How to install the 52n SAS

Version 2.0-SNAPSHOT

Jan Torben Heuer <jan.heuer@uni-muenster.de>

Requirements

1. Tomcat 5.5 from http://tomcat.apache.org/download-55.cgi
2. The Java Standard Edition 5 or better. http://java.sun.com/javase/downloads/index_jdk5.jsp

Optional components

1. A xmpp server implementation, for example: http://www.igniterealtime.org/projects/openfire/index.jsp
2. A WNS server implementation. This can be the 52North implementation: http://52north.org/index.php?option=com_projects&task=showProject&id=5&Itemid=127

Installation

The following steps describe the installation of the binary. If you want to build the SAS from sources, please refer to the section “Manual build”

Make sure, that you have installed the Tomcat 5.5 and the server is **not** running.

1. Put the 52N SAS war-file into the TOMCAT_HOME/webapps directory and (re-) start the Tomcat.
2. The Tomcat should now extract the war-file and create a sas-communication-http directory.
3. From that directory, open the file WEB-INF/config.txt and change the properties. Comments start with “#”. Properties starting with “Module.” define lava-classes. Most important is the location of the Database file, which is per default in the current directory. It is highly recommended to set an absolute path (e.g. C:/tmp/sas_db4o.db – always use “/” not “\”; folders must exist)! Please verify the bold-printed entries:

```
# SAS - Sensor Alert Service
# Copyright 2007 52North Initiative for Geospatial Open Source Software GmbH
#
============================================================
=============
# automatically generated configfile
# date: Fri Jan 11 12:54:38 CET 2008
# sas version: 2.0 $Revision: 362 $
```

```
## The database client.
Module.DAO=org.n52.swe.sas.dao.db4o.Db4oConnector
Db4o.DatabaseFileLocation=sas_db4o.db

## The event stream processing engine.
Module.Event=org.n52.swe.sas.event.esper.EsperEventHandler

## The messaging communicator, i.e. an xmpp client.
Module.MessagingCommunicator=org.n52.swe.sas.mock.MockMessagingCommunicator

## The WNS client. Select the class here if you want to support
## WNS capabilities (requires a WNS server).
## values: [ NULL org.n52.swe.sas.communication.wns.WNSCommunicator ]
## default: NULL
Module.WNS=org.n52.swe.sas.communication.wns.WNSCommunicator
## Configuration for WNS-Support
##
## The url of the used WNS
WNS_URL=
## the url of your sas for global access
OWN_SAS_URL=
## Short message text that is sended to e.g. an SMS number
WNS_SHORT_TEXT=

## The SOS Adapter. If you want to use the SOS Adapter then
## define its class here.
## values: [ NULL org.n52.swe.sas.sosadapter.SOSConnector ]
## default: NULL
Module.SOSConnector=org.n52.swe.sas.sosadapter.SOSConnector

## Should all XML Documents be validated? (Recommended)
## values: [ TRUE FALSE ]
## default: TRUE
VALIDATE_XML_PROPERTY=TRUE
```

4. Restart the Tomcat or only the SAS from the Tomcat Management Interface (TOMCAT_URL/manager/)
5. Go to TOMCAT_URL/sas/. You should see a welcome HTML page.
6.

Manual Build

1. Install a subversion client
2. Get the SAS sources from https://incubator-52n.svn.sourceforge.net/svnroot/incubator-52n/swe/sas/trunk/. with your preferred subversion client.
3. Make shure you have Maven installed. You can get it from http://maven.apache.org
4. Add the 52North maven repository to the settings xml. Below is a sample file. Up-to-date maven information can be found in the 52North maven wiki: https://52north.org/twiki/bin/view/Documentation/MavenHome .

```
TWiki.TwistyPlugin.init("twistyIdDocumentationMavenHome1hide");TWiki.TwistyPlugin.
init("twistyIdDocumentationMavenHome1toggle");<?xml version="1.0"?>
<settings>
 <profiles>

  <profile>
   <id>52n-start</id>
   <repositories>
      <repository>
         <id>n52-releases</id>
         <name>52n Releases</name>
         <url>http://incubator.52north.org/maven/maven-repo/releases/</url>
         <releases>
            <enabled>true</enabled>
         </releases>
         <snapshots>
            <enabled>false</enabled>
         </snapshots>
      </repository>

      <repository>
         <id>n52-releases</id>
         <name>52n Snapshots</name>
         <url>http://incubator.52north.org/maven/maven-repo/snapshots/</url>
         <releases>
            <enabled>false</enabled>
         </releases>
         <snapshots>
            <enabled>true</enabled>
         </snapshots>
      </repository>


      <repository>
         <id>apache</id>
         <name>apache Releases</name>
         <url>http://people.apache.org/repo/m2-snapshot-repository/</url>
         <releases>
            <enabled>false</enabled>
         </releases>
         <snapshots>
            <enabled>true</enabled>
         </snapshots>
      </repository>

   </repositories>
  </profile>
 </profiles>
 <activeProfiles>
   <activeProfile>52n-start</activeProfile>
 </activeProfiles>
```

5. Open a terminal window and navigate to the directory of the sources.
6. Run “mvn compile”. Maven downloads required libraries automatically. This step may take a long time.
7. The final war file will be created at sas-communication-http/target/sas-communication-http-2.0-SNAPSHOT.war.

SAS Operations

The following section describes operations that have been changed or extended. All SAS operations can be executed with a web interface. Browse to your SAS url to get an

overview. (`For example: http://host:8080/sas/`)

SOSAdvertise

This operation registers a sensor of a sos. You have to change the bold-printed parameters below:

```
<?xml version="1.0" encoding="UTF-8"?>
<AdvertiseSOS xmlns="http://www.opengis.net/sas/0.0"
xmlns:sml="http://www.opengis.net/sensorML/1.0" xmlns:swe="http://www.opengis.net/swe/1.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas/0.0
../sasAdvertise.xsd" service="SAS" version="1.0.0">
  <SOSInformation>
    <SOSURL>http://mars.uni-muenster.de:8080/52nSOSv2/sos</SOSURL>
    <offering>wv.offering-2w</offering>
    <procedure>wv.sensor-15@22</procedure>
    <observedProperty>wv.phenomenon-10</observedProperty>
  </SOSInformation>
  <RetrievalFrequency>PT2S</RetrievalFrequency>
</AdvertiseSOS>
```

You can obtain them from the SOS getCapabilites request
You can try `http://sosurl/sos?request=GetCapabilites&service=SOS`

The response is a normal AdvertiseResponse. Keep the sensor id, because you need it to unregister the sensor, **add the sensor to the client** or subscribe manually. However you can retrieve the id later with the sas's getCapabilites operation.

Subscribe

This operation registers a user for different filter criteria. The following example shows a subscribe with SensorID, Location and ValueFilter set. Those three elements (here: bold italic) are optional. If you do not set one of it, all incoming data will match your subscription. The SensorID, in difference to the current OGC Schema, is now located inside the EventFilter.

The attribute “alertOnChange” of the element ValueFilterList is also optional and is not yet added in the OGC Schema. If set to true, incoming alerts will only match the subscription when a change from (value) filter match to no match (and vice versa) has been detected. In such a case, the alert which inflicted the changing condition is sent to the subscriber (e.g., when a subscriber is interested in alerts from sensor A that have a temperature value greater than 40 °C, then the sequence of values sent by sensor A - 50, 45, 41, 35, 40, 45 - would generate the following sequence of alerts which are sent to the subscriber: 35, 45).

```
<?xml version="1.0" encoding="UTF-8"?>
<Subscribe xmlns="http://www.opengis.net/sas/0.0"
xmlns:swe="http://www.opengis.net/swe/1.0" xmlns:wns="http://www.opengis.net/wns/0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sas/0.0 ../sasSubscribe.xsd" service="SAS"
version="1.0.0">
        <EventFilter>
```

```
            <SensorID>2</SensorID>

            <Location>
            <swe:Position>
             <swe:location>
              <swe:Vector>
                    <swe:coordinate name="lat">
                     <swe:Quantity>
                            <swe:value>
                             52.0
                            </swe:value>
                     </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="lon">
                     <swe:Quantity>
                            <swe:value>
                             12.0
                            </swe:value>
                     </swe:Quantity>
                    </swe:coordinate>
              </swe:Vector>
             </swe:location>
            </swe:Position>
            </Location>

            <ValueFilterList alertOnChange="true">
              <member>
                    <ValueFilter definition="urn:x-ogc:def:phenomenon:OGC:Temperature">
                     <filterCriteria>
                            <isGreaterThanOrEqualTo>20.4</isGreaterThanOrEqualTo>
                     </filterCriteria>
                     <uom code="%"/>
                    </ValueFilter>
              </member>
            </ValueFilterList>

        </EventFilter>

        <ResultRecipient>
          <wns:NotificationTarget>
                <wns:NotificationChannel>
                    <wns:Email>test@server.com</wns:Email>
                </wns:NotificationChannel>
                <wns:NotificationFormat>basic</wns:NotificationFormat>
          </wns:NotificationTarget>
        </ResultRecipient>
</Subscribe>
```

The ResultRecipient defines a communication-endpoint where matching data should be sent to. This can be a WNS NotificationChannel, like in this example, or an XMPPURI (e.g. a normal Jabber-ID or MultiUserChat). If the NotificationChannel is set the SAS will register the user with the given data at the WNS it is using.

Bugs and Feedback

Please report bugs with bugzilla: https://52north.org/bugzilla/ .

The account is your twiki account https://52north.org/twiki/bin/view/Main/WebHome which can be created here: https://52north.org/twiki/bin/config/register.pl .

Please join our sas-user list at https://52north.org/mailman/listinfo/sas-user .

(Modified) Schema

sasAdvertise.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:sas="http://www.opengis.net/sas/0.0"
xmlns:swe="http://www.opengis.net/swe/1.0"
xmlns:sml="http://www.opengis.net/sensorML/1.0"
xmlns:ogc="http://www.opengis.net/ogc"
targetNamespace="http://www.opengis.net/sas/0.0" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="0.0.0" xml:lang="en">
 <!-- ============================================================
      includes and imports
      ============================================================ -->
 <xs:include schemaLocation="./sasCommon.xsd"/>
 <xs:import namespace="http://www.opengis.net/ogc"
schemaLocation="http://schemas.opengis.net/filter/1.1.0/filter.xsd"/>
 <!-- ============================================================

      ============================================================ -->
 <xs:element name="Advertise">
   <xs:annotation>
     <xs:documentation>Request to a SAS to allow a publisher to publish
alerts.</xs:documentation>
   </xs:annotation>
   <xs:complexType>
     <xs:complexContent>
       <xs:extension base="sas:RequestBaseType">
         <xs:sequence>
           <xs:element ref="sas:messageStructure"/>
           <xs:element ref="sas:sensorDescription"/>
           <xs:element ref="sas:reportingFrequency" minOccurs="0"/>
           <xs:element ref="sas:desiredPublicationExpiration" minOccurs="0"/>
           <xs:element ref="sas:Location" minOccurs="0"/>
           <xs:element ref="sas:XMPPCredentials" minOccurs="0"/>
           <xs:element ref="sas:ReliableCommunication" minOccurs="0"/>
         </xs:sequence>
       </xs:extension>
     </xs:complexContent>
   </xs:complexType>
 </xs:element>
 <xs:element name="AdvertiseResponse">
   <xs:complexType>
     <xs:sequence>
       <xs:element ref="sas:SensorID">
         <xs:annotation>
           <xs:documentation>unique ID, submitted in response to an advertise-
```

```
request. Is used in cancel- or renewAdvertisement requests.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="sas:AlertChannel"/>
        <xs:element ref="sas:XMPPCredentials" minOccurs="0"/>
        <xs:element ref="sas:AcknowledgementChannel" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="expires" type="xs:dateTime" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="AdvertiseSOS">
    <xs:annotation>
      <xs:documentation>Create a new Advertisement with a virutal sensor that
takes the data out of an SOS. The information provided is needed to specify
which data should be retrieved by the SAS when accessing the
SOS.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="sas:RequestBaseType">
          <xs:sequence>
            <xs:element name="SOSInformation">
              <xs:annotation>
                <xs:documentation>Contains information required to identify the
SOS from which to retrieve data as well as the SOS offering, procedure and
observed properties.</xs:documentation>
              </xs:annotation>
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="SOSURL" type="xs:anyURI">
                    <xs:annotation>
                      <xs:documentation>URL of the SOS to retrieve data
from.</xs:documentation>
                    </xs:annotation>
                  </xs:element>
                  <xs:element name="offering" type="xs:anyURI">
                    <xs:annotation>
                      <xs:documentation>ID of an offering advertised in the
capabilities of the referenced SOS.</xs:documentation>
                    </xs:annotation>
                  </xs:element>
                  <xs:element name="procedure" type="xs:anyURI">
                    <xs:annotation>
                      <xs:documentation>Identifies one sensor / procedure that
belongs to the given offering. Data shall be retrieved for exactly this
procedure.</xs:documentation>
                    </xs:annotation>
                  </xs:element>
                  <xs:element name="observedProperty" type="xs:anyURI"
maxOccurs="unbounded">
                    <xs:annotation>
                      <xs:documentation>ID of a phenomenon which shall be
retrieved by the SAS. It is advertised in the capabilities document of the
referenced SOS.</xs:documentation>
                    </xs:annotation>
                  </xs:element>
                  <xs:element name="featureOfInterest" minOccurs="0">
                    <xs:annotation>
                      <xs:documentation>Provides functionality as the
featureOfInterest element in the sos:GetObservation request.
may be used to restrict data retrieval through SAS to only get data from SOS for
certain stations (by providing their IDs) or regions (by providing a
BBox).</xs:documentation>
                    </xs:annotation>
```

```
                    <xs:complexType>
                      <xs:choice>
                        <xs:element ref="ogc:spatialOps"/>
                        <xs:element name="ObjectID" type="xs:anyURI"/>
                      </xs:choice>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="RetrievalFrequency" type="xs:duration">
              <xs:annotation>
                <xs:documentation>Provide the duration of the interval (i.e., the
frequency) with which to retrieve data from the SOS. E.g., a value of PT2S would
retrieve data every two seconds.</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="AdvertiseSOSResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="sas:SensorID"/>
      </xs:sequence>
      <xs:attribute name="expires" type="xs:dateTime" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="CancelAdvertisement">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="sas:RequestBaseType">
          <xs:sequence>
            <xs:element ref="sas:SensorID"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="CancelAdvertisementResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="sas:SensorID"/>
        <xs:element name="CancellationStatus">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="confirmed"/>
              <xs:enumeration value="expired"/>
              <xs:enumeration value="invalidSensorID"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="RenewAdvertisement">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="sas:RequestBaseType">
          <xs:sequence>
            <xs:element ref="sas:SensorID"/>
            <xs:element ref="sas:desiredPublicationExpiration" minOccurs="0"/>
```

```
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
 </xs:element>
 <xs:element name="RenewAdvertisementResponse">
   <xs:complexType>
     <xs:sequence>
       <xs:element ref="sas:SensorID"/>
       <xs:element name="renewalStatus">
         <xs:annotation>
           <xs:documentation>Indicator shows if renewal request was confirmed or
rejected.</xs:documentation>
         </xs:annotation>
         <xs:simpleType>
           <xs:restriction base="xs:string">
             <xs:enumeration value="confirmed"/>
             <xs:enumeration value="rejected"/>
           </xs:restriction>
         </xs:simpleType>
       </xs:element>
     </xs:sequence>
     <xs:attribute name="expires" type="xs:dateTime" use="required"/>
   </xs:complexType>
 </xs:element>
</xs:schema>
```

sasSubscribe.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:sas="http://www.opengis.net/sas/0.0"
xmlns:wns="http://www.opengis.net/wns/0.0"
xmlns:swe="http://www.opengis.net/swe/1.0"
targetNamespace="http://www.opengis.net/sas/0.0" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="0.0.0" xml:lang="en">
 <!-- ================================================================
      and imports
      ================================================================ -->
 <xs:import namespace="http://www.opengis.net/wns/0.0"
schemaLocation="../../wns/0.0.0/wnsShared.xsd"/>
 <xs:import namespace="http://www.opengis.net/swe/1.0"
schemaLocation="../../sweCommon/1.0.0/swe.xsd"/>
 <xs:include schemaLocation="./sasCommon.xsd"/>
 <!-- ================================================================

      ================================================================ -->
 <xs:element name="Subscribe">
   <xs:annotation>
     <xs:documentation>Request to a SAS to subscribe to
alerts.</xs:documentation>
   </xs:annotation>
   <xs:complexType>
     <xs:complexContent>
       <xs:extension base="sas:RequestBaseType">
         <xs:sequence>
           <xs:element name="EventFilter" minOccurs="0">
             <xs:annotation>
               <xs:documentation>The client may choose to subscribe for all
events of a single sensor (in which case he should be referred directly to the
```

```
MUC where the sensor publishes its data) or he can specify in which events he is
interested in.</xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:annotation>
                  <xs:documentation>There is an implicit AND between the
conditions defined by the following elements.</xs:documentation>
                </xs:annotation>
                <xs:element ref="sas:SensorID" minOccurs="0"/>
                <xs:element ref="sas:SubscriptionOfferingID" minOccurs="0"/>
                <xs:element ref="sas:Location" minOccurs="0"/>
                <xs:element name="ValueFilterList" minOccurs="0">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="member" maxOccurs="unbounded">
                        <xs:complexType>
                          <xs:sequence>
                            <xs:element name="ValueFilter">
                              <xs:annotation>
                                <xs:documentation>Used to indicate in which
observed properties (indicated by the definition attribute) the user is really
interested. Also allows the definition of simple value filters and provision of
uom.</xs:documentation>
                              </xs:annotation>
                              <xs:complexType>
                                <xs:sequence>
                                  <xs:element name="filterCriteria"
minOccurs="0">
                                    <xs:complexType>
                                      <xs:choice>
                                        <xs:element name="isLessThan"
type="xs:double"/>
                                        <xs:element
name="isGreaterThanOrEqualTo" type="xs:double"/>
                                        <xs:element name="isGreaterThan"
type="xs:double"/>
                                        <xs:element name="isLessThanOrEqualTo"
type="xs:double"/>
                                        <xs:element name="isEqual">
                                          <xs:simpleType>
                                            <xs:union memberTypes="xs:string
xs:double xs:boolean"/>
                                          </xs:simpleType>
                                        </xs:element>
                                        <xs:element name="isNotEqualTo">
                                          <xs:simpleType>
                                            <xs:union memberTypes="xs:string
xs:double"/>
                                          </xs:simpleType>
                                        </xs:element>
                                        <xs:element name="isBetween">
                                          <xs:complexType>
                                            <xs:sequence>
                                              <xs:element name="lowerBoundary"
type="xs:double"/>
                                              <xs:element name="upperBoundary"
type="xs:double"/>
                                            </xs:sequence>
                                          </xs:complexType>
                                        </xs:element>
                                      </xs:choice>
                                    </xs:complexType>
                                  </xs:element>
```

```
                                    <xs:element name="uom"
type="swe:UomPropertyType" minOccurs="0">
                                      <xs:annotation>
                                        <xs:documentation>Used to indicate which
uom the value provided in the filterCriteria has, so that the service can
transform it to the uom of the actual event property (or vice versa). This only
makes sense if the definition provided in the ResultFilter references a
swe:Quantity property (swe:Count, swe:Boolean, swe:Category and swe:Text do not
have a uom - swe:Time has, but SAS does not perform temporal filtering right
now).</xs:documentation>
                                      </xs:annotation>
                                    </xs:element>
                                  </xs:sequence>
                                  <xs:attribute name="definition"
type="xs:anyURI" use="required">
                                    <xs:annotation>
                                      <xs:documentation>Points to semantics
information defining the precise nature of the component</xs:documentation>
                                    </xs:annotation>
                                  </xs:attribute>
                                </xs:complexType>
                              </xs:element>
                            </xs:sequence>
                          </xs:complexType>
                        </xs:element>
                      </xs:sequence>
                      <xs:attribute name="alertOnChange" type="xs:boolean"
default="false">
                        <xs:annotation>
                          <xs:documentation>If set to true, incoming alerts will
only match the subscription when a change from (value) filter match to no match
(and vice versa) has been detected. In such a case, the alert which inflicted
the changing condition is sent to the subscriber.
.g., when a subscriber is interested in alerts from sensor A that have a
temperature value greater than 40 °C, then the sequence of values sent by sensor
A - 50, 45, 41, 35, 40, 45 - would generate the following sequence of alerts
which are sent to the subscriber: 35, 45.</xs:documentation>
                        </xs:annotation>
                      </xs:attribute>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="ResultRecipient" minOccurs="0">
              <xs:annotation>
                <xs:documentation>Allows to specify where to the SAS should send
the data . If this element is used, the SAS will not open a MUC and provide the
data to the client, but will forward any alert to the specified remote MUC or
uses the WNS (in which case SASMessages will be sent to the specified
target(s)). This allows to use the SAS in "last-mile-mode".</xs:documentation>
              </xs:annotation>
              <xs:complexType>
                <xs:choice>
                  <xs:element name="XMPPURI" type="xs:anyURI">
                    <xs:annotation>
                      <xs:documentation>The excplicit URI where the xmpp messages
should be send to. Also implies that simple Alerts, not SASMessages are
published on this MUC.</xs:documentation>
                    </xs:annotation>
                  </xs:element>
                  <xs:element ref="wns:NotificationTarget"/>
                </xs:choice>
              </xs:complexType>
```

```
            </xs:element>
            <xs:element ref="sas:ReliableCommunication" minOccurs="0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="SubscribeResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="sas:AcknowledgementChannel" minOccurs="0"/>
        <xs:element ref="sas:AlertChannel" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="SubscriptionID" type="xs:token" use="required">
        <xs:annotation>
          <xs:documentation>ID of the new subscription.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="expires" type="xs:dateTime" use="required">
        <xs:annotation>
          <xs:documentation>Point in time when the subscription will
expire.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="CancelSubscription">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="sas:RequestBaseType">
          <xs:sequence>
            <xs:element name="SubscriptionID" type="xs:token"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="CancelSubscriptionResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="SubscriptionID" type="xs:token"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="RenewSubscription">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="sas:RequestBaseType">
          <xs:sequence>
            <xs:element name="SubscriptionID" type="xs:token"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:element name="RenewSubscriptionResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="SubscriptionID" type="xs:token"/>
      </xs:sequence>
      <xs:attribute name="expires" type="xs:dateTime" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```