



deegree SOS

(Bonn, May 5th 2006)

latlon GmbH

Aennchenstr. 19

53177 Bonn

Germany

Tel ++49 - 228 - 184 96-0

Fax ++49 - 228 - 184 96-29

info@lat-lon.de

www.lat-lon.de

Dept. of Geography

Bonn University

Meckenheimer Allee 166

53115 Bonn

Tel. ++49 228 732098

www.giub.uni-bonn.de

Table of Contents

1 Introduction.....	3
2 Get deegree SOS up and running.....	4
2.1 Download and archive extraction.....	4
2.2 Tomcat integration.....	4
2.3 Required libraries.....	6
2.4 Architecture.....	7
2.5 Configuration.....	8
2.6 deegree Parameters.....	8
2.7 OWS Commons parameters.....	10
2.7.1 ServiceIdentification.....	10
2.7.2 OperationsMetadata.....	11
2.7.3 PlatformList.....	12
2.7.4 SensorList	13
3 Appendix A: Example SOS configuration/capabilities document.....	15
4 Appendix C: Third party libraries.....	19

1 Introduction

deegree is a Java Framework offering the main building blocks for Spatial Data Infrastructures (SDIs). Its entire architecture is developed using standards of the Open Geospatial Consortium (OGC) and ISO Technical Committee 211 – Geographic information / Geoinformatics (ISO/TC 211). deegree encompasses OGC Web Services as well as clients. deegree is Free Software protected by the GNU Lesser General Public License (GNU LGPL) and accessible at www.deegree.org.

deegree2 is the new release of deegree supporting a number of features deegree1 was not able to handle. This documentation describes setup and configuration of deegree2 Sensor Observation Service (SOS), an implementation of OGC's Sensor Collection Service Specification 0.9 (an internal OGC paper).

The deegree Sensor Observation Service (SOS) is an implementation of the Open Geospatial Consortium's (OGC) Sensor Observation Service, which is, in turn, a central service specification of the OGC's Sensor Web Enablement (SWE) Initiative. SWE is concerned with the development of concepts and specifications for the creation of an open platform, through which any type of sensor can be accessed. Application fields are varied and range from satellite-based remote sensing to personal, mobile sensors.

Currently three specifications of the SWE Initiative are relevant to the deegree SOS:

- Sensor Model Language (SensorML), a XML definition for Sensors (OGC Recommendation Paper #04-019r2),
- Observations and Measurements (O&M), a XML definition for measurements (OGC Recommendation Paper #03-022r3), and
- Sensor Observation Service (SOS, formerly known as Sensor Collection Service, SCS), a service description, which is tightly coupled with the Web Feature Service (WFS) and the Web Coverage Service (WCS) (Internal OGC-Specification draft #03-023r1)

The deegree SOS permits the interoperable access to measurements. As of 2005-10-20, the deegree SOS is feature-based (WFS).

Besides a SOS, deegree comprises a number of additional services and clients. A complete list of deegree components can be found at:

<http://www.lat-lon.de> → Products

Downloads of packaged deegree components can be found at:

<http://www.deegree.org> → Download

The web services of deegree are realized as Java modules controlled by one central servlet (a “dispatcher”). This servlet has to be integrated into the respective web server/servlet engine. Most of the common web servers support servlet technology, thus making deegree a universal product. The Apache-Tomcat 5.5 Servlet-Engine is recommended due to its widespread use and its status as an open-source product.

2 Get deegree SOS up and running

2.1 Download and archive extraction

Visit <http://www.deegree.org> and download the Sensor Observation Service package appropriate to your platform.

Extract the zip archive (using WinZip or a similar software) to your hard disk, e.g.:

LINUX: /home/deegree or /opt/deegree or /usr/local/deegree

Windows: /program files/deegree

In the remainder of this document, the directory you extracted the zip archive to is referred to as `$root_directory$`.

Your `$root_directory$` will contain the following structure:

directory	Content
./WEB-INF	Required by Tomcat, containing all libraries, configuration- and data-files
./WEB-INF/conf/sos	SOS configuration files
./WEB-INF/data/db/scripts	SQL scripts for creating and filling example database

2.2 Tomcat integration

(Download and install a current release of Tomcat, you also need a JDK 5 installed for this).

The location of deegree's libraries and the central deegree configuration file `sos_capabilities.xml` should be registered with the Servlet Engine (in this case Apache Tomcat 5.5). Tomcat offers several possibilities to register and configure web contexts.

The easiest way to register deegree web services with Tomcat is to use the Tomcat manager that can be accessed from your Tomcat starting page (at: <http://my.server.domain>). Use the "WAR or directory URL" field to point to your installation in the following way: **file:\$root-directory\$**.

If you want to do the Tomcat installation process manually use the steps described in the following.

First Tomcat needs information about the root directory of the SOS (and other deegree web services if installed in the same server context). This has to be realized by adding an additional <Context> tag under the already existing <Host> tag in `server.xml`, situated in the conf directory of the Tomcat installation.

```
<Service>
  <Engine>
    <Host>
      <Context path="/deegree"
        docBase="$root_directory$"
        crossContext="false"
        debug="0"
        reloadable="false"/>
    </Host>
  </Engine>
</Service>
```

The `path` attribute describes the virtual location of the main directory of the deegree web service. In the example, the root directory of the service is accessible at `http://my.server.domain/deegree/`. The `docbase` attribute reflects the physical location of the deegree service in the file system. The other attributes and their values are essential for good performance. It is possible to store the <Context> element in its own XML-file in directory `$TOMCAT_HOME/conf/Catalina/localhost`. For further information have a look at the Tomcat documentation included with the installation.

The name of the main directory is arbitrary whereas Tomcat definitely looks for a subdirectory WEB-INF (in capital letters – even on a Windows system) in the root directory. This directory will be automatically created after unpacking the zip archive. Here the Deployment-Descriptor (`web.xml`) is located, which is analysed by Tomcat to identify the servlet(s) belonging to the application, their names, the parameters that are delivered to the servlet(s) and information about the existing access restrictions.

Before starting deegree SOS the following entry in `web.xml` is essential. The relevant elements of the `$deegreeWebApp/WEB-INF/web.xml` (also known as deployment descriptor) are described below.

Make sure the `org.deegree.enterprise.servlet.OGCServletController` servlet is available, has the following initial parameter, and has a handler, as given by the elements below:

```
<web-app>
  <servlet>
    <servlet-name>ogcwebservice</servlet-name>
    <servlet-class>
      org.deegree.enterprise.servlet.OGCServletController
    </servlet-class>
    <init-param>
      <param-name>services</param-name>
      <param-value>SOS</param-value>
    </init-param>
    <init-param>
      <param-name>SOS.handler</param-name>
      <param-value>org.deegree.enterprise.servlet.SOSHandler</param-value>
    </init-param>
    <init-param>
      <param-name>SOS.config</param-name>
      <param-value>WEB-INF/conf/sos/sos_capabilities.xml</param-value>
    </init-param>
  <!-- skip other elements... -->
```

Note that the servlet name can be chosen arbitrarily (see below). Also note that the capabilities file `sos_capabilities.xml` is the basis of the capabilities document delivered for a `GetCapabilities` request and is, at the same time, a configuration file. This document has to be adapted to your needs. The servlet mentioned above is included in the `deegree.jar`, which you should have put in the `$deegreeWebApp/WEB-INF/lib` directory.

Further down the `web.xml` you need to include a servlet mapping:

```
<servlet-mapping>
  <servlet-name>ogcwebservice</servlet-name>
  <url-pattern>/ogcwebservice</url-pattern>
</servlet-mapping>
```

The `servlet-name` can be any, but has to match the one defined for the `org.deegree.enterprise.servlet.OGCServletController`. The `servlet-name` pattern can be any valid URL fragment. Your web application will be accessed through this url. In case you're using a `deegree WFS` as the data source for your `SOS`, you can include the relevant piece of information in the `web.xml`, without the need to set up another Tomcat Web Application. Please consult the introductory documentation on how to do that.

2.3 Required libraries

Make sure the following libraries have been added to `$deegreeWebApp/WEB-INF/libs`

- `deegree.jar`
- `commons-httpclient-2.0.2.jar`

- commons-logging.jar
- commons-logging-api.jar
- j3dcore.jar
- j3dutils.jar
- vecmath.jar
- jts.jar

You might also need other libraries, in case you have other deegree services, e.g. a WMS, running in the same Tomcat context.

2.4 Architecture

The deegree SOS like deegree CSW or Gazetteer represents a facade on top of the deegree Web Feature Service (WFS). The WFS is deegree's central accessing unit for vector data of any kind. So configuration of a SOS instance requires configuration of at least one (internal) WFS and a SOS modul on top of the WFS. The WFS instance(s) used by a SOS are referenced in the central configuration file of SOS (see below).

Figure 1: associations between SOS and WFS configuration files

Notice that the assigned Web Feature Services do not need to be available through a web interface. They will be accessed by deegree SOS through the deegree API. As an alternative you can use remote WFS through the WFS web interface as defined by the OGC but we do not recommend this in because it is significantly slower.

2.5 Configuration

As mentioned, the `sos_capabilities.xml` is the main configuration file for the deegree Sensor Observation Service. This file is a valid XML document including a subset that will be delivered as response to a `GetCapabilities` request. The element `deegree:deegreeParams` contains deegree-specific configuration information and is removed from the document prior to the `GetCapabilities` request. These deegree-specific parameters are described as follows.

2.6 deegree Parameters

The sub-elements of the `deegree:deegreeParams` are listed and explained below:

DefaultOnlineResource

```
<deegree:DefaultOnlineResource  
  xlink:type="simple" xlink:href="http://localhost:8081/deegree/services"/>
```

This parameter defines a simple link to a default online resource. It will be used if a required `OnlineResource` is not defined.

CacheSize

```
<deegree:CacheSize>1</deegree:CacheSize>
```

With this parameter you can define the size of the cache used by the SOS. At the moment caching is not defined. This is a placeholder for future deegree releases.

RequestTimeLimit

```
<deegree:RequestTimeLimit>60</deegree:RequestTimeLimit>
```

This parameter defines the timeout value in seconds for a request. That is, how long the service is allowed to wait till it declares the service unreachable.

SourceServerTimeLimit

```
<deegree:SourceServerTimeLimit>60</deegree:SourceServerTimeLimit>
```

Like above, but concerning the underlying WFS.

Encoding

```
<degree:Encoding>UTF-8</degree:Encoding>
```

Defines the encoding value of the requests send to a WFS and the SOS's responses.

SourceServerList

```
<degree:SourceServerList>
  <degree:SourceServer id="wfs1" service="OGC:WFS" version="1.0.0">
    <ogc:OnlineResource xlink:type="simple"
      xlink:href="file:///e:/sos/WEB-INF/conf/sos/wfs_capabilities.xml"/>
    <degree:PlatformDescription>
      <degree:FeatureType>scs_platforms</degree:FeatureType>
      <degree:IdPropertyName>ID</degree:IdPropertyName>
      <degree:CoordPropertyName>
        location/coordinates
      </degree:CoordPropertyName>
      <degree:XSLTSource>
        transform_describe_platform.xml
      </degree:XSLTSource>
    </degree:PlatformDescription>
    <degree:SensorDescription>
      <degree:FeatureType>scs_sensors</degree:FeatureType>
      <degree:IdPropertyName>ID</degree:IdPropertyName>
      <degree:XSLTSource>transform_describe_sensor.xml</degree:XSLTSource>
    </degree:SensorDescription>
  </degree:SourceServer>
<!-- skip other source servers -->
</degree:SourceServerList>
```

This somewhat longer section is divided into three parts: (1) the data source (the WFS), (2) which FeatureTypes of the source WFS are mapped to which SOS type, that is Platform or Sensor, and (3) how the WFS response (GML) is to be transformed so that it represents valid SensorML. The sub-elements of SourceServerList are detailed as follows.

SourceServer

This element needs a unique id attribute, and attributes defining the service and its version. Currently the only accepted service value is „OGC:WFS“.

OnlineResource defines a simple link to the WFS capabilities document of this data source. Note that the xlink:href must be a valid URL. If a file URL is used degree assumes that it points to the capabilities/configuration of a degree WFS. In this case the SOS will call the WFS via the degree API. If a HTTP URL is used the SOS will access the WFS by the web interface defined by OGC WFS specification.

PlatformDescription: defines the relation between a SOS platform and its underlying FeatureType.

The `FeatureType` element is the WFS `FeatureType` that represents the platform.

The `IdPropertyName` is the unique identifier of the above `FeatureType`. The `CoordPropertyName` is the (geometry) property of the above `FeatureType`. It must be a valid geometry property of Point type. The `XSLTSource` element defines how the WFS output is to be transformed into a valid SensorML. This elements must point to a XSLT file in your system. It's up to you to adapt this transformation script and make sure it is valid SensorML. The following XSLT fragment shows an example for a script to transform a response of a `GetFeature` request (GML) to `SensorDescriptionML`:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:gml="http://www.opengis.net/gml" xmlns:sml="http://www.opengis.net/sensorML"
  xmlns:iso19115="http://www.opengis.net/iso19115"
  xmlns:deegreewfs="http://www.deegree.org/wfs" version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="deegreewfs:FeatureCollection">
    <xsl:element name="sml:Sensors">
      <xsl:apply-templates select="./gml:featureMember"/>
    </xsl:element>
  </xsl:template>
  <xsl:template match="gml:featureMember/scs_sensors">
    <xsl:element name="sml:Sensor">
      <xsl:attribute name="id"><xsl:value-of select="ID"/></xsl:attribute>
      <sml:identifiedAs>
        <sml:Identifier>
          <xsl:value-of select="ID"/>
        </sml:Identifier>
      </sml:identifiedAs>
      <sml:attachedTo>
        <sml:Component>21001</sml:Component>
      </sml:attachedTo>
    </xsl:element>
  </xsl:template>
</xsl:stylesheet>
```

The `SensorDescription` serves the same use as `PlatformDescription`, but for the sensors.

Note that you can define any number of `SourceServers`. The advantage of this approach is to have a distributed system, with, perhaps a WFS serving metadata (Platform and Sensor data) and another serving observation data.

2.7 OWS Commons parameters

For a detailed description of the following parameters – if not in deegree namespace - please have a look at the OGC commons specification.

2.7.1 ServiceIdentification

The `ServiceIdentification` element encloses elements for the service type description, `ServiceType`, here always 'SOS', the `ServiceTypeVersion`, as well as a title and an abstract. These elements are shown below:

```
<ows:ServiceIdentification>
  <ows:ServiceType>SOS</ows:ServiceType>
```

```

<ows:ServiceTypeVersion>1.0</ows:ServiceTypeVersion>
<ows:Title>deegree Sensor Collection Service</ows:Title>
<ows:Abstract>
  A Sensor Collection Service implemented with
  the deegree Framework.
</ows:Abstract>
</ows:ServiceIdentification>

```

The `ServiceProvider` element encloses elements to describe the provider and his contact data. You may want to modify these to match your data. The element is shown below:

```

<ows:ServiceProvider>
  <ows:ProviderName>lat/lon</ows:ProviderName>
  <ows:ProviderSite xlink:href="http://www.lat-lon.de"/>
  <ows:ServiceContact>
    <ows:IndividualName>Andreas Poth</ows:IndividualName>
    <ows:PositionName>Technical Director</ows:PositionName>
    <ows:ContactInfo>
      <ows:Phone>
        <ows:Voice>+49 0228 184960</ows:Voice>
        <ows:Facsimile>+49 0228 1849629</ows:Facsimile>
      </ows:Phone>
      <ows:Address>
        <ows:DeliveryPoint>Aennchenstr. 19</ows:DeliveryPoint>
        <ows:City>Bonn</ows:City>
        <ows:AdministrativeArea>NRW</ows:AdministrativeArea>
        <ows:PostalCode>53177</ows:PostalCode>
        <ows:Country>Germany</ows:Country>
        <ows:ElectronicMailAddress>
          poth@lat-lon.de
        </ows:ElectronicMailAddress>
      </ows:Address>
    </ows:ContactInfo>
  </ows:ServiceContact>
</ows:ServiceProvider>

```

2.7.2 OperationsMetadata

This element lists all operations implemented by the SOS. It is possible to define the protocols allowed for each the operations. Currently, deegree SOS only supports HTTP GET requests for the GetCapabilities operation. Note that you can also define the format of the exception.

```

<ows:OperationsMetadata>
  <ows:Operation name="GetCapabilities">
    <ows:DCP>
      <ows:HTTP>
        <ows:Get xlink:type="simple"
          xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
        <ows:Post xlink:type="simple"
          xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
      </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="Format">
      <ows:Value>text/xml</ows:Value>
    </ows:Parameter>
    <ows:Parameter name="ExceptionFormat">
      <ows:Value>any other exception type</ows:Value>
    </ows:Parameter>
  </ows:Operation>
  <ows:Operation name="GetObservation">
    <ows:DCP>

```

```

        <ows:HTTP>
            <ows:Post xlink:type="simple"
                xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
        </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="Format">
        <ows:Value>text/xml</ows:Value>
    </ows:Parameter>
</ows:Operation>
<ows:Operation name="DescribeSensor">
    <ows:DCP>
        <ows:HTTP>
            <ows:Get xlink:type="simple"
                xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
            <ows:Post xlink:type="simple"
                xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
        </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="Format">
        <ows:Value>text/xml</ows:Value>
    </ows:Parameter>
</ows:Operation>
<ows:Operation name="DescribePlatform">
    <ows:DCP>
        <ows:HTTP>
            <ows:Get xlink:type="simple"
                xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
            <ows:Post xlink:type="simple"
                xlink:href="http://localhost:8081/deegree2/ogcwebservice"/>
        </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="Format">
        <ows:Value>text/xml</ows:Value>
    </ows:Parameter>
</ows:Operation>
<ows:Parameter name="ExceptionFormat">
    <ows:Value>text/xml</ows:Value>
</ows:Parameter>
</ows:OperationsMetadata>

```

2.7.3 PlatformList

The PlatformList element defines a list of Platform elements each of which describing a platform.

```

<scs:PlatformList>
    <scs:Platform Id="Bad Godesberg rain gage"
        Description="Bad Godesberg Rain gaging station">
        <deegree:DescriptionSource>
            <deegree:SourceServerID>wfs1</deegree:SourceServerID>
            <deegree:IdPropertyValue>123456</deegree:IdPropertyValue>
        </deegree:DescriptionSource>
    </scs:Platform>
<!-- skip other platforms -->
</scs:PlatformList>

```

Note that, for each platform, you have to input a platform identity (`Id` attribute) as well as a description of this platform. The enclosed `deegree:DescriptionSource` is deegree specific and has the purpose of identifying the data source (`deegree:sourceServerID` element), that is, the WFS, and the identity of the Feature representing the platform (`IdPropertyValue`). The `Id` need not be a integer number, but the unique identifier of this underlying `FeatureType`. You may add any number of platforms to this list.

2.7.4 SensorList

The `SensorList` element is similar to the `PlatformList`.

```
<scs:SensorList>
  <scs:Sensor Id="Bad Godesberg barometer" Description="Barometer on Godesburg">
    <deegree:DescriptionSource>
      <deegree:SourceServerID>wfs1</deegree:SourceServerID>
      <deegree:IdPropertyValue>789123</deegree:IdPropertyValue>
    </deegree:DescriptionSource>
    <deegree:MeasurementList>
      <deegree:Measurement id="air_pressure" phenomenon="air pressure">
        <deegree:SourceServerID>wfs1</deegree:SourceServerID>
        <deegree:XSLTSource>transform_observation.xsl</deegree:XSLTSource>
        <deegree:FeatureTypeName>air_pressure</deegree:FeatureTypeName>
        <deegree:Constraint>
          <ogc:Filter>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>LOG_ID</ogc:PropertyName>
              <ogc:Literal>23001</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Filter>
        </deegree:Constraint>
        <deegree:TimePropertyName>TIME</deegree:TimePropertyName>
        <deegree:MeasurandPropertyName>
          MEASURAND
        </deegree:MeasurandPropertyName>
        <deegree:TimeResolution type="hour">1</deegree:TimeResolution>
      </deegree:Measurement>
    </deegree:MeasurementList>
  </scs:Sensor>
</scs:SensorList>
```

Note the deegree parameters defining the XSLT document responsible for the transformation from `FeatureType` (GML, delivered by the WFS) to `Observation` (SensorML). It is therefore necessary to adapt the XSLT script to transform the WFS output into one that is conformant to the SOS.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:gml="http://www.opengis.net/gml" xmlns:sml="http://www.opengis.net/sensorML"
  xmlns:iso19115="http://www.opengis.net/iso19115"
  xmlns:om="http://www.opengis.net/om" version="1.0">
  <xsl:output method="xml"/>
  <xsl:template match="ResultCollection">
    <om:ObservationArray>
      <om:observationMembers>
        <xsl:apply-templates select="./gml:featureMember"/>
      </om:observationMembers>
    </om:ObservationArray>
```

```
</xsl:template>
<xsl:template match="gml:featureMember">
  <gml:Observation>
    <gml:timeStamp>
      <gml:TimeInstant>
        <gml:timePosition>
          <xsl:value-of select="./observation/TI_LIMNO_PHYS_CHEM_LOG_H1.TIME"/>
        </gml:timePosition>
      </gml:TimeInstant>
    </gml:timeStamp>
    <gml:resultOf>
      <gml:QuantityList>
        <xsl:value-of select="./observation/TI_LIMNO_PHYS_CHEM_LOG_H1.WERT"/>
      </gml:QuantityList>
    </gml:resultOf>
  </gml:Observation>
</xsl:template>
</xsl:stylesheet>
```

3 Appendix A: Example SOS configuration/capabilities document

Example for a full deegree SOS configuration/capabilities document:

```
<?xml version="1.0" encoding="UTF-8"?>
<scs:SCS_Capabilities xmlns="http://www.opengis.net/scs"
  xmlns:deegree="http://www.deegree.org/sos" xmlns:scs="http://www.opengis.net/sos"
  xmlns:ows="http://www.opengis.net/ows" xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" updateSequence="0"
  version="1.0">
  <deegree:deegreeParams>
    <deegree:RootDirectory>D:/sos/webapps/services</deegree:RootDirectory>
    <deegree:DefaultOnlineResource xlink:type="simple"
      xlink:href="http://localhost:8081/deegree/services"/>
    <deegree:CacheSize>1</deegree:CacheSize>
    <deegree:RequestTimeLimit>60</deegree:RequestTimeLimit>
    <!-- set the timeout for the wfs requests-->
    <deegree:SourceServerTimeLimit>60</deegree:SourceServerTimeLimit>
    <deegree:Encoding>UTF-8</deegree:Encoding>
    <!-- set all requestable WFS Servers-->
    <deegree:SourceServerList>
      <deegree:SourceServer id="wfs1" service="OGC:WFS" version="1.0.0">
        <ogc:OnlineResource xlink:type="simple"
          xlink:href="file:///D:/sos/WEB-INF/conf/sos/wfs_capabilities.xml"/>
        <deegree:PlatformDescription>
          <deegree:FeatureType>scs_platforms</deegree:FeatureType>
          <deegree:IdPropertyName>ID</deegree:IdPropertyName>
          <deegree:CoordPropertyName>
            coordinates
          </deegree:CoordPropertyName>
          <deegree:XSLTSource>
            file:///D:/sos/WEB-INF/conf/sos/transform_describe_platform.xml
          </deegree:XSLTSource>
        </deegree:PlatformDescription>
        <deegree:SensorDescription>
          <deegree:FeatureType>scs_sensors</deegree:FeatureType>
          <deegree:IdPropertyName>ID</deegree:IdPropertyName>
          <deegree:XSLTSource>
            file:///D:/sos/WEB-INF/conf/sos/transform_describe_sensor.xml
          </deegree:XSLTSource>
        </deegree:SensorDescription>
      </deegree:SourceServer>
    </deegree:SourceServerList>
  </deegree:deegreeParams>
  <ows:ServiceIdentification>
    <ows:ServiceType>SOS</ows:ServiceType>
    <ows:ServiceTypeVersion>1.0</ows:ServiceTypeVersion>
    <ows:Title>deegree Sensor Collection Service</ows:Title>
    <ows:Abstract>A Sensor Collection Service implemented in the deegree
      Framework.</ows:Abstract>
  </ows:ServiceIdentification>
  <ows:ServiceProvider>
    <ows:ProviderName>lat/lon</ows:ProviderName>
    <ows:ProviderSite xlink:href="http://www.lat-lon.de"/>
    <ows:ServiceContact>
      <ows:IndividualName>Andreas Poth</ows:IndividualName>
      <ows:PositionName>Technical Director</ows:PositionName>
    </ows:ServiceContact>
  </ows:ServiceProvider>
</scs:SCS_Capabilities>
```

```

<ows:ContactInfo>
  <ows:Phone>
    <ows:Voice>0228 184960</ows:Voice>
    <ows:Facsimile>0228 1849623</ows:Facsimile>
  </ows:Phone>
  <ows:Address>
    <ows:DeliveryPoint>Aennchenstr. 19</ows:DeliveryPoint>
    <ows:City>Bonn</ows:City>
    <ows:AdministrativeArea>NRW</ows:AdministrativeArea>
    <ows:PostalCode>53177</ows:PostalCode>
    <ows:Country>Germany</ows:Country>
    <ows:ElectronicMailAddress>
      <a href="mailto:poth@lat-lon.de">poth@lat-lon.de</a>
    </ows:ElectronicMailAddress>
  </ows:Address>
</ows:ContactInfo>
</ows:ServiceContact>
</ows:ServiceProvider>
<ows:OperationsMetadata>
  <ows:Operation name="GetCapabilities">
    <ows:DCP>
      <ows:HTTP>
        <ows:Get xlink:type="simple"
          xlink:href="http://localhost:8081/deegree/services/">
        <ows:Post xlink:type="simple"
          xlink:href="http://localhost:8081/deegree/services/">
        </ows:HTTP>
      </ows:DCP>
      <ows:Parameter name="Format">
        <ows:Value>text/xml</ows:Value>
      </ows:Parameter>
      <ows:Parameter name="ExceptionFormat">
        <ows:Value>any other exception type</ows:Value>
      </ows:Parameter>
    </ows:Operation>
    <ows:Operation name="GetObservation">
      <ows:DCP>
        <ows:HTTP>
          <ows:Post xlink:type="simple"
            xlink:href="http://localhost:8081/deegree/services/">
          </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="Format">
          <ows:Value>text/xml</ows:Value>
        </ows:Parameter>
      </ows:Operation>
    <ows:Operation name="DescribeSensor">
      <ows:DCP>
        <ows:HTTP>
          <ows:Get xlink:type="simple"
            xlink:href="http://localhost:8081/deegree/services/">
          <ows:Post xlink:type="simple"
            xlink:href="http://localhost:8081/deegree/services/">
          </ows:HTTP>
        </ows:DCP>
        <ows:Parameter name="Format">
          <ows:Value>text/xml</ows:Value>
        </ows:Parameter>
      </ows:Operation>
    <ows:Operation name="DescribePlatform">
      <ows:DCP>
        <ows:HTTP>
          <ows:Get xlink:type="simple"
            xlink:href="http://localhost:8081/deegree/services/">
          <ows:Post xlink:type="simple"
            xlink:href="http://localhost:8081/deegree/services/">

```



```

        </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="Format">
        <ows:Value>text/xml</ows:Value>
    </ows:Parameter>
</ows:Operation>
<ows:Parameter name="ExceptionFormat">
    <ows:Value>text/xml</ows:Value>
</ows:Parameter>
</ows:OperationsMetadata>
<scs:PlatformList>
    <scs:Platform Id="21001" Description="Wuppertalsperre_Ablauf_Abruf">
        <deegree:DescriptionSource>
            <deegree:SourceServerID>wfs1</deegree:SourceServerID>
            <deegree:IdPropertyValue>21001</deegree:IdPropertyValue>
        </deegree:DescriptionSource>
    </scs:Platform>
    <scs:Platform Id="21002" Description="Dhünn_Rohdaten">
        <deegree:DescriptionSource>
            <deegree:SourceServerID>wfs1</deegree:SourceServerID>
            <deegree:IdPropertyValue>21002</deegree:IdPropertyValue>
        </deegree:DescriptionSource>
    </scs:Platform>
</scs:PlatformList>
<scs:SensorList>
    <scs:Sensor Id="22001" Description="Wuppertalsperre Ablauf Temperatur">
        <deegree:DescriptionSource>
            <deegree:SourceServerID>wfs1</deegree:SourceServerID>
            <deegree:IdPropertyValue>22001</deegree:IdPropertyValue>
        </deegree:DescriptionSource>
        <deegree:MeasurementList>
            <deegree:Measurement id="Wuppertalsperre Ablauf Temperatur"
                phenomenon="tempreture">
                <deegree:SourceServerID>wfs1</deegree:SourceServerID>
                <deegree:XSLTSource>
                    <a href="file:///D:/sos/transform_observation.xsl">file:///D:/sos/transform_observation.xsl</a>
                </deegree:XSLTSource>
                <deegree:FeatureTypeName>
                    observation1H
                </deegree:FeatureTypeName>
                <deegree:Constraint>
                    <ogc:Filter>
                        <ogc:PropertyIsEqualTo>
                            <ogc:PropertyName>LOG_ID</ogc:PropertyName>
                            <ogc:Literal>23001</ogc:Literal>
                        </ogc:PropertyIsEqualTo>
                    </ogc:Filter>
                </deegree:Constraint>
                <deegree:TimePropertyName>TIME</deegree:TimePropertyName>
                <deegree:MeasurandPropertyName>
                    MEASURAND
                </deegree:MeasurandPropertyName>
                <deegree:TimeResolution type="hour">1</deegree:TimeResolution>
            </deegree:Measurement>
        </deegree:MeasurementList>
    </scs:Sensor>
    <scs:Sensor Id="22002" Description="Wuppertalsperre Ablauf ph-Wert">
        <deegree:DescriptionSource>
            <deegree:SourceServerID>wfs1</deegree:SourceServerID>
            <deegree:IdPropertyValue>22002</deegree:IdPropertyValue>
        </deegree:DescriptionSource>
        <deegree:MeasurementList>
            <deegree:Measurement id="Wuppertalsperre Ablauf ph-Wert"
                phenomenon="ph">
                <deegree:SourceServerID>wfs1</deegree:SourceServerID>
                <deegree:XSLTSource>

```

```

file:///D:/sos/WEB-INF/conf/sos/transform_observation.xsl
</degree:XSLTSource>
<degree:FeatureTypeName>
  observation1H
</degree:FeatureTypeName>
<degree:Constraint>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>LOG_ID</ogc:PropertyName>
      <ogc:Literal>23002</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
</degree:Constraint>
<degree:TimePropertyName>TIME</degree:TimePropertyName>
<degree:MeasurandPropertyName>
  MEASURAND
</degree:MeasurandPropertyName>
<degree:TimeResolution type="hour">1</degree:TimeResolution>
</degree:Measurement>
</degree:MeasurementList>
</scs:Sensor>
</scs:SensorList>
</scs:SCS_Capabilities>

```

4 Appendix C: Third party libraries

deegree uses and redistributes the following third party libraries:

JTS - Java Topology Suite:

<http://www.vividsolutions.com/jts/jtshome.htm>

The JTS Topology Suite is an API of 2D spatial predicates and functions. It has the following design goals:

- JTS conforms to the Simple Features Specification for SQL published by the Open GIS Consortium
- JTS provides a complete, consistent, robust implementation of fundamental 2D spatial algorithms
- JTS is fast enough for production use
- JTS is written in 100% pure Java™

JTS is open source (under the LGPL license)

ACME Java - Software

<http://www.acme.com/java/software/>

All of this is completely free for any use, educational commercial or whatever. I do have to earn my grocery money, though. I'm available for consulting, and would be very interested in work that builds on these Java utilities. And of course if your company wants to sponsor ACME Labs with equipment or large cash donations, that would be ok too!

This software is written for JDK1.0.2, but with one exception it also compiles under JDK1.1. You'll get a lot of warnings about deprecated methods, but it'll work. The only exception is StubToolkit, where I had to use some of the internal sun.* classes.

Apache Batik

<http://xml.apache.org/batik/index.html>

Batik is a Java(tm) technology based toolkit for applications or applets that want to use images in the [Scalable Vector Graphics \(SVG\)](#) format for various purposes, such as viewing, generation or manipulation.

1. Applications of Batik
2. The SVG Specification
3. What is Batik's Implementation Status?
4. Downloading the Batik distribution (source and binary)
5. Examples of projects and products using Batik

The project's ambition is to give developers a set of [core modules](#) which can be used together or individually to support specific SVG solutions. Examples of [modules](#) are the [SVG Parser](#), the [SVG Generator](#) and the [SVG DOM](#). Another ambition for the Batik project is to make it highly [extensible](#) (for example, Batik allows the developer to handle custom SVG tags). Even though the goal of the project is to provide a set of core modules, one of the deliverables is a full fledged [SVG browser](#) implementation which validates the various modules and their interoperability.

Jakarta Commons

<http://jakarta.apache.org/commons/index.html>

The Commons is a Jakarta subproject focused on all aspects of reusable Java components.

The Jakarta Commons project is composed of two parts:

- [The Commons Proper](#) - A repository of reusable Java components.
- [The Commons Sandbox](#) - A workspace for Java component development.

Apache log4j

Apache log4j is a logging api provided by the apache project.

PROJ.4 - Cartographic Projections Library

<http://proj.maptools.org/>

Program proj release is a standard Unix function which converts geographic longitude and latitude coordinates into cartesian coordinates by means of a wide variety of cartographic projection functions. For many of the projection functions the inverse conversion can also be performed.