



Open architecture for Smart and Interoperable networks
In Risk management based on In-situ Sensors

www.osiris-fp6.eu

Integrated Project - Contract Number 033475

WORKPACKAGE 5400 Deliverable 5400

Home Weather Station Report

Issue 01

Prepared for the:

COMMISSION OF THE EUROPEAN COMMUNITIES

INFORMATION SOCIETY AND MEDIA DIRECTORATE-GENERAL



THALES



Document prepared by
Westfälische Wilhelms-Universität Münster (Institute for Geoinformatics)

Reviewed and Approved by	Representative
THALES Communications Westfälische Wilhelms-Universität Münster Vlaamse Instelling voor Technologisch Onderzoek N.V	Danielle TACYNIAK Simon JIRKA Nicolas LEWYCKYJ

ISSUE	DATE	OBJECT
Issue 01	29/02/08	Report for submission to EC

Copyright

© Copyright 2008 The OSIRIS Consortium

Consisting of:

- THALES Communications SA
- Westfälische Wilhelms-Universität Münster (Institute for Geoinformatics)
- APS GmbH
- Vlaamse Instelling voor Technologisch Onderzoek N.V.
- GMV Aerospace and Defense SA
- Thales Research & Technology (UK) Ltd
- Fondazione per il Clima e la Sostenibilita -Laboratorio per la Meteorologia e la Modellistica Ambientale
- Hydrogeotechnika SP.ZOO
- ESYS PLC
- Réseau Euro-Méditerranéen d'Information et de FORmation à la gestion des risques
- Stadt Aachen (Feuerwehr Aachen)
- Autobuses Urbanos de Valladolid S.A
- Regione Toscana

This document may not be copied, reproduced, or modified in whole or in part for any purpose without written permission from the OSIRIS Consortium. In addition to such written permission to copy, reproduce or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

Table of contents

<u>1. INTRODUCTION</u>	6
1.1 GENERAL	6
1.2 DOCUMENT STRUCTURE	6
1.3 GLOSSARY OF TERMS AND ABBREVIATIONS	7
1.4 APPLICABLE DOCUMENTS AND REFERENCES	7
<u>2. SCOPE OF WP5400: IN-SITU SENSORS AT HOME</u>	8
2.1 SCOPE OF WP5400	8
2.2 WP5400 RELATIONSHIP WITH OTHER WP	8
<u>3. PROCESS OVERVIEW</u>	9
<u>4. HOME WEATHER STATION STATE-OF-THE-ART REPORT</u>	10
4.1 GENERAL DESIGN OF HOME WEATHER STATIONS	10
4.2 OVERVIEW OF VARIOUS HOME WEATHER STATIONS	10
4.3 OVERVIEW OF AVAILABLE SOFTWARE	14
4.4 SELECTION OF AN ADEQUATE STATION	15
<u>5. HOME WEATHER STATION BRIDGE SPECIFICATION AND DESIGN</u>	19
5.1 INTRODUCTION	19
5.2 GLOBAL DESIGN	20
5.3 INTERFACE SPECIFICATIONS	21
<u>6. SUMMARY AND CONCLUSIONS</u>	31
6.1 KEY ISSUES ADDRESSED	31
6.2 LESSONS IDENTIFIED	31
<u>APPENDIX.A. HOME WEATHER STATION BRIDGE DEMONSTRATOR REPORT</u>	32
A.1 AIMS OF THE HOME WEATHER STATION BRIDGE DEMONSTRATOR.....	32
A.2 DEVELOPMENT	33
A.3 DEPLOYMENT AND INTEGRATION INTO THE OSIRIS WEB PAGE.....	34
<u>APPENDIX.B. XML-SCHEMATA</u>	36
B.1 REGISTERSENSOR.XSD	36
B.2 REGISTERSENSORRESPONSE.XSD.....	36
B.3 PUSHDATA.XSD	37
B.4 COMMON.XSD	37
B.5 SWE4HWSBRIDGE.XSD	39
<u>APPENDIX.C. XML-EXAMPLES</u>	40
<u>APPENDIX.D. REFERENCES</u>	46

List of figures

Figure 4-1: WS2500 base station.....	17
Figure 4-2: WS2500 rain sensor	17
Figure 4-3: WS2500 wind sensor	18
Figure 5-1: Use of interfaces and protocols between the components	19
Figure 5-2: General structure of the RegisterSensor element	23
Figure 5-3: The StationMetadata element.....	29
Figure 5-4: The RegisterSensorResponse element	30
Figure A-1: User Interface of the Station Component.....	34

List of tables

<i>Table 4-1: Summarised overview of various weather stations.....</i>	<i>11</i>
<i>Table 4-2: Summarised overview of the features of the home weather stations.....</i>	<i>13</i>
<i>Table 4-3: Overview of available software.....</i>	<i>15</i>
<i>Table 5-1: SAS operations.....</i>	<i>21</i>
<i>Table 5-2: SWE Common data types for the message structure element</i>	<i>24</i>
<i>Table 5-3: SWE Common data types for the location element</i>	<i>25</i>
<i>Table 5-4: SensorML data types for the SensorSystem element.....</i>	<i>28</i>
<i>Table 5-5: Elements used in the StationMetadata element</i>	<i>29</i>
<i>Table 5-6: Elements of RegisterSensorResponse.....</i>	<i>30</i>

1. INTRODUCTION

1.1 GENERAL

This document produced under the OSIRIS project (Contract number 033475) constitutes the deliverable D5400.

1.2 DOCUMENT STRUCTURE

This paragraph gives an overview of the document structure and the relationship between the different chapters.

Chapter 1 is a general introduction.

Chapter 2 presents the Scope of Work for the WP5400 and its relationship with other work packages within OSIRIS.

Chapter 3 presents the process that was used to achieve the stated objectives for WP5400.

Chapter 4 describes the State of the Art of Home Weather Stations.

Chapter 5 describes the Home Weather Station Bridge Specification and Design.

The document body finishes with:

Chapter 6 which summarises the document and concludes the findings.

In the appendixes the following information will be found:

Appendix A – Home Weather Station Bridge Demonstrator report

Appendix B – XML-SCHEMATA

Appendix C – XML EXAMPLES

Appendix D– References

1.3 GLOSSARY OF TERMS AND ABBREVIATIONS

Term	Meaning
EPSG	European Petroleum Survey Group
FP6	6 th Framework Program
IETF	The Internet Engineering Task Force
Observed value	A value describing a natural phenomenon, which may use one of a variety of scales including nominal, ordinal, ratio and interval. The term is used regardless of whether the value is due to an instrumental observation, a subjective assignment or some other method of estimation or assignment (see [OGC2006])
OGC	Open Geospatial Consortium, Inc.
OSIRIS	Open architecture for Smart and Interoperable networks In Risk management based on In-situ Sensors
SAS	Sensor Alert Service
SensorML	Sensor Model Language
SOS	Sensor Observation Service
SWE	Sensor Web Enablement
TCF	THALES Communications SA
URN	Uniform Resource Name
USB	Universal Serial Bus
VITO	Vlaamse Instelling voor Technologisch Onderzoek n.v. (Flemish Institute for Technological Research)
WWU	Westfälische Wilhelms-Universität Münster

1.4 APPLICABLE DOCUMENTS AND REFERENCES

Document	Title
Annex 1 of the OSIRIS contract	Description of work

2. SCOPE OF WP5400: IN-SITU SENSORS AT HOME

2.1 SCOPE OF WP5400

The scope of WP5400 is the development of a home weather station bridge demonstrator which illustrates basic principles of the OSIRIS architecture.

In order to achieve this goal first a state-of-the-art analysis on home weather stations had to be performed. This was the basis for the selection of the home weather station type that was used during the further development.

The second important step was the design of the interfaces of the home weather station system. This includes protocols for registering weather stations on a central server, transmitting data to this server and making the data available to all interested clients. These interfaces were based on OGC standards like SAS, SOS and SensorML which also form a foundation of the OSIRIS architecture.

After the specification of the interfaces a client component that is running on those computers to which the home weather stations are connected was developed. This client component is used for registering the weather stations and submitting the data to a central server. Furthermore also the server component which manages the weather stations and their data was produced.

2.2 WP5400 RELATIONSHIP WITH OTHER WP

There is a close relationship to WP 8000 dealing with dissemination of the OSIRIS results. This is because in WP 8110 the integration of the home weather station system into a web site will be performed. The most important aspect of this is the production of a client component that allows users to access and visualise the weather data.

Furthermore there is a global connection to WP 6000 defining the OSIRIS global sensor web architecture as basic concepts of the OSIRIS architecture were used for developing the home weather station bridge demonstrator.

3. PROCESS OVERVIEW

As defined in the DoW (Description of Work) the work in this WP was performed by WWU which was responsible also for writing this report.

Comments and suggestions to this WP were contributed by the partners TCF and VITO.

	WWU
Home Weather Station State-of-the-art report	X
Home Weather Station Bridge Specification and Design	X
Home Weather Station Bridge Demonstrator report	X

The results were exposed to the partners during the different common meetings.

4. HOME WEATHER STATION STATE-OF-THE-ART REPORT

This section describes the state-of-the-art weather stations. Since the weather stations should be integrated into the in-situ network only the weather stations which can be connected to a PC are evaluated.

4.1 GENERAL DESIGN OF HOME WEATHER STATIONS

Most weather stations consist of a base station and one or more sensors. The base station is normally positioned inside a building and receives the data from the sensors. Its main purpose is to visualise the received data, but some base stations also contain an internal memory which is used to store records. The amount of records which can be stored varies from 200 to 4000 records. The connection to a computer is established via an USB or RS232 interface. Through this connection the computer has access to on-line or stored records.

The weather stations can be equipped with different kind of outdoor sensors. A common sensor combination consists of a wind direction sensor, a wind speed sensor, a temperature sensor, a humidity sensor and a rain collector. Additionally, some base stations contain internal sensors which measure indoor temperature, pressure and humidity. As the outdoor sensors are remote they communicate cable-based or wirelessly with the base station.

4.2 OVERVIEW OF VARIOUS HOME WEATHER STATIONS

In this part a detailed overview of various weather stations is given. The description includes general information about the base stations and the sensors as well as some technical specifications. The stations evaluated are taken from a selection of six manufacturers. The following table gives a first, summarised overview.

Table 4-1: Summarised overview of various weather stations

Manufacturer (Distributor)	Station	Interface	Price
Davis	Vantage Pro 2	USB	900 €
ELV	WS 2500	RS232	400 €
Hideki (Honeywell)	TE923W	USB	170 €
LaCrosse	Mega II + Data Logger	RS232	365 €
Oregon Scientific	WMR 100	USB	160 €
Peet Bros	Ultimeter 800	RS232	200 €

4.2.1 VANTAGE PRO 2

[Sources: <http://www.davisnet.com/weather/products/vantage2.asp>,
http://www.davisnet.com/product_documents/weather/spec_sheets/6152-62-53-63_VP2Spec_C.pdf , Davis 2006]

The Vantage Pro 2 consists of a base station and a sensor suite which includes a rain collector, a temperature sensor, a humidity sensor, an anemometer, a barometric pressure sensor and a wind direction sensor. The accuracy for the sensors is stated as follows: (+/- 5% accuracy) for the rain collector, (+/- 0.5°C accuracy) for the temperature sensor, (+/- 3% accuracy) for the humidity sensor, (+/- 5% accuracy) for the anemometer, (+/- 0.04 Hg accuracy) for the barometric pressure sensor and (+/- 4° accuracy) for the wind direction sensor. The measurements are sent wirelessly to the base station whereas the distance between base station and sensors shouldn't exceed 150 m. The base station visualises the measurements numerically or plots diagrams. Additionally, it contains a pressure sensor which measures the barometric pressure. Since the base station can't be connected to a PC directly, an additional logger is needed which is able to store up to 2560 records. The logger can be connected to a PC via the USB interface. The protocol for communication between the logger and the PC is available freely.

4.2.2 WS 2500

[Sources: <http://shop.elv.de/output/controller.aspx?cid=74&detail=10&detail2=9096>]

The WS 2500 is shipped with a substantial sensor suite. It contains a temperature sensor (+/- 0.5°C accuracy), a humidity sensor (+/- 3.5% accuracy), a rain collector (+/- 2% accuracy), a wind speed sensor (+/- 2% accuracy), a wind direction sensor (numerical resolution 5°), a barometric pressure sensor (+/- 1 hPa) and a luminosity sensor (+/- 10% accuracy). In addition to these outdoor sensors the base station contains a pressure sensor which has an accuracy of +/- 1 hPa. Communication between the base station and the outdoor sensors takes

place wirelessly whereas the maximal distance between the components is 100 m. The base station itself offers a touch-screen display which shows the current weather data. The included memory is able to store weather records up to 8 days. The communication protocol is available freely.

4.2.3 TE923W

[Sources: Honeywell 2006]

Hideki's TE923W weather station is shipped with a variety of different sensors. A speciality of this sensor suite is that it contains an UV-sensor which is able to measure ultraviolet light levels. In addition to that the suite includes a temperature sensor (1°C accuracy), a humidity sensor (5% accuracy), a rain collector (5% accuracy), a wind direction sensor (11.25° accuracy), a barometric pressure sensor (0.015 inHg accuracy) and a wind speed sensor (2 mph accuracy). All data measured by the remote sensors are transmitted to the base station wirelessly with the operating range up to 100 m in the open. The base station provides the usual functionality for visualising and plotting the received data. Furthermore, it is equipped with an internal memory which has a maximum capacity of 200 records. The protocol for the communication between the base station and the PC is not available.

4.2.4 MEGA II + DATA LOGGER

[Sources: La Crosse 2007]

The Mega II station is connected to the PC through an external data logger whereas the communication protocol between the logger and the PC is non-public. Depending on the number of sensors and the record interval, the data logger stores records continuously between 3 and 42 days without loss of data. The data logger is able to receive the sensor data autonomously which means that the base station is not needed inevitably to run the station. The sensor suite consists of a temperature sensor, a humidity sensor, a wind speed sensor, a wind direction sensor, a rain collector and a barometric pressure sensor. All outdoor sensors are equipped with solar cells and communicate wirelessly with the data logger and the base station. Their accuracy is stated as follows: (+/- 1°C accuracy) for the temperature sensor, (+/- 8 % accuracy) for the humidity sensor, (+/- 1 hPa accuracy) for the pressure sensor, (+/- 1 mm accuracy) for the rain collector and (+/- 1 km/h accuracy) for the wind speed sensor. There is no accuracy data available for the wind direction sensor.

4.2.5 WMR-100

[Sources: <http://www.oregonscientific.de/assets/manuals/WMR%20100.pdf>]

Besides the visualisation functionality, the base station of WMR-100 offers no storage of records, i.e. the station has to be connected to a PC all the time in order not to lose data. The

sensor suite consists of temperature sensor (+/- 1°C accuracy), a humidity sensor (+/- 7% accuracy), a wind speed sensor (+/- 3 m/s accuracy), a wind direction sensor, a rain collector (+/- 1 mm accuracy) and a barometric pressure sensor (+/- 10 mb/hPa accuracy). The sensors can be installed up to 100 m from the base station whereas the communication between the base station and the sensors takes place wirelessly. The communication protocol is non-public.

4.2.6 ULTIMETER 800

[Sources: <http://www.peetbros.com/pdf/ULTIMETER800Manual.pdf>]

The basic setup of the Ultimeter 800 consists of a base station, a wind speed and wind direction sensor and a temperature sensor. Optionally, it can be equipped with a combined humidity/temperature sensor and with a rain gauge. For the standard sensors Peet Bros indicates an accuracy of +/- 3 km/h, +/- 5% and +/- 0.5°C. In contrast to the other weather stations, the sensors of the Ultimeter are connected to the base station by cable. As the provided cables are only 12 m long, the cable based connection has the disadvantage that the base station and sensors have to be installed in direct vicinity to each other. The base station itself provides basic visualisation and is able to store the highest and lowest values from the different sensors (the maximum and minimum values can be reset manually). A memory for storage of records is not included, but the protocol for the communication between the base station and the PC is public.

4.2.7 SUMMARY

The following table summarises the features of the home weather stations.

Table 4-2: Summarised overview of the features of the home weather stations

Station	Sensors	Transmission to base station	Storage capacity	Protocol available
Vantage Pro 2	rain collector, temperature sensor, humidity sensor, anemometer, wind direction sensor, barometric pressure sensor	wirelessly	2560 records	yes

WS 2500	temperature sensor, humidity sensor, rain collector, wind speed sensor, wind direction sensor, luminosity sensor, barometric pressure sensor	wirelessly	8 days	yes
TE923W	UV-sensor, temperature sensor, humidity sensor, rain collector, wind speed sensor, wind direction sensor, barometric pressure sensor	wirelessly	200 records	no
Mega II + Data Logger	temperature sensor, humidity sensor, rain collector, wind speed sensor, wind direction sensor, barometric pressure sensor	wirelessly	3 – 42 days	no
WMR 100	temperature sensor, humidity sensor, rain collector, wind speed sensor, wind direction sensor, barometric pressure sensor	wirelessly	n/a	no
Ultimeter 800	wind speed, wind direction sensor, temperature sensor	cable	n/a	yes

4.3 OVERVIEW OF AVAILABLE SOFTWARE

As the weather stations presented in this chapter can be accessed by PCs, there is usually a software available which allows to process the measured weather data. The software can be used for different purposes like visualising the data or automatically creating html code from current measurements. The goal of this chapter is to evaluate the different tools aptitude to automatically store the measurements in an easy-accessible, non-proprietary format. The following table gives an overview of the software available for the given stations. Some soft-

ware packages can be downloaded freely while other have to be purchased in order to use the full functionality.

Table 4-3: Overview of available software

Name	Automatic download	Storage	Price	Comments
WeatherProfessional	Yes	Professional database (PostgreSQL)	Free	Supports ELV's station; Configurable time intervals for storage (5 – 60 minutes)
WeatherCapture	No	No export functionality	Free	Supports Honeywell stations
Virtual Weather Station	Yes	csv-file	40 €	Supports Oregon and Peet Bros stations; Configurable time intervals for storage; Automatic backup
PC-Wetterstation	Yes	csv-file	30 €	Supports Davis and Honeywell stations; timer for automatic start-up and shut-down; timer for automatic execution of other programmes
Heavy Weather	Yes	Proprietary format	Free	Supports La Crosse stations; export into a text-file has to be done manually

4.4 SELECTION OF AN ADEQUATE STATION

For the development of the Home Weather Station Demonstrator it was necessary to select a station type which will be used for the developments. This is the type that will initially be supported by the Home Weather Station system. However the structure of the software developed for the demonstrator will possess open interfaces so than an adaptation to other station types can be performed at a later point.

The home weather station bridge is going to be adapted for one type of home weather station. Ideally the station type should fulfil the following requirements in order to allow the development of the software as well as raising the chance that users use this weather station type:

1. the price for the station should not be bigger than 500€ in order to keep the price reasonable for potential users. It is not desirable to develop a system that works only with very expensive station types,
2. an USB interface for connecting the weather station to the PC should be available,
3. the protocol of the communication between station and PC should be public so that the bridge can access the data from the station directly,
4. easy installation of hardware and, if required, software,
5. little configuration effort.

The station type which was finally selected is ELV's WS 2500 weather station. The reasons for selecting the WS 2500 are diverse. Obviously, the main reason is that it fulfils four of the five requirements which are stated above. Its price is about 400€, the protocol is available freely and, once the sensors have been activated, it installs and configures itself automatically, so the fourth and fifth requirements are fulfilled too. The only lack is that it does not provide a USB interface, but with USB/RS-232 adapters available this lack is bearable. Another reason for the selection is that the WS2500 station offers a substantial sensor suite. As all sensors are equipped with a solar panel, the sensors are even autonomous from external power supply. Additionally, up to 8 extra temperature/humidity sensors can be included. Furthermore, it provides a storage which is able to store records up to eight days. This fact is important as it is probable that the user's PC is not running permanently so that the station has to offer storage in order to avoid loss of data.

Below three images showing the base station, a wind sensor and a rain sensor can be found.



Figure 4-1: WS2500 base station



Figure 4-2: WS2500 rain sensor



Figure 4-3: WS2500 wind sensor

5. HOME WEATHER STATION BRIDGE SPECIFICATION AND DESIGN

5.1 INTRODUCTION

The aim of the work package 5400 is to build a bridge between those home weather station sensors connected to a PC and the in-situ network. The general design of the bridge consists of the following three components:

- Station component,
- Registry component,
- Feeder component.

The station component is installed on the PC which is connected to the weather station. Its purpose is to read out observed values from the station and send them to the feeder component which feeds the values into a database. The feeder component, as well as the registry component, is a remote component, i.e. it is situated on a server. The purpose of the registry component is to register weather stations and provide them with an ID which is unique within the network.

The components are able to communicate with each other through their interfaces. The following figure illustrates the use of the interfaces as well as the use of different protocols.

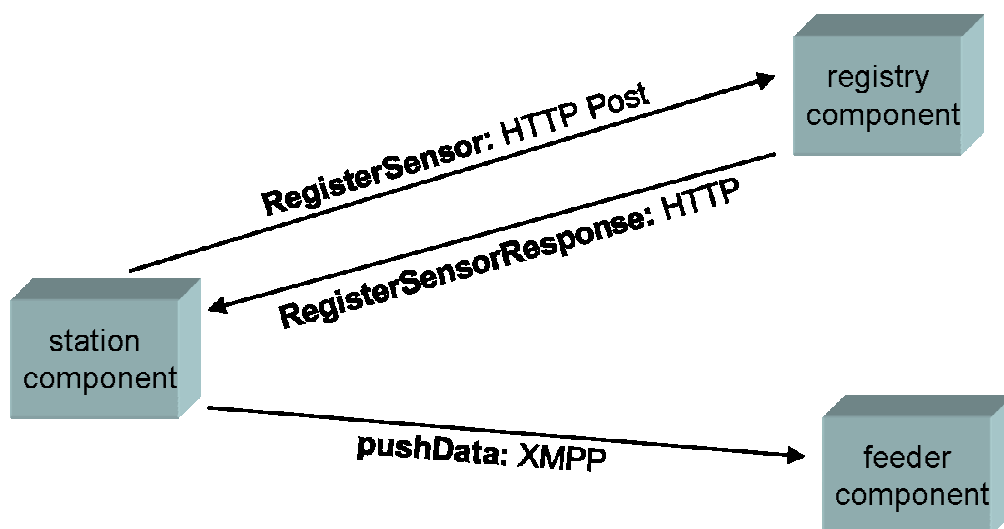


Figure 5-1: Use of interfaces and protocols between the components

The station component utilizes the RegisterSensor request for registering its station. The registry component receives the request via HTTP Post and responds with a RegisterSensor-Response. Beside the unique ID for the station, the response contains a reference to the feeder component. With these two parameters available the station component is able to send observed values to the feeder component which is done through the pushData request. The underlying protocol used here is the Extensible Messaging and Presence Protocol (XMPP¹). XMPP is a protocol for near-real-time XML streaming between two endpoints. It was developed by the Jabber community starting in 1999 and was formalized by IETF in 2002/2003 as a standard for instant messaging and presence. Various extension are available, e.g. the “Multi-User-Chat” which allows a many-to-many chat. XMPP is used here in order to provide the push-based send functionality.

After receiving the data send from the station component, the feeder stores the observed values in a database where it is accumulated for later use, e.g. in a SOS.

5.2 GLOBAL DESIGN

The design of the home weather station system is based on several OGC Sensor Web Enablement standards and the SWE developments of OSIRIS. As shown below in section 5.3 the following components are being used:

- Sensor Observation Service: This service is used for providing the measured weather data to interested clients,
- Sensor Alert Service: From this component the schema for transmitting alerts/measured data is taken,
- SWE Common: This SWE specification provides basic data types that are used for the home weather station system,
- SensorML: This encoding is used for describing the metadata of the home weather stations.

A detailed description of these components and the general SWE architecture can be found in D6001.

¹ See <http://www.xmpp.org/>

5.3 INTERFACE SPECIFICATIONS

5.3.1 ENCODING

All requests and responses have to be encoded in XML. The design of the schemata for the XML-encoding is based on elements from the OGC SWE components SAS, SWE Common and SensorML.

5.3.1.1 SAS

SAS is an event notification service which notifies registered consumers if a certain threshold for an event is reached. Sensors can advertise their services and publish new events while consumers subscribe to events that fulfil certain conditions and are notified by the system once these conditions hold true. SAS supports the following operations:

Table 5-1: SAS operations

Operation	Purpose
getCapabilities	Called to get metadata about the service
Advertise	Called by a sensor to advertise what kind of data could be published
RenewAdvertisement	Allows sensors to renew their previously made advertisement
CancelAdvertisement	Allows sensors to cancel their previously made advertisement
Subscribe	Called by a client to subscribe to alerts advertised and published at the SAS
RenewSubscription	Allows clients to renew their previously made subscription
CancelSubscription	Allows clients to cancel their previously made subscription

In addition an AlertMessage document is defined which is used to send alerts from the sensor to the SAS. The request and response encodings of the bridge are to a great extent adopted from the SAS encodings for “Advertise” and “AlertMessage” as well as “AdvertiseResponse”, which is the response sent from the SAS to the sensor when an “Advertise” request has been processed.

5.3.1.2 SWE COMMON

SWE Common provides elements for describing structure and encoding of data. It can be used to specify:

- Semantics of scalar quantities,
- Units of scalar quantities,
- Reference frame and projection axis for scalar quantities,
- Quantity by scalar values,
- Constraints on scalar values.

The definitions made in SWE Common are expected to be shared among all SWE encodings and services.

5.3.1.3 SENSORML

SensorML provides a way of modelling and encoding metadata about sensors and observation processing. Sensor measurements are modelled as processes in SensorML and contain the following elements:

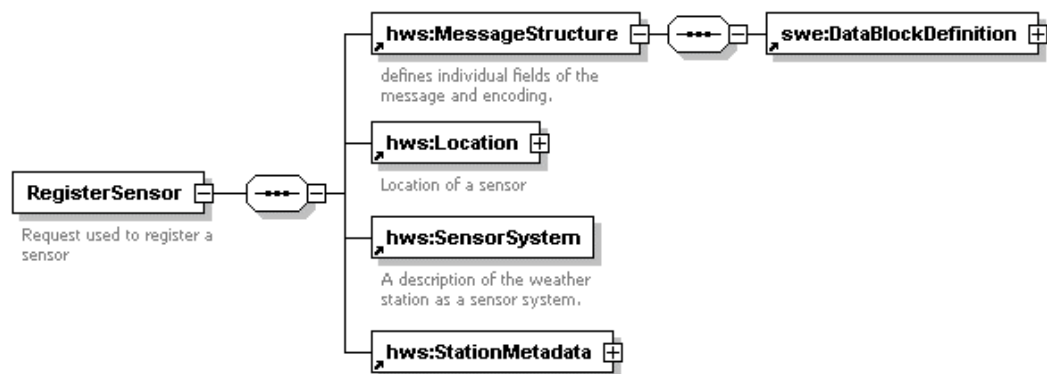
- Inputs of the process, e.g. atmospheric temperature,
- Outputs of the process, e.g. temperature,
- Parameters of the process,
- Method which is used to transform inputs into outputs,
- Reference systems that are used,
- Metadata.

Additionally, process metadata, like general information or contacts, can be specified.

5.3.2 REGISTER SENSOR

The purpose of the RegisterSensor request is to register the weather station and to provide the registry component with additional information about the station, e.g. the location of the station. Another very important purpose of the request is to specify the structure of the data package which is used to send observed values from the station component to the feeder.

Generally, the RegisterSensor document consists of four main elements as one can see in the following figure.



Generated with XMLSpy Schema Editor www.altova.com

Figure 5-2: General structure of the RegisterSensor element

The MessageStructure is the element which defines the structure of the data package, i.e. it defines the ordering in which the observed values will be sent to the feeder. The following extract shows an example of this element. The message described here consists of a temperature field, whose value is measured in degrees Celsius, and a wind speed field, measured in meters per second. Please note, that the URNs like `urn:x-ogc:def:phenomenon:OGC:temperature` are defined in a special dictionary. The most interesting additional SWE elements used in this example are explained in the subsequent table.

```

<hws:MessageStructure>
  <swe:DataBlockDefinition>
    <swe:components name="">
      <swe:DataRecord>
        <swe:field name="temperature">
          <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:temperature">
            <swe:uom code="cel"/>
          </swe:Quantity>
        </swe:field>
        <swe:field name="windSpeed">
          <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:windSpeed">
            <swe:uom code="m/s"/>
          </swe:Quantity>
        </swe:field>
      </swe:DataRecord>
    </swe:components>
    <swe:encoding>
      <swe:TextBlock tokenSeparator=" " decimalSeparator="." blockSeparator=" "/>
    </swe:encoding>
  </swe:DataBlockDefinition>
</hws:MessageStructure>
  
```

Table 5-2: SWE Common data types for the message structure element

Name	Definition	Multiplicity
DataBlockDefinition	Defines data which consists of different data elements that are combined to a data block	1
DataRecord	Defines some logical collection of data values of any type (see [OGC2007])	1
Field	Defines a data value	many
Quantity	Represents a numerical that can be quantified using a decimal value ⁴ ; definition-attribute should include the URN of the phenomenon	1
UOM	Unified Code for Units of Measure; defines the unit of measure. See http://aurora.regenstrief.org/UCUM/ for further information	1
Encoding	Defines how the data is encoded; the separator attributes define which tokens are used for separating different elements	1

The location element contains the location of the weather station. The location can be interpreted either as a single point or an envelope with a given extent. An envelope can be used, for example, if the sensors are distributed over a bigger area. An example of the location element as well as a table of important SWE Common data types is listed in the following two elements. The example shows the three-dimensional location at 8.55° longitude, 52.45° latitude and 200 meters above sea level.

```

<hws:Location>
  <swe:Position referenceFrame="EPSG:31467">
    <swe:location>
      <swe:Vector>
        <swe:coordinate name="x-coordinate">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>8.55</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="y-coordinate">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>52.45</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="z-coordinate">
          <swe:Quantity>
            <swe:uom code="m"/>
            <swe:value>200</swe:value>
          </swe:Quantity>
        </swe:coordinate>
      </swe:Vector>
    </swe:location>
  </swe:Position>
</hws:Location>

```

Table 5-3: SWE Common data types for the location element

Name	Definition	Multiplicity
Position	Includes not only the location but also orientation, velocity (linear), acceleration (linear), angular velocity, and angular acceleration, as well as a time tag (see [OGC2007]); the referenceFrame attribute contains the EPSG code for the reference frame in which the coordinates are defined	1
Vector	Defines a point in space; as the vector is three-dimensional it contains three coordinates	1
Coordinate	A coordinate of the point in space	many

SensorSystem describes the weather station as a sensor system by means of SensorML. This element contains information about the sensors as well as the input and output parameters. A SensorSystem element may look like the following:

```
<hws:SensorSystem version="1.0">
  <sml:member>
    <sml:System>
      <sml:identification>
        <sml:IdentifierList>
          <sml:identifier name="longName">
            <sml:Term definition="urn:x-ogc:def:identifier:longName">
              <sml:value>Touch Screen Radio Weather Station WS 2500</sml:value>
            </sml:Term>
          </sml:identifier>
          <sml:identifier name="shortName">
            <sml:Term definition="urn:x-ogc:def:identifier:shortName">
              <sml:value>WS2500 Weather Station</sml:value>
            </sml:Term>
          </sml:identifier>
          <sml:identifier name="modelNumber">
            <sml:Term definition="urn:x-ogc:def:identifier:modelNumber">
              <sml:value>53759</sml:value>
            </sml:Term>
          </sml:identifier>
          <sml:identifier name="manufacturer">
            <sml:Term definition="urn:xogc:def:identifier:manufacturer">
              <sml:value>ELV Elektro-nik AG</sml:value>
            </sml:Term>
          </sml:identifier>
        </sml:IdentifierList>
      </sml:identification>
      <sml:classification>
        <sml:ClassifierList>
          <sml:classifier name="intendedApplication">
            <sml:Term definition="urn:x-ogc:def:sensor:application">
              <sml:value>weather</sml:value>
            </sml:Term>
          </sml:classifier>
          <sml:classifier name="thermometerType">
            <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
              <sml:value>thermometer</sml:value>
            </sml:Term>
          </sml:classifier>
        </sml:ClassifierList>
      </sml:classification>
    </sml:System>
  </sml:member>
</hws:SensorSystem>
```

```
</sml:Term>
</sml:classifier>
<sml:classifier name="anemometerType">
  <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
    <sml:value>anemometer</sml:value>
  </sml:Term>
</sml:classifier>
<!--this part is abbreviated-->
</sml:ClassifierList>
</sml:classification>
<sml:position name="stationPosition">
  <swe:Position referenceFrame="EPSG:31467">
    <swe:location>
      <swe:Vector>
        <swe:coordinate name="x">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>8.55</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="y">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>52.45</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="z-coordinate">
          <swe:Quantity>
            <swe:uom code="m"/>
            <swe:value>200</swe:value>
          </swe:Quantity>
        </swe:coordinate>
      </swe:Vector>
    </swe:location>
  </swe:Position>
</sml:position>
<sml:interfaces>
  <sml:InterfaceList>
    <sml:interface name="RS-232">
      <sml:InterfaceDefinition>
        <sml:physicalLayer>
          <swe:DataRecord definition="urn:ogc:def:protocol:RS232">
            <swe:field name="num-bits">
              <swe:Count definition="urn:ogc:def:protocol:numberOfBits">
                <swe:value>8</swe:value>
              </swe:Count>
            </swe:field>
            <swe:field name="stop-bits">
              <swe:Count definition="urn:ogc:def:protocol:stopBits">
                <swe:value>2</swe:value>
              </swe:Count>
            </swe:field>
            <swe:field name="baudrate">
              <swe:Quantity definition="urn:ogc:def:protocol:baudrate">
                <swe:value>19200</swe:value>
              </swe:Quantity>
            </swe:field>
            <swe:field name="parity">
              <swe:Boolean definition="urn:ogc:def:protocol:parity">
```

```
<swe:value>true</swe:value>
</swe:Boolean>
</swe:field>
</swe:DataRecord>
</sml:physicalLayer>
</sml:InterfaceDefinition>
</sml:interface>
</sml:InterfaceList>
</sml:interfaces>
<sml:inputs>
<sml:InputList>
<sml:input name="atmosphericTemperature">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:temperature"/>
</sml:input>
<sml:input name="wind">
<swe:Quantity definition="urn:x-ogcdef:phenomenon:OGC:windSpeed"/>
</sml:input>
<!--this part is abbreviated-->
</sml:InputList>
</sml:inputs>
<sml:outputs>
<sml:OutputList>
<sml:output name="WeatherMeasurements">
<swe:DataRecord gml:id="outputGroup">
<swe:field name="temperature">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:temperature">
<swe:uom code="cel"/>
</swe:Quantity>
</swe:field>
<swe:field name="windSpeed">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:windSpeed">
<swe:uom code="m/s"/>
</swe:Quantity>
</swe:field>
<!--this part is abbreviated-->
</swe:DataRecord>
</sml:output>
</sml:OutputList>
</sml:outputs>
<sml:components>
<sml:ComponentList>
<sml:component name="thermometer" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2001A"/>
<sml:component name="windSensor" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2000W"/>
<!--this part is abbreviated-->
</sml:ComponentList>
</sml:components>
</sml:System>
</sml:member>
</hws:SensorSystem>
```

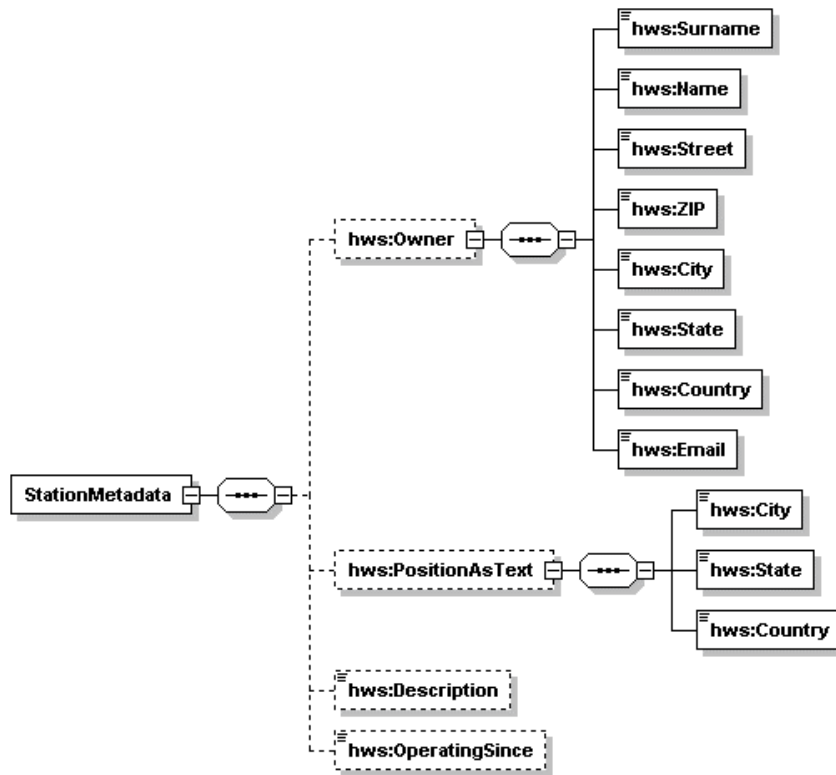
The example defines a sensor system which is called “Touch Screen Radio Weather Station WS 2500” and whose intended application is to observe weather parameters. It has two sensors types and is located at 8.55° longitude, 52.45 ° latitude and 200 meters above sea level. The interface provided is a RS-232 serial interface with 19200 baud-rate, 8 bits, 2 stop-bits

and an even parity. The inputs are wind speed and atmospheric temperature and the outputs are wind speed and temperature. The components used to measure are the S2001A temperature sensor and the S2000W wind sensor. The most interesting SensorML elements used here are described in the following table.

Table 5-4: SensorML data types for the SensorSystem element

Name	Definition	Multiplicity
Identification	General information about the sensor system	many
Classification	Provides a list of possible classifiers that might aid in the rapid discovery of processes, sensors, or sensor systems; classifiers should be considered as information suitable for the discovery process (see [OGC2007])	many
Position	Position of the sensor system	1
Interface	Interface of the sensor system which is used to communicate with the system	1
Input	A property which is observed, e.g. atmospheric pressure	many
Output	The observation made by the sensor	many
Component	A component of the sensor system, i.e. the sensor	many

The last element of the XML-encoded RegisterSensor request is the StationMetadata element. Its purpose is to provide the user of the observations with additional information which might be interesting for him or her, like a description of the station or information about the owner of the station. The following figure illustrates the composition of the StationMetadata element whereas *Table 5-4* describes the elements used inside the main element.



Generated with XMLSpy Schema Editor www.altova.com

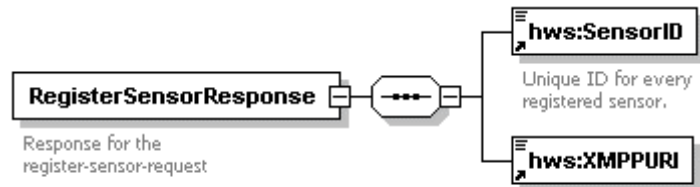
Figure 5-3: The StationMetadata element

Table 5-5: Elements used in the StationMetadata element

Name	Definition	Multiplicity
Owner	The owner of the weather station	1
PositionAsText	The position of the weather station. The elements may only contain text. This element is included since the position of the station and the residence of the owner can be distinct	1
Description	General description of the weather station	1
OperatingSince	Date since the station is operating	1

5.3.3 REGISTER SENSOR RESPONSE

The normal response to a RegisterSensor request is to send a RegisterSensorResponse. The response is XML-encoded and shall include the elements shown in the following figure.



Generated with XMLSpy Schema Editor www.altova.com

Figure 5-4: The RegisterSensorResponse element

Table 5-6: Elements of RegisterSensorResponse

Name	Definition	Multiplicity
SensorID	The ID of the sensor system which has to be unique within the sensor network	1
XMPPURI	The uri of the xmpp server; the structure of the uri is: xmpp:room@server, where room is a multi-user chat room	1

5.3.4 PUSH DATA

The PushData request is used to send observations to the feeder component. Its XML-encoding is an adaptation of SAS's SASAlert element and contains an AlertData element and a timeStamp element which stores the timestamp of the observation. The AlertData element encloses the observations made. Please note that the observations have to be in the same order as defined in the messageStructure element in RegisterSensor. A PushData request may look like the following:

```
<PushData xmlns:sas="http://www.opengis.net/sas/0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="..\hwsBridgePushData.xsd" SID=" urn:x-ogc:object:sensor:IFGI:ws4823">
  <!--values for the phenomena. same order as defined in the registerSensor request-->
  <sas:AlertData>30 5.6 24</sas:AlertData>
  <sas:timeStamp>2007-06-19T13:39:29+02:00</sas:timeStamp>
</PushData>
```

The SID attribute contains the ID, which was received through RegisterSensorResponse. It is mandatory to fill SID because otherwise it is impossible to assign the observations to the sensor system.

6. SUMMARY AND CONCLUSIONS

6.1 KEY ISSUES ADDRESSED

In this report the Home Weather Station Bridge Demonstrator was presented. First an overview of the state-of-the-art of home weather stations was given. This was necessary in order to analyse the market and to find a weather station type that is suited to the aims of the demonstrator. After this the protocols and interfaces used in the Home Weather Station Bridge Demonstrator were described. This comprises a client component which acts as an adapter between the individual home weather stations and the distributed system as a whole and a registry component which allows registering home weather stations, collecting their observation data and making the data available to other users.

6.2 LESSONS IDENTIFIED

One important lesson of the home weather station state-of-the-art analysis was the finding that it is difficult to find home weather stations with open and well documented interfaces that allow the development according adapters. Many manufacturers refuse to make their interface specifications available due to legal and patent concerns. Thus the selection of home weather stations suited for the purposes of the demonstrator was limited.

The development of the home weather station bridge specification has shown that SWE standards (i.e. OGC SAS and OGC SOS) which form a big part of the OSIRIS architecture can be efficiently used for realizing the aims of the Weather Station Bridge Demonstrator.

Appendix.A. HOME WEATHER STATION BRIDGE DEMONSTRATOR REPORT

A.1 AIMS OF THE HOME WEATHER STATION BRIDGE DEMONSTRATOR

The home weather station bridge demonstrator aims at showing basic capabilities and features of OSIRIS technology to the public. Basic principles of the OSIRIS architecture are implemented and applied to the domain of home weather stations which is a quite popular technology.

The first aim is to make the OSIRIS and SWE technology tangible for all interested parties by applying it to a domain which is known to a broad audience. This means that parts of the OSIRIS infrastructure are used for showing how the data of a system of weather stations can be collected and made available on the internet. Thus any user of the system can see that the OSIRIS system works and what it is capable of. In the end every user of the system can see that OSIRIS is not just an abstract concept but even more a system which can be used for building useful applications.

The home weather station bridge demonstrator is furthermore designed as an open platform which allows the easy integration of new weather stations. This is another benefit of the OSIRIS technology which is illustrated by the demonstrator: the plug and play of sensors into existing infrastructures. This will also make it possible that every interested party (e.g. companies, organisations, private users) is able to install their own weather station and add it to the system. This will enhance the value of the system as the number of weather stations increases. At the same time an increasing number of weather stations can raise the attractiveness for users to add their weather stations.

Closely related to this is also the intention to create a community which uses the software and which can also contribute to further enhancements and improvements. Thus the code developed for the demonstrator will be made available as Open Source Software. This idea of creating a community is especially important for supporting additional weather station types besides the type that is initially supported as it is not possible to provide support for every weather station type that is available on the market. Thus the first version of the demonstrator contains the support for one station type which is relatively broadly used. In order to facilitate the development of adapters for additional weather station types a detailed documentation of the bridge and its interfaces will be made available.

Besides these aspects which are more focussed on the OSIRIS dissemination activities the home weather station bridge demonstrator will also provide a basis for further experiments not only related to the OSIRIS architecture and SWE but also based on the data that are delivered by the stations. For example one interesting question that could be investigated is how to handle data that are coming from different sources (i.e. different station types and owners) with a special focus on the reliability of the data.

If the long-term goal of a broad acceptance of the Home Weather Station system is achieved the whole system would also be able to provide a broad range of acquired weather data on a no cost basis. This means that the availability of weather data would also have a practical value to the public.

A.2 DEVELOPMENT

As already shown in section 5.1 the home weather station bridge system consists of two important components:

- Station component
- Registry component

The station component is a piece of software that is installed on those computers to which the home weather stations are connected. This component receives the weather data from the home weather station and transmits them to the registry component. Thus the part of the station component which handles the connection to the weather station is specific to the home weather station type that is used. In this work package the station component was developed to support the home weather station type “ELV WS 2500”. However the station software is designed in such a way that the support for other weather station types can be added easily. Besides the connector to the home weather station the station component contains a module for transmitting the data to the registry component using the protocol described in section 5.3.4. Finally the client component also supports the registration of home weather stations at the registry component. This is handled through an input form through which the user is able to enter information about the weather station, its location and also personal information about the owner of the station.

The registry component is used for collecting the data and making them available. The protocols for this are described in section 5.3. The collected weather data are made available through the OGC SOS interface so that every user application which supports the OGC SOS can access the data.

In order to provide a portal which allows interested users to access the weather data a web site will be set up that contains a client for browsing the different weather stations and displaying their data. The development of this web site will be subject of WP 8110.

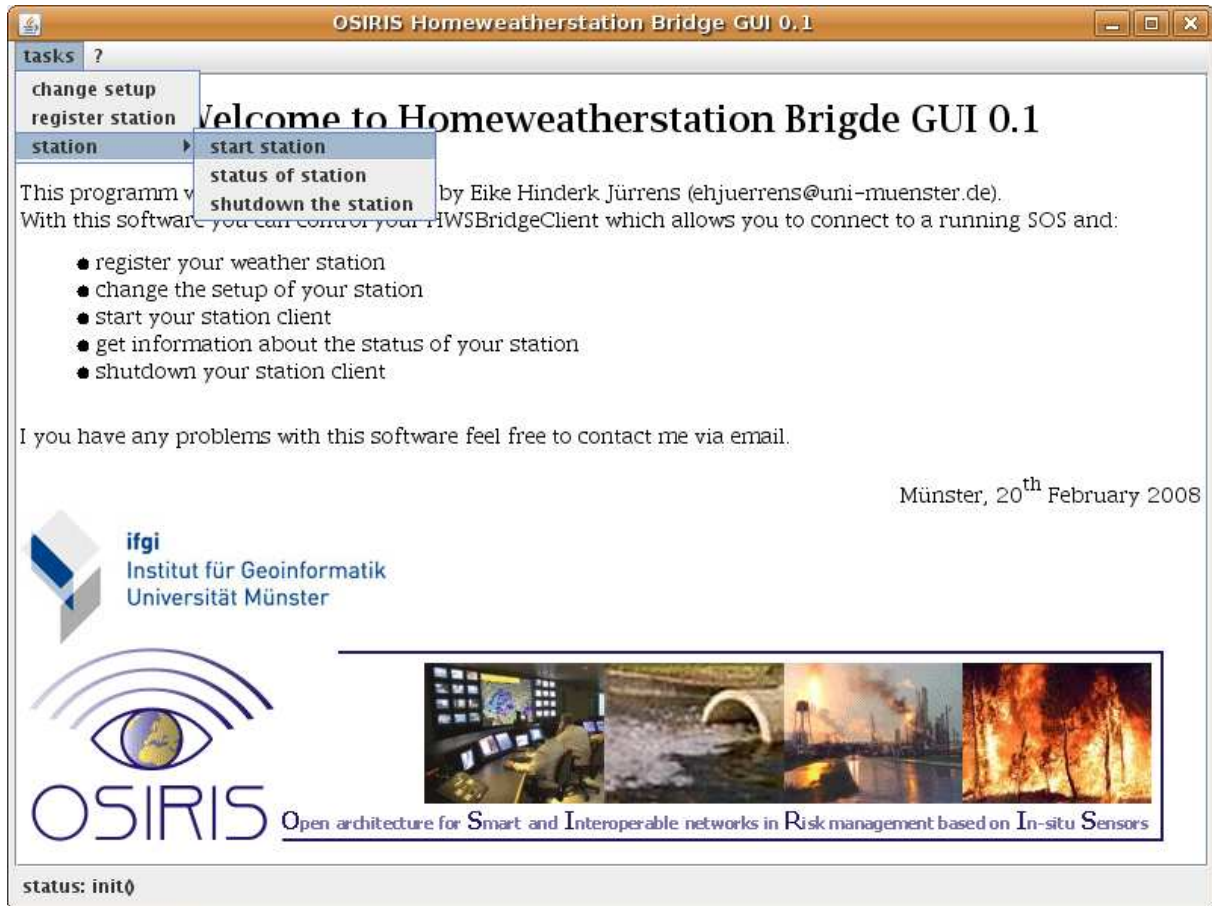


Figure A-1: User Interface of the Station Component

A.3 DEPLOYMENT AND INTEGRATION INTO THE OSIRIS WEB PAGE

An initial set of five home weather stations is deployed by WWU and is distributed across several locations in Münster and other German cities. These stations form the initial network which can be enhanced by additional stations.

The integration into the OSIRIS web site and thus making the demonstrator available to the public is the subject of another work package which is scheduled to be started and completed at a later point (Task T8114 in WP 8110). This web site will comprise the following items:

- General description of the Home Weather Station Bridge Demonstrator
- Downloadable version of the Home Weather Station Bridge Demonstrator software that allows users to integrate their weather station into the system

- Documentation of the interfaces and information intended for developers wanting to provide enhancements and improvements of the current software version
- Client application that allows the access to the weather data provided by the home weather stations.

According to the description of work this web site integration will be completed at T0+27.

Appendix.B. XML-SCHEMATA

B.1 REGISTERSENSOR.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:hws="hwsBridgeCommon.hwsBridge.n52.org"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <annotation>
    <appinfo>hwsBridgeRegisterSensor.xsd</appinfo>
    <documentation>
      <description>This XML Schema encodes the elements and types that are used to register a sen-
sor.</description>
    </documentation>
  </annotation>
  <!-- =====
imports
  ===== -->
  <import namespace="hwsBridgeCommon.hwsBridge.n52.org" schemaLocation="hwsBridgeCommon.xsd"/>
  <!-- =====
types
  ===== -->
  <element name="RegisterSensor">
    <annotation>
      <documentation>Request used to register a sensor</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element ref="hws:MessageStructure"/>
        <element ref="hws:Location"/>
        <element ref="hws:SensorSystem"/>
        <element ref="hws:StationMetadata"/>
      </sequence>
    </complexType>
  </element>
</schema>
```

B.2 REGISTERSENSORRESPONSE.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:hws="hwsBridgeCommon.hwsBridge.n52.org"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <annotation>
    <appinfo>hwsBridgeRegisterSensorResponse.xsd</appinfo>
    <documentation>
      <description>This XML Schema encodes the elements and types that are used to send a response for a regis-
ter-sensor-request</description>
    </documentation>
  </annotation>
  <!-- =====
imports
  ===== -->
  <import namespace="hwsBridgeCommon.hwsBridge.n52.org" schemaLocation="hwsBridgeCommon.xsd"/>
```

```
<!-- =====
types
===== -->
<element name="RegisterSensorResponse">
  <annotation>
    <documentation>Response for the register-sensor-request</documentation>
  </annotation>
  <complexType>
    <sequence>
      <element ref="hws:SensorID"/>
      <element ref="hws:XMPPURI"/>
    </sequence>
  </complexType>
</element>
</schema>
```

B.3 PUSHDATA.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:hws="hwsBridgeCommon.hwsBridge.n52.org"
xmlns:sas="http://www.opengis.net/sas/0.0" xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDe-
fault="qualified" attributeFormDefault="unqualified">
  <annotation>
    <appinfo>hwsBridgePushData.xsd</appinfo>
    <documentation>
      <description>This XML Schema encodes the elements and types for the push data document which is used to
transmit data to the database server.</description>
    </documentation>
  </annotation>
  <!-- =====
imports
===== -->
  <import namespace="http://www.opengis.net/sas/0.0" schemaLocation="../sas/current/sasCommon.xsd"/>
  <!-- =====
types
===== -->
  <element name="PushData" substitutionGroup="sas:Alert">
    <annotation>
      <documentation>The xml element used for sending data from the weather sta-
tion pc to the data
server.</documentation>
    </annotation>
    <!--<complexType>
      <sequence>
        <element ref="hws:SensorID"/>
        <element ref="hws:Timestamp"/>
        <element ref="hws:Data"/>
      </sequence>
    </complexType-->
  </element>
</schema>
```

B.4 COMMON.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:hws="hwsBridgeCommon.hwsBridge.n52.org"
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:xlink="http://www.w3.org/1999/xlink" targetNa-
```

```
namespace="hwsBridgeCommon.hwsBridge.n52.org" elementFormDefault="qualified" attributeFormDe-
fault="unqualified">
<annotation>
<appinfo>hwsBridgeRegisterSensor.xsd</appinfo>
<documentation>
<description>This XML Schema encodes the elements and types that are used to register a sen-
sor.</description>
</documentation>
</annotation>
<!-- =====
imports
===== -->
<import namespace="http://www.opengis.net/swe/1.0" schemaLocation="../sweCommon/1.0.0/swe.xsd"/>
<import namespace="http://www.opengis.net/sensorML/1.0" schemaLoca-
tion="../sensorML/1.0.0/base/sensorML.xsd"/>
<import namespace="http://www.opengis.net/sas/0.0" schemaLocation="../sas/current/sasCommon.xsd"/>
<!-- =====
types
===== -->
<element name="Location">
<annotation>
<documentation>Location of a sensor</documentation>
</annotation>
<complexType>
<choice>
<element ref="swe:Envelope"/>
<element ref="swe:Position"/>
</choice>
</complexType>
</element>
<element name="MessageStructure">
<annotation>
<documentation>defines individual fields of the message and encoding. </documentation>
</annotation>
<complexType>
<sequence>
<element ref="swe:DataBlockDefinition"/>
</sequence>
</complexType>
</element>
<element name="SensorID" type="token">
<annotation>
<documentation>Unique ID for every registered sensor.</documentation>
</annotation>
</element>
<element name="SensorSystem" substitutionGroup="sml:SensorML">
<annotation>
<documentation>A description of the weather station as a sensor sys-tem.</documentation>
</annotation>
</element>
<element name="XMPPURI" type="anyURI"/>
<element name="StationMetadata">
<complexType>
<sequence>
<element name="Owner" minOccurs="0">
<complexType>
<sequence>
<element name="Surname" type="string"/>
<element name="Name" type="string"/>

```

```
<element name="Street" type="string"/>
<element name="ZIP" type="string"/>
<element name="City" type="string"/>
<element name="State" type="string"/>
<element name="Country" type="string"/>
<element name="Email" type="string"/>
</sequence>
</complexType>
</element>
<element name="Description" type="string" minOccurs="0"/>
<element name="OperatingSince" type="date" minOccurs="0"/>
</sequence>
</complexType>
</element>
</schema>
```

B.5 SWE4HWSBRIDGE.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" targetName-
space="http://www.opengis.net/swe/0.0" elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:annotation>
    <xs:documentation>
      swe4hwsBridge.xsd
      Utility schema which simply includes the SweCommon schema documents required for the OGC SAS sche-
      mas</xs:documentation>
    </xs:annotation>
    <!-- =====
      includes and imports
      ===== -->
    <xs:include schemaLocation="../sweCommon/current/swe.xsd"/>
  </xs:schema>
```

Appendix.C. XML-EXAMPLES

C.1 REGISTERSENSOR

```
<?xml version="1.0" encoding="UTF-8"?>
<RegisterSensor xmlns:hws="hwsBridgeCommon.hwsBridge.n52.org"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="C:\Dokumente
und Einstellungen\lordeman\Eigene Datei-
en\Osiris\Bridge\schemata\BridgeSchemata\hwsBridgeRegisterSensor.xsd"
xmlns:swe="http://www.opengis.net/swe/1.0" xmlns:sml="http://www.opengis.net/sensorML/1.0"
xmlns:gml="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink">
<hws:MessageStructure>
<swe:DataBlockDefinition>
<swe:components name="">
<swe:DataRecord>
<swe:field name="temperature">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:temperature">
<swe:uom code="cel"/>
</swe:Quantity>
</swe:field>
<swe:field name="windSpeed">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:windSpeed">
<swe:uom code="m/s"/>
</swe:Quantity>
</swe:field>
<swe:field name="windDirection">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:windDirection">
<swe:uom code="deg"/>
</swe:Quantity>
</swe:field>
<swe:field name="precipitation">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:precipitation">
<swe:uom code="mm"/>
</swe:Quantity>
</swe:field>
<swe:field name="barometricPressure">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:barometricPressure">
<swe:uom code="hPa"/>
</swe:Quantity>
</swe:field>
<swe:field name="radiation">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:Radiation">
<swe:uom code="lx"/>
</swe:Quantity>
</swe:field>
<swe:field name="relativeHumidity">
<swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:RelativeHumidity">
<swe:uom code="%"/>
</swe:Quantity>
</swe:field>
</swe:DataRecord>
</swe:components>
<swe:encoding>
<swe:TextBlock tokenSeparator=" " decimalSeparator="." blockSeparator=" "/>
</swe:encoding>
```

```
</swe:DataBlockDefinition>
</hws:MessageStructure>
<hws:Location>
  <swe:Position>
    <swe:location>
      <swe:Vector>
        <swe:coordinate name="x-coordinate">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>8.55</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="y-coordinate">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>52.45</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="z-coordinate">
          <swe:Quantity>
            <swe:uom code="m"/>
            <swe:value>200</swe:value>
          </swe:Quantity>
        </swe:coordinate>
      </swe:Vector>
    </swe:location>
  </swe:Position>
</hws:Location>
<hws:SensorSystem version="1.0">
  <sml:member>
    <sml:System>
      <sml:identification>
        <sml:IdentifierList>
          <sml:identifier name="longName">
            <sml:Term definition="urn:x-ogc:def:identifier:longName">
              <sml:value>Touch Screen Radio Weather Station WS 2500</sml:value>
            </sml:Term>
          </sml:identifier>
          <sml:identifier name="shortName">
            <sml:Term definition="urn:x-ogc:def:identifier:shortName">
              <sml:value>WS2500 Weather Station</sml:value>
            </sml:Term>
          </sml:identifier>
          <sml:identifier name="modelName">
            <sml:Term definition="urn:x-ogc:def:identifier:modelNumber">
              <sml:value>53759</sml:value>
            </sml:Term>
          </sml:identifier>
          <sml:identifier name="manufacturer">
            <sml:Term definition="urn:x-ogc:def:identifier:manufacturer">
              <sml:value>ELV Elektro-nik AG</sml:value>
            </sml:Term>
          </sml:identifier>
        </sml:IdentifierList>
      </sml:identification>
      <sml:classification>
        <sml:ClassifierList>
          <sml:classifier name="intendedApplication">
            <sml:Term definition="urn:x-ogc:def:sensor:application">
```

```
<sml:value>weather</sml:value>
</sml:Term>
</sml:classifier>
<sml:classifier name="thermometerType">
  <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
    <sml:value>thermometer</sml:value>
  </sml:Term>
</sml:classifier>
<sml:classifier name="barometerType">
  <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
    <sml:value>barometer</sml:value>
  </sml:Term>
</sml:classifier>
<sml:classifier name="anemometerType">
  <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
    <sml:value>anemometer</sml:value>
  </sml:Term>
</sml:classifier>
<sml:classifier name="rainGaugeType">
  <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
    <sml:value>rainGauge</sml:value>
  </sml:Term>
</sml:classifier>
<sml:classifier name="radiationSensorType">
  <sml:Term definition="urn:x-ogc:def:sensor:sensorType">
    <sml:value>radiationSensor</sml:value>
  </sml:Term>
</sml:classifier>
</sml:ClassifierList>
</sml:classification>
<sml:position name="stationPosition">
  <swe:Position referenceFrame="EPSG:31467">
    <swe:location>
      <swe:Vector>
        <swe:coordinate name="x">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>8.55</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="y">
          <swe:Quantity>
            <swe:uom code="deg"/>
            <swe:value>52.45</swe:value>
          </swe:Quantity>
        </swe:coordinate>
        <swe:coordinate name="z-coordinate">
          <swe:Quantity>
            <swe:uom code="m"/>
            <swe:value>200</swe:value>
          </swe:Quantity>
        </swe:coordinate>
      </swe:Vector>
    </swe:location>
  </swe:Position>
</sml:position>
<sml:interfaces>
  <sml:InterfaceList>
    <sml:interface name="RS-232">
```

```
<sml:InterfaceDefinition>
  <sml:physicalLayer>
    <swe:DataRecord definition="urn:ogc:def:protocol:RS232">
      <swe:field name="num-bits">
        <swe:Count definition="urn:ogc:def:protocol:numberOfBits">
          <swe:value>8</swe:value>
        </swe:Count>
      </swe:field>
      <swe:field name="stop-bits">
        <swe:Count definition="urn:ogc:def:protocol:stopBits">
          <swe:value>2</swe:value>
        </swe:Count>
      </swe:field>
      <swe:field name="baudrate">
        <swe:Quantity definition="urn:ogc:def:protocol:baudrate">
          <swe:value>19200</swe:value>
        </swe:Quantity>
      </swe:field>
      <swe:field name="parity">
        <swe:Boolean definition="urn:ogc:def:protocol:parity">
          <swe:value>true</swe:value>
        </swe:Boolean>
      </swe:field>
    </swe:DataRecord>
  </sml:physicalLayer>
</sml:InterfaceDefinition>
</sml:interface>
</sml:InterfaceList>
</sml:interfaces>
<sml:inputs>
  <sml:InputList>
    <sml:input name="atmosphericTemperature">
      <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:temperature"/>
    </sml:input>
    <sml:input name="wind">
      <swe:Quantity definition="urn:x-ogcdef:phenomenon:OGC:windSpeed"/>
    </sml:input>
    <sml:input name="precipitation">
      <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:precipitation"/>
    </sml:input>
    <sml:input name="atmosphericPressure">
      <swe:Quantity definition="urn:x-ogcdef:phenomenon:OGC:pressure"/>
    </sml:input>
    <sml:input name="radiation">
      <swe:Quantity definition="urn:x-ogcdef:phenomenon:OGC:Radiation"/>
    </sml:input>
    <sml:input name="humidity">
      <swe:Quantity definition="urn:x-ogcdef:phenomenon:OGC:RelativeHumidity"/>
    </sml:input>
  </sml:InputList>
</sml:inputs>
<sml:outputs>
  <sml:OutputList>
    <sml:output name="WeatherMeasurements">
      <swe:DataRecord gml:id="outputGroup">
        <swe:field name="temperature">
          <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:temperature">
            <swe:uom code="cel"/>
          </swe:Quantity>
        </swe:field>
      </swe:DataRecord>
    </sml:output>
  </sml:OutputList>
</sml:outputs>
</sml:inputs>
```

```
</swe:field>
<swe:field name="windSpeed">
  <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:windSpeed">
    <swe:uom code="m/s"/>
  </swe:Quantity>
</swe:field>
<swe:field name="windDirection">
  <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:windDirection">
    <swe:uom code="deg"/>
  </swe:Quantity>
</swe:field>
<swe:field name="precipitation">
  <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:precipitation">
    <swe:uom code="mm"/>
  </swe:Quantity>
</swe:field>
<swe:field name="barometricPressure">
  <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:barometricPressure">
    <swe:uom code="hPa"/>
  </swe:Quantity>
</swe:field>
<swe:field name="radiation">
  <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:Radiation">
    <swe:uom code="lx"/>
  </swe:Quantity>
</swe:field>
<swe:field name="relativeHumidity">
  <swe:Quantity definition="urn:x-ogc:def:phenomenon:OGC:RelativeHumidity">
    <swe:uom code="%"/>
  </swe:Quantity>
</swe:field>
</swe:DataRecord>
</sml:output>
</sml:OutputList>
</sml:outputs>
<sml:components>
  <sml:ComponentList>
    <sml:component name="thermometer" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2001A"/>
    <sml:component name="humiditySensor" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2001A"/>
    <sml:component name="radiationSensor" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2500H"/>
    <sml:component name="rainGauge" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2000R"/>
    <sml:component name="windSensor" xlink:role="urn:x-ogc:def:sensor:detector" xlink:href="urn:x-ogc:sensor:ELV:S2000W"/>
    <sml:component name="barometer" xlink:role="urn:x-ogc:def:sensor:detector"/>
  </sml:ComponentList>
</sml:components>
</sml:System>
</sml:member>
</hws:SensorSystem>
<hws:StationMetadata>
  <hws:Owner>
    <hws:Surname>Mustermann</hws:Surname>
    <hws:Name>Max</hws:Name>
    <hws:Street>Robert-Koch-Straße 26</hws:Street>
    <hws:ZIP>48149</hws:ZIP>
```

```
<hws:City>Münster</hws:City>
<hws:State>North Rhine-Westfalia</hws:State>
<hws:Country>Germany</hws:Country>
<hws:Email>max.mustermann@uni-muenster.de</hws:Email>
</hws:Owner>
<hws:PositionAsText>
  <hws:City>Münster</hws:City>
  <hws:State>North Rhine-Westfalia</hws:State>
  <hws:Country>Germany</hws:Country>
</hws:PositionAsText>
<hws:Description>This is a WS2500 weather station setup at the Institute for Geoinformat-ics, Münster Uni-
versity.</hws:Description>
<hws:OperatingSince>2007-06-01</hws:OperatingSince>
</hws:StationMetadata>
</RegisterSensor>
```

C.2 REGISTERSENSORRESPONSE

```
<?xml version="1.0" encoding="UTF-8"?>
<RegisterSensorResponse xmlns:hws="hwsBridgeCommon.hwsBridge.n52.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="..\hwsBridgeRegisterSensorResponse.xsd">
  <hws:SensorID>urn:x-ogc:object:sensor:IFGI:ws4823</hws:SensorID>
  <hws:XMPPURI>xmpp:pub301@conference.52North.org</hws:XMPPURI>
</RegisterSensorResponse>
```

C.3 PUSHDATA

```
<?xml version="1.0" encoding="UTF-8"?>
<PushData xmlns:sas="http://www.opengis.net/sas/0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="..\hwsBridgePushData.xsd" SID="urn:x-
ogc:object:sensor:IFGI:ws4823">
  <!--values for the phenomena. same order as defined in the registerSensor request-->
  <sas:AlertData>30 5.6 24</sas:AlertData>
  <sas:timeStamp>2007-06-19T13:39:29+02:00</sas:timeStamp>
</PushData>
```

Appendix.D. REFERENCES

Davis 2006, Vantage Pro 2 Console Manual, Retrieved: May 10, 2007, from http://www.davisnet.com/support/weather/support_docs.asp?dtype=1

Honeywell 2006, Honeywell professional weather station with remote control, Retrieved: May 15, 2007, from <http://www.honeywellweatherstations.com/Manuals/TE923W-ENG.pdf>

La Crosse 2007, Profi-Funk-Wetterstation Bedienungsanleitung, Retrieved: May 23, 2007, from http://www.tfa-dostmann.de/Bedienungsanleitungen/35.1023_multi.pdf

OGC 2006, OpenGIS Sensor Model Language (SensorML) Implementation Specification, OGC Document Number OGC 05-086r2

OGC 2007, OpenGIS Sensor Model Language (SensorML) Implementation Specification, pre-approval version, OGC Document Number OGC 07-000