imports data from a WaterMl2 webserver:

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews"
                     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                     xsi:schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0
/timeSeriesImportRun.xsd">
    <!-- This is an example import configuration file for importing WaterML data from a WaterMl server    -->
    <import>
        <general>
                        <importTypeStandard>wml2_server</importTypeStandard>
            <serverUrl>http://nwisvaws02.er.usgs.gov/ogc-swie/wml2/uv/sos</serverUrl>
                        <relativeViewPeriod unit="hour" end="0" start="-3"/>
                        <idMapId>IdImportWaterML2_usgs</idMapId>
            <importTimeZone>
                <timeZoneOffset>-06:00</timeZoneOffset>
            </importTimeZone>
        </general>
        <properties>
            <!-- Optional: Use this option to write request and response messages to file. -->
            <string key="RequestsOutputDirectory" value="c:/temp/"/>
            <!-- Optional: Use this option to define what the request string must look like. This depends on
how the WaterML2 webservice is setup. By using tags in the
                 request template it is possible to insert the import parameters at run-time.
                 The following tags can be set:
                                @starttime@ = start time request period
                                @endtime@ = end time of request period
                                @locationid@ = location identifier
                                @parameterid@ = parameter identifier
                                @qualifier0@ = qualifier value 0 (can be used freely for additional purposes
if required)
                                @qualifier1@ = qualifier value 1
                                @qualifier2@ = qualifier value 2
                                @qualifier3@ = qualifier value 3
            -->
            <string key="requestTemplate" value="format=wml2&amp;from=@starttime@&amp;to=@endtime@&amp;
loc_id=@locationid@@&amp;par_id=@parameterid@"/>
            <!-- Optional: Maximum time in millis before a read action times-out. Read time is defined by the
time betweend sending a request and start reading the response -->
            <int key="ReadTimeoutMillis" value="10000"/>
            <!-- Optional: Maximum time in millis for making the connection to the WaterML2 service. -->
            <int key="ConnectionTimeoutMillis" value="1000"/>
            <!-- Optional: Option to import multiple locations in single request. -->
            <bool key="RequestMultipleLocations" value="true"/>
        </properties>
        <timeSeriesSet>
            <moduleInstanceId>ImportWaterML2_usgs</moduleInstanceId>
            <valueType>scalar</valueType>
            <parameterId>MyPar</parameterId>
            <locationSetId>MyLocSet</locationSetId>
            <timeSeriesType>external historical</timeSeriesType>
            <timeStep unit="nonequidistant"/>
            <readWriteMode>add originals</readWriteMode>
            <synchLevel>1</synchLevel>
        </timeSeriesSet>
    </import>
</timeSeriesImportRun>
```

Note that when using qualifiers in the timeSeriesSet, in the idMapping both an internal and external qualifier need to be defined (with the same value):

<map externalParameter="206194010" internalParameter="DP.obs" internalQualifier="normal" externalQualifier="normal"/>

Versions before 2017.02

Here is a pre-2017.02 example import module configuration file that imports data from a WaterMl2 webserver:

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews"
                     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                     xsi:schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0
/timeSeriesImportRun.xsd">
    <!-- This is an example import configuration file for importing WaterML data from a WaterMl server    -->
    <import>
        <general>
           <!-- Class name of WaterML server parser -->
            <parserClassName>nl.wldelft.waterml.timeseriesparsers.WaterMlServerParser</parserClassName>

            <!-- Path to directory containing libraries -->
            <binDir>%REGION_HOME%/Modules/waterml-bin</binDir>

            <!-- Directory from which CSV files are to be imported -->
            <serverUrl>http://nwisvaws02.er.usgs.gov/ogc-swie/wml2/uv/sos</serverUrl>
            <idMapId>IdImportWaterML2_usgs</idMapId>
            <importTimeZone>
                <timeZoneOffset>-06:00</timeZoneOffset>
            </importTimeZone>
        </general>
        <properties>
            <!-- Optional: Use this option to write request and response messages to file. -->
            <string key="RequestsOutputDirectory" value="c:/temp/"/>
            <!-- Optional: Use this option to define what the request string must look like. This depends on
how the WaterML2 webservice is setup. By using tags in the
                  request template it is possible to insert the import parameters at run-time.
                  The following tags can be set:
                                 @starttime@ = start time request period
                                 @endtime@ = end time of request period
                                 @locationid@ = location identifier
                                 @parameterid@ = parameter identifier
                                 @qualifier0@ = qualifier value 0 (can be used freely for additional purposes
if required)
                                 @qualifier1@ = qualifier value 1
                                 @qualifier2@ = qualifier value 2
                                 @qualifier3@ = qualifier value 3
            -->
            <string key="requestTemplate" value="format=wml2&amp;from=@starttime@&amp;to=@endtime@&amp;
loc_id=@locationid@@&amp;par_id=@parameterid@"/>
            <!-- Optional: Maximum time in millis before a read action times-out. Read time is defined by the
time betweend sending a request and start reading the response -->
            <int key="ReadTimeoutMillis" value="10000"/>
            <!-- Optional: Maximum time in millis for making the connection to the WaterML2 service. -->
            <int key="ConnectionTimeoutMillis" value="1000"/>
            <!-- Optional: Option to import multiple locations in single request. -->
            <bool key="RequestMultipleLocations" value="true"/>

        </properties>
        <timeSeriesSet>
            <moduleInstanceId>ImportWaterML2_usgs</moduleInstanceId>
            <valueType>scalar</valueType>
            <parameterId>MyPar</parameterId>
            <locationSetId>MyLocSet</locationSetId>
            <timeSeriesType>external historical</timeSeriesType>
            <timeStep unit="nonequidistant"/>
            <readWriteMode>add originals</readWriteMode>
            <synchLevel>1</synchLevel>
        </timeSeriesSet>
    </import>
</timeSeriesImportRun>
```

WaterML2 file import

Version 2017.02 and later

Here is a 2017.02 and later example import module configuration file that imports data from a directory

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews"
                     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                     xsi:schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0
/timeSeriesImportRun.xsd">
    <!-- This is an example import configuration file for importing WaterML data from a WaterMl server    -->
    <import>
        <general>
            <importTypeStandard>wml2</importTypeStandard>

            <!-- Directory from which CSV files are to be imported -->
                  <folder>$IMPORT_FOLDER_WATERML$</folder>
            <idMapId>IdImportWaterML2_usgs</idMapId>
            <importTimeZone>
                <timeZoneOffset>-06:00</timeZoneOffset>
            </importTimeZone>
        </general>
        <timeSeriesSet>
            <moduleInstanceId>ImportWaterML2_usgs</moduleInstanceId>
            <valueType>scalar</valueType>
            <parameterId>MyPar</parameterId>
            <locationSetId>MyLocSet</locationSetId>
            <timeSeriesType>external historical</timeSeriesType>
            <timeStep unit="nonequidistant"/>
            <readWriteMode>add originals</readWriteMode>
            <synchLevel>1</synchLevel>
        </timeSeriesSet>
    </import>
</timeSeriesImportRun>
```

Versions before 2017.02

Here is a pre-2017.02 example import module configuration file that imports data from a directory:

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews"
                     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                     xsi:schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0
/timeSeriesImportRun.xsd">
    <!-- This is an example import configuration file for importing WaterML data from a WaterMl server    -->
    <import>
        <general>
           <!-- Class name of WaterML server parser -->
            <parserClassName>nl.wldelft.waterml.timeseriesparsers.WaterMlTimeSeriesParser</parserClassName>

            <!-- Path to directory containing libraries -->
            <binDir>%REGION_HOME%/Modules/waterml-bin</binDir>

            <!-- Directory from which CSV files are to be imported -->
            <folder>$IMPORT_FOLDER_WATERML$</folder>
            <idMapId>IdImportWaterML2_usgs</idMapId>
            <importTimeZone>
                <timeZoneOffset>-06:00</timeZoneOffset>
            </importTimeZone>
        </general>
        <timeSeriesSet>
            <moduleInstanceId>ImportWaterML2_usgs</moduleInstanceId>
            <valueType>scalar</valueType>
            <parameterId>MyPar</parameterId>
            <locationSetId>MyLocSet</locationSetId>
            <timeSeriesType>external historical</timeSeriesType>
            <timeStep unit="nonequidistant"/>
            <readWriteMode>add originals</readWriteMode>
            <synchLevel>1</synchLevel>
        </timeSeriesSet>
    </import>
</timeSeriesImportRun>
```

WaterML2 Authentication

Currently the server import support two types of authentication: Basic Authentication and OAuth2.

Basic Authentication

Here is an configuration example showing how to configure Basic Authentication. Only a user name and password need to be configured.

```
<general>
        <importTypeStandard>wml2_server</importTypeStandard>
    <serverUrl>https://dummy_hostname/KiWIS/KiWIS</serverUrl>
    <user>dummy_username</user>
    <password>dummy_password</password>
        ...
 </general>
```

As of release 2017.02 it is also possible to configure OAuth2 authentication. This functionality has been backported to Stable 2016.02 and 2017.01, however the way to configure this in the older releases differs.

OAuth2 Authentication version 2017.02 and later

```
<general>
        <importTypeStandard>wml2_server</importTypeStandard>
    <serverUrl>https://dummy_hostname/KiWIS/KiWIS</serverUrl>
    <!-- <user>dummy_username</user> -->                                <!-- normally not required for OAuth2 --
>
    <!-- <password>dummy_password</password> -->        <!-- normally not required for OAuth2 -->
    <oauth2Config>
                <!-- Required: URL from which to receive the access token -->
            <authUrl>https://dummy_hostname/auth</authUrl>
                 <!-- Optional: For proper OAuth2 authentication a client_id and client_secret are required.
However in some cases the authentication URL does not require this.
                          Instead Basic Authentication is required to access the authUrl. Here the user and
password fields shown above are required -->
                <clientId>openid client id</clientId>
                <clientSecret>openid client secret</clientSecret>
                <!-- Optional Array: Scope of request -->
        <scope>openid</scope>
        <scope>email</scope>
                <!-- Optional Array: Audience for whom request is intended. Used to validate response. If
omitted the clientId and username become are used -->
        <audience>audienceId</audience>
        <audience>audienceId2</audience>
       <!-- Optional: Issuer of the access token. Used to validate response. If omitted the root url of authUrl
is used. -->
        <issuer>http://localhost:8080/KiWebPortal/auth</issuer>
       <!-- Optional: A refresh token can be used if provider supports this. -->
        <refreshToken>refresh access token</refreshToken>
        </oauth2Config>
        ...
 </general>
```

OAuth2 Authentication versions before 2017.02

```
<general>
        <importTypeStandard>wml2_server</importTypeStandard>
    <serverUrl>https://dummy_hostname/KiWIS/KiWIS</serverUrl>
    <!-- <user>dummy_username</user> -->                                <!-- normally not required for OAuth2 --
>
    <!-- <password>dummy_password</password> -->        <!-- normally not required for OAuth2 -->
        ...
 </general>
<properties>
   <string key="authUrl" value="https://dummy_hostname/auth" />
   <string key="issuer" value="http://dummy_hostname_2:8080/KiWebPortal/auth" />
   <string key="clientId" value="id...." />
   <string key="clientSecret" value="*****" />
</properties>
```

WaterML2 IdMap configuration

Here is an example id-map file:

```
<?xml version="1.0" encoding="UTF-8"?>
<idMap xmlns="http://www.wldelft.nl/fews" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0/idMap.xsd" version="1.1">
    <!-- Id mapping for import from WaterML 2 service
        Required fields:
         externalLocation: maps to featureId. Enter value that is inserted in URL request
         externalParameter: maps to observedProperty. Enter value that is inserted in URL request

       Optional fields:
         externalQualifier: maps to procedure. Enter value that is inserted in URL request.
         If not required but 'offering' is required then fill in 'unknown'.
         externalQualifier1: maps to offering.  Enter value that is inserted in URL request.
         If not required can be omitted or fill in 'unknown'.
     -->
    <map internalLocation="LOC-001" internalParameter="PAR-001" externalLocation="featureId"
         externalParameter="observedProperty" externalQualifier="procedure" externalQualifier1="offering" />

    <!-- Here some examples to configure Procedure and Offering -->
    <map internalLocation="LOC-001" internalParameter="PAR-001" externalLocation="featureId"
         externalParameter="observedProperty" externalQualifier="procedure" />

    <map internalLocation="LOC-001" internalParameter="PAR-001" externalLocation="featureId"
         externalParameter="observedProperty" externalQualifier="unknown" externalQualifier1="offering" />

    <map internalLocation="LOC-001" internalParameter="PAR-001" externalLocation="featureId"
         externalParameter="observedProperty" />


 </idMap>
```

Id-Mapping goes as follows:

Internal timeseries location and parameter combinations are mapped to external location, parameter and qualifier sets as follows:

- externalLocation values are written as Feature Id
- externalParameter values are written as ObservedProperty
- 1st externalQualifier values are written as procedure (if not required fill in 'unknown')
- 2nd externalQualifier values are written as offering (if not required fill in 'unknown' or omit)

WaterML2 FlagConversions configuration

Only required when importing WaterML2 files containing 'qualifier' elements. These elements will be used to map timeseries flag values. When the import can contain 'qualifier' values but their values are not known then it is important to define a 'defaultOutputFlag' value in order to avoid errors in the log.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--NFFS EA TimeSeriesDataExchangeFormat flag conversion file for Import-->
<flagConversions xmlns="http://www.wldelft.nl/fews" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0/flagConversions.xsd">
    <flagConversion>
        <inputFlag>
            <name>External01</name>
            <value>E01</value>
        </inputFlag>
        <outputFlag>
            <name>ORIGINAL_RELIABLE</name>
            <value>0</value>
            <description>Observed value retrieved from external data source. Value is valid, marked as original
reliable as validation is yet to be done</description>
        </outputFlag>
    </flagConversion>
    <flagConversion>
        <inputFlag>
            <name>External02</name>
            <value>E02</value>
        </inputFlag>
        <outputFlag>
            <name>COMPLETED_RELIABLE</name>
            <value>2</value>
            <description>Original value was missing. Value has been filled in through byteerpolation,
transformation (e.g. stage discharge) or a model</description>
        </outputFlag>
    </flagConversion>
    <defaultOuputFlag>
        <name>ORIGINAL_RELIABLE</name>
        <value>0</value>
        <description>The data value is the original value retrieved from an external source and it successfully
passes all validation criteria set.</description>
    </defaultOuputFlag>
    <missingValueFlag>
        <name>ORIGINAL_MISSING</name>
        <value>9</value>
    </missingValueFlag>
</flagConversions>
```

Parsing of response HRef items

Due to the nature of the WaterML standard, requested items such as featureOfInterest, observedProperties and procedure are returned in the form of **xlink :href** values. Such values give an URI representation of the original requested value. In order for the FEWS system to extract the original information from these URI's the response value is parsed. The following parsing scheme has been implemented for the given URI formats:

1.Fragments

example: http://sweet.jpl.nasa.gov/2.2/phenHydro.owl#StreamDischarge

parse: new URI("http://sweet.jpl.nasa.gov/2.2/phenHydro.owl#StreamDischarge").getFragment()

result: StreamDischarge

2.Queries

example: http://nwis.waterdata.usgs.gov/usa/nwis/pmcodes?00065

_parse:_new URI("http://nwis.waterdata.usgs.gov/usa/nwis/pmcodes?00065").getQuery()

result: 00065

3. Path

example: http://kiwis.kisters.de/parameters/Q

parse: new File(new URI("http://kiwis.kisters.de/parameters/Q").getPath()).getName()

result: Q (element after last '/')

4. Text

example: myObservedProperty

parse: new File(new URI("myObsProperty").getPath()).getName();

result: myObsProperty

5. **SchemaPart**

example: urn:ogc:def:phenomenon:OGC:GageHeight

parse: new URI("urn:ogc:def:phenomenon:OGC:GageHeight").getSchemeSpecificPart()

result: GageHeight (element after last ':')