Free and Open Source Software for Geospatial (FOSS4G) Conference Proceedings

Volume 15 *Seoul, South Korea* Article 39

2015

OPENSENSEMAP - A Citizen Science Platform For Publishing And Exploring Sensor Data as Open Data

Matthias Pfeil
Institute for Geoinformatics, University of Münster

Thomas Bartoschek
University of Münster, Germany

Jan Alexander Wirwahn
Institute for Geoinformatics, University of Münster

AT4-01

OPENSENSEMAP - A CITIZEN SCIENCE PLATFORM FOR PUBLISHING AND EXPLORING SENSOR DATA AS OPEN DATA

Matthias Pfeil, Thomas Bartoschek and Jan Alexander Wirwahn

Institute for Geoinformatics, University of Münster
Heisenbergstr. 2, 48149 Münster, Germany
Email: matthias.pfeil | bartoschek | jan.wirwahn @uni-muenster.de

ABSTRACT

A plethora of citizen science sensor platforms for dierent use-cases already exist. They provide cheap, precongured, plug and playable hardware and software solutions. Using data from multiple platforms and resources can be a challenging task in respect of discovering, exploring, downloading and converting. This work aims to implement a one-stop-shop for sensor data that tries to tackle these problems. Therefore a basic data schema capable of metadata is established that allows publishing generic sensor platforms and sensor data. For exploration a web platform is implemented based on common web standards. This work describes the requirements, design and implementation of the OpenSenseMap. OpenSenseMap is following the clientserver model and its complete application stack is implemented in JavaScript. It describes the basic components and the underlying data model. All components are using open-source technology and are published under an open-source license too. The RESTful service is the heart of OpenSenseMap and implements the data publish and data retrieval functions. Developers may use it to build custom applications around OpenSenseMap or integrate OpenSenseMap into existing projects. The NodeJS application runs its own web server and implements the different HTTP request types. The OpenSenseMap concept is being evaluated with a technical and a user survey.

1. INTRODUCTION

Citizen science is often called "public participation in scientific research" (Hand, 2010) and describes the engagement by non-professional scientists in collecting and analysing data, decision making, developing technology and publication of these on a voluntary basis. The idea of involving citizen in scientific projects is not new. Two examples are the Christmas Bird Count (Silvertown, 2009) by the National Audubon Society in the USA, which started back in 1900 and is still ongoing, and the Galaxy Zoo project, which identifies and classifies galaxies on sky images taken at the Sloan Digital Sky Survey. An always discussed concern about citizen science is data quality (Wiggins et al., 2011). Due to limited knowledge of the volunteers and missing or questionable metadata, scientists often characterise data collected by citizens as valueless. In the beginning citizen science was connected with activities where humans were used as sensors (Goodchild, 2007). Nevertheless, the scientific community is becoming more and more interested in citizen science today and citizen science projects are part of complex research projects (Conrad & Hilchey, 2011); i.e. ambient environment monitoring. We think two reasons boosted this trend. First, citizens are getting more interested in their environment and its effects on daily life. Also many popular citizen science projects are related to ambient environment monitoring and topics around smart cities; i.e. the AirQualityEgg project (http://airqualityegg.com) and the SmartCitizen project (https://smartcitizen.me). Second, the vast development of technology makes it easy and aordable to build the necessary tools for a citizen science project. Microcontroller platforms like Arduino or tessel.io and the quality and price of sensors make it nowadays easy to build sensor platforms, which are easy to use and affordable in price. By using open-source hardware and software, it is easier - especially for traditional non-technical research communities - to create custom sensor platforms. Furthermore, the web makes it easier to share and discuss data at global scale.

This work is closely related to SenseBox, which is a generic sensor platform for the Web Of Things (WoT) proposed by Bröring et al. (2012). The SenseBox is a tiny, lightweight and low-cost sensor platform being developed at the Institute for Geoinformatics, University of Münster. It is an ongoing citizen science project and enables citizen scientists to run private sensor stations for environmental observations like temperature and wind speed. Up to now, SenseBoxes are rather isolated: it is up to its owner to share the collected data with the world. Sharing the collected data for scientific purposes is a challenging task especially for non-professional scientists as the data needs to be structured and enriched with additional metadata. Structuring and annotating the collected data is not only a prerequisite to make the data usable for scientific research but also for making the data easily explorable for the involved citizen scientists. An active and involved community is one key for a successful citizen science project.

The main objective of this work is the design and implementation of a web application for sharing the data collected by sensor platforms like SenseBox; we call it OpenSenseMap. OpenSenseMap defines an extensible but yet tiny and lightweight encoding standard for spatio-temporal sensor data and metadata. With its RESTful API, OpenSenseMap fits the characteristics of WoT. These key features and its openness make it easy to adopt Open-SenseMap to many other sensor platforms and citizen science projects.

2. RELATED WORK

2.1 Internet of Things and Web of Things

The term Internet of Things (IoT) describes the mash-up of real-world objects with the virtualworld and was coined by David Brook (Kopetz, 2011 & Uckelmann et al. 2011]. Everyday life objects (things) like household devices are equipped with embedded computing systems. These embedded computing systems make things capable of sensing, processing and connectivity and are the building blocks of IoT (Kortuem et al., 2010) Most IoT services are custom implementations and require expertise for integrating and interoperating "which prevents users from creating small tactical, ad-hoc applications using sensor networks" (Guinard & Trifa, 2009).

To tackle this the next evolutionary step of IoT is theWeb of Things (WoT) (Gurnard et al., 2011). TheWoT paradigm proposes to use well-known web standards to realize IoT applications. Using web standards has the advantage that all things speak the same language and makes it easy to integrate them into the existing web. Each WoT enabled device is identifiable by its own unique Uniform Resource Identifier (URI).

A typical used web standard for communication is REST. REST stands for Representational State Transfer and is a common architectural pattern for building web applications (Fielding & Taylor, 2002). This approach uses the standardized Hypertext Transfer Protocol (HTTP) to transport data between a client and server. HTTP handles the communication by listening for requests and answering with responses. To identify the target of a request, REST uses the aforementioned URIs. The whole communication is stateless; that is, each request contains all necessary information to respond to a request. By using these standards all “things” speak the same language and make the custom IoT solutions redundant. FurthermoreWoT is more user-friendly and can be easily integrated into web projects.

2.2 Sensor networks

A sensor network (Faludi, 2010) typically contains two to many sensor nodes and uses a network to exchange data. Each sensor node is able of measuring distinct characteristics of its ambient environment and publishes its measurements inside the network. The amount of connected sensors is theoretically unlimited but limited in real-world applications by the technical specifications of the used hard- and software. According to Soharby et al. (2007) typical applications of sensor networks are data collection, monitoring or surveillance. Most sensor networks contain a logical entity called processing node or gateway (cf. Figure 1). The gateway is responsible to aggregate the measurements of the network’s sensor nodes and provide them to interested clients. A client may be a real user or another software platform. In general a gateway is more powerful than the sensor nodes.

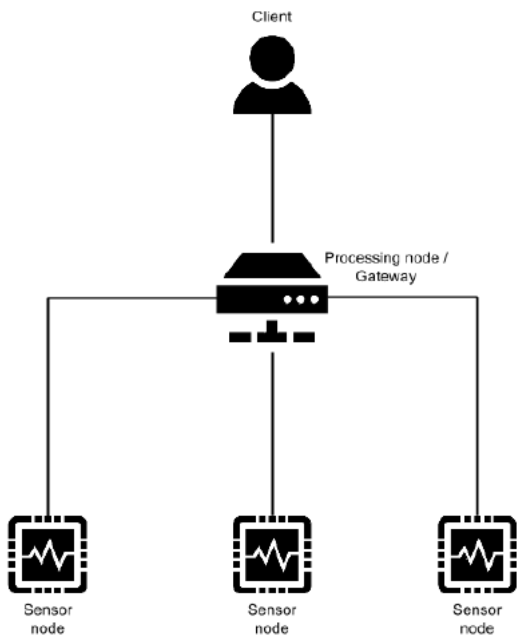


Figure 1. Sensor network with the central processing node / gateway and connected sensor platforms

Two approaches to realize a gateway are the Sensor Web Enablement (SWE) and Sensor Web portals; both will be described more detailed in the following sections.

2.3 Sensor Web Enablement

The Sensor Web Enablement (SWE) is a standardized framework developed by the Open Geospatial Consortium (OGC). SWE enables developers to make all types of sensors and their data discoverable, accessible and useable by standardized web technologies. The

SWE framework proposes the following characteristics for sensors (Bröring et al., 2011):

- sensors are annotated with a geo-location
- sensors are connected to the web
- sensors have adequate metadata
- sensors are readable remotely
- sensors may be controllable remotely

The SWE framework includes different standards to realize the proposed characteristics of sensors. The most relevant standards will be described in the following sections.

2.3.1 *Sensor Observation Service*

The Sensor Observation Service (SOS) is a standardized web service interface and represents the gateway of a SWE based sensor network. It provides services to store, request and filter sensor data. Furthermore, the SOS offers management capabilities for the sensor network itself and the connected sensors. This includes functionalities to register new sensors and to add new and delete existing observations.

The SOS builds upon further standards dened by the OGC and International Organization for Standardization (ISO) and provides a comprehensive variety of interoperable capabilities to interconnect in-situ or dynamic sensor platforms (Florence, 2012).

2.3.2 *Sensor Observation Service*

The Observation & Measurement (O&M) model is an essential component of the SOS. O&M is a standard to represent and exchange domain independent and spatio-temporal observations and measurements of a sensor. An observation is an act that takes place at a certain time and geo-location (spatio-temporal) and produces a measurement for the observed feature of interest.

2.3.3 *Sensor Model Language*

The SWE calls for sensors to have adequate metadata to understand how the sensor works and generates observations. To describe sensor metadata the OGC has dened the Sensor Model Language (SensorML). With SensorML it is possible to describe “geometric, dynamic, and observational characteristics of sensors and sensor systems” (OGC, 2007). Beside describing sensors, SensorML provides a way to apply additional sensor information and process descriptions. These information are a prerequisite to discover the sensor network and its sensors and the semantics of the observations. SensorML is encoded as XML.

2.4 Sensor Web Portals

Bröring et al. (2012) claim that so-called sensor web portals are emerging. These platforms act like the gateway of a sensor network and allow users to register sensors and publish the collected data. We currently see two kinds of sensor web portals: generic sensor web portals, which provide the architecture and infrastructure to connect any sensor to the web, and specialized sensor web portals, which focus on one specific use case. Usually they offer an API for registering sensors and publishing sensor data.

2.4.1 *OpenSensors.io*

OpenSensors.io is an IoT platform supported by the Open Data Institute (ODI)8. The ODI promotes the evolution of an open data culture. Open data empowers people to action on problems that concern themselves. Furthermore open data creates and spreads knowledge about local and global issues.

OpenSensors.io usesMQTT, which is a publish-subscribe based lightweight message protocol. Data is being published on OpenSensors.io in so-called topics. One can think of topics as the equivalent to Twitter hashtags. Devices publish data on OpenSensors.io by sending a MQTT message to a topic. Those interested may subscribe to a topic. Topics are the only (and rather limited) metadata supported by OpenSensors.io.

The user has full control over the gathered data and can restrict the data privacy. Publisher gain full public domain license over the published data and other users can make use of the data by attributing the work with the ODC Open Database License (ODbL) (http://opendatacommons.org/licenses/odbl/summary/)

OpenSensors.io is an example of a generic sensor web portal. It is a commercial sensor web portal but offers a free personal plan with a limited number of devices and storage.

2.4.2 *Xively*

Xively (http://xively.com) is also an example of a generic sensorweb portal. It is comparable to OpenSensors.io but offers extended functionality regarding connectivity and metadata handling. Additionally to MQTT, Xively supports REST and Sockets. Furthermore, it is possible to store and process metadata like geo-location on Xively. A popular citizen science project hosted on Xively is the AirQualityEgg10 project.

Primarily Xively is a commercial sensor web portal but offers a free plan for personal use only with limited capabilities regarding exploring and sharing data.

2.4.2 *Smartcitizen.me*

Smartcitizen.me is the sensor web platform of the Smart Citizen project and focusses on urban ambient monitoring. The Smart Citizen project offers the so-called Smart Citizen Kit, which is an Arduino based sensor platform that can be purchased. It allows volunteers to capture data on temperature, humidity, light, sound, carbon monoxide and nitrogen dioxide.

The collected data may be explored by everyone but only owners of a Smart Citizen Kit can publish data on the Smartcitizien.me Sensor Web portal. Clearly, the SmartCitizen.me sensor web portal is an example of a specialized sensor web portal. Currently 1003 sensor platforms with 5015 sensors are online.

3. PRINCIPLES AND REQUIREMENTS

OpenSenseMap is meant to be a platform for professionals and non-professionals where it is possible to store, publish and explore sensor data. Therefore the requirements are split up into two sections. Section 3.1 focuses on the technical requirements for a gateway in a sensor network. Section 3.2 focuses on the non-professionals and what is needed to include citizens regarding design and interaction.

3.1 Technical Requirements

Steenkamp et al. (2009) state that a main task of a gateway in a sensor network is to connect contradictory networks. In this case that means to connect the sensor network and the client. Registered sensor platforms shall be able to publish their collected data on Open-SenseMap. Users shall be able to register new sensor platforms and to administer existing sensor platforms. Administration includes deleting sensor platforms and modifying the setup of a sensor platform. Table 1 lists all use cases for the user. This leads to the first requirement: users shall be able to register new sensor platforms and administer already created sensor platforms.

Table 1. List of interactions between user and OpenSenseMap

Use Case	Description
register new sensor platform	A new sensor platform is registered including metadata and connected sensors
unregister sensor platform	Remove a sensor platform from the sensor network
alter sensor platform properties	Alter the provided metadata for the sensor platform
add sensor	Add a new sensor to a sensor platform
remove sensor	Remove a sensor from a sensor platform
alter sensor properties	Alter the provided metadata for the sensor
get data	Retrieve data from sensor platform and sensors

Currently a lot of citizen science projects implement own infrastructures and are bound to a specific topic (Wiggins & Crouton, 2011). Approaches to build platforms to provide infrastructures for generic citizen science projects as described by Dickinson et al. (2012) exist but with a different focus. They focus on being a one-stop-shop for discovering citizen science projects and to get information about the project but not offering a standardized way to connect and publish sensor platforms and its data. Bröring et al. (2012) propose to separate the sensor platform logic from the gateway logic. By applying the Web of Things paradigm to the gateway all sensor platforms of the network get discoverable by other systems and users. Unfortunately, the unique addresses of the sensor platforms does not help to understand the underlying dataset, because important metadata information like the unit of measurement are not encoded in the address. Thus it is necessary to dene a data model that can handle the actual data and is capable of metadata. This puts two important requirements on the data model. It must be flexible to handle any kind of sensor data but also must be lightweight to allow low-cost and not well equipped microcontrollers to be part of the sensor network, leading to the requirement of a data model definition capable of storing actual data and metadata.

The gateway is the central entity of a sensor network. It should be connected to the internet all time and handles the requests of the sensor platforms (when publishing their collected data) and of the users (when exploring the collected data and/or performing administrative tasks). Therefore a reliable, robust and high-performance back-end is required. The back-end must handle bad requests without causing internal server errors and respond with appropriate error messages. Furthermore an authentication method is needed to ensure only registered sensor platforms are allowed to post data to the platform. Though the back-end shall be reliable, robust and performant.

All gathered data must be stored on the gateway. Therefore a reasonable database which is capable to handle geo-data and oer geospatial features for indexing and querying is needed. Additionally it must support a flexible schema. Considering the possible amount of gathered data, the size of the stored objects should not take up to much space. The database shall be capable of a flexible schema, small storage size and geo-data.

3.2 Citizen Science & Design

Running a successful citizen science project is not that simple. In the past, citizens were often allowed to contribute only to selected phases of a research project. Current evaluations of citizen science projects have shown that the stage of inquiry is one key for a successful citizen science project (Wiggins & Crowston, 2012). It should be the goal of every citizen science project to establish an active and involved community. In addition to that, Newman et al. (2012) recommend to build citizen science projects upon open-source solutions and encourage to pick-up the idea of open data. These recommendations lead to the requirement that the collected data shall be published as open data and the implementation shall use open-source technologies. The collected data shall be published as open data and open-source technologies should be used.

A typical characteristic of citizen science is often the heterogeneity of the participants. This is often an advantage but also a challenge. In this case, the Sensor Web portal will be used by users with different experience levels of web applications. Nielsen (1994) defines software usability as the sum of learnability, efficiency, memorability, errors, and satisfaction. This implies that software must be easy to learn, efficient to use, easy to remember, should not allow to make errors and should be fun to use. In case of OpenSenseMap, the user shall be guided through the registration of a new sensor platform and information on usage shall be provided throughout the Sensor Web portal. OpenSenseMap shall focus on usability by non-professionals.

To make the collected data available to a wide audience and encourage the participation of interested citizens, OpenSenseMap shall be useable free of charge. Free of charge means not only that no money is charged but also that OpenSenseMap is usable without prior registration.

4. OPENSENSEMAP - DESIGN AND IMPLEMENTATION

This chapter focuses on the design and implementation of OpenSenseMap. It describes the basic components and the underlying data model. OpenSenseMap is following the clientserver model (Berson, 1992) and its complete application stack is implemented in JavaScript (cf. Figure 2). All source code is published on GitHub. The server-side source code is available in the OpenSenseMap-API repository12 and the client-side source code is available in the OpenSenseMap repository13. All components are using open-source technology and are published under an open-source license too. By this all interested citizens are invited to take the current state of the project and to contribute in either implementing new features, fixing bugs or suggesting and discussing future development of OpenSenseMap.

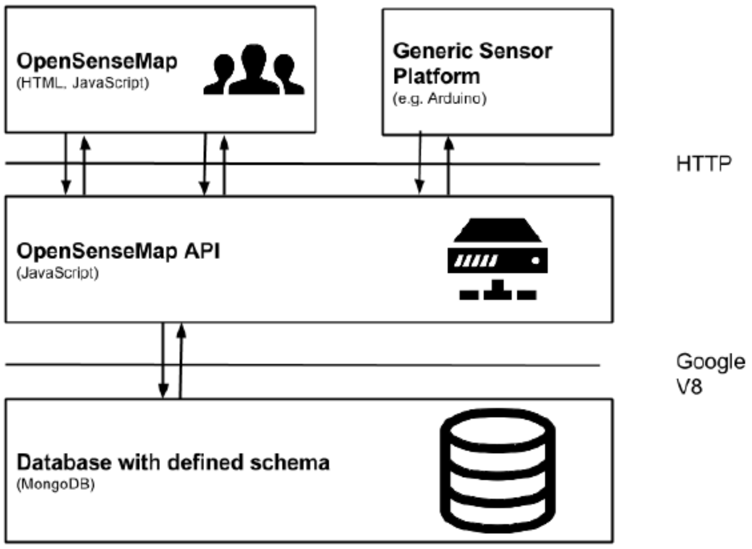


Figure 2. OpenSenseMap system architecture.

The performance of web browsers immensely increased over the last years. Today, it is possible to realize desktop-like web applications (Gutwin et al, 2011) due to highly efficient JavaScript engines and hardware acceleration. OpenSenseMap is completely written in HTML5 and JavaScript. This makes it runnable in any modern web browser on almost any device. As stated in chapter 3 not all provided functionality by the SWE framework are used and required by the OpenSenseMap. Therefore Bröring et al. (2012) advise that it may be easier to go with theWeb of Things approach. Additionally Demuth et al. (2013) stated that setting up OGC services and providing the right publishing methods are a challenging and cost-intensive task for a community. Therefore the application was developed from scratch and picks up the WoT paradigm in order to be applicable in citizen science projects. The development process of OpenSenseMap followed an agile development paradigm with early feedback from its potential users.

4.1 Database & Data model

As shown in Figure 3, OpenSenseMap uses MongoDB to store its data. MongoDB is a NoSQL Database. NoSQL stands for ‘not only SQL databases’ and are typically categorized into five different categories (Nayak et al., 2013) whereas MongoDB is a Document Database. Document Databases, store data in standard formats like JavaScript Object Notation (JSON) or XML. NoSQL Databases are well-suited for WoT applications: besides horizontal scaling, NoSQL databases support dynamic schemas. In contrast to traditional SQL databases, NoSQL databases require no structural definition of the data that is going to be stored. This makes NoSQL databases highly flexible because each data record in the database may be different in structure. This enables OpenSenseMap to keep pace with upcoming sensor platforms, which may include new sensors. OpenSenseMap's data model (cf. Figure 3) is derived from the O&M and SOS data model. It contains three collections and each collection is dened with a minimal schema; Table 2 summarizes the complete data model. Collections can be seen as tables in relational databases. Furthermore it is eye-catching in Figure 3, that the sensors collection is inside the sensor platform collection; this is called an embedded document. An embedded document is one way of modeling relations in NoSQL databases;

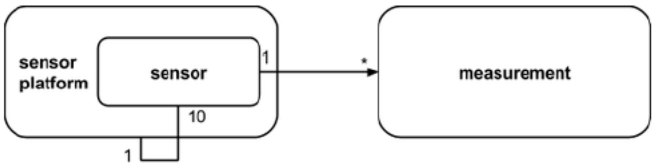


Figure 3. OpenSenseMap data model as collections.

an alternative is referencing documents by unique identifiers. A good rule (Iacobelli, 2014) to decide between embedded and referenced documents is as follows: if the amount of subdocuments has no limit go with references, otherwise go with embedded documents. The sensors of a sensor platform are modeled as an embedded document because the number of sensors is limited to ten for each sensor platform. The number of measurements is unlimited; therefore, the sensor's measurement collection is referenced. Furthermore, the entities sensor and sensor platform can be extended to include additional metadata. This could be detailed information such as the sampling rate of a sensor. In contrast, the entity measurement can not be extended.

Table 2. Complete OpenSenseMap schema for all collections

Collection	Field	Required
Sensor Platform	name	yes
Sensor Platform	location	yes
Sensor Platform	boxType	yes
Sensor Platform	exposure	yes
Sensor Platform	grouptag	no
Sensor Platform	sensors	yes
Sensor Platform	image	no
Sensor	title	yes
Sensor	unit	yes
Sensor	sensorType	no
Sensor	lastMeasurement	no

Measurement	value	yes
Measurement	sensor_id	yes

All data is stored as JSON in the database. JSON was selected because it was originally derived from JavaScript and is nowadays a lightweight data-exchange format. JSON can be used in JavaScript without implementing parsing and transformation operations. Furthermore, JSON is well-suited for use in environments with restricted resource and performance capabilities (Nurseitov, 2009).

4.2 Back-end / API

This section describes the back-end implementation of OpenSenseMap and its exposed API. OpenSenseMap makes use of NodeJS. NodeJS enables us to use JavaScript on the server-side and its event-driven and non-blocking I/O is well-suited for building real-time and data-intensive applications. Furthermore, NodeJS supports asynchronous I/O for sockets, files and HTTP communication; this makes a traditional web sever like Apache obsolete. NodeJS is actively developed and supported by its community and Joyent.

4.2.1 *RESTful Service*

The RESTful service is the heart of OpenSenseMap and implements the data publish and data retrieval functions. Developers may use it to build custom applications around OpenSenseMap or integrate OpenSenseMap into existing projects. The NodeJS application runs its own web server and implements the different HTTP request types. This makes the registered sensor platforms with their sensors and measurements accessible by unique URIs; Figure 4 shows the implemented URI schema.

```
http://opensensemap.org:8000/boxes/<platform-id>/sensors/<sensor-id>
```

Figure 1. URI schema to address single sensor platforms, its sensors and its measurements

The API currently implements the common HTTP-methods GET, POST, PUT, DELETE. These HTTP-methods realize the use-cases defined in section 3.1. If a resource hits the API and requests a resource, the request is processed at the NodeJS application and it responds with the data for the requested resource or an HTTP error code and message[20]. The GET-request is used to return different data from OpenSenseMap. Executed at root level, it returns a list of all registered sensor platforms. By adding the identifier of a selected sensor platform, it returns detailed information of the selected sensor platform and its latest measurements. To reduce the response size it is also possible to query individual sensors. The POST-request is used to register new sensor platforms and to publish new measurements. The PUT-request is used to update or modify an already registered sensor platform. The DELETE-request is used to delete a registered sensor platform and all of its published data. OpenSenseMap does not support a traditional user management but implements a simplified per sensor platform user management. To prevent abuse of the service, each modifying HTTP-method is authenticated using an unique api-key (Babal, 2015). The unique api-key identifies and authenticates the owner of a sensor platform.

4.2.2 *Data formats*

OpenSenseMap currently supports two different response formats: JSON and GeoJSON. By default, the response is formatted in JSON. By adding .geojson to a GET-request, the response is formatted in GeoJSON. GeoJSON is always a JSON object but encodes geographic data structures in a standardized way. Most Geographic Information Systems (GIS) are capable of processing GeoJSON. Both formats - JSON and GeoJSON - are an alternative to XML for data exchange on the web due to its compact representation (Crockford, 2006). That means it is more human-friendly in case of read- and writability and as well as for machines to parse and to generate. JSON is an open and language-independent data format.

Data is published on OpenSenseMap under the ODC Public Domain Dedication License (PDDL). The PDDL allows the reuse and modication of the original data without any restrictions.

4.3 Front-end

This section describes the graphical user interface (GUI) to interact with the back-end implementation, which was introduced in the previous section. The user-interface (cf. Figure 5) allows users to easily explore and contribute to OpenSenseMap. It fullfils two usecases: Exploring the already registered sensor platforms and their measurements and registering new sensor platforms. The functionality required for the two use-cases is implemented in separate modules, which will be described in the following two sections.

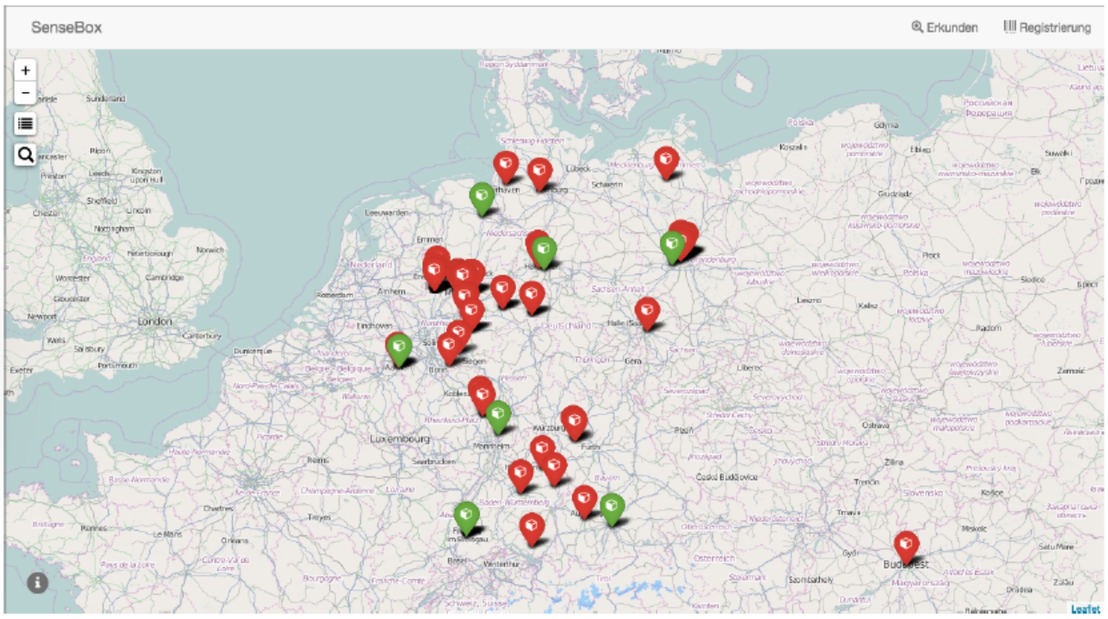
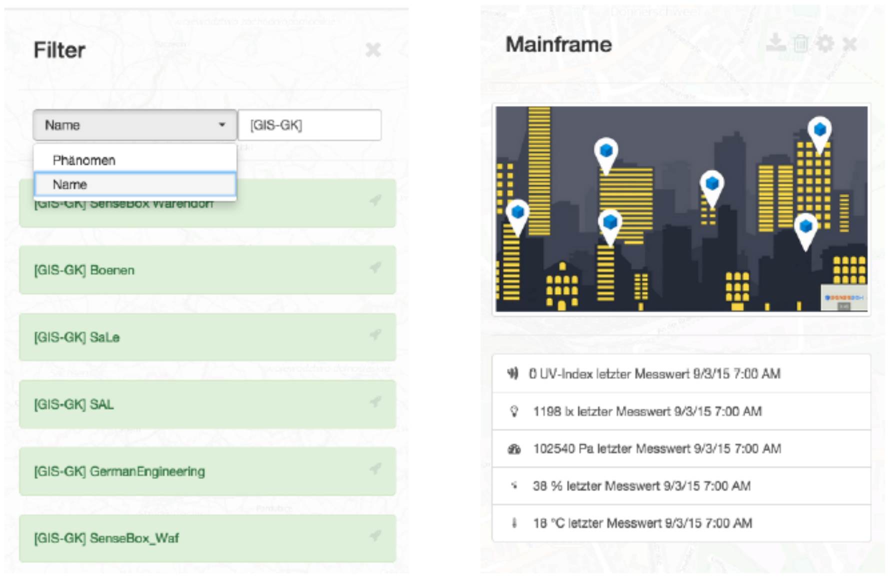


Figure 5. Explore view of the OpenSenseMap platform.

4.3.1 *Explore*

The explore view allows users to discover the registered sensor platforms and their published data; it is the default view when entering the OpenSenseMap website. It is a map based interface and supports basic spatial filtering using standard map interactions like panning and zooming. Additionally by clicking on the "magnifying glass" icon an address can be entered and searched for. Furthermore, users can browse the registered sensor platforms by searching for a sensor platform name or the measured phenomenons (cf. Figure 6a). While typing the filter term, the results are filtered and displayed as a list. To get detailed information about a sensor platform, a sub-view was implemented and can be accessed by clicking on a marker on the map.



[GIS-GK] SenseBox Warendorf
[GIS-GK] Boenen
[GIS-GK] SaLe
[GIS-GK] SAL
[GIS-GK] GermanEngineering
[GIS-GK] SenseBox_Waf

0 UV-Index letzter Messwert 9/3/15 7:00 AM
1198 lx letzter Messwert 9/3/15 7:00 AM
102540 Pa letzter Messwert 9/3/15 7:00 AM
38 % letzter Messwert 9/3/15 7:00 AM
18 °C letzter Messwert 9/3/15 7:00 AM

(a) Filter view with listed results.

(b) Detail view of a selected sensor platform.

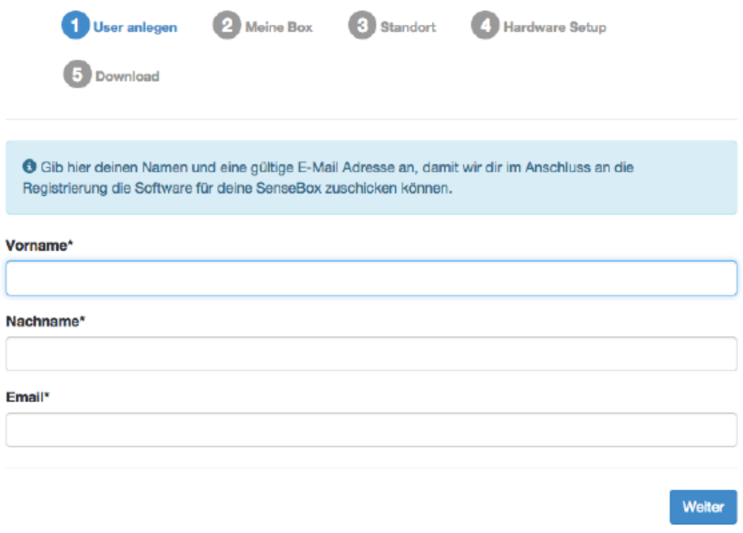
Figure 6. Sub-views of explore view.

The detail-view (Figure 6b) shows all metadata, all sensors and their lastest measurements. The measurements are updated in real-time. Furthermore, the explore-view follows an URI pattern to make the boxes uniquely identifiable and open up the detail-view directly. According to the requirements of usability the front-end interface was build with a clean and easy to use design in mind. The main used framework is AngularJS[22] to encapsulate the single modules and to provide a way to easily extend the modules. The map interface is provided by Leaet.js that oers an easy way to display JSON and GeoJSON data.

4.3.2 *Registration*

Users can add new sensor platforms using a guided step-by-step registration procedure (cf. Figure 7). The registration procedure is divided into five single steps and all required fields must be entered in order to proceed with the next step. According to Nielsen (1994) that users should not be able to make mistakes, this mechanism preserves the user against registering a non-functional sensor platform. As mentioned in section 3.2, no prior registration is needed to use the platform. The first step of the registration procedure requests the user to enter his name and email address. This information is only needed for registering a new sensor platform and to prevent people to upload spam and useless data. The email address is used to send the necessary api-key for authentication and a personalized sketch, described later in this section, to the owner of the sensor platform. In the following two steps of the registration procedure, the user has to specify the sensor platform by providing a name, exposure, geo-location and further optional metadata. The sensors of the sensor platform can be specified in two different ways:

- manual configuration
- plug & play solution


1 User anlegen 2 Meine Box 3 Standort 4 Hardware Setup 5 Download

Gib hier deinen Namen und eine gültige E-Mail Adresse an, damit wir dir im Anschluss an die Registrierung die Software für deine SenseBox zuschicken können.

Vorname*

Nachname*

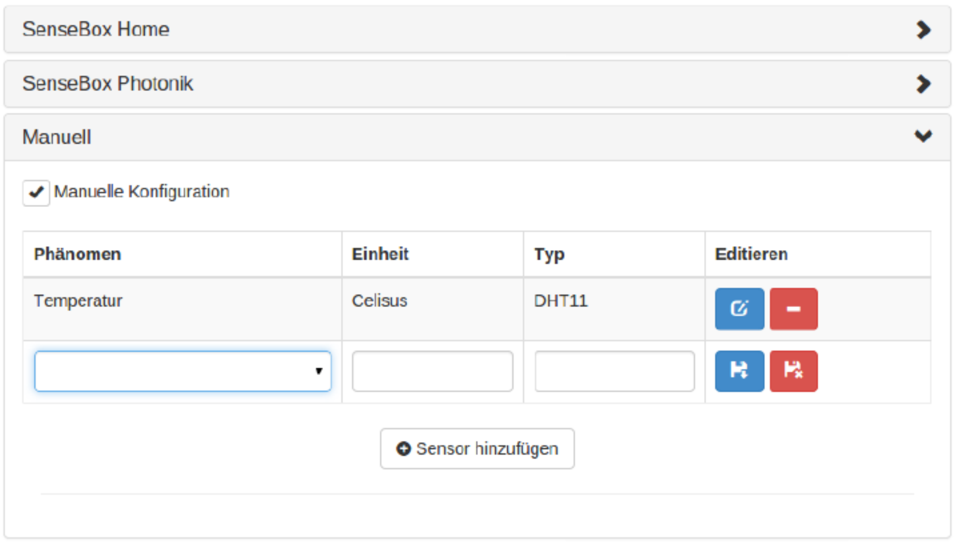
Email*

Weiter

Figure 7. Step-by-step registration form.

If choosing manual configuration, the user has to specify for each connected sensor the measured phenomenon, the unit of measurement and the sensor type (cf. Figure 8). The number of connected sensors is limited to ten sensors per sensor platform. After registering a sensor platform using the manual configuration option, the user is able to upload data to OpenSenseMap using OpenSenseMap's RESTful API. Of course, the user needs programming knowledge to implement this functionality.

If choosing the plug & play configuration, the user can select from a list of supported sensor platforms. The plug & play configuration is based on pre-configured templates, which integrate the information made by the user during registration into an executable program for the selected sensor platform. An example for SenseBoxes, which are custom Arduino boards, shall illustrate this. Arduino requires a so-called Sketch, which is the program that runs on an Arduino board. It is responsible to interface with the connected sensors. The missing part is to push the collected data to OpenSenseMap using the exposed API. With knowledge of the underlying hardware setup and the user information provided during the registration procedure, a personalized sketch, which is capable of interfacing the connected sensors and pushing data to OpenSenseMap, can be generated automatically.


SenseBox Home

SenseBox Photonik

Manuell

✓ Manuelle Konfiguration

Phänomen	Einheit	Typ	Editieren
Temperatur	Celisus	DHT11	

Figure 8. Registration of a generic sensor platform (manual conguration).

The described plug & play mechanism was implemented for the different SenseBoxes. In the last step where the sensor setup is specified, the user can select one of the four preconfigured SenseBoxes. By choosing one of these the registration procedure identifies the corresponding sketch template. The template is copied and personalized by inserting the IDs to reference the sensor platform and sensors. The proposed plug & play mechanism enables citizens without programming knowledge to participate in complex projects. What was exemplarily shown for the SenseBox hardware setup can be easily applied to different sensor platforms. To our knowledge, no Sensor Web portal offers this kind of configuration. The code generation for manual hardware setups is not possible right now and is a challenging task concerning the diversity of available microcontroller and sensors.

5. EVALUATION

This chapter evaluates the user interface design of OpenSenseMap. The user interface was evaluated by conducting a web-based survey. After a short description of principles and the survey design, the survey results will be analysed.

5.1 Principles and survey design

A web-based survey was conducted in order to evaluate the user interface design of OpenSenseMap. The survey follows the design principles for web-based surveys (Couper et al., 2001) and uses standardized survey components like scale, free-text and multiple choice inputs. Different statements on the user interface design of OpenSenseMap were questioned and the survey participants have to agree or disagree. The survey participants have to rate their agreement or disagreement on a six-level Likert scale (Wuensch, 2005) hence, no neutral rating is possible and a tendency must be stated by the survey participants. Furthermore, the survey participants have to perform two tasks and rate how easy it was to solve the task. The survey participants can give detailed feedback on each task by using the provided free-text input eld.

The survey contains 21 questions in total and is split into the following categories:

- General information about participant
- First impressions of OpenSenseMap
- Functionalities of OpenSenseMap (Explore and Registration)
- Open Data
- Suggestions

The first section on general information asks for age, gender, profession, expertise in web technologies and participation in other citizen science projects. The next section focusses on the first impression of OpenSenseMap. The third section includes two tasks, which have to be performed by the survey participants. First, the survey participants have to register a new sensor platform with an integrated temperature and humidity sensor. Second, the survey participants have to use the filter functionalities and the detailed view. The next section focusses on open data and privacy. The survey finishes with an open question on general suggestions and missing functionalities.

The user study was shared with all SenseBox customers without any knowledge about the person. The survey questionnaire and the results are attached in the Appendix.

5.2 Analysis

The web-based survey was conducted by 15 participants. The group of survey participants was made up of eleven males, three females and one unspecified gender participant. The average age of the survey participants was 34 years with a standard deviation s = 13;85. The age of the survey participants is in the range of 19 to 65 years. Eight survey participants are employed, five are students and two are pensioners. Seven survey participants say that they have no technical background. All survey participants claim to have experience with web technologies; 40% describe themselves as experts.

All survey participants rated OpenSenseMap’s user interface design as well-arranged. Only the used icons were rated as being not intuitive. Three survey participants answered that "the red icons suggest that the corresponding sensor platform is offline or that an error occurred at the corresponding sensor platform". The legend icon, which shows up an explanation of the icons, in the left lower corner of the map is inconspicuous and should be placed more prominently on the site. Furthermore, the survey participants asked for an optional quick-tour when using OpenSenseMap for the first time.

All survey participants said that it was easy to register a new sensor platform. 67% of the survey participants fully agreed with the guided step-by-step registration procedure. The remaining 33% agreed with the guided step-by-step procedure but suggested to reduce the number of steps. Specifying the configuration of the new sensor platform was rated to be easy by 14 of the 15 survey participants. But one survey participant was not able to complete this task because of insufficient information; more detailed information should be added to the registration procedure. The perception of the explore view is similar. The survey participants mentioned that they had problems finding the filter function. After finding the filter function (magnifying glass icon), all survey participants were able to solve the task. The filter-as-you-type realization was perceived positively because of the instant feedback of the user interface.

Concluding with the questions regarding open data and privacy, 80% of the survey participants completely agree with the statement that they appreciate the free of charge model of OpenSenseMap. According to that 93% have no problem that all data is published as open data and is accessible and reusable by anyone. However, nearly 50% of the survey participants would favour the opportunity to mark data as private and not to publish it as open data.

6. CONCLUSION AND FUTURE WORK

This work presented OpenSenseMap - an open sensorweb portal for citizen science projects. OpenSenseMap is open in many ways. Open in that it builds upon an open-source software stack. Open in that the collected data is published as open data. Open in that its RESTful interface and lightweight data model make it possible to connect many different WoT-capable sensor platforms; we have already seen other sensor platforms[23] connecting to OpenSenseMap. Open in that selecting a NoSQL document database and its flexible data model make it highly extensible.

The user-interface of OpenSenseMap was carefully designed in that it shall be easy and is fun to use. The survey confirmed that OpenSenseMap is easy to use but also showed what needs to be improved and gave ideas for future work. The users asked for an optional quick-tour when using OpenSenseMap for the first time, additional information when using the manual setup of a yet unknown sensor platform and more intuitive icons. Furthermore the survey revealed a contradictory meaning about open data: even if welcoming open data, users asked for some kind of privacy settings. A further analysis of this conict may be interesting and valuable.

Future work on OpenSenseMap should include the extension of the plug&play mechanism to further sensor platforms, data export from OpenSenseMap and exploration functionalities. We currently see two approaches to extend the plug&play mechanism: a configurator-based approach, which is capable of the most common microcontrollers and sensors, or a community-based approach, which allows users to create recipes for new hardware and provide them to the other users. Data export from OpenSenseMap should support geospatial formats like Shapefiles and common-use file formats like CSV or Excel. For exploration the user needs more functionalities like diagrams of historic data or advanced filter functionalities.

7. REFERENCES

Babal, H., 2015. Token-based authentication with angularjs & nodejs. http://code.tutsplus.com/tutorials/token-based-authenticationwith-angularjs-nodejs--cms-22543. last accessed 7.3.2015.

Berson, A., 1992. *Client/Server Architecture*. McGraw-Hill, Inc., New York, NY, USA, 1992.

Bröring, A., Echterhoff, J., Jirka, S., Simonis, I., Everding, T., Stasch, C., Liang, S., and Lemmens, R., 2011. New generation sensor web enablement. *Sensors* 11, 3 (2011), 2652–2699.

Bröring, A., Remke, A., and Lasnia, D., 2012. Sensebox–a generic sensor platform for the web of things. In *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer Berlin Heidelberg, 2012, pp. 186–196.

Conrad, C., and Hilchey, K., 2011. A review of citizen science and community-based environmental monitoring: issues and opportunities. *Environmental Monitoring and Assessment* 176, 1-4

(2011), 273–291.

Couper, M. P., Traugott, M. W., and Lamias, M. J., 2001. Web survey design and administration. *Public Opinion Quarterly* 65, 2 (2001), 230–253.

Crockford, D. 2006 The application/json media type for javascript object notation (json).

Demuth, D., Bröring, A., and Remke, A. A citizen science sensor platform as a live link from gis to the internet of things. Proceedings of AGILE 2013.

Dickinson, J. L., Shirk, J., Bonter, D., Bonney, R., Crain, R. L., Martin, J., Phillips, T., and Purcell, K., 2012. The current state of citizen science as a tool for ecological research and public engagement. *Frontiers in Ecology and the Environment* 10, 6 (2012), 291–297.

Faludi, R., 2010. *Building Wireless Sensor Networks: With ZigBee, XBee, Arduino, and Processing, 1st ed.* O'Reilly Media, Inc., 2010.

Fielding, R. T., and Taylor, R. N., 2012. Principled design of the modern web architecture. *ACM Transactions on Internet Technology* (TOIT) 2, 2 (2002), 115–150.

Florence, T., 2012. Sensor observation service. https://www.seegrid.csiro.au/wiki/Siss/SensorObservationService. last accessed 01.03.2015.

Goodchild, M. F., 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal* 69, 4 (Aug. 2007), 211–221.

Guinard, D., and Trifa, V., 2009. Towards the web of things: Web mashups for embedded devices. InWorkshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences) (Madrid, Spain, Apr. 2009).

Guinard, D., Trifa, V., Mattern, F., and Wilde, E., 2011. From the internet of things to the web of things: Resource-oriented architecture and best practices. In *Architecting the Internet of Things*, D. Uckelmann, M. Harrison, and F. Michahelles, Eds. Springer Berlin Heidelberg, 2011, pp. 97–129.

Gutwin, C. A., Lippold, M., and Graham, T., 2011. Real-time groupware in the browser: testing the performance of web-based networking. In Proceedings of the ACM 2011 conference on Computer supported cooperative work (2011), ACM, pp. 167–176.

Hand, E., 2010. Citizen science: People power. *Nature* 466, 7307 (2010), 685–687.

Iacobelli, G., 2014. Mongodb schema design: Embedded vs references. https://coderwall.com/p/px3c7g/mongodb-schema-designembedded-vs-references. last accessed 16.02.2015.

Kopetz, H., 2011. Internet of things. In *Real-Time Systems*, Real-Time Systems Series. Springer US, 2011, pp. 307–323.

Kortuem, G., Kawsar, F., Fitton, D., and Sundramoorthy, V., 2010. Smart objects as building blocks for the internet of things. *Internet Computing,* IEEE 14, 1 (2010), 44–51.

Nayak, A., Poriya, A., and Poojary, D., 2013. Type of nosql databases and its comparison with relational databases. *International Journal of Applied Information Systems* 5, 4 (2013).

Newman, G., Wiggins, A., Crall, A., Graham, E., Newman, S., and Crowston, K., 2012. The future of citizen science: emerging technologies and shifting paradigms. *Frontiers in Ecology and the Environment* 10, 6 (2012), 298–304.

Nielsen, J., 1994. *Usability engineering*. Elsevier, 1994.

Nurseitov, N., Paulson, M., Reynolds, R., and Izurieta, C., 2009. Comparison of json and xml data interchange formats: A case study. Caine 2009 (2009), 157–162.

OGC, 2007. OPEN GEOSPATIAL CONSORTIUM. OpenGIS Sensor Model Language (SensorML) Implementation Specication. http://portal.opengeospatial.org/files/?artifact_id=21273. Version: 2007.

Silvertown, J., 2009. A new dawn for citizen science. *Trends in Ecology & Evolution* 24, 9 (2009), 467 – 471.

Sohraby, K., Minoli, D., and Znati, T., 2007. *Wireless Sensor Networks: Technology, Protocols, and Applications*. Wiley-Interscience, 2007.

Steenkamp, L., Kaplan, S., andWilkinson, R. Wireless sensor network gateway. In AFRICON, 2009. AFRICON ’09. (Sept 2009), pp. 1–6.

Uckelmann, D., Harrison, M., and Michahelles, F., 2011. An architectural approach towards the future internet of things. In *Architecting the Internet of Things*, D. Uckelmann, M. Harrison, and F. Michahelles, Eds. Springer Berlin Heidelberg, 2011, pp. 1–24.

Wiggins, A., and Crowston, K., 2011. From conservation to crowdsourcing: A typology of citizen science. In System Sciences (HICSS), 2011 44th Hawaii International Conference on (Jan 2011), pp. 1–10.

Wiggins, A., and Crowston, K., 2012. Goals and tasks: Two typologies of citizen science projects. In System Science (HICSS), 2012 45th Hawaii International Conference on (2012), IEEE, pp. 3426–3435.

Wiggins, A., Newman, G., Stevenson, R., and Crowston, K., 2011. Mechanisms for data quality and validation in citizen science. In e-ScienceWorkshops (eScienceW), 2011 IEEE Seventh International Conference on (Dec 2011), pp. 14–19.

Wuensch, K. L., 2005. What is a likert scale? and how do you pronounce’likert?’. East Carolina University (2005).