Sensor Web Enablement

Installation and Developement Guide for uDig-SOS-Plugin 1.0

Document Change Control

Revision Number	*Date Of Issue*	*Author(s)*	*Brief Description Of Changes*
1.0	18.09.09	Martin Kiesow	initial version

Editors

Martin Kiesow

Email: martin.kiesow@uni-muenster.de

Licence

Contact: Andreas Wytzisk,
52 North Initiative for Geospatial Open Source Software GmbH,
Martin-Luther-King-Weg 24,
48155 Muenster, Germany, info@52north.org

For more information, contact:

52°North Initiative for Geospatial Open Source Software GmbH
Martin-Luther-King-Weg 24
48155 Münster, Germany

www.52north.org

Table of contents

1 Introduction

1.1 Scope

This document describes the general architecture and the different ways to install the uDig-SOS-Plugin.

1.2 Characteristics of the UDIG-SOS-Plugin

The uDig-SOS Plugin offers an uDig-/Geotoolsplugin to represent a client for Sensor Observation Service (SOS) communication using the 52° North OX-Framework (OXF). The plug-in enables the user to define requests, send them to a SOS and unmarshal the returned result to geotools compatible features.

It supports the following SOS operations :

- *GetCapabilities*: Requests information about the SOS, containing general descriptive information about the service and request parameters
- *GetObservation*: for requesting the sensor data encoded in Observation&Measurements (O&M)
- *GetObservationByID*: for requesting the pure sensor data for a given ObservationID encoded in Observation&Measurements (O&M)
- *GetFeatureOfInterest*: for requesting the GML encoded representation of the feature that is the target of an observation.

1.3 Architecture

A DataStore is the Geotools-API for data source access and it is used to access both file- and service-informations. This plugin uses the OX-Framework for SOS communication and creates DataStores offering access to features delivered by the OXF using the Geotools FeatureReader interface, which offers sequential access to geotools-compatible features.

For more specific information about Datastores, Features, Layers and the Udig Pluginsystem, refer to the uDig Developer's Guide and the Geotools Developers Guide.

The plugin interprets every field of an Observation(-Collection) or Measurement obtained from a SOS as a single Feature. Therefore in addition to the result of a single field additional information (i.e. the SamplingTime, or the FeatureOfInterest (FOI)) is stored within the feature. A complete list with name and used data type in Java can be found in Table 1.

As for this version the general structure of features obtained by the SOSFeatureReader is as follows:

Table 1: uDig SOS-Plugin: Feature Data Model

name	type
description	class java.lang.String
name	class java.lang.String

name	type
location	class com.vividsolutions.jts.geom.Geometry
procedure	class java.lang.String
observedProperty	class org.n52.oxf.feature.PhenomenonPropertyType
featureOfInterest	interface org.geotools.feature.Feature
samplingTime	interface org.n52.oxf.owsCommon.capabilities.ITime
quality	interface org.n52.oxf.feature.IQuantity
result	class java.lang.Object

The attribute with the name „result“ represents the result of the single field delivered by the SOS. As they may have different types and complexity it is necessary to store them in different Java types depending on their XML type. This mapping is described in Table 2.

Table 2: ResultTypes

XML type	Java Type
Quantity	class org.n52.oxf.feature.MeasureType
Boolean	class java.lang.Boolean
Time	interface org.n52.oxf.owsCommon.capabilities.ITime
Count	class java.lang.Integer
Category	Class org.n52.oxf.feature.ScopedName

2 Requirements

- User-Friendly Desktop Internet GIS (UDIG) 1.1.RC14 or later

  [tested with 1.1.RC14, 1.1.SC3]
- JRE/JDK 1.5 or later [tested with 1.6]
- Eclipse RCP 3.2.2 Europa Winter or later [tested with 3.2.2, 3.4.1]

  (for building the JAR; make sure you choose the RCP/Plug-in Developers version)
- SVN-Client (already contained Eclipse 3.4)

3 Installation Procedure

3.1 Installation

This chapter describes the installation process:

1. Follow 3.1.1 to install the plugin from the jar, or follow 3.1.2 to run the plugin from Eclipse.
2. Proceed with configuration in 3.3

3.1.1 Installation with jar

- Download User-Friendly Desktop Internet GIS (uDig) Release 1.1-SC3 from:

  http://udig.refractions.net

  Follow the installation instructions on the website and on the installer to install User-Friendly Desktop Internet GIS.
- Copy the UDIG-SOSPlugin_v1.0.jar into the *plugins* folder of your UDIG installation directory

3.1.2 Installation from SVN in eclipse

3.1.2.1 Set up your development environment

- Download the UDIG SDK consistent with your UDIG program version from

  http://udig.refractions.net/files/downloads, e.g.

  http://udig.refractions.net/files/downloads/uDig11/udig-1.1-SC3-sdk.zip

  and unzip the package to a directory of your choice.
- Download the extra file package consistent with your Eclipse version from

  http://udig.refractions.net/files/downloads/extras

  and unzip it to your Eclipse directory in a way that *features* and *plugins* match the corresponding directories.

- In Eclipse add your UDIG's JRE to your *Installed JREs* under *Window → Preferences → Java → Installed JREs → Add... → Standard VM*, e.g.

  C:\Program Files\uDig\1.1-SC3\eclipse\jre

- Select your UDIG SDK directory under *Window → Preferences → Plug-in Development → Target Platform → Browse...*
- For further information download the UDIG SDK Quickstart Guide from:

  http://udig.refractions.net/tutorials/SDKQuickstartStable.pdf

3.1.2.2 Check out the plug-in

- Get the plugin from SVN:

  https://svn.52north.org/svn/swe/main/Clients/uDig/SOS/trunk

  Choose *Check out as a project configured using the new Project Wizard: → Plug-in Project*, choose a name and don't select any template.

  You may uncheck *Generate an activator, a Java class that controls the plug-in's life cycle* which is redundant.

- In your new project open META-INF\MANIFEST.MF → *Runtime* tag. Remove all *.jar archives from the *Classpath* section and add them again from the \lib directory (don't add the whole directory, just the contained archives). Save MANIFEST.MF.
- Change your build path's JRE: *Window → Preferences → Java Build Path → Libraries → Add Library... → JRE System Library → Alternate JRE:* and select your UDIG's JRE.

3.1.2.3 Build and start the plug-in

Now you have three different ways to proceed:

- Run UDIG from Eclipse:

  Run → Run Configurations... → Eclipse Application → New → check *Run a product:* net.refractions.udig.product and *Runtime JRE:* your UDIG's JRE → *Run*

- Create a jar with *File → Export... → Plug-in Development → Deployable plug-ins and fragments*, which can be deployed to the „plugins“ folder of your UDIG installation directory (see 3.1.1 Installation with jar).
- In Eclipse execute the target in the buildplugin.xml file with an Ant-Builder where target defines the target of the ant build file, that should be executed. Leave the target value blank or use „plugin_export“ to to start the creation of the jar file. The result of the build procedure is listed in *Listing 1*.

  In the case Ant cannot identify pde.exportPlugins you might have to change the JRE by right clicking the Ant task → *Run As → 2 Ant Build → JRE → Run in the same JRE as the workspace → Run*.

  The jar will be copied to <currentdirectory>/dist/plugins

Listing 1: Ant output: plugin_export

```
Buildfile: D:\Java\workspace\net.refractions.udig.catalog.sos\buildPlugin.xml

plugin_export:
BUILD SUCCESSFUL
Total time: 221 milliseconds
```

3.2 Directory structure

The directory structure of the Udig-SOS-Plugin is as follows (see Figure 2):

- `doc:` installation guide, base directory for the source code documentation generated by javadoc
- `lib:` required libraries
- `src:` source files
- `example:` examples as like configuration files and requests
- `schema:` additional xml schema descriptions, like sosConfiguration.xsd

3.3 Configuration

The basic configuration is suited for most use cases, however the configuration options in this section offer possibilities to adapt the plugins' behaviour to specific needs of the user.

3.3.1 Configure UDIG-SOSPlugin.properties

The file UDIG-SOSPlugin.properties located in the user-directory offers basic configuration options. If the file is missing, or an option is not set in the configuration file, the default value will be used. In case of a missing file, changing one or more parameters will result in the creation of a new configuration file.

The configuration may also be done with in UDIG with the „Preferences-GUI“ (needs to be enabled, see 3.3.2).

Pattern: „property=value“

Table 3: Udig-SOSPlugin Properties

Property	Explanation
log4jpath	The path leading to the log4j configuration file. *Default*: ./Java/log4j.properties
preferedSOSVersion	If a SOS supports more than one „Service version“, this value allows a selection which version should be prefered. *Possible values*: 0.0.0, 1.0.0 *Default*: 1.0.0
sosConfigurationFilename	Location of the sosConfiguration-File, which is described in 3.3.3 . *Default:* „sosConfiguration.xml“

Property	Explanation
defaultTimeToCacheCapabilities	This value (in *ms*) allows to set how long the capablities of a single SOS will be cached. A value of 0 means no caching. *Default*: 30000
defaultTimeToCacheDatastore	This value (in *ms*) allows to set how long a Datastore of a single SOS will be cached. A value of 0 means no caching. *Default*: 30000
fixErrors	Setting this option to true, the plugin is allowed to modify request as a workaround some known issus with current SOS implementations. Possible values: true, false *Default*: true
proxyHost	If a proxy is used for connecting to the network, this option defines the hostname of the used proxyserver. Default: “”
proxyPort	If a proxy is used for connecting to the network, this option defines the port of the used proxyserver. Default: “”
seismicContainsProcedure	This String is used to identify seismic observations . The value of the xml-tag “procedure” is checked with .contains(s) for the value of this option. Default: “seismology”
simulationContainsProcedure	This String is used to identify simulation observations . The value of the xml-tag “procedure” is checked with .contains(s) for the value of this option. Default: “ScenarioSelection”

3.3.1.1 Example:

The file „examples/UDIG-SOSPlugin_properties.txt“ offers an example which sets the path of the sos configuration file to „c:\config\sosconf.xml“, the time to cache capabilities to 40 seconds and the fixErrors option to false.

3.3.2 Configure plugin.xml

The file plugin.xml in the plugin's rootdirectory or the jar offers the option to enable and disable features of the plugin using extension points. If using the jar, you'll need to extract the plugin.xml from the .jar with an compression tool which is capable of zip-archives like 7-zip. Then change the file and put it back into the jar.

For additional information regarding „Extension Points“ refer to Eclipse documentation about the „Pde Schema Editor“.

- The basic extension, do not remove!

```
<extension
      point="net.refractions.udig.catalog.ServiceExtension">
   <service
      class="org.n52.udig.catalog.internal.sos.SOSServiceExtension"
      id="sos"
      name="SensorObservationService">
   </service>
</extension>
```

- This extension enables three sequential wizardpages (“Select SOS“ → “Select operation“ → “Select parameters“) that represent the catalog-GUI.

  Without this extension the GUI is not available to create connections to Sensor Observation Services.

```
<extension
      point="net.refractions.udig.catalog.ui.connectionFactory">
   <factory
      class="org.n52.udig.catalog.internal.sos.ui.SOSConnectionFactory"
      id="org.n52.udig.catalog.internal.sos.ui">
   </factory>
   <wizardPage
      class="org.n52.udig.catalog.internal.sos.ui.SOSWizardPage"
      description="%wizard.description"
      name="SensorObservationService">
   </wizardPage>
   <wizardPage
      class="org.n52.udig.catalog.internal.sos.ui.SOSSelectOperationPage"
      description="%wizard.description"
      name="SelectOperation">
   </wizardPage>
   <wizardPage
      class="org.n52.udig.catalog.internal.sos.ui.SOSParameterConfigurationPage"
      description="&amp;quot;left: select your parameters; right select parameter values; next button is
disabled untli all parameters are configured&amp;quot;"
      name="ConfigureParameters">
   </wizardPage>
</extension>
```

- This extension enables the preference windows inside uDig, allowing the configuration of UDIG-SOSPlugin.properties (3.3.1) and SOSconfiguration.xml (3.3.3). The pages can be found in uDig under „Window → Preferences → Catalog → SOS“

```
<extension point="org.eclipse.ui.preferencePages">
   <page category="net.refractions.udig.catalog.ui.preferences.CatalogPreferencePage"
      class="org.n52.udig.catalog.internal.sos.ui.SOSPreferencePage"
      id="org.n52.udig.catalog.internal.sos.ui.SOSPreferencePage"
      name="SOS">
   </page>
   <page
      category="org.n52.udig.catalog.internal.sos.ui.SOSPreferencePage"
      class="org.n52.udig.catalog.internal.sos.ui.SOSPreferencePage1"
      id="org.n52.udig.catalog.internal.sos.ui.SOSPreferencePage1"
      name="SOS Connection Properties">
   </page>
</extension>
```

3.3.2.1 Example

The file „/examples/plugin.xml“ offers an example which enables all extensions described in this section.

3.3.3 SOSConfiguration.xml

Configuration files following the xml-schema “/schema/SOSConfiguration.xsd” can be used to preconfigure SOS-Requests and reduce the configuration effort for single request.

For example a GetObservationById request can be preconfigured so the user only has to select a proper ObservationId.

Configuration may be done:

- using the preference pages (if enabled, see 3.2.2)
- manualyl by creating an applicatable xml, refer to the examples and the comments in the xsd.
- using the XML-Beans classes directly in Java (org.x52N.schema.xmlConfigSchema.UDigSOSPluginDocument)

3.3.3.1 Example:

The file „/examples/SOSConfiguration.xml“ describes an example that configures requests for „http://mars.uni-muenster.de:8080/OWS5SOS/sos“.

The operation „DescribeSensor“ will not be shown in the GUI. In requests the parameter „service“ for the operation „GetObservationById“ is set to „SOS“ and the parameter „outputFormat“ will not be send to the SensorObservationService.

4 Troubleshooting

If you have any question concerning the installation process, feel free to contact the SWE mailing list. Information can be found on http://52north.org.

5 Appendix

5.1 How to access a SOS programaticly

The file „examples/ExampleAccessSOSProgramaticly.java“ offers an example which explains how to create a GetObservation request using the SOSConfiguration.xml described in 3.3.3, send it to a SOS and how to access the features generated from the SOSresponse.

5.2 Known issues

- Requests containing time instants or time periods with an accuracy higher than seconds are currently not available as this is not supported by the current version of the 52N SOS.
- 3-dimensional Points are stored as 2-D Points with an additional height attribute. To get the 3rd dimension you`ll have to work directly with the coordinates from the geometry .getCoordinate()[3]
- ResultType: ResultTemplate not supported
- ResultType application/zip not supported
- Layers and Servercapabilities do not contain contact informations from the service`s capabilities

5.3 References

- Priest, M. & Na, A. (2007): Sensor Observation Service, Implementation Specification, Version 0.1.5, OGC document 06-009r4
- Cox, S. (2007): Observations and Measurements (O&M) Implementation Specification, Version 1.0, OGC document 07-022r1