GSN Web Service Interface

Initial WSDL

```
http://<GSN-IP>:<GSN-PORT>/services/GSNWebService?wsdl
http://localhost:22001/services/GSNWebService?wsdl
```

If the AXIS Web Service is not already activated, uncomment configuration in:

```
webapp/WEB-INF/web.xml
```

Methods

listVirtualSensorNames

Definition

Returns the list of all the loaded Virtual Sensors names.

Input Parameters

None

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body/>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:listVirtualSensorNamesResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:virtualSensorName>memoryplotvs</ns3:virtualSensorName>
         <ns3:virtualSensorName>memorymonitorvs</ns3:virtualSensorName>
      </ns3:listVirtualSensorNamesResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

listWrapperURLs

Definition

Returns the list of all loaded wrappers urls. A wrapper URL is defined by the following fields:

- **String virtualSensor** – The name of the Virtual Sensor embedding the wrapper.
- **String stream** – The name of the input stream embedding the wrapper.
- **String source** – The alias of the source embedding the wrapper.
- **String wrapper** – The name of the wrapper.

Input Parameters

None

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body/>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:listWrapperURLsResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:wrapperURLs ns1:virtualSensor="memoryplotvs" ns1:source="source1" ns1:stream="DATA" ns1:wrapper="local"
xmlns:ns1="http://standard.webservice.gsn/xsd"/>
         <ns3:wrapperURLs ns1:virtualSensor="memorymonitorvs" ns1:source="source1" ns1:stream="input1" ns1:wrapper="memory-usage"
xmlns:ns1="http://standard.webservice.gsn/xsd"/>
      </ns3:listWrapperURLsResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

getContainerInfo

Definition

Returns the GSN container informations.

Input Parameters

None

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header/>
   <soap:Body/>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:getContainerInfoResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:containerDetails>
            <ns1:port xmlns:ns1="http://standard.webservice.gsn/xsd">22001</ns1:port>
            <ns1:author xmlns:ns1="http://standard.webservice.gsn/xsd">GSN Development Team.</ns1:author>
            <ns1:description xmlns:ns1="http://standard.webservice.gsn/xsd">Not Provided</ns1:description>
            <ns1:email xmlns:ns1="http://standard.webservice.gsn/xsd">my email address</ns1:email>
            <ns1:name xmlns:ns1="http://standard.webservice.gsn/xsd">GSN Server</ns1:name>
            <ns1:timeFormat xmlns:ns1="http://standard.webservice.gsn/xsd">dd/MM/yyyy HH:mm:ss Z</ns1:timeFormat>
         </ns3:containerDetails>
      </ns3:getContainerInfoResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

getMultiData

Definition

Returns the data for a specific set of Virtual Sensors and their field. In order to limit the memory consumption when requesting a large amount of data, the data is returned by pages. Each page contains a maximum of 1000 stream elements per each virtual sensor.
Each virtual sensor resultset is attached to a unique sequence number which will be used to fetch the next pages (with the getNextData method) in case (check the hasNext flag) there is more data for the query.

Output Selection

The selection of the data returned can be made either by the virtual sensor names, or the virtual sensor field names.

Select all the fields from all the Virtual Sensors	`<stan:fieldSelector xsd:vsname="ALL">` `</stan:fieldSelector>`
Select all the fields from a specific Virtual Sensor	`<stan:fieldSelector xsd:vsname="memorymonitorvs">` `</stan:fieldSelector>`

Select a specific set of fields from a specific Virtual Sensor	`<stan:fieldSelector xsd:vsname="memorymonitorvs">`
	`  <xsd:fieldNames>heap</xsd:fieldNames>`
	`  <xsd:fieldNames>non_heap</xsd:fieldNames>`
	`</stan:fieldSelector>`

Select a specific set of fields from all the virtual sensors	`<stan:fieldSelector xsd:vsname="ALL">`
	`  <xsd:fieldNames>heap</xsd:fieldNames>`
	`  <xsd:fieldNames>non_heap</xsd:fieldNames>`
	`</stan:fieldSelector>`

Standard Criterion

The data can be filtered according to a set of conditions, each of them composed of the following parameters:

- **String critJoin** (Mandatory) – An operator to join the condition to the query. Must be in { “AND”, “OR” }.
- **String critNeg** (Mandatory) – An operator to enable the negation of this condition. Must be in { “”, “NOT” }.
- **String critField** (Mandatory) – Defines the field (or all the fields) on which this condition apply. Must be in { <fieldname>, “ALL” }.
- **String critVsname** (Mandatory) – Defines the virtual sensors (or all the virtual sensors) on which this condition apply. Must be in { <vsname>, “ALL” }.

- **String critValue** (Mandatory) – Sets the value for this condition.
- **String critOperator** (Mandatory) – Sets the operator for this condition.
  Must be in { “le” (<), “leq” (<=), “ge” (>), “geq” (>=), “eq” (equal), “like” }.

Select only the tuples for which the heap field is greater than 1000 ...	`<stan:conditions xsd1:critJoin="AND" xsd1:field="heap" xsd1:negation="" xsd1:operator="ge" xsd1:value="10000" xsd1:vsname="ALL"/>`
… or the heap field is smaller than 0 ...	`<stan:conditions xsd1:critJoin="OR" xsd1:field="heap" xsd1:negation="" xsd1:operator="le" xsd1:value="0" xsd1:vsname="ALL"/>`

Aggregation

The data can be aggregated over the time, according to the following functions:

- **String groupOperator** (Mandatory) – The aggregation function. Must be in { “MAX”, “MIN”, “AVG” }
- **String timeRange** (Mandatory) – The aggregation period, in millisecond.

Aggregate the data by getting the average over one second.	`<stan:aggregation xsd1:groupOperator="AVG" xsd1:timeRange="1000"/>`

Input Parameters

- **FieldSelector[] virtualSensors** (Mandatory) – the data selection.
- **long from** (Optional) – Defines a lower time (epoch) for the data to be returned.
- **long to** (Optional) – Defines a upper time (epoch) for the data to be returned.
- **long nb** (Optional) – Define the maximal number of data to be returned. Note that this is not related to the pagination mecanism.
- **StandardCriterion[] conditions** (Optional) – A set of conditions for the query.
- **AggregationCriterion aggregation** (Optional) – An aggregation function applied to all the fields.

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn"
xmlns:xsd="http://standard.webservice.gsn/xsd" xmlns:xsd1="http://datarequest.http.gsn/xsd">
   <soap:Header/>
   <soap:Body>
      <stan:getMultiData stan:from="1271638922851" stan:to="1281638922851" stan:nb="2" stan:timeFormat="unix">
         <!--Zero or more repetitions:-->
         <stan:fieldSelector xsd:vsname="ALL">
            <!--Zero or more repetitions:-->
            <xsd:fieldNames>heap</xsd:fieldNames>
            <xsd:fieldNames>non_heap</xsd:fieldNames>
         </stan:fieldSelector>
         <!--Zero or more repetitions:-->
         <stan:conditions xsd1:critJoin="AND" xsd1:field="heap" xsd1:negation="" xsd1:operator="ge" xsd1:value="10000"
xsd1:vsname="ALL"/>
         <!--Optional:-->
         <stan:aggregation xsd1:groupOperator="AVG" xsd1:timeRange="1000"/>
      </stan:getMultiData>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:getMultiDataResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:queryResult ns1:hasNext="false" ns1:vsname="memorymonitorvs" ns1:sid="477ea289-5197-41b4-8af1-407379c5dedc"
xmlns:ns1="http://standard.webservice.gsn/xsd">
            <ns1:format>
               <ns1:field ns1:name="heap" ns1:type="Double"/>
               <ns1:field ns1:name="non_heap" ns1:type="Double"/>
               <ns1:field ns1:name="aggregation_interval" ns1:type="Double"/>
            </ns1:format>
            <ns1:streamElements ns1:timed="1272641983386">
               <ns1:field>1.0505856E7</ns1:field>
               <ns1:field>3.1908552E7</ns1:field>
               <ns1:field>1.272641983E9</ns1:field>
            </ns1:streamElements>
            <ns1:streamElements ns1:timed="1272641973364">
               <ns1:field>9758904.0</ns1:field>
               <ns1:field>3.1908552E7</ns1:field>
               <ns1:field>1.272641973E9</ns1:field>
            </ns1:streamElements>
            <ns1:executedQuery>select avg(pk) as pk, avg(heap) as heap, avg(non_heap) as non_heap, avg(timed) as timed,
floor(timed/1000) as aggregation_interval from memorymonitorvs where  timed > 1271638922851 and  timed &lt;= 1281638922851 and
heap > 10000 and  pk &lt; 9223372036854775807 group by aggregation_interval desc (size: 2 offset: 0)</ns1:executedQuery>
```

```
         </ns3:queryResult>
      </ns3:getMultiDataResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

getLatestMultiData

Definition

Returns the last Stream Element for a selection of virtual sensors and fields.

Input Parameters

- **FieldSelector[] virtualSensors** (Mandatory) – the data selection.

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn"
xmlns:xsd="http://standard.webservice.gsn/xsd">
   <soap:Header/>
   <soap:Body>
      <stan:getLatestMultiData stan:timeFormat="unix">
         <!--Zero or more repetitions:-->
         <stan:fieldSelector xsd:vsname="ALL">
         </stan:fieldSelector>
      </stan:getLatestMultiData>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:getLatestMultiDataResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:queryResult ns1:hasNext="false" ns1:vsname="memoryplotvs" ns1:sid="d4fc8518-74da-4749-bb93-d8348acec79d"
xmlns:ns1="http://standard.webservice.gsn/xsd">
            <ns1:format>
               <ns1:field ns1:name="data" ns1:type="Binary"/>
            </ns1:format>
            <ns1:streamElements ns1:timed="1272642205705">
               <ns1:field>[B@1767d16</ns1:field>
            </ns1:streamElements>
            <ns1:executedQuery>select pk, data, timed from memoryplotvs where  pk &lt; 9223372036854775807 order by timed desc
(size: 1 offset: 0)</ns1:executedQuery>
         </ns3:queryResult>
         <ns3:queryResult ns1:hasNext="false" ns1:vsname="memorymonitorvs" ns1:sid="7b87675b-dbcd-423c-b785-1b3bf4466ad3"
xmlns:ns1="http://standard.webservice.gsn/xsd">
            <ns1:format>
               <ns1:field ns1:name="heap" ns1:type="Double"/>
               <ns1:field ns1:name="non_heap" ns1:type="Double"/>
               <ns1:field ns1:name="pending_finalization_count" ns1:type="Double"/>
            </ns1:format>
            <ns1:streamElements ns1:timed="1272642225543">
               <ns1:field>7153616.0</ns1:field>
               <ns1:field>3.1935048E7</ns1:field>
               <ns1:field>0.0</ns1:field>
            </ns1:streamElements>
            <ns1:executedQuery>select pk, heap, non_heap, pending_finalization_count, timed from memorymonitorvs where  pk &lt;
9223372036854775807 order by timed desc (size: 1 offset: 0)</ns1:executedQuery>
         </ns3:queryResult>
      </ns3:getLatestMultiDataResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

getVirtualSensorDetails

Definition

Returns a combinaison of the virtual sensor information available.

Input Parameters

- **FieldSelector[] virtualSensors** (Mandatory) – the data selection.
- **String[] detailsType** (Mandatory) – The types of details returned. Must be in { “ADDRESSING”, “PROCESSOR”, “INFO”, “OUTPUTSTRUCTURE”, “WRAPPER” }.

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn"
xmlns:xsd="http://standard.webservice.gsn/xsd">
   <soap:Header/>
   <soap:Body>
      <stan:getVirtualSensorsDetails>
         <!--Zero or more repetitions:-->
         <stan:fieldSelector xsd:vsname="ALL"/>
         <!--Zero or more repetitions:-->
         <stan:detailsType>ADDRESSING</stan:detailsType>
         <stan:detailsType>INFO</stan:detailsType>
      </stan:getVirtualSensorsDetails>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:getVirtualSensorsDetailsResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:virtualSensorDetails ns1:vsname="memoryplotvs" xmlns:ns1="http://standard.webservice.gsn/xsd">
            <ns1:addressing>
               <ns1:predicates ns1:name="geographical">Not Specified, please edit the file
                         $INSTALL_PATH/virtual-sensors/memoryPlotVS.xml</ns1:predicates>
            </ns1:addressing>
            <ns1:info>
               <ns1:description>Not Specified, please edit the file
                 $INSTALL_PATH/virtual-sensors/memoryPlotVS.xml</ns1:description>
            </ns1:info>
         </ns3:virtualSensorDetails>
         <ns3:virtualSensorDetails ns1:vsname="memorymonitorvs" xmlns:ns1="http://standard.webservice.gsn/xsd">
            <ns1:addressing>
               <ns1:predicates ns1:name="geographical">Not Specified, please edit the file
                         $INSTALL_PATH/virtual-sensors/memoryDataVS.xml</ns1:predicates>
               <ns1:predicates ns1:name="LATITUDE">37.5419</ns1:predicates>
               <ns1:predicates ns1:name="LONGITUDE">-122.1419</ns1:predicates>
            </ns1:addressing>
            <ns1:info>
               <ns1:description>Not Specified, please edit the file
                 $INSTALL_PATH/virtual-sensors/memoryDataVS.xml</ns1:description>
            </ns1:info>
         </ns3:virtualSensorDetails>
      </ns3:getVirtualSensorsDetailsResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

createVirtualSensor

Definition

Creates and loads a Virtual Sensor in GSN. Returns false if GSN failed to load the virtual sensor. This could be because of the following reasons

1. There is an error in the xml configuration file sent.
2. The virtual sensor already exists.

Input Parameters

- **String vsname** – The name of the Virtual Sensor XML description file to be created (without the extension).
- **String descriptionFileContent** – The XML content of the description file to be created.

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn">
   <soap:Header/>
   <soap:Body>
      <stan:createVirtualSensor stan:vsname="memoryusage3">
         <stan:descriptionFileContent>
<![CDATA[
<virtual-sensor name="memoryusage3" priority="10" >
    <processing-class>
       <class-name>gsn.vsensor.BridgeVirtualSensor</class-name>
       <init-params/>
    <output-structure>
        <field name="HEAP" type="bigint"/>
```

```
    </output-structure>
    </processing-class>
    <description>this VS reports the memory usage at the local machine every 1100 ms</description>
    <life-cycle pool-size="10" />
    <addressing>
          </addressing>
    <storage />
    <streams>
        <stream name="data">
            <source alias="mem" storage-size="1" sampling-rate="1">
                <address wrapper="memory-usage">
                    <predicate key="HOST">localhost</predicate>
                    <predicate key="PORT">22001</predicate>
                    <predicate key="sampling-rate">1100</predicate>
                </address>
                <query>select HEAP from wrapper</query>
            </source>
        <query>select * from mem</query>
        </stream>
    </streams>
</virtual-sensor>
]]>
         </stan:descriptionFileContent>
      </stan:createVirtualSensor>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:createVirtualSensorResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:status>true</ns3:status>
      </ns3:createVirtualSensorResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

deleteVirtualSensor

Definition

Unload a Virtual Sensor and delete its XML description file. Returns false if GSN failed to delete the file because it did not exist.

Input Parameters

- **String vsname** – The name of the Virtual Sensor XML description file to be deleted (without the extension).

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn">
   <soap:Header/>
   <soap:Body>
      <stan:deleteVirtualSensor stan:vsname="memoryusage3"/>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:deleteVirtualSensorResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:status>true</ns3:status>
      </ns3:deleteVirtualSensorResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

registerQuery

Definition

Register a query over a set of already existing virtual sensors. The query is implemented by creating a new virtual sensor doing the bridge between the set of virtual sensors in the query.

Input Parameters

- **String queryName** – The name of the query (this name is also used for creating the virtual sensor implementing the query)
- **String query** – The query to be executed over the set of virtual sensors.
- **OutputStructure outputStructure** – The output structure of the query.
- **String[] vsnames** – The names of the virtual sensors involved in the query.

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn"
xmlns:xsd="http://standard.webservice.gsn/xsd">
   <soap:Header/>
   <soap:Body>
      <stan:registerQuery stan:queryName="myquery1" stan:query="select memorymonitorvs.heap as heap1, memoryusage3.heap as heap2
from memorymonitorvs, memoryusage3">
         <!--Zero or more repetitions:-->
         <stan:outputStructure xsd:name="heap1" xsd:type="bigint" xsd:description="This field comes from the memorymonitorvs"/>
         <stan:outputStructure xsd:name="heap2" xsd:type="bigint" xsd:description="This field comes from the memoryusage3"/>
         <!--Zero or more repetitions:-->
         <stan:vsnames>memorymonitorvs</stan:vsnames>
         <stan:vsnames>memoryusage3</stan:vsnames>
      </stan:registerQuery>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:registerQueryResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:status>true</ns3:status>
      </ns3:registerQueryResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

unregisterQuery

Definition

Unregister a query by deleting the virtual sensor associated to it. Rturn false if the query did not exist or if GSN failed to delete it.

Input Parameters

- **String queryName** – The name of the query (this name is also used for creating the virtual sensor implementing the query)

Example

Request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:stan="http://standard.webservice.gsn">
   <soap:Header/>
   <soap:Body>
      <stan:unregisterQuery>
         <stan:queryName>myquery1</stan:queryName>
      </stan:unregisterQuery>
   </soap:Body>
</soap:Envelope>
```

Response

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <ns3:unregisterQueryResponse xmlns:ns3="http://standard.webservice.gsn">
         <ns3:status>true</ns3:status>
      </ns3:unregisterQueryResponse>
   </soapenv:Body>
</soapenv:Envelope>
```