International Journal of Geo-Information

MDPI

Article

A Methodology for Heterogeneous Sensor Data Organization and Near Real-Time Data Sharing by Adopting OGC SWE Standards

Bartolomeo Ventura [1,*], Andrea Vianello [2], Daniel Frisinghelli [2,3], Mattia Rossi [1,4], Roberto Monsorno [2] and Armin Costa [2]

1 Institute for Earth Observation—Eurac Research, Viale Druso 1, 39100 Bolzano, Italy; mattia.rossi@eurac.edu
2 Center for Sensing Solutions—Via Alessandro Volta, 13, 39100 Bolzano, Italy; andrea.vianello@eurac.edu (A.V.); Daniel.Frisinghelli@student.uibk.ac.at (D.F.); roberto.monsorno@eurac.edu (R.M.); armin.costa@eurac.edu (A.C.)
3 Institute of Atmospheric and Cryospheric Sciences, University of Innsbruck, Innrain 52, 6020 Innsbruck, Austria
4 Faculty of Science and Technology, Free University of Bolzano, Piazza Università, 1, 39100 Bolzano, Italy
* Correspondence: bartolomeo.ventura@eurac.edu

Received: 23 January 2019; Accepted: 29 March 2019; Published: 2 April 2019

Abstract: Finding a solution to collect, analyze, and share, in near real-time, data acquired by heterogeneous sensors, such as traffic, air pollution, soil moisture, or weather data, represents a great challenge. This paper describes the solution developed at Eurac Research to automatically upload data, in near real-time, by adopting Open Geospatial Consortium (OGC) Sensor Web Enablement (SWE) standards to guarantee interoperability. We set up a methodology capable of ingesting heterogeneous datasets to automatize observation uploading and sensor registration, with minimum interaction required of the user. This solution has been successfully tested and applied in the Long Term (Socio-)Ecological Research (LT(S)ER) Matsch-Mazia initiative, and the code is accessible under the CC BY 4.0 license.

Keywords: sensor web technologies; standard; OGC; interoperability; SOS; LT(S)ER; near-real-time data; automatization

1. Introduction

The possibility to collect and ingest data in near real-time for the monitoring of events belonging to different fields of our life is a big challenge. One issue is being able to collect the data, while another is being able to access the data quickly enough and with sufficient ease to allow visualization and analysis.

In recent years, the market for sensors has grown quickly and it is possible to find more and more efficient and intelligent devices, with a subsequent notable decrease in their costs. Hydrology [1], oceanography [2], risk monitoring [3], weather forecasting or citizen- and urban-sensing [4] and the monitor of the essential climate variables (ECV) [5] represent some of the fields of interest. In fact, the possibility of defining a real-time solution for data acquisition could improve the understanding of physical processes and their evolution.

Due to the large number of sensor manufacturers and the related diversity of their output formats, the first problem to face is finding a uniform way to organize data for analysis and, as a further step, to share them on the web. In this context, the idea of the Sensor Web represents a solution to standardize this process.

The definition of the "Sensor Web" was introduced by Delin et al. [6] and Delin [7], as an idea to create sensor nodes for collecting data by means of a set of shared and robust protocols. The Open Geospatial Consortium (OGC) defines interoperability interfaces and metadata encodings that enable the real-time integration of heterogeneous sensor webs and the Internet of Things into the information infrastructure by means of the Sensor Web Enablement (SWE) initiative [8–10].

OGC is an international, non-profit, and voluntary consensus standards organization. It consists of more than 380 companies, government agencies, and universities. OGC introduced the following SWE standards:

- A standard for encoding sensor observation: Observations and Measurements (O&M) [11];
- a standard for describing sensor characteristics: Sensor Model Language (SensorML) [12]; and
- a web service to access SensorML O&M: Sensor Observation Service (SOS) [13].

These standards allow sensor interoperation and service synergy, based on the possibility of discovering and exchanging data from sensor observations, facilitating process automation. Furthermore, the availability of the respective metadata guarantees enough information provided to allow sensors to be selected properly and to give an initial overview of their specific histories and characteristics.

This paper describes a solution to automatically upload data, in near real-time, in any standard SOS implementation, to simplify data sharing and interoperability and to speed up the output of research.

2. Related Work

The diffusion of the OGC SWE standards led to the creation of sensor observation management systems, such as istSOS (http://istsos.org/) [14] and the 52°North SOS project (http://52north.org/about/52north). These implementations form a basis for organizing datasets and sharing them on the internet in a real-time system.

Of these two solutions, we decided to adopt the latter because it is robust, largely adopted in research institutes, and is well-supported by the developers. Addditionally, the GET-IT (http://www.get-it.it/) system integrates the 52°North SOS project for distributing observation data and metadata of sensors, according to the INSPIRE directive. It offers a user-friendly web interface which can effectievly simplify the use of the SWE standards, especially for non-expert users [15,16]. As mentioned before, the heterogeneous and dynamic market of sensors does not fit with the concept of standardization. This complicates the harmonization of datasets and increases the effort before uploading into the SOS service. This is why it is also necessary to create an intermediate phase, to make sensor data ready for the SWE initiative. Martínez et al. [17] considered a valuable solution in this direction, by means of a plug-and-play sensor integrated into research data infrastructures. To facilitate the integration of multiple data providers, it is also possible to use a Lightweight SOS Profile [18]. The basic idea is to comprise those basic SOS operations (GetCapabilities, DescribeSensor, GetObservation, and GetFeatureOfInterest), so that they must be implemented by every standard compliant SOS server to support typical functionalities, such as sensor data access and visualization.

These solutions show that data standardization represents a great opportunity to take into account, as it allows the movement from local to distributed and cloud computing, in which information and computing resources are provided as web services [19].

This paper focuses on the development of a flexible procedure which, with a minimum interaction of the user, automatically uploads data, in near real-time, in any standard SOS implementation. In particular, the main idea is to adopt the 52°North SOS implementation and to integrate a tool able to automatize observation uploading and sensor registration (InsertSensor operation), with the least effort possible required by the user. This automatization is, in our opinion, still missing in the available opensource solutions and represents a big problem for researchers. In our work, we try to fill this gap by means of a Python-based tool that is efficient and easy to use.

Furthermore, this solution offers the possibility of simplifying the creation of sensor metadata in a standard format (i.e., OGC, INSPIRE, or ISO) or, whenever necessary, in a project-specific format. In fact, considering the increasing number of sensors, it is extremely important to adopt a common standard for metadata, in order to obtain an organized and interoperable federated system.

The paper contains the following sections: Section 3 introduces the rationale behind the implementation design. Section 4 describes the concept of the data life cycle, which is fundamental in describing all of the phases, from acquisition through to the sharing of data. In particular, the procedure we set up to have the data running in a SOS service is described. Section 5 presents an application of this procedure used in the LT(S)ER initiative. Finally, the conclusion section summarizes our results and defines the future steps.

In the initial stage of the implementation phase, we realized that the user may not even know the OGC standards or have competence in writing code; either of which could increase errors and production time. Therefore, the previous HCI factors were directly discussed with the user because it is important to know, already in the test phase, his/her perception of the methodology and, above all, if it meets the requirements. As a consequence, the possibility to automatize, as much as possible, the entire data flow, together with a minimization of user interaction, aims to guarantee an overall level of satisfaction with respect to the final goal represented by the possibility to have data in near real-time. This is the general concept of the rationale of our approach for the data life cycle.

In particular, ground stations have internet connections. In this way, the observed data are locally stored initially and then, after establishing a remote control and connection, they are directly accessible and downloadable. The harmonization phase guarantees quality-checked data that can be organized to be ready for the automatic upload in a SOS system. The SOS environment offers the possibility to access sensor metadata and their observations through an Applied Programming Interface (API) by means of GET requests. This system has the advantage that it is accessible with any programming languages able to deal with simple HTTP requests. Each data point at a given time stamp can be identified, downloaded, and included in further scientific analyses, resulting in a simplified human-data interaction. The data sharing is meant to connect users to the data stored in the database in a direct way. Depending on the user's experience with programming languages and the equivalent complex systems of data retrieval and analysis, it is useful to create systems able to expose the data in a very intuitive way. Such approaches could be web-based applications directly interacting with a back-end client and giving the possibility to access, wrangle, and visualize the data on the fly by a potential data user.

3. Data Life Cycle

The concept of the data life cycle consists of a sequence of phases that data goes through, from its initial generation or capture through to its eventual archival and/or deletion at the end of its useful life. It provides a high-level overview about the management of data, subdivided into different stages according to specific tasks, with the aim of the successful organization and preservation of data for use and sharing. Multiple versions of a data life cycle exist [20–22], where the main differences are related to the variation in practices across fields of interests or communities.

The phases of the data life cycle we adopted are represented in Figure 1. Data are collected (acquisition phase) by ground stations running proprietary software tools. To fulfill the remaining phases (harmonization, organization, analysis, and sharing), we combined open-source solutions with Python and R scripts to simplify the entire procedure.

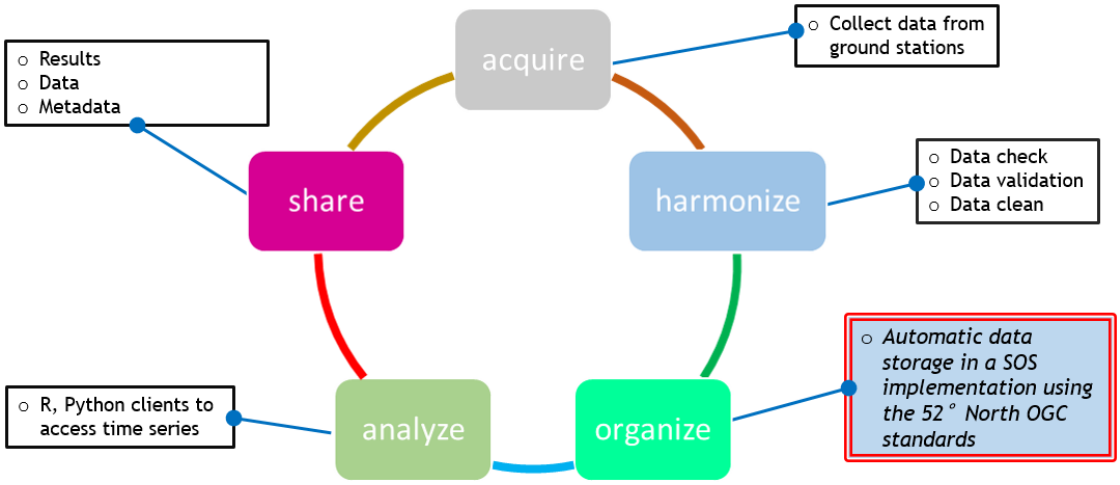


Figure 1. The adopted data life cycle for implementing our methodology. In the double-framed box, our contribution is summarized.

4. Implementation Design

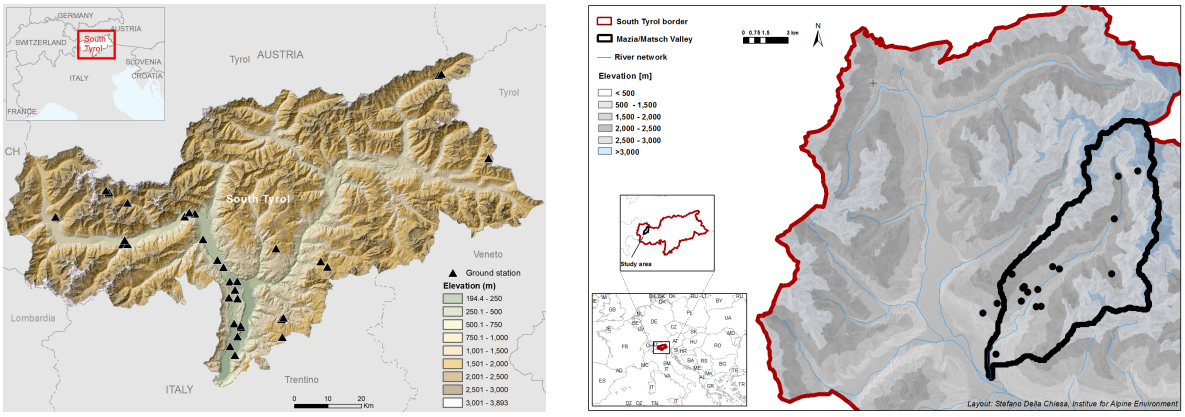
In the early phase of the development of the methodology, we tried to combine the concept of usability together with the factors of Human-Computer Interaction (HCI) to create a system capable of supporting the research in a functional way. Usability refers to the quality of the interaction, in terms of parameters such as time taken to perform tasks, number of errors made, and the time to become a competent user [23]. The word "usability" also refers to methods for improving ease of use during the design process [24]. Among the different factors involved in HCI, we focused on the following [25]:

- The User—motivation, enjoyment, satisfaction, personality, and experience level;
- Task Factors—easy, complex, novel, task allocation, repetitive, monitoring, skills, and components;
- Productivity factors—increase output, increase quality, decrease cost, decrease errors, decrease labor requirements, decrease production time, and increase creative and innovative ideas leading to new products.

4.1. Acquisition

The Institute for Alpine Environment, with the technical collaboration of the Center for Sensing Solution group, at Eurac Research in Bolzano, Italy, installed and maintained ground stations for collecting both images (RGB and Infrared) and 75 other physical parameters, such as Normalized Difference Vegetation Index (NDVI), Surface Temperature, or Photosynthetically Active Radiation. In particular, digital cameras taking repeated images (Digital Repeat Photography, DRP) of the landscape at high frequencies (several images per day) helped to overcome the limitations of field observations by individuals, such as continuity and logistics [26]. The geographical locations of these ground stations are shown in Figure 2a,b. The main goal of creating such a network is the possibility of having long-term observations with automated systems at high temporal (and, in the case of DRP, also high spatial) resolutions.

The solution we introduce in this work uses the data collected by this network of ground stations.



(**a**) Ground station network used for data collection.

(**b**) Ground station network overview in Val Mazia, Matsch area.

Figure 2. South Tyrol network of ground stations.

4.2. Harmonization

Harmonization consists of three steps: Checking, validating, and cleaning the acquired data. It is not unusual for data to have discrepancies in consistency or format and, as a consequence, it is important to create a cohesive dataset that is ready to be organized. In Figure 3, the details of the evaluated quality dimensions are listed.

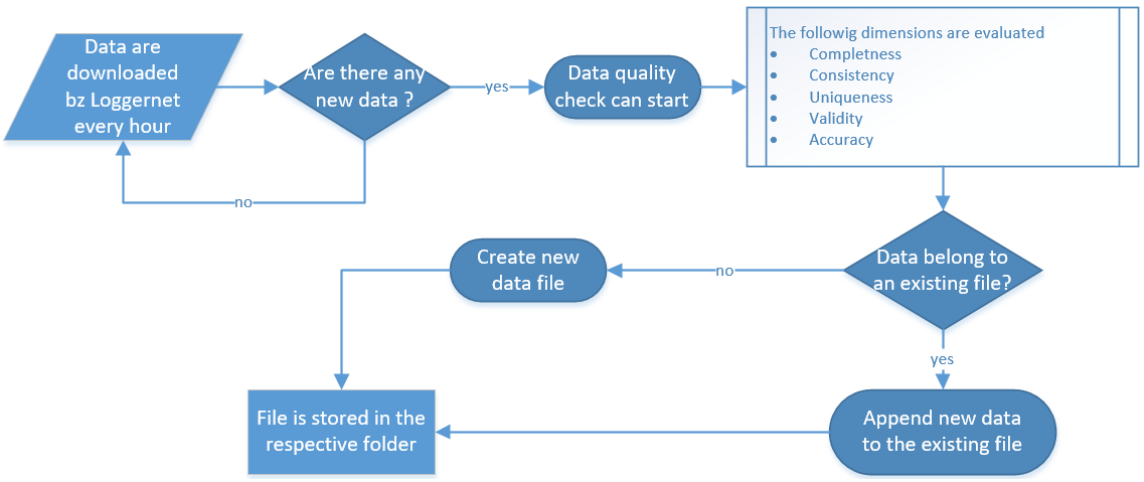


Figure 3. Flowchart describing the data quality-check performed to harmonize the collected data.

In particular [27,28]:

- Data completeness refers to whether there are any gaps in the data, between what was expected to be collected and what was actually collected.
- Data consistency refers to whether the types of data align with the expected versions of the data that should be coming in.
- Data uniqueness ensures that each record should be unique.
- Validity of data is determined by whether the data measures that which it was intended to measure.
- Data accuracy refers to whether the collected data is correct, and accurately represents what it should.

The tests for these different dimensions are performed using scripts, mainly R-codes, written by the researchers of the Institute for Alpine Environment.

4.3. Organization

Collected data are stored in a relational database management system (RDBMS) using the database model implementation (https://wiki.52north.org/SensorWeb/SensorObservationServiceDatabaseModel) of the 52°North SOS project (version 4.1), based on the OGC standard PostgreSQL [29]. This database has all the necessary tables to provide the SOS service, without the need to develop a new one unless specific project customizations are required.

The 52°North SOS provides a RESTful API for inserting data in the database using the POST method and the operations defined in the SOS standard:

- *InsertSensor*: Register sensors in the DataBase (DB) following the SensorML standard.
- *InsertObservation*: Upload measurements into the DB following the O&M standard.

The InsertSensor operation is done once and contains the following information:

- *PhysicalSystem*: Identifier for the physical instrument in one of the platforms (stations).
- *Offering*: A group of parameters measured by one sensor. It is worth noting that one procedure (PhysicalSystem) can have multiple offerings, but the same offering cannot be applied to multiple procedures.
- *Observable property*: The output parameters of the sensors.
- *Feature Of Interest* (FOI): The coordinates and altitude of the station or platform where the sensor is installed.

The InsertObservation operation allows the client to insert new observations for a sensor system formerly registered with the SOS. By means of this last operation, the records are inserted in the database.

Considering the big number of records to be collected, it is necessary to have a solution able to speed up not only the InsertSensor, but also the InsertObservation operations.

4.3.1. Methodology

The availability of an automation process gives the chance to upload new observations into the database as soon as they are collected. The innovation of the methodology is to facilitate the insertion of the observations into the database with a semi-automatic procedure. In fact, only a minimal interaction of the user and/or data provider is required, see Figure 4. Furthermore, the possibility to have a methodology based on OGC standards can improve the comparability of data collected by any other institutions, offering the possibility of their reuse and, thus, avoiding duplication.

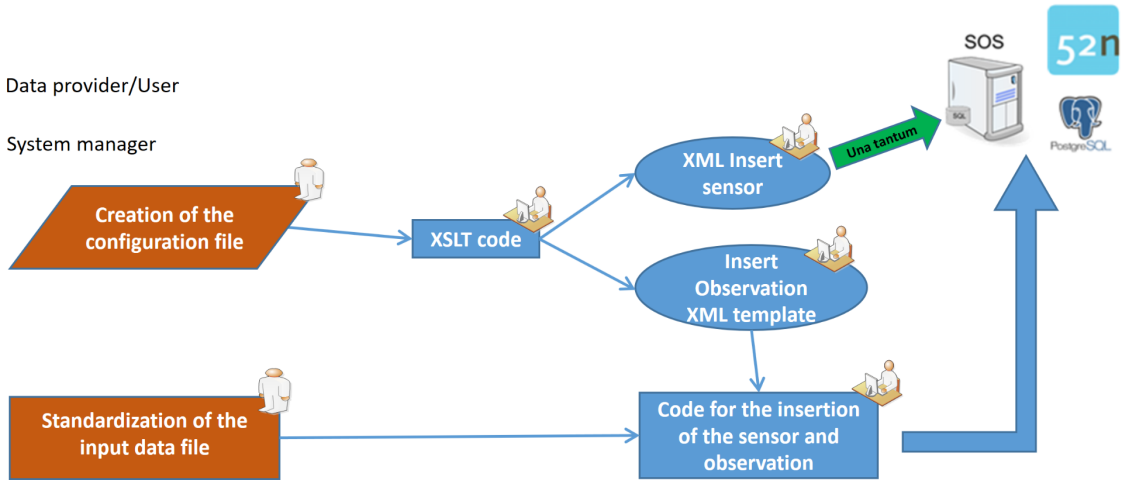


Figure 4. Schema for the insertion of the sensors and observations. From the data provider/user point of view, only two actions are requested (in orange): Create a configuration file and standardize the input data file.

The first step to be performed by the user is the standardization of the input file. The user has to define a univocal structure for the input file, as every station has its own data-logger using a proprietary standard.

The structure we propose is the following, see Figure 5:

- A header describing the data column;
- an established date and time format, (i.e., *YYYY-MM-DDThh:mm*);
- the time-stamp is always stored in the first column; and
- a predetermined naming convention (i.e., *XXX_FOI_XXX.csv*).

```
TIMESTAMP,T_Air,RH_Air
2017-01-01T00:00,-1.938,35.83
2017-01-01T00:15,-2.316,35.7
2017-01-01T00:30,-2.745,37.67
2017-01-01T00:45,-3.246,39.88
...
```

Figure 5. Standardization of the input file: Example of univocal structure for the input data file *EuracID00216A_M2paf2200_20170101000020171231234 5.csv*.

A fixed structure for the input data allows us to know, in advance, how to correctly retrieve all the data to be inserted into the database. A well-established naming convention is necessary for correct use of a predefined folder structure. In fact, the code uses the naming convention, extracting the FOI string, to automatically create paths to locate and access files for the entire process. This will avoid modifying the script for each new data insertion.

The second step is the creation of the configuration file.

The configuration file is an XML file, which has to be filled by the user because it contains all the information for describing the sensor metadata and observed properties list; see Figure 6. The idea of creating a configuration file is essentially twofold: Firstly, it will contain all the information useful for automatically creating both the Insert Sensor and Insert Observation XML files, by means of a XSL Transformation (XSLT); secondly, it can be seen as a raw version of the metadata for the considered FOI. The XSLT is strictly connected to the configuration file. In fact, all the necessary information to be inserted into the Insert Sensor and Insert Observation XML files will be retrieved from it. As a consequence, it is not possible to use a standard SensorML document because it will not be useful for the automation of the aforementioned SOS operations. Furthermore, the definition of a custom XML document offers the possibility to include information not foreseen by the standard model.

After the creation of the configuration files, the procedure can execute the following steps:

1. Run the environment_generator.sh script. This code creates the data folder structure and all the auxiliary files and folders for the insertion process.
2. Save the the .csv input data files in the Data folder.
3. Run the SOS_Insert_Sensor.py script once to register the sensors in the SOS service.
4. Run the SOS_Insert_Observation.py script to upload the observation into the SOS service, every time new data are available.

The implemented Python script uses the Insert Observation XML template file to read a sensor-specific .csv file. Afterwards, it extracts the information to store in databases (time and real values registered) and posts them to the database server. During the process of data insertion, the correct folder structure is highly relevant. The Data folder is organized in the following tree structure: Station category folder $\rightarrow$ sensors folders $\rightarrow$ folder containing the .csv data file of observation. The folder structure, created with the environment_generator.sh script, already foresees this organization to guide the user in the preliminary process of input data-file organization.

```
<?xml version="1.0" encoding="UTF-8"?>
<static>
   <FOI_list>Tramin13er,Unterrain2FuchsangerEppan,
       Gries2NeufeldGreifensteinerweg,Girlan1Lamm,Lana6</FOI_list>
   <FOI_coordinates_list>46.3291770 11.2445070,46.5016200
       11.2504780,46.5017570 11.2802830,46.4558950 11.2795330,46.6165530
        11.1517880</FOI_coordinates_list>
   ...
<observedProperty>
   <short_name>NDVI_UpRed_Avg</short_name>
   <long_name>Normalized Differenced Vegetation Index Up Red</long_name>
   <long_name_def>
       Normalized_Differenced_Vegetation_Index_UpRed_Avg</long_name_def>
   <uom>None</uom>
   <id_sensor_num>1</id_sensor_num>
</observedProperty>
   ...
<coordinate>
   <easting>11.2445070</easting>
   <northing>46.3291770</northing>
   <altitude>239</altitude>
</coordinate>
```

Figure 6. Creation of the configuration file: The XML fragment shows an example of some tags to be filled with the required parameters.

Two versions of the SOS_Insert_Observation.py script have been implemented:

1. Serial approach: The process reads every folder and .csv file of the sensor consequently, one after the other; and
2. Parallel approach: The process analyzes multiple folders simultaneously and inserts multiple observations at the same time, assigning to each of the available computer's cores the first available station category folder.

The different performances of these two versions were compared, in a benchmark test, to test the efficiency and the speed in uploading data. We used 14 input data files, describing one entire year of acquisition of the corresponding 14 station categories. In particular, we tested both versions with respect to an increasing number of input files, checking the time required to complete the process. As shown in Figure 7, it transpires that the choice between the two versions is almost time-independent, when considering less than four station categories. When the number of station categories starts to increase, the parallel approach can be about 80 min faster than the serial.

As a consequence, we suggest using the serial version when uploading data belonging to a station category number less than ten and if it is necessary to debug the code. If the number of the sensors is greater than 10, the parallel approach will save time.

Further technical information about the scripts are available on the dedicated GIT repository (https://gitlab.inf.unibz.it/SOS/SOS-data_upload/tree/master), under a GNU GENERAL PUBLIC LICENSE v3.0 (https://choosealicense.com/licenses/gpl-3.0/).

4.3.2. Metadata

Metadata is structured information that describes data. With well-defined metadata, users should be able to get basic information about data without the need to have knowledge about its entire content.

The availability of metadata in the sensor web can be very useful for discovering the right sensor, together with its spatial and temporal information. When the sensor is described in a standard way, a more expressive representation, advanced access, and formal analysis of its resources can be granted.

Sensor metadata provides descriptions of the properties of the sensors and the related infrastructure; it includes a description of the sensor, platform, time of acquisition, location, and other relevant information to characterize the measurement. After inserting the sensor into the database, the DescribeSensor operation allows the client to retrieve metadata about the sensors, encapsulated by

a SOS instance. It is a mandatory element of the SOS specification. The request of the DescribeSensor operation is very simple, as the only parameter consists of the identifier of the sensor for which a metadata document is requested. The response is a SensorML document [30].

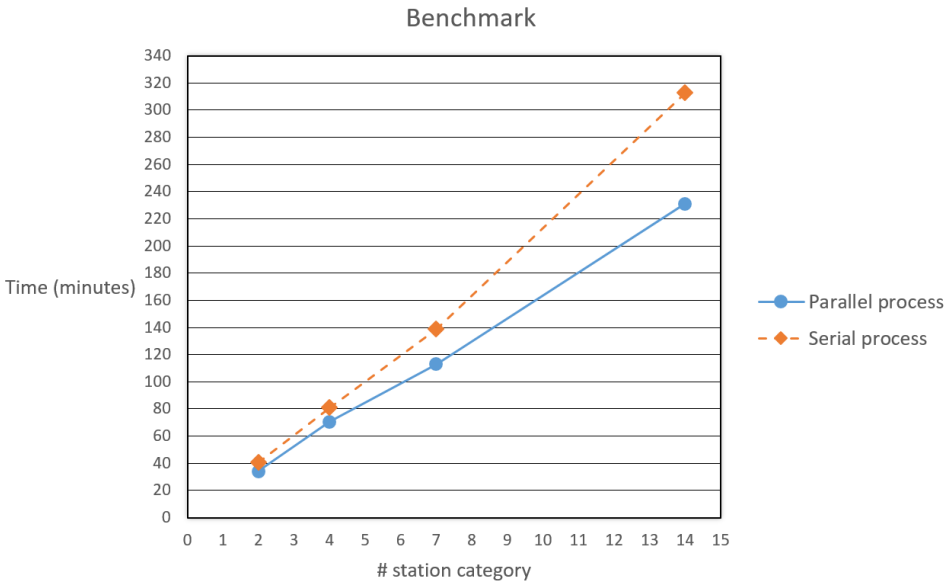


Figure 7. Performance analysis of the serial and parallel versions of the SOS_Insert_Observation.py script.

In addition to this, we created metadata in our GeoNetwork (http://sdi.eurac.edu) catalogue, which implements the Catalogue Service for the Web (CSW) standards. An extension of the SensorML document can be the creation of a metadata file that can be ingested by a Catalogue Service for the Web (CSW), such as GeoNetwork. As Geonetwork is not able to harvest metadata from an SOS service, we use an XSL transformation, starting from the configuration file and a XSLT-style sheet defining the transformation rules. The use of an XSLT is fundamental. XSL stands for Extensible Stylesheet Language, and it is a stylesheet language for XML documents [31]. The main advantage of using an XSLT consists in having the possibility of adapting all the information dealing with the sensor in the corresponding standard to be used or directly customized to the requirements of a specific project; see Figure 8.

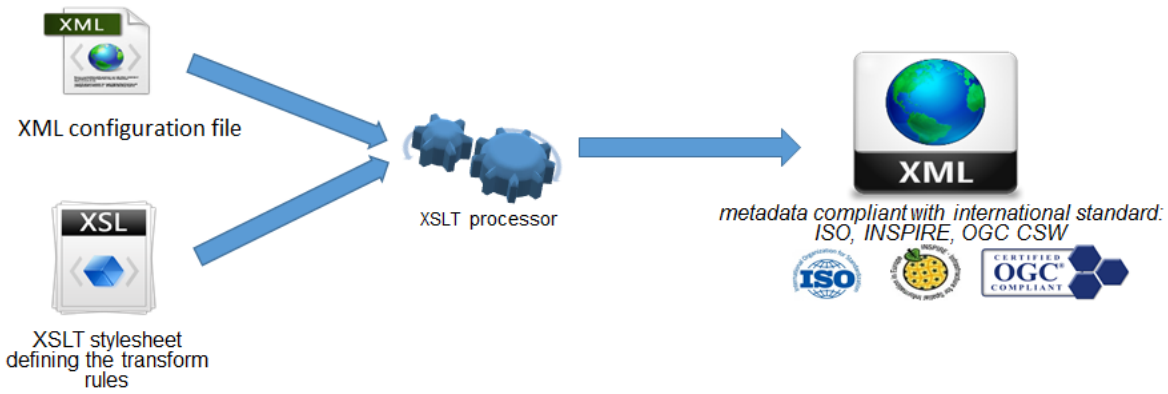


Figure 8. Metadata creation process, by means of the configuration file and an XSL transformation.

In this way, a unique configuration file allows us to have both the XML file for the InsertSensor operation in the SOS service, and an XML file for a CSW catalogue, compliant with the OGC standards as well as INSPIRE and ISO-19139.

4.4. Analysis

The RESTful API, provided by the SOS service, allows access to a complete dataset by means of various clients. We use R because it easily allows the handling of large data tables and the creation of a visualization client. We created the R package MonalisR (https://gitlab.inf.unibz.it/earth_observation_public/MonalisR), which is able to communicate with the SOS API to download data in JSON format, giving a complete overview of the database content; see Figure 9.

It consists of four main functions: Setup, Exploration, Download, and Visualization. To start using this package, it is necessary to set the URL path to interface-API, in order to explore the database.

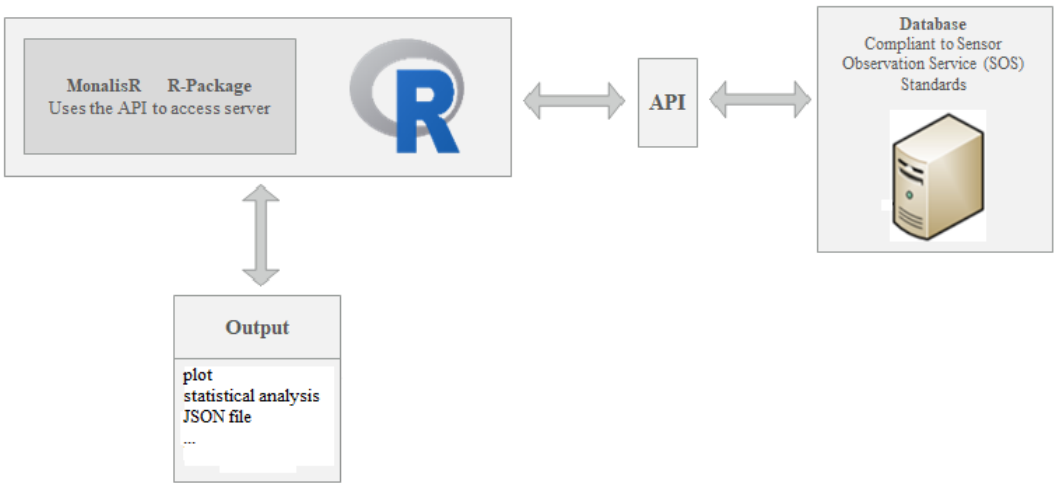


Figure 9. Schematic representation of the workflow to visualize SOS data using R.

Once connected, it is possible to extract data, filtering by time span, FOI, property, or procedure—the MonalisR package creates the URL path to query the SOS service. The visualization functions allow us to plot the spatial location of the measurement stations or to visualize the downloaded time-series by simple lines or box plots. In addition, basic statistics are computed and NaN values are automatically excluded, if present. Users can also directly interact with the RESTful API by creating individual queries and adding their own R scripts for other data analysis. This package can also query any other APIs which provide data in JSON format.

4.5. Share

As a further development, we have created a basic web application which has all the functionalities of the MonalisR package, as listed above. We used Shiny, an R package, to develop this interactive web application. The main advantage of using Shiny is the possibility of creating a customizable Graphical User Interface (GUI), through which the user can perform queries using the available drop-down menus to select the input criteria, as well as an interactive map to geographically locate the ground stations. The web application communicates with the SOS service in real time and allows interactive visualizations of the extracted dataset by selecting different upper tabs, as shown in Figure 10. As this web application is installed on a server, all of the users have access to it by means of a web browser, facilitating popularization of the data. At its most basic level, the web application provides the means to make data available to the researcher in a format that can be more easily handled. While repositories physically hold datasets, a web portal offers a faster way to search and visualize data by creating ad-hoc queries using GUI, without writing codes.

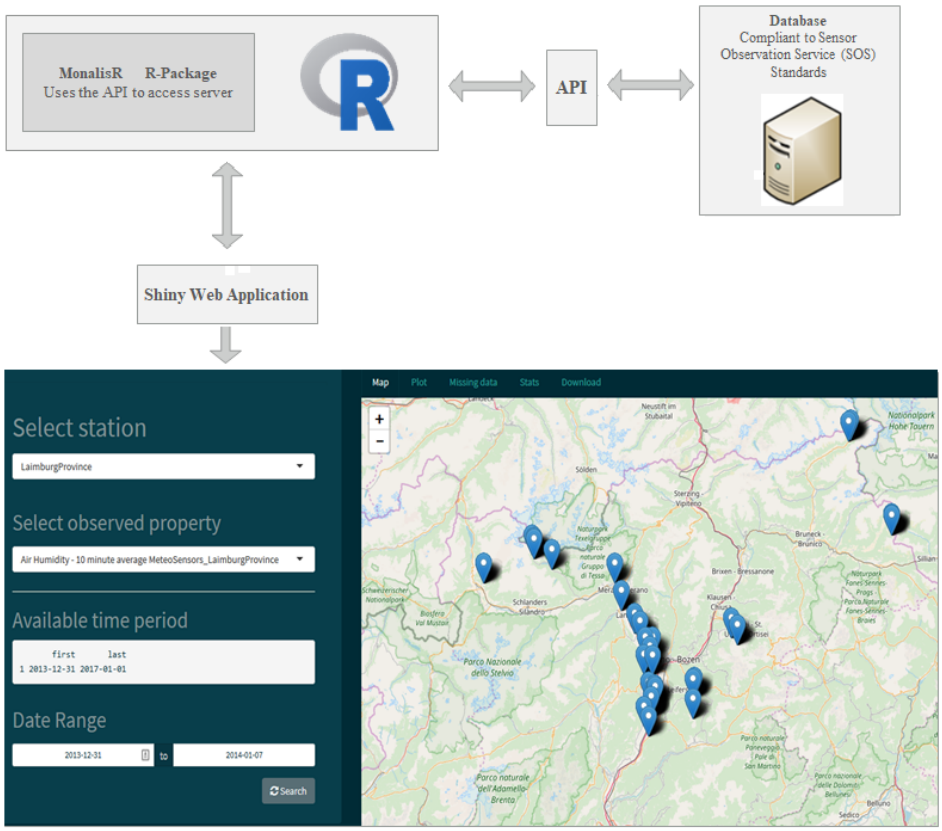


Figure 10. Shiny web application to access data using a Graphical User Interface (GUI).

5. A Case Study: LT(S)ER Matsch-Mazia

5.1. Study Area

The Long-Term Ecosystem Research (LTER) Europe (http://www.lter-europe.net/) is an initiative which aims to study how ecosystems react, in the long term, when influenced by different factors (belonging to different fields) at different scales (local, regional, continental, and global). Different countries are involved in this project, providing research analysis dealing with their typical environmental typologies. In the case of Italy, freshwater and marine ecosystems have been analyzed in the LTER italian network (http://www.lteritalia.it/).

The Institute for Alpine Environment, with the technical collaboration of the Institute for Earth Observation at Eurac Research in Bolzano, Italy is involved in this network, within the framework of the initiative LT(S)ER Matsch-Mazia ("S" stands for Social) (http://www.lteritalia.it/macrositi/it25). At present, this network has collected about 80 million observations from 30 ground stations, corresponding to 500 sensors spread over the Val Mazia-Matsch area in South Tyrol, as shown in Figure 2b. This valley is characterized by a low anthropic impact, relative dryness, and its hydrographic basin does not extend very far. In general, Val Venosta is the reference arid valley, but it is too large. The position of the ground stations reflects the need to cover a range of altitudes (between 1000–2700 m) where typical mountain agriculture is developed and the collected data will drive the studies to correlate how the environmental conditions affect functional trait variability within communities and, thus, shape ecosystem properties ([32,33]).

5.2. Sensors

The LTER Stations in Val Mazia-Matsch are individually equipped with diverse sensors, detecting various key environmental parameters. Environmental parameters measured and transferred automatically by the LTER stations are the surface and soil temperatures, wind fluctuations

and speed, relative humidity, soil water content, and potential of precipitation measurements. Furthermore, sensors for the analysis of the spectral characteristics near the stations have been installed. These detect variables such as the total solar radiation and the photo-synthetically active radiance (PAR), or vegetation indexes such as the Normalized Difference Vegetation Index (NDVI) and the Photochemical Reflectance Index (PRI). In Table 1 list of the sensors used in the LT(S)ER initiative is shown. For the sake of brevity, this list is not fully representative. For complete information, please refer to the LT(S)ER website (http://www.lteritalia.it/macrositi/it25).

Table 1. List of the main sensors used in the LT(S)ER initiative.

LT(S)ER Sensors		
Manufacturer	***Type***	***Model Factsheet***
Apogee	Infrared Radiometer	SI111
Apogee	Pyranometer	SP110
Apogee	Quantum Sensor (PAR)	SQ110
Campbell Scientific	Temperature Sensor	107
Campbell Scientific	Data Logger	CR1000
Campbell Scientific	Soil Moisture	CS655
Campbell Scientific	Temperature and Relative Humidity	HC2S3
Campbell Scientific	Temperature and Relative Humidity	HMP45C-L
Campbell Scientific	Acoustic Distance Sensor	SR50AT-L
Campbell Scientific	Average Soil Thermocouple Probe, 6–8 cm	TCAV-L
Decagon	Large Volume Soil Moisture Sensor	10HS
Decagon	Leaf Moisture Sensor	LWS
Decagon	Soil Water Potential Sensor	MP6S
Decagon	Spectral Reflectance Sensors (NDVI and PRI measurements)	NDVIPRI
Decagon	Spectral Reflectance Sensors (NDVI and PRI measurements)	10HS
Decagon	Spectral Reflectance Sensors (NDVI and PRI measurements)	NiHemispherical
Decagon	Spectral Reflectance Sensors (NDVI and PRI measurements)	NrFIeldStop
Decagon	Spectral Reflectance Sensors (NDVI and PRI measurements)	PiHemispherical
Gill	Wind Sensor	Windsonic
Huskeflux	Heat Flux Sensors	HFP01
Huskeflux	Surface Fluxes	NR01
Onset	Solar Radiation Sensor Pyranometer	SLIBM003
Onset	Rainfall Smart Sensor (Bucket)	SRGBM002
Ott	Precipitation Detector	Pluvio2

5.3. Results

In this case study, SOS represents the common platform for sharing the collected data using SWE standards. As described before, it is fundamental to create the necessary configuration files for describing the entire network of ground stations. For this project, 30 configuration files are necessary. Once they are ready, the process to insert the corresponding sensors into the SOS can start and the remaining steps are completely automatic. The system is able to verify if new input files arrive in the dedicated data folder. If new data are available, the corresponding Insert Observation file is created and automatically inserted in the database. Considering the frequency of ground measurements, we set this automatic and customizable routine to start every fifteen minutes, allowing a near real-time database update, as well as analysis-ready data for researchers.

Due to the data policy of the LT(S)ER initiative, we developed two different web applications: Public and internal. Both are based on the same SOS, 52°North, and regularly updated with the incoming observations. The public one contains only a restricted subset of the data that can be visualized on the dedicated LT(S)ER website (http://sdi.eurac.edu/lter_sos/static/client/helgoland/index.html); see Figure 11.

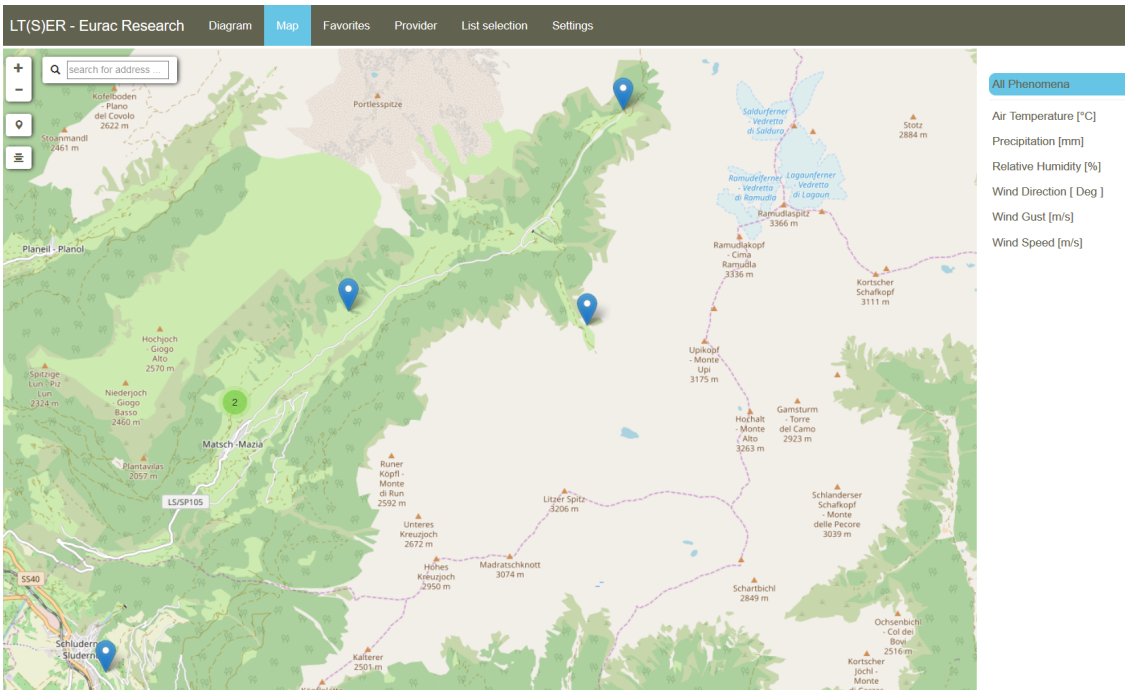


Figure 11. Public LT(S)ER web application, view of the Map section.

Here, the user can have a first look at the data and compare the different trends in the latest collected data. If further data are needed, please refer to the contact page in the website. The internal one contains the full dataset, and is accessible only by authorized researchers involved in the project.

The creation of such an infrastructure can be helpful for research purposes. In fact, once the database is filled with the observations it is possible to query for specific values, such as NDVI, and to extract time series belonging to different sensors which are ready to be analyzed [34].

6. Conclusions

This paper describes our solution to automatically upload data, in near real-time, in any standard SOS implementation. In particular, we jointly used Python scripts, R codes, and the SOS service 52°North, which implements the SWE OGC standards. The advantage of this solution consists in having a lightweight system requiring only minor actions from the user. These interactions comprise of the creation of a configuration file describing all the necessary information about the sensors, and to format the input data files following a few requirements. Consequently, once the sensors have been registered in the SOS service, the system is ready to automatically check for new data and to insert them using Python scripts.

Another important result is the possibility of simplifying the sensor metadata creation, compliant with the OGC standards (as well as the INSPIRE and ISO-19139 standards). The development of an R client to dynamically interact with the database, as well as a Shiny web interface, represents another further improvement important for data visualization and analysis.

This solution has been applied in the framework of the LT(S)ER Matsch-Mazia initiative, producing significant time savings in managing about 80 million observations. This, in turn, facilitates the sharing of collected time-series data. Some of the most significant advantages that we have achieved are:

- High performance, thanks to the availability of a parallelized process which reduces the time for inserting new measurements in a SOS implementation when the number of station categories is greater than ten.
- An OGC standard-compliant methodology, allowing the reuse of data outside the research context and, thus, avoiding duplication.

- A flexible methodology, which can be applied to upload data into any SOS implementation.
- Open source and custom software: Our python scripts can be improved, with respect to a more dynamic way to handle the parallelization processes.
- Collected data are available in near real-time and accessible through a web application, offering a faster way to search through and visualize data by creating ad-hoc queries using GUI without writing codes.
- Source code available in a dedicated GIT repository.

Future work will focus on the optimization of the creation of the configuration file by developing a GUI. We plan to develop a driver to connect the OpenEO API (http://openeo.org/) to the SOS API, in order to access different services from different clients (such as R, Python, and Javascript) for big earth observation cloud back-ends, in a simple and unified way.

Author Contributions: Conceptualization, Bartolomeo Ventura and Andrea Vianello; methodology, Bartolomeo Ventura; software, Bartolomeo Ventura, Mattia Rossi, and Daniel Frisinghelli; validation, Bartolomeo Ventura and Andrea Vianello; formal analysis, Bartolomeo Ventura and Andrea Vianello; investigation, Bartolomeo Ventura; resources, Roberto Monsorno and Armin Costa; data curation, Bartolomeo Ventura; writing—original draft preparation, Bartolomeo Ventura; writing—review and editing, Bartolomeo Ventura, Andrea Vianello, and Mattia Rossi; visualization, Bartolomeo Ventura; supervision, Roberto Monsorno.

Funding: This research has been funded by the Autonomous Province of Bolzano, Innovation, Research and University Department, through the projects *Environmental Sensing LAB at NOI technology Park, LT(S)ER site Matsch/Mazia (I)*, *MONALISA*.

Acknowledgments: The authors would like to thank our colleagues Notarnicola Claudia, Petitta Marcello, Abraham Meja-Aguilar, and Della Chiesa Stefano for comments that greatly improved the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

OGC	Open Geospatial Consortium
SWE	Sensor Web Enablement
API	Application Programming Interface
HCI	Human-Computer Interaction
CC BY	Creative Commons Attribution only
O&M	Observations&Measurements
SensorML	Sensor Model Language
SOS	Sensor Observation Service
FOI	Feature Of Interest
DRP	Digital Repeat Photography
NDVI	Normalized Differenced Vegetation Index
DB	Database
XML	eXtensible Markup Language
XSLT	Extensible Stylesheet Language
CSW	Catalogue Service for the Web
JSON	JavaScript Object Notation
RDBMS	Relational DataBase Management System
GUI	Graphical User Interface
LTER	Long Term Ecological Research
LT(S)ER	Long Term Socio-Ecological Research

References

1. Zhang, Z.; Glaser, S.D.; Bales, R.C.; Conklin, M.; Rice, R.; Marks, D.G. Technical report: The design and evaluation of a basin-scale wireless sensor network for mountain hydrology. *Water Resour. Res.* **2017**, *53*, 4487–4498. [CrossRef]

2. Partescano, E.; Brosich, A.; Lipizer, M.; Cardin, V.; Giorgetti, A. Open geospatial data, softw. stand. *Open Geospatial Data Softw. Stand.* **2017**, *2*, 22. [CrossRef]
3. Jirka, S.; Bröring, A.; Stasch, C. Applying OGC Sensor Web Enablement to Risk Monitoring and Disaster Management. In Proceedings of the GSDI 11 World Conference, Workshop on Sensorweb Enablement: Strengthening the SDI, Rotterdam, The Netherlands, June 2009. Available online: http://www.gsdi.org/gsdiconf/gsdi11/papers/pdf/96.pdf (accessed on 25 February 2011).
4. Kamel Boulos, M.N.; Resch, B.; Crowley, D.N.; Breslin, J.G.; Sohn, G.; Burtner, R.; Pike, W.A.; Jezierski, E.; Chuang, K.-Y.S. Crowdsourcing, citizen sensing and sensor web technologies for public and environmental health surveillance and crisis management: Trends, OGC standards and application examples. *Int. J. Health Geogr.* **2011**, *10*, 67. [CrossRef] [PubMed]
5. Bojinski, S.; Verstraete, M.; Peterson, T.; Richter, C.; Simmons, A.; Zemp, M. The concept of essential climate variables in support of climate research, applications, and policy. *Bull. Am. Meteorol. Soc.* **2014**, *95*, 1431–1443. [CrossRef]
6. Delin, K.; Jackson, S.; Some, R. Sensor Webs. *NASA Tech. Briefs* **1999**, *23*, 90.
7. Delin, K. The Sensor Web: A Macro—Instrument for Coordinated Sensing. *Sensors* **2001**, *2*, 270–285. [CrossRef]
8. Cox, S. *Observations and Measurements—Part 1—Observation Schema*, version 1.0; Open Geospatial Consortium: Wayland, MA, USA, 2007; p. 07-022r1.
9. Cox, S. *Observations and Measurements—Part 2—Observation Schema*, version 1.0; Open Geospatial Consortium: Wayland, MA, USA, 2007; p. 07-022r3.
10. Cox, S. *OGC Implementation Specification: Observations and Measurements (O&M)—XML Implementation 2.0*; Open Geospatial Consortium: Wayland, MA, USA, 2011; p. 10-025r1.
11. Open Geospatial Consortium 2011. Observations and Measurements—XML Implementation. Available online: http://portal.opengeospatial.org/files/?artifact_id=41510 (accessed on 1 January 2019).
12. Open Geospatial Consortium 2011. OpenGIS Sensor Model Language (SensorML) Implementation Specification. Available online: http://portal.opengeospatial.org/files/?artifact_id=21273 (accessed on 1 January 2019).
13. Open Geospatial Consortium 2011. OGC Sensor Observation Service Interface Standard. Available online: https://portal.opengeospatial.org/files/?artifact_id=47599 (accessed on 1 January 2019).
14. Cannata, M.; Antonovic, M.; Molinari, M.; Pozzoni, M. istSOS, a new sensor observation management system: Software architecture and a real-case application for flood protection. *Geomat. Nat. Hazards Risk* **2015**, *6*, 635–650. [CrossRef]
15. Lanucara, S.; Zilioli, M.; Oggioni, A.; Carrara, P. GET-IT, a software suite for easy, interoperable sharing of ecological data in the Long Term Ecological Research Network. In Proceedings of the Conference: EnviroInfo, Luxembourg, 13–15 September 2017.
16. Lanucara, S.; Carrara, P.; Oggioni, A.; Rogora, M.; Kamburska, L.; Rossetti, G. Exploiting observations and measurement data standard for distributed LTER-Italy freshwater sites. Water quality issues. *PeerJ Prepr.* **2016**, *4*, e2233v2.
17. Martínez, E.; Toma, D.M.; Jirka, S.; Del Río, J. Middleware for Plug and Play Integration of Heterogeneous Sensor Resources into the Sensor Web. *Sensors* **2017**, *17*, 2923. [CrossRef] [PubMed]
18. Jirka, S.; Bröring, A.; Kjeld, P.; Maidens, J.; Wytzisk, A. A Lightweight Ap-proach for the Sensor Observation Service to Share Environmental Data across Europe. *Trans. GIS* **2012**, *16*, 293–312. [CrossRef]
19. Bychkov, I.V.; Ruzhnikov, G.M.; Paramonov, V.V.; Shumilov, A.S.; Fedorov, R.K.; Levi, K.G.; Demberel, S. Infrastructural approach and geospatial data processing services in the tasks of territorial development management. *Earth Environ. Sci.* **2018**, *190*, 012048. [CrossRef]
20. Bishop, W.; Grubesic, T.H. Data Lifecycle. In *Geographic Information: Organization, Access, and Use*; Springer: Berlin, Germany, 2016; pp. 169–186.
21. Hook, L.A.; Vannan, S.K.S.; Beaty, T.W.; Cook, R.B.; Wilson, B.E. Best Practices for Preparing Environmental Data Sets to Share and Archive. In *Oak Ridge National Laboratory Distributed Active Archive Center*; Environmental Sciences Division, Oak Ridge National Laboratory: Oak Ridge, TN, USA, 2010.
22. Jones, A.S.; Horsburgh, J.S.; Reeder, S.L.; Ramírez, M.; Caraballo, J. A data management and publication workflow for a large-scale, heterogeneous sensor network. *Environ. Monit. Assess.* **2015**, *187*, 348–367. [CrossRef]

23. Benyon, D.; Turner, P.; Turner, S. *Designing Interactive Systems: People, Activities, Contexts, Technologies*; Pearson Education Limited: Essex, UK, 2005; p. 52.
24. Nielsen, J. Usability101: IntroductiontoUsability. *Jakob Nielsen'sAlertbox*; 21 May 2011. Available online: http://www.useit.com/alertbox/20030825.html (accessed on 1 January 2019).
25. Preece, J.; Rogers, Y.; Benyon, D.; Holland, S.; Carey, T. *Human Computer Interaction*; Addison-Wesley: Wokingham, UK, 1994; p. 31.
26. Toda, M.; Richardson, A.D. Estimation of plant area index and phenological transition dates from digital repeat photography and radiometric approaches in a hardwood forest in the Northeastern United States. *Agric. For. Meteorol.* **2018**, *249*, 457–466. [CrossRef]
27. Sidi, F.; Shariat Panahy, P.H.; Affendey, L.S.; Jabar, M.A.; Ibrahim, H.; Mustapha, A. Data quality: A survey of data quality dimensions. In Proceedings of theInternational Conference on Information Retrieval and Knowledge Management, Kuala Lumpur, Malaysia, 13–15 March 2012; pp. 300–304.
28. Wand, Y.; Wang, R.Y. Anchoring data quality dimensions in ontological foundations. *Commun. ACM* **1996**, *39*, 86–96. [CrossRef]
29. Pfeiffer, T.; Wenk, A. *PostgreSQL: Das Praxisbuch*; Galileo-Press: Bonn, Germany, 2010.
30. Open Geospatial Consortium. SensorML: Model and XML Encoding Standard. 2014. Available online: http://www.opengis.net/doc/IS/SensorML/2.0 (accessed on 1 January 2019).
31. Clark, J. (Ed.) *XSL Transformations (XSLT) Version 1.0*; World Wide Web Consortium: Cambridge, MA, USA, 1999.
32. Niedrist, G.; Tasser, E.; Bertoldi, G.; Della Chiesa, S.; Obojes, N.; Egarter-Vigl, L.; Tappeiner, U. Down to future: Transplanted mountain meadows react with increasing phytomass or shifting species composition. *Flora* **2016**, *224*, 172–182. [CrossRef]
33. Fontana, V.; Kohler, M.; Niedrist, G.; Bahn, M.; Tappeiner, U.; Frenck, G. Decomposing the land-use specific response of plant functional traits along environmental gradients. *Sci. Total Environ.* **2017**, *599–600*, 750–759. [CrossRef] [PubMed]
34. Rossi, M.; Niedrist, G.; Asam, S.; Tonon, G.; Tomelleri, E.; Zebisch, M. A Comparison of the Signal from Diverse Optical Sensors for Monitoring Alpine Grassland Dynamics. *Remote Sens.* **2019**, *11*, 296. [CrossRef]