Deliverable D4.1.2 Concerted Approach V2

Appendix 5: OGC documentation: Rainfall downscaling

Revision History			
Version	**Date**	**Modified by**	**Comments**
0.1	2011-11-14	Martin Scholl	Initial document template
0.2	2011-12-19	Mihai Bartha	Contents and chapters
0.3	2011-12-20	Peter Kutschera	Review
0.4	2012-01-17	Sascha Schlobinski	Revision moved to appendix

Copyright © SUDPLAN

Table of Contents

Table of Figures

Glossary

Information product	Raw data, such as the results of mathematical modelling, and the analysis thereof, will often need to be packaged in such a way as to be accessible to the various stakeholders of an analysis. The medium can be one of a wide variety, such as print, photo, video, slides, or web pages. The term *information product* refers to such an entity.
Model	A *model* is a simplified representation of a system, usually intended to facilitate analysis of the system through manipulation of the model. In the SUDPLAN context the term can be used to refer to mathematical models of processes or spatial models of geographical entities.
Profile	Within SUDPLAN a *profile* is a set of configuration parameters which are associated with an individual or group, and which are remembered in order to facilitate repeated use of the system.
Report	A *report* is a particular type of information product which is usually static and might integrate still images, static data representations, mathematical expressions, and narrative to communicate an analytical result to others.
Scenario	A *scenario* is a set of parameters, variables and other conditions which represent a hypothetical situation, and which can be analysed through the use of models in order to produce hypothetical outcomes.
Scenario Management System	*Scenario Management System* is synonymous with SUDPLAN platform
SUDPLAN application	A *SUDPLAN application* is a decision support system crafted by using the SUDPLAN platform and integrating models, data, sensors, and other services to meet the requirements of the particular application.
SUDPLAN platform	The *SUDPLAN platform* is an ensemble of software components which support the development of SUDPLAN applications.
SUDPLAN system	*SUDPLAN system* is synonymous with SUDPLAN application

User	The term *user* refers to people who have a more or less direct involvement with a system. Primary users are directly and frequently involved, while secondary users may interact with the system only occasionally or through an intermediary. Tertiary users may not interact with the system but have a direct interest in the performance of the system.
Web-based	Computer applications are said to be *web-based* if they rely on or take advantage of data and/or services which are accessible via the World Wide Web using the Internet.

Acronyms

CL/DoW	The Call and the Description of Work for SUDPLAN
CS	Combined Sewer
CSO	Combined Sewer Overflows
CSV	Comma Separated Values
LT	The literature
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
ORCHESTRA	Open Architecture and Spatial Data Infrastructure for Risk Management
PI	The four pilot applications
PR	Other projects
SANY	Sensors ANYwhere
SISE	Single Information Space in Europe
SULFV	Stockholm - Uppsala Air Quality Management Association
SW	General software engineering principles
W3C	World Wide Web Consortium

1. Management Summary

This document contains the local Rainfall Downscaling CS model specific instructions needed to access the model. General information can be found in [SUDPLAN specific OGC services - abstract spec], which is a required reading.

This is a living document; the actual version is available from Sudplan.EU.

This version reflects the implementation at the end of 2011 and concentrates to the usage of the services using Timeseries-Toolbox, as this is the method used in the project. The OGC-related part will be completed in 2012.

Copyright © SUDPLAN

2. Component usage

The rainfall downscaling service layer contains two service components:

1. The model itself is encapsulated by a SPS interface. The SPS is used to control model execution.
2. The SOS interface is used to provide access to model results and as mechanism to provide model input time series. Input time series data is usually some long (>10 years) rain time series with high temporal resolution (<= 10 minutes).

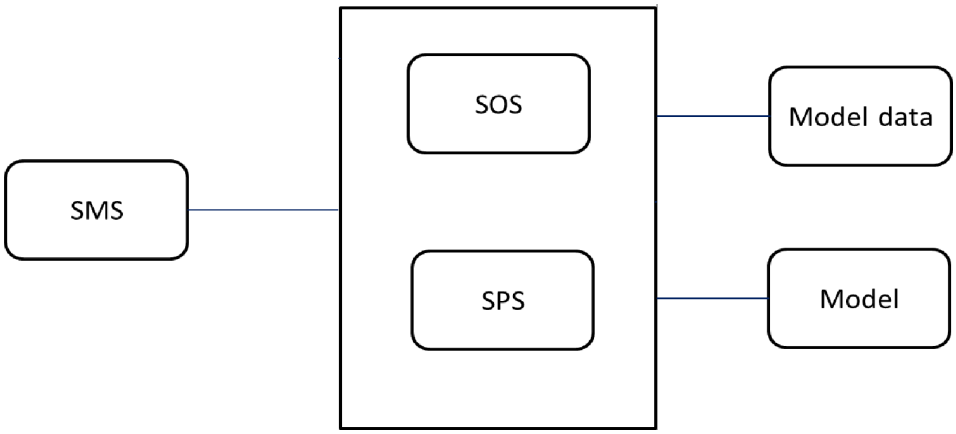

Figure 1 - Rainfall Downscaling Model Interfaces

3. Rainfall Downscaling Model workflow

There are two typical workflows regarding the Rainfall Downscaling model.

3.1. Model-run with historical data

The model run with historical data is the typical model invocation having as input a historical rain timeseries for a given geographic location.

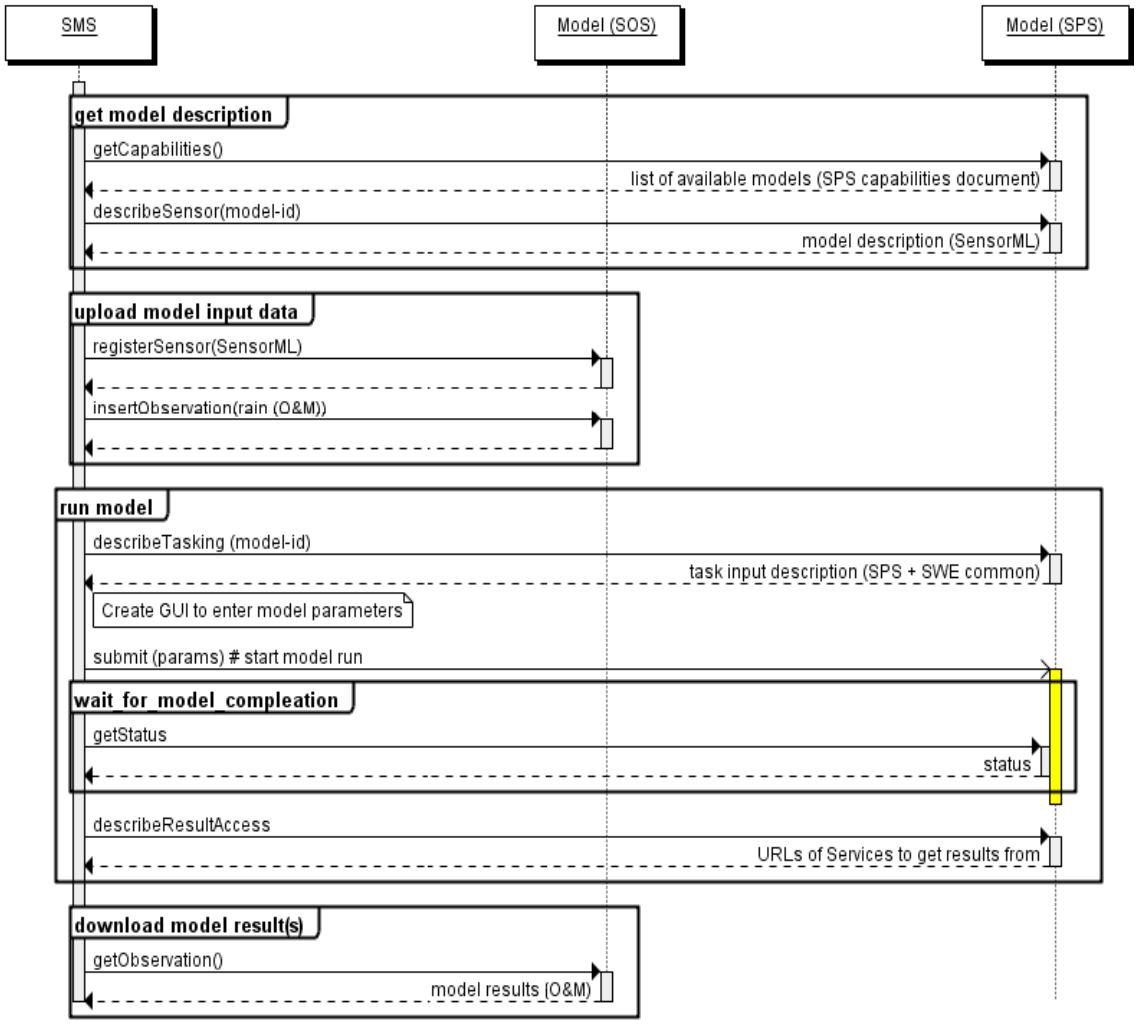

Figure 2 - Rainfall Downscaling based on Historical Data

The steps in this workflow are as follows:

1. Get the model description
   Ask for available models and get the description of the model. The description can be used to inform a user on the model. This step may be skipped if the information is not needed.

2. Upload model input data
   Upload new or choose an existing historical rain time series as model input.

3. Run model
   Get the list of needed model parameters with their data types and valid value ranges (tasking description). This step may be skipped if the information is already known. There is not necessary to upload a rain time series as model input over and over again for each model run. The returned model parameter descriptions contain a list of already available input time series in the form of an enumeration. The submit operation starts the model run, which can continue in the background even if the SMS is closed. Expect run times of some hours depending on the length of time processed (This might be shorter than the uploaded time series). The SMS can subscribe to notifications on task status. The last step is getting the information result location. The result are exposed through the SOS interface and organised as offerings.

4. Download model results
   Use the information from the model run to locate the results on the models SOS interface and download them.

3.2. Model-run with IDF data

This further model invocation type is based on IDF records as opposed to historical timeseries. The IDF is treated as model run parameter and not priory uploaded to the SOS.

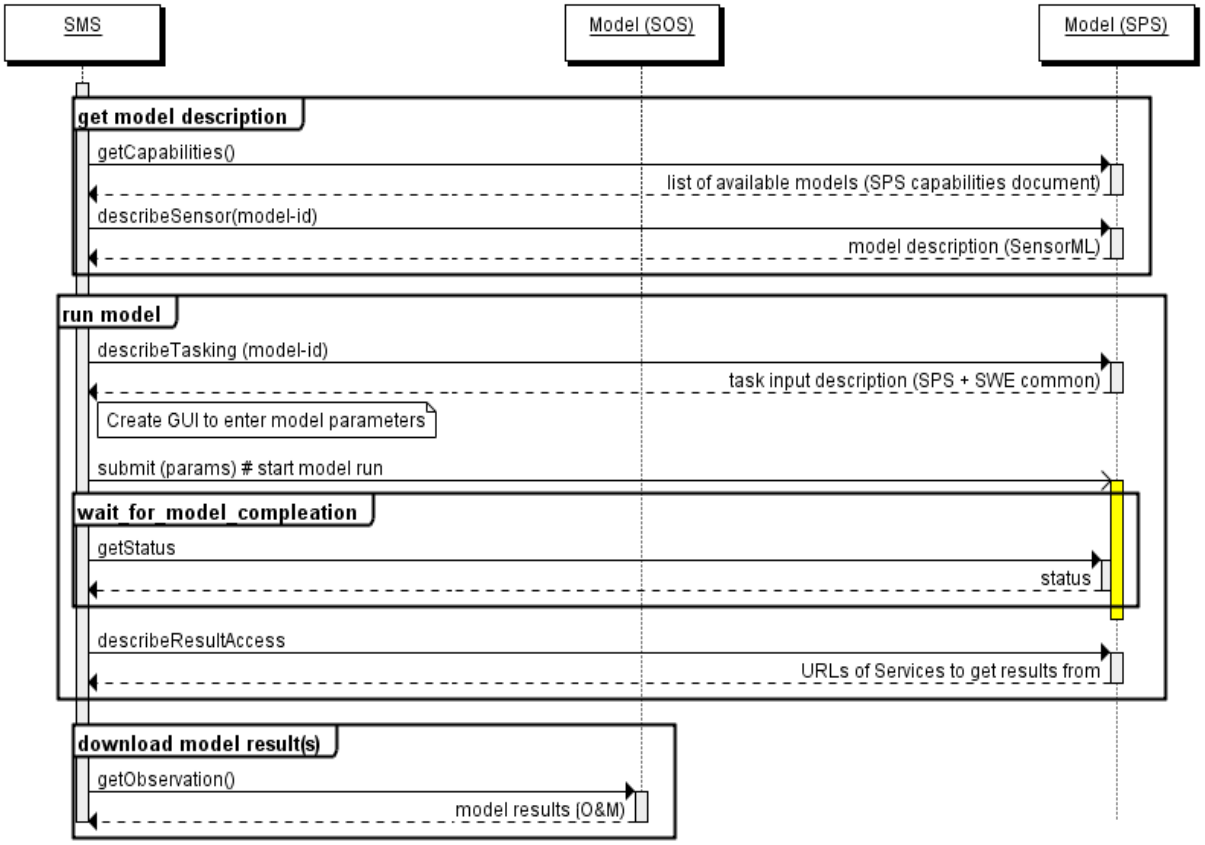

Figure 3 - Rainfall Downscaling Based on IDF

Compared to the workflow in **Fel! Hittar inte referenskälla.** the only difference is that the IDF data is passed in as model run parameter and no input timeseries is uploaded to the SOS.

1. Get the model description
   Ask for available models and get the description of the model. The description can be used to inform a user on the model. This step may be skipped if the information is not needed

2. Run model
   Get the list of needed model parameters with their data types and valid value ranges (tasking description). This step may be skipped if the information is already known. The submit operation starts the model run, which can continue in the background even if the SMS is closed. Expect run times of some hours depending on the length of time processed. The SMS can subscribe to notifications on task status. The last step is getting the information result location. The result are exposed through the SOS interface and organised as offerings

3. Download model results
   Use the information from the model run to locate the results on the models SOS interface and download them

Copyright © SUDPLAN

4. Rainfall Downscaling service interfaces

This section describes the OGC service interfaces to the Rainfall Downscaling services. It is assumed that the reader knows about the OGC standards, especially SOS, SPS, O&M and SensorML, so only Rainfall Downscaling specific information is given.

4.1. SOS with Model Data

The SOS service is used to upload model input timeseries and download model results. For uploading input timeseries the SOS implements the necessary operations of the SOS-T (SOS Transactional Profile).

4.1.1 Metadata

The service is located at http://sudplan.ait.ac.at:8084/

4.1.2 GetCapabilities

In **Fel! Hittar inte referenskälla.** the client (integral part of the SMS) upon instantiation invokes the GetCapabilities operation and retrieves from the server the list of the available timeseries including their meta-information. The SOS service follows the SUDPLAN convention of having exactly one offering by available timeseries. The list of available timeseries can be obtained from the ObservationOfferingList element of the Capabilities document.

4.1.3 GetObservation

The getObservation request does not differ from ref (OGC SOS implementation spec). It provides access to all model results. The SOS client generates and sends a GetObservation request to the server. Such a request contains filters on the requested data such as SOS offering name, procedure, featureOfInterest, observed property as well as temporal and spatial filters.

GetObservation request containg temporal and spatial filters:

```
<sos:GetObservation xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sos="http://www.opengis.net/sos/1.0" ... service="SOS" version="1.0.0">
 <sos:offering>EUROPE-O3-A1B3-coverage-10y</sos:offering>
 <sos:eventTime>
  <ogc:TM_During>
   <ogc:PropertyName>urn:ogc:data:time:iso8601</ogc:PropertyName>
   <gml:TimePeriod>
    <gml:beginPosition>1965-01-01T00:00:00+0100</gml:beginPosition>
    <gml:endPosition>2000-01-01T21:00:00+0100</gml:endPosition>
   </gml:TimePeriod>
  </ogc:TM_During>
 </sos:eventTime>
 <sos:procedure>urn:ogc:object:EUROPE:O3:A1B3:10y</sos:procedure>
 <sos:observedProperty>urn:ogc:def:property:OGC:O3</sos:observedProperty>
 <sos:featureOfInterest>
```

```
  <ogc:BBOX>
   <ogc:PropertyName>gml:location</ogc:PropertyName>
    <gml:Envelope>
     <gml:lowerCorner>14.18 48.24</gml:lowerCorner>
     <gml:upperCorner>14.18 48.24</gml:upperCorner>
    </gml:Envelope>
   </ogc:BBOX>
  </sos:featureOfInterest>
<sos:responseFormat>text/xml;subtype=&quot;om/1.0.0&quot;</sos:responseFormat
>
 <sos:resultModel>om:Observation</sos:resultModel>
 <sos:responseMode>inline</sos:responseMode>
</sos:GetObservation>
```

GetObservation response:

```
<om:ObservationCollection xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sos="http://www.opengis.net/sos/1.0"...>
 <om:member xlink:type="simple">
  <om:Observation>
   <gml:description>none</gml:description>
   <gml:boundedBy>
    <gml:Envelope srsName="EPSG:4326">
     <gml:lowerCorner>18.06 59.32</gml:lowerCorner>
     <gml:upperCorner>18.06 59.32</gml:upperCorner>
    </gml:Envelope>
   </gml:boundedBy>
   <om:procedure xlink:type="simple"
xlink:href="urn:ogc:object:STHLM:temp:A1B3"/>
   <om:observedProperty xlink:type="simple"
xlink:href="urn:ogc:def:property:OGC:temp"/>
   <om:featureOfInterest>
    <sa:SamplingPoint>
     <sa:sampledFeature xlink:href=""/>
     <sa:position>
      <gml:Point srsName="EPSG:4326">
       <gml:pos srsName="EPSG:4326">18.06 59.32</gml:pos>
      </gml:Point>
     </sa:position>
    </sa:SamplingPoint>
   </om:featureOfInterest>
   <om:result>
    <swe:DataArray>
     <swe:elementCount>
      <swe:Count>
       <swe:value>10</swe:value>
      </swe:Count>
     </swe:elementCount>
     <swe:elementType name="RootRecord">
      <swe:DataRecord>
       <swe:field name="Timestamp">
        <swe:Time definition="urn:ogc:data:time:iso8601"/>
       </swe:field>
       <swe:field name="value">
        <swe:Quantity definition="urn:ogc:def:property:OGC:temp">
         <swe:uom code="urn:ogc:def:uom:OGC::K"/>
        </swe:Quantity>
```

Copyright © SUDPLAN

```
       </swe:field>
      </swe:DataRecord>
     </swe:elementType>
     <swe:encoding>
      <swe:TextBlock blockSeparator="@" decimalSeparator="." tokenSeparator="
"/>
     </swe:encoding>
     <swe:values>1961-01-01T00:00:00+0100 280.086 @1962-01-01T00:00:00+0100
279.513 @1963-01-01T00:00:00+0100 279.681 @1964-01-01T00:00:00+0100 280.5
@1965-01-01T00:00:00+0100 279.966 @1966-01-01T00:00:00+0100 279.726 @1967-01-
01T00:00:00+0100 278.475 @1968-01-01T00:00:00+0100 279.025 @1969-01-
01T00:00:00+0100 279.5 @1970-01-01T00:00:00+0100 278.138 @</swe:values>
    </swe:DataArray>
   </om:result>
  </om:Observation>
 </om:member>
</om:ObservationCollection>
```

4.1.4 GetGeatureOfInterest

The GetFeatureOfInterest request is sometimes generated by the client and sent to the server in order to retrieve the samplingFeature of the timeseries. For most situations the sampling feature is integral part of the Observation response. Our implementation uses the sa:samplingPoint feature defined in SA Sampling and defines one additional samplingFeature type, namely ait:samplingGrid. The samplingGrid element contains a gml:RectifiedGrid element that describes the origin, offset vectors, and dimensions of the grid for which values are available. At the same time it contains a reference to the sampled feature through the sa:sampledFeature element. The samplingGrid schema is listed as an example of the DescribeFeatureType request of the next chapter.

Sampling point example:

```
<sa:SamplingPoint>
    <sa:sampledFeature xlink:href=""/>
    <sa:position>
        <gml:Point srsName="EPSG:4326">
            <gml:pos srsName="EPSG:4326">18.06 59.32</gml:pos>
        </gml:Point>
    </sa:position>
</sa:SamplingPoint>
```

Sampling grid example

```
<ait:SamplingGrid gml:id="AITSGID0">
    <gml:description>grid</gml:description>
    <sa:sampledFeature xlink:href=""/>
    <gml:RectifiedGrid srsName="EPSG:4326">
        <gml:limits>
            <gml:GridEnvelope>
                <gml:low>0 0</gml:low>
                <gml:high>95 77</gml:high>
            </gml:GridEnvelope>
        </gml:limits>
        <gml:axisName>x</gml:axisName>
```

```
        <gml:axisName>y</gml:axisName>
        <gml:origin>
            <gml:Point srsName="EPSG:4326">
                <gml:pos>-12.25 32.75</gml:pos>
            </gml:Point>
        </gml:origin>
        <gml:offsetVector srsName="EPSG:4326">0.5 0.0</gml:offsetVector>
        <gml:offsetVector srsName="EPSG:4326">0.0 0.5</gml:offsetVector>
    </gml:RectifiedGrid>
</ait:SamplingGrid>
```

4.1.5 DescribeFeatureType

DescribeFeatureType returns the XML schema for the specified GML feature advertised in GetCapabilities or part of the Observation result. This may be used to obtain a description of the type of a feature.

The following contains the schema (featureType) of the ait:samplingGrid as returned by the DescribeFeatureType operation:

```
<schema xmlns=http://www.w3.org/2001/XMLSchema
xmlns:gml=http://www.opengis.net/gml
xmlns:sa=http://www.opengis.net/sampling/1.0
xmlns:ait="http://www.ait.ac.at/sampling"
targetNamespace=http://www.ait.ac.at/sampling elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0.0">
 <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"/>
 <import namespace="http://www.opengis.net/sampling/1.0"
schemaLocation="http://schemas.opengis.net/sampling/1.0.0/sampling.xsd"/>
 <complexType name="SamplingGridType">
  <complexContent>
   <extension base="sa:SpatiallyExtensiveSamplingFeatureType">
    <sequence>
     <element ref="gml:RectifiedGrid"/>
    </sequence>
   </extension>
  </complexContent>
 </complexType>
 <element name="SamplingGrid" type="ait:SamplingGridType"
substitutionGroup="gml:_Feature"/>
</schema>
```

4.1.6 RegisterSensor

The registerSensor operation is used to create room for a new model input dataset and requires a valid SensorML description.This method returns a new sensor id which will be used in the insertObservation operation.

Example RegisterSensor request:

```
<sos:RegisterSensor xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sos="http://www.opengis.net/sos/1.0"
xmlns:gml="http://www.opengis.net/gml" ...service="SOS" version="1.0.0">
 <sos:SensorDescription>
  <sml:System gml:id="SUDPLAN_A1B3">
  <gml:description>Simple rain downscaling model</gml:description>
  <sml:identification xlink:type="simple">
   <sml:IdentifierList>
    <sml:identifier name="UID">
     <sml:Term definition="urn:x-ogc:def:identifier:OGC:uuid">
      <sml:value>urn:x-ogc:object:model:SUDPLAN:prec:A1B3</sml:value>
     </sml:Term>
    </sml:identifier>
    <sml:identifier>
     <sml:Term definition="urn:x-ogc:def:identifier:OGC:shortName">
      <sml:value>SUDPLAN A1B3</sml:value>
     </sml:Term>
    </sml:identifier>
    </sml:IdentifierList>
   </sml:identification>
   <sml:inputs xlink:type="simple">
    <sml:InputList>
     <sml:input name="ObservationOfferingName" xlink:type="simple">
      <swe:Text/>
     </sml:input>
     <sml:input name="centerTime" xlink:type="simple">
      <swe:Time/>
     </sml:input>
    </sml:InputList>
   </sml:inputs>
   <sml:outputs xlink:type="simple">
    <sml:OutputList>
     <sml:output name="Downscaled_rain" xlink:type="simple">
      <swe:DataArray>
       <swe:elementCount>
        <swe:Count/>
       </swe:elementCount>
       <swe:elementType name="SimpleType" xlink:type="simple">
        <swe:Quantity
definition="urn:ogc:def:property:OGC:1.0:precipitation">
         <swe:uom code="mm" xlink:type="simple"/>
        </swe:Quantity>
       </swe:elementType>
      </swe:DataArray>
     </sml:output>
    </sml:OutputList>
   </sml:outputs>
  </sml:System>
 </sos:SensorDescription>
 <sos:ObservationTemplate>
  <om:Observation>
   <om:samplingTime xlink:type="simple"/>
   <om:procedure xlink:type="simple"/>
   <om:observedProperty xlink:type="simple"/>
   <om:featureOfInterest/>
   </om:Observation>
 </sos:ObservationTemplate>
</sos:RegisterSensor>
```

Example RegisterSensor response:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sos:RegisterSensorResponse xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sos="http://www.opengis.net/sos/1.0"
xmlns:gml="http://www.opengis.net/gml" ...>
  <sos:AssignedSensorId>urn:ogc:object:0</sos:AssignedSensorId>
</sos:RegisterSensorResponse>
```

4.1.7 InsertObservation

The InsertObservation operation is used to upload model input data. This is a simple timeseries of float values, encoded according to the O&M standard. The InsertObservation makes use of the SamplingPoint and SamplingGrid elements as described in section 4.1.4 to describe the observation data.

Example InsertObservation request:

```
<sos:InsertObservation xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sos="http://www.opengis.net/sos/1.0"
xmlns:gml="http://www.opengis.net/gml" ... service="SOS" version="1.0.0">
 <sos:AssignedSensorId>urn:ogc:object:0</sos:AssignedSensorId>
 <om:Observation>
  <gml:description>Data from urn:ogc:object:0</gml:description>
   <om:samplingTime xlink:type="simple">
    <gml:TimePeriod>
     <gml:beginPosition>2011-11-25T01:15:56+0100</gml:beginPosition>
     <gml:endPosition>2011-11-25T01:15:56+0100</gml:endPosition>
    </gml:TimePeriod>
   </om:samplingTime>
   <om:resultTime xlink:type="simple">
    <gml:TimePeriod>
     <gml:beginPosition>2011-11-25T01:15:56+0100</gml:beginPosition>
     <gml:endPosition>2011-11-25T01:15:56+0100</gml:endPosition>
    </gml:TimePeriod>
   </om:resultTime>
   <om:procedure xlink:type="simple" xlink:href="urn:ogc:object:0"/>
   <om:observedProperty xlink:type="simple"
xlink:href="urn:ogc:def:property:OGC:prec"/>
   <om:featureOfInterest>
    <ait:SamplingGrid gml:id="AITSGID0">
     <gml:description>grid</gml:description>
     <sa:sampledFeature xlink:href=""/>
     <gml:RectifiedGrid srsName="EPSG:4326">
      <gml:limits>
       <gml:GridEnvelope>
        <gml:low>0 0</gml:low>
        <gml:high>1 1</gml:high>
       </gml:GridEnvelope>
      </gml:limits>
      <gml:axisName>x</gml:axisName>
      <gml:axisName>y</gml:axisName>
      <gml:origin>
       <gml:Point srsName="EPSG:4326">
        <gml:pos>14.18 48.24</gml:pos>
       </gml:Point>
```

```
      </gml:origin>
      <gml:offsetVector srsName="EPSG:4326">0.20000000000000107
0.0</gml:offsetVector>
      <gml:offsetVector srsName="EPSG:4326">0.0
0.10000000000000142</gml:offsetVector>
     </gml:RectifiedGrid>
    </ait:SamplingGrid>
   </om:featureOfInterest>
   <om:result>
    <swe:DataArray>
     <swe:elementCount>
      <swe:Count>
       <swe:value>1</swe:value>
      </swe:Count>
     </swe:elementCount>
     <swe:elementType name="RootRecord">
      <swe:DataRecord>
       <swe:field name="Timestamp">
        <swe:Time definition="urn:ogc:data:time:iso8601"/>
       </swe:field>
       <swe:field name="value">
        <swe:Quantity definition="urn:ogc:def:property:OGC:prec">
         <swe:uom code="urn:ogc:def:uom:OGC::mm"/>
          </swe:Quantity>
       </swe:field>
      </swe:DataRecord>
     </swe:elementType>
    <swe:encoding>
     <swe:TextBlock blockSeparator="@" decimalSeparator="." tokenSeparator="
"/>
    </swe:encoding>
    <swe:values>2011-11-25T01:15:56+0100 1.0 @</swe:values>
   </swe:DataArray>
  </om:result>
 </om:Observation>
</sos:InsertObservation>
```

Example InsertObservation response:

```
<sos:InsertObservationResponse xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:sos="http://www.opengis.net/sos/1.0"
xmlns:gml="http://www.opengis.net/gml" ...
  <sos:AssignedObservationId>assigned_obj_id</sos:AssignedObservationId>
</sos:InsertObservationResponse>
```

4.2. SPS controlling the model

The SPS service is used to control model runs on a background system. This service provides a standard OGC SPS interface.

4.2.1 Metadata

The service is located at http://sudplan.ait.ac.at:8085/

4.2.2 Timeseries API usage

See [SPEC-A] for a description of the Timeseries-Toolbox usage in the SUDPLAN context. The following text contains only the rainfall downscaling specific information. There are two flavours of rainfall downscaling:

1. Downscaling based on a historical rain timeseries.
   A historical rainfall timeseries needs to be uploaded in advance to the SOS and will be used as input for downscaling according to selectable future climate scenario conditions. There is no need to upload one timeseries more than once. The result of the downscaling is a timeseries with the same length as the input timeseries. But the client can request to download parts of the result only. This result is valid for one specific geographic point, but this information is implicitly contained in the metadata of the timeseries.

2. Downscaling based on an IDF table specific for one position.
   The provided IDF data is downscaled for a selectable future climate scenario. The IDF data is small and can be passed as model parameter to the SPS at every run. Since there is no implicit metadata some additional information has to be provided explicitly as parameters for the model run. The result of this downscaling is an IDF curve of the same size as the input curve.

4.2.3 DataHandler instantiation

In the rainfall downscaling there are two services involved:

1. A SOS-T to upload timeseries and to download results (Timeseries and IDF tables)
   The service endpoint is http://sudplan.ait.ac.at:8084/

2. A SPS for model control. The service endpoint is http://sudplan.ait.ac.at:8085/

Both services are used for both flavours of rainfall downscaling, timeseries as well as IDF based.

4.2.4 Historical rain timeseries based downscaling

4.2.4.1 Input timeseries upload

As described in 4.2 this type of downscaling is based on priori uploaded historical rainfall data, exposed by an SOS service. For the upload no filter is needed, a unique sensor (procedure) id will be generated by the SOS service. After creation of a new DataPoint using *createDatapoint()* use *getFilter().getProperty(TimeSeries.PROCEDURE)* to retrieve this uniqe id. Clearly it is not necessary to upload the same a timeseries multiple times. In the model run all already uploaded

timeseries will be available to be used as model input. The offering name is also generated by the server upon writing the timeseries data. It is a good practice to generate offering names according to a given schema.

Creating datapoint:

```
Properties dpFilter = new Properties();
Map<String,Object> dpProps = new HashMap<String, Object>();
dpProps.put(TimeSeries.SENSORML, readResource("smlSensor.xml"));
Datapoint dp = client.createDatapoint(dpFilter, dpProps, Access.READ);
```

Retrieving and listing assigned procedure id:

```
String proc = dp.getFilter().getProperty(TimeSeries.PROCEDURE);
System.out.println("AssignedSensorId " + TimeSeries.PROCEDURE + " " + proc);
```

Setting up timeseries:

```
HashMap<String, Object> tsProps = new HashMap<String, Object>();
tsProps.put(PropertyNames.DESCRIPTION, "Data from " + proc);
tsProps.put(TimeSeries.VALUE_KEYS, new String[]{PropertyNames.VALUE});
tsProps.put(TimeSeries.VALUE_JAVA_CLASS_NAMES,
   new String[]{Float.class.getName()});
tsProps.put(TimeSeries.VALUE_TYPES,
   new String[]{TimeSeries.VALUE_TYPE_NUMBER});
tsProps.put(TimeSeries.VALUE_OBSERVED_PROPERTY_URNS,
   new String[]{ "urn:ogc:def:property:OGC:prec"});
tsProps.put(TimeSeries.VALUE_UNITS, new String[]{"urn:ogc:def:uom:OGC:mm"});
tsProps.put(TimeSeries.GEOMETRY, new Envelope(29.02057234081306,
29.02057234081306, 46.945455467553245, 46.945455467553245 ));
tsProps.put(PropertyNames.SPATIAL_RESOLUTION, new Integer[]{1, 1});
tsProps.put(PropertyNames.TEMPORAL_RESOLUTION, "NONE");
tsProps.put(PropertyNames.COORDINATE_SYSTEM, "EPSG:4326");
TimeSeries ts = new LargeTimeseriesImpl(tsProps);
TimeStamp t = new TimeStamp();
long millis = t.asMilis();
for(int i =0; i<100; i++) {
   ts.setValue(new TimeStamp(millis += 1000), PropertyNames.VALUE,
      new Float(i));
}
```

Uploading the timeseries:

```
dp.putTimeSeries(ts);
```

4.2.4.2 Input timeseries properties

As seen in the example above the following properties are required upon invocation of a *putTimeseries* request. These properties are transported to the SOS service as various elements of the sos:InsertObservation request.

Name	Type	Description
VALUE_KEYS	String Array	Name of the value component. In some cases the value name is simply “value”.
UNITS	String Array	Units of Measurements corresponding to the VALUE_KEYS. They have to have the same length and order. Example: “mm”.
OBSERVED_PROPERTY_URNS	String Array	URN of the property corresponding to the VALUE_KEYS. They have to have the same length and order. Example: “urn:ogc:def:property:OGC:1.0:precipitation”
DESCRIPTION	String	Some text describing this time series
COORDINATE_SYSTEM	String	Describes the reference system of the geometry. Example: "EPSG:3423"
GEOMETRY	Envelope	Envelope(14.18, 14.38, 48.24, 48.34)
SPATIAL_RESOLUTION	Integer Array	Describes data fields. Example: one value “[1]”, array with 5 values “[5]”, two dimensional array “[72, 55]”.
TEMPORAL_RESOLUTION	String	Specifies the temporal resolution (time between two measurements.). If the time differs the minimum time would be a good choice. The format is a number followed by a single letter. The letter indicates time unit and can be one of the following: s Seconds m Minute h Hour d Day M Month Y Year “None” if unknown.

4.2.4.3 Model execution

In order to start a new model run a new datapoint has to be created. The datapoint is the equivalent to a model run. This can be accomplished by providing a filter containing the TimeSeries.PROCEDURE value “Rain_Timeseries_Downscaling”.

```
Properties filter = new Properties();
filter.put(TimeSeries.PROCEDURE, "Rain_Timeseries_Downscaling");
Datapoint dp = client.createDatapoint(filter, null, Access.READ);
```

In order to start a new model run (task) the tasking description must be retrieved. This can be accomplished by listing the description properties of the newly created datapoint: The datapoint properties contains a property called TimeSeries.DESCRIPTION_KEYS of type String[]. This string array contains the keys of the model parameter description properties.

```
String [] desc_keys =
(String[])dp.getProperties().get(TimeSeries.DESCRIPTION_KEYS);
for(String desc_key : desc_keys) {
   System.out.println(dp.getProperties().get(desc_key));
}
```

The property values corresponding to the DESCRIPTION_KEYS are of type String and contain the String representation of a SWE Common encoded model parameter description (XML fragment). Additionally the property JAXB_DESCRIPTION_KEYS contains the keys pointing to the same parameter descriptions only as type JAXBElement.

Parameter description example:

Key:

```
desc:climate_scenario:
```

Value:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns6:InputDescriptor parameterID="climate_scenario" use="required"
updateable="false" xmlns:smillang="http://www.w3.org/2001/SMIL20/Language"
xmlns:smil="http://www.w3.org/2001/SMIL20/"
xmlns:ows="http://www.opengis.net/ows"
xmlns:sml="http://www.opengis.net/sensorML/1.0.1"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:ic2="urn:us:gov:ic:ism:v2"
xmlns:gml="http://www.opengis.net/gml"
xmlns:ns5="http://www.opengis.net/swe/1.0.1"
xmlns:ns6="http://www.opengis.net/sps/1.0"
xmlns:swe="http://www.opengis.net/swe/1.0">
 <ns6:definition>
  <ns6:commonData>
   <swe:Category>
    <swe:constraint xlink:type="simple">
     <swe:AllowedTokens>
      <swe:valueList>
      </swe:valueList>
```

```
      </swe:AllowedTokens>
     </swe:constraint>
    </swe:Category>
   </ns6:commonData>
  </ns6:definition>
</ns6:InputDescriptor>
```

In order to start the model a timeseries containing the parameters according to the parameter descriptions has to be written (putTimeseries) on the newly created Datapoint. Additionally the timeseries value PropertyNames.TaskAction has to set to PropertyNames.TaskActionStart. Upon starting a new model run the SPS service generates a unique taskId that is used throughout the interaction with the model. It can be retrieved as a filter property PropertyNames.TASK_ID.

```
TimeSeries ts = new LargeTimeseriesImpl(dp.getProperties());
TimeStamp now = new TimeStamp();
ts.setValue(now, CLIMATE_SCENARIO,          paramClimateScenario);
ts.setValue(now, SOURCE_RAIN,               paramSourceRain);
ts.setValue(now, CENTER_TIME,               paramCenterTime);
ts.setValue(now, PropertyNames.TaskAction,  PropertyNames.TaskActionStart);
```

Writing the timeseries on the Datapoint:

```
dp.putTimeSeries(ts);
```

Retrieving the status of the model run can be accomplished by retrieving a timeseries from the datapoint. In the following example all data available is retrieved.

```
TimeInterval ti = new TimeInterval(TimeInterval.Openness.OPEN,
TimeStamp.NEGATIVE_INFINITY, TimeStamp.POSITIVE_INFINITY,
TimeInterval.Openness.OPEN);
TimeSeries statusTs = dp.getTimeSeries(ti);
```

4.2.4.4 Model parameters:

Name	Description	Notes
climate_scenario	<ns6:definition> <ns6:commonData> <swe:Category> <swe:constraint xlink:type="simple"> <swe:AllowedTokens> <swe:valueList> </swe:valueList> </swe:AllowedTokens> </swe:constraint> </swe:Category> </ns6:commonData> </ns6:definition>	<swe:AllowedTokens> valueList based on available climate scenarios provided by the backend In the moment the following climate scenarios are available: • echam5a1b3

		• hadleya1b
source_rain	<ns6:definition> <ns6:commonData> <swe:Category> <swe:constraint xlink:type="simple"> <swe:AllowedTokens> <swe:valueList> </swe:valueList> </swe:AllowedTokens> </swe:constraint> </swe:Category> </ns6:commonData> </ns6:definition>	<swe:AllowedTokens> valueList based on available uploaded input timeseries This year is interpreted as the center of a future period of the same length as the input timeseries.
center_time	<ns6:definition> <ns6:commonData> <swe:Time> <swe:constraint xlink:type="simple"> <swe:AllowedTimes> <swe:interval>2050-01-01T00:00:00 2080-01-01T00:00:00</swe:interval> </swe:AllowedTimes> </swe:constraint> </swe:Time> </ns6:commonData> </ns6:definition> </ns6:InputDescriptor>	

4.2.4.5 Model timeseries properties

The following properties will be available:

Name	Type, Value	Comment
TimeSeries.VALUE_KEYS	String[] {“climate_scenario”, “source_rain”, “center_time” “status”, “errors”, “results”}	ValueKeys of timeseries data
TimeSeries.DESCRIPTION_KEYS	String[]	PropertyNames of descriptions of data with matching valueKey
“desc:climate_scenario”	XML-Schema	Climate scenario to use

“desc:source_rain”	XML-Schema	Reference to uploaded rain timeseries
“desc:centertime”	XML-Schema	Timestamp in the middle of the downscaled result.

The data is organized as follows:

valueKey	Type, Values	Description
action	String, only “start” is relevant	Command to the model task
state	String: “not yet started”, “in operation”, “finished”	Actual model state
errors	String[]	Error messages from the model
results	String[] Format: ts:result_service_url=<Service-Endpoint>; ts:result_service_type=SOS; ts:offering=<Offering>;	Describes where to get the results from.

4.2.4.6 Result retrieval

The following properties are available on downloaded rain timeseries:

```
TimeSeries ts = dp.getTimeSeries(ti);
TimeStamp last = ts.getTimeStamps().last();
String status = (String)ts.getValue(last, PropertyNames.TaskStatus);
if(status!=null) {
if(TaskOperationStatus.Finished.equals(TaskOperationStatus.getStatus(status))
) {
   String[] results= (String[])ts.getValue(last, PropertyNames.TaskResults);
   for(String result : results) {
      Properties prop = string2Properties(String s);
      String url = prop.getProperty(PropertyNames.RESULT_SERVICE_URL);
      String type = prop.getProperty(PropertyNames.RESULT_SERVICE_TYPE);
      String offering = props.getProperty(TimeSeries.OFFERING);
   }
}
```

Based on the url, type and offering one can connect to the SOS pointed to by the url and retrieve the timeseries corresponding to the offering point to by the value of offering. This can be accomplished es described in the [SPEC-A].

Copyright © SUDPLAN

4.2.4.7 Result time series properties

Following timeseries properties are available on downloaded model run results.

Name	Type, Value	Comment, Example
TimeSeries.VALUE_KEYS	String[], {TimeSeries.VALUE}	ValueKey of the rain values
TimeSeries.VALUE_UNITS	String[], {“mm”}	Units matching the valueKeys
TimeSeries. VALUE_OBSERVED_PROPERTY _URNS	String[], {“urn:ogc:def:property:OGC:1.0: precipitation”}	URN of observed property matching valueKeys
TimeSeries.VALUE_TYPES	String[], {TimeSeries. VALUE_TYPE_NUMBER}	Values are Numbers. Not set in the moment. See VALUE_CLASSES
TimeSeries. VALUE_JAVA_CLASS_NAMES	String[], {java.lang.Float}	Java-Type of values.
PropertyNames. SPATIAL_RESOLUTION	Float[], {1}	Values are scalar values
PropertyNames. TEMPORAL_RESOLUTION	Sting	Time between values (if known)
TimeSeries. DESCRIPTION_KEYS	String[]	PropertyNames of properties holding descriptions matching valueKeys. Not set.
PropertyNames. DESCRIPTION	String	Human readable description of the timeseries.
PropertyNames. COORDINATE_SYSTEM	String, “EPSG:3423”	Coordinate system for GEOMETRY
Timeseries.GEOMETRY	Envelope, [14.17 : 48.18, 14.17 : 48.18]	Bounding box.
TimeSeries. AVAILABLE_DATA_MIN	TimeStamp	TimeStamp of first available value
TimeSeries. AVAILABLE_DATA_MAX	TimeStamp	TimeStamp of last available value

The data are organized as follows:

valueKey	Type	Description
Timeseries.VALUE_KEY	One float value	mm rain (mm per measurement interval length).

4.2.4.8 Timeseries statistics data download

Not currently available.

4.2.5 IDF based rainfall downscaling

This type of model run is very similar to the one based on historical rainfall timeseries. It differs only in the input parameters and that it does not rely on historical rainfall data. Such a model run can be started as described in section 4.2.4.3 but with parameters as described in the following. Similarly the results can be retrieved as described in section 4.2.4.6 and the result timeseries will be stored on a SOS and can be retrieved as in 4.2.4.7.

4.2.5.1 Model parameters

Parameter Name	Parameter Description	Notes
climate_scenario	<ns6:definition> <ns6:commonData> <swe:Category> <swe:constraint xlink:type="simple"> <swe:AllowedTokens> <swe:valueList> </swe:valueList> </swe:AllowedTokens> </swe:constraint> </swe:Category> </ns6:commonData> </ns6:definition>	The climate scenario for which the model shall be run. <swe:AllowedTokens> valueList based on available climate scenarios provided by the backend In the moment the following climate scenarios are available: • echam5a1b3 • hadleya1b
historical_year	<sps:definition> <sps:commonData> <swe:Time> </swe:Time> </sps:commonData> </sps:definition>	Historical year for which the provided IDF curve is valid This is assumed as the center of a 30 year period.
future_year	<sps:definition> <sps:commonData> <swe:Time> <swe:constraint> <swe:AllowedTimes> <swe:interval>2050-01-01T00:00:00 2080-01-01T00:00:00</swe:interval>	Future year for which the downscaling shall be executed. <swe:AllowedTimes> interval based on model restrictions.

	</swe:AllowedTimes> </swe:constraint> </swe:Time> </sps:commonData> </sps:definition>	This year is interpreted as the center of a future 30 year period.
coordinate_system	<sps:definition> <sps:commonData> <swe:Text> </swe:Text> </sps:commonData> </sps:definition>	Name of the coordinate system of the coordinate for which the downscaling should be performed. E.g: "EPSG:4326"
coordinate_x	<sps:definition> <sps:commonData> <swe:Quantity> </swe:Quantity> </sps:commonData> </sps:definition>	The X coordinate of the point for which downscaling shall be performed.
coordinate_y	<sps:definition> <sps:commonData> <swe:Quantity> </swe:Quantity> </sps:commonData> </sps:definition>	The Y coordinate of the point for which downscaling shall be performed.
idf_data	<sps:definition> <sps:commonData> <swe:Text> </swe:Text> </sps:commonData> </sps:definition>	The textual representation of the IDF curve that shall be used as input for the rainfall downscaling. The encoding is in the form: Duration column of the IDF table (values are given in minutes, Range [30..?], separated by ":") Frequency column of the IDF table (values are given in years) Intensity column of the IDF table. Values are given in mm/h.
		e.g: 30:30:60:60 10:100:10:100 122:165:81:114

4.2.5.2 Model results

The result from the IDF downscaling is again an IDF table. This table can be downloaded from the SOS service. The exact location is described in the results – value. This corresponds to the values returned from the DescribeResultAccess operation of the SOS service.

Beside the usual meta information like coordinate system and geometry the following meta information is returned from the SOS:

Name	Type, Value	Comment
TimeSeries.VALUE_KEYS	String[] {“Duration”, “ReturnPeriod”, “Intensity”}	ValueKeys of timeseries data
TimeSeries.DESCRIPTION	String	Human-Readable text

The data is organized as follows:

valueKey	Type, Values	Description
Duraton	Float[]	Duration-column of the result IDF table, values in minutes
ReturnPeriod	Float[]	Return-period column of the result IDF table, values in years
Intensity	Float[]	Intensity column of the result IDF table, values in mm/h

5. References

[SPEC-A] SUDPLAN specific OGC services - abstract spec
Will be available from SUDPLAN.EU.