Bachelor Thesis in Geoinformatics

A Web of Things integrated Sensor Platform for Precision Agriculture

Dustin Demuth
d.demuth@uni-muenster.de
MatNr.: 350461

University of Münster
Institute for Geoinformatics
March 19, 2012

Advisor: Arne Bröring
Second Advisor: Dr. Albert Remke

Abstract

This thesis introduces the *AgriSenseBox*, a Web of Things integrated Sensor Platform for Precision Agriculture. Based upon open hardware, the AgriSenseBox provides a web server containing a RESTful interface. The sensor platform is deployable in the field and makes sensor data processable without caring of differing formats. The AgriSenseBox is encoding its measurements as JSON using the structure proposed by O&M and hands out links to descriptions of the attached sensors. Following the principles of the WoT, sensor data is browseable by using web standards, such as HTTP and URI.

Contents

1 Introduction

The evolution of microcontroller platforms is continuously advancing. Open hardware projects like Arduino[1], Netduino[2] or nanode[3] are cheap and can be programmed to fit needs without previous knowledge of microcontroller-programming. An interested community of enthusiasts is the engine behind those projects. The renunciation of black-boxed hardware enables professional and even amateur developers to change the hardware for their own goals and to improve it. Existing hardware is enhanced and extended. Like in open source software, information about those extensions is brought back to the community. Most of these open hardware platforms are capable of providing open standards like TCP/IP or HTTP and therefore are able to serve as content-generators (pushing data to services like pachube[4]) or even hosting data as a web server [24].

With the open hardware movement and the internet-enablement of hardware, the *Internet of Things* (IoT) has arrived at the sensor-markets [3]. Micro web servers can be able to provide RESTful services [22] as well as deliver all sorts of Internet Media Types (MIME) and even work as file servers. Using these standardized methods, the hardware can be fully integrated into the web, expanding the IoT to the *Web of Things* (WoT) [23]. Limited in Random Access Memory (RAM), Read-Only Memory (ROM) and bandwidth they are not the best option to serve massive metadata-refined Extensible Markup Language (XML) documents, but they are capable of describing the basic attributes of an attached sensor in a more lightweight, text-based format. Data gathered by the devices is in most cases neither annotated nor metadata-refined.

1.1 Use Case

Agricultural methods can be improved with better understanding of the surrounding environment. Manuring and moistening of plants can be fitted to values where maximum crop is possible. In ecology the (primary) location factors are defined as light, temperature, availability of water, chemical substances (nutrients) and mechanical influences (wind, fire, animals)[19, p.415]. Optimal location factors lead to optimal plant growth. Therefore in agriculture knowledge on brightness, temperature, soil moisture, soil carbon and nitrogen is important to support good growth of plants. Oerke et al. [32] define Precision Agriculture or Precision Farming as an

[1] http://www.arduino.cc
[2] http://www.netduino.com
[3] http://www.nanode.eu
[4] http://www.pachube.com

> "agricultural management system using computerized information technologies (IT) – global navigation satellite systems (GNSS), geographic information systems, remote sensing devices, data management systems and telecommunications – for optimal use of nutrients, water, seed, pesticides and energy in heterogeneous field situations."

A network of, possibly location-aware, small wireless sensors, with low power consumption is able to provide that information [5]. The resulting data can be used to create an interpolation map of the indicators [11]. Reasoning from this information helps in decision making and leads to improvements in watering, manuring and harvesting. In an agricultural environment an agriculturist needs to make the right decisions and act correspondingly to produce maximal crop. The data collected by a sensor network is supposed to support him in decision making. In an autonomous agriculture system the system would make decisions based on the collected data and might even act accordingly, for instance by turning on the sprinklers to moisturize the area of interest.

1.2 Motivation

Currently, there are a lot of research projects which evaluate related technologies, like wireless transmission protocols [37], routing-protocols [1] and power consumption [29] of sensor networks. In most cases the raw sensor data is transmitted by the sensor platform to a gateway, processed and sent to applications which are using the data. When accessing the platform between gateway and sensor platform it remains unclear what kind of data is sent, how it is formatted or how it is accessed. Data translation and annotation is done in the gateway or even later during data processing. Solutions for annotating sensor data directly on the sensor platform in the field are still missing or very rare. When not knowing the properties of a sensor platform, it is complicated to find out which sensors are provided and how the sensor data can be accessed. Adding the WoT paradigm to this situation would make each sensor platform a thing which is directly connected to a network and using web standards to access the sensor data [23]. This could also provide a solution for the data and sensor annotation problem.

The goal of this thesis is to carry the paradigms of the Web of Things into the applications of precision agriculture. Web-enablement should simplify the integration of sensor platforms into applications. Less complications in accessing sensor data from a sensor platforms should occur, as the platforms should be using common web stan-

dards to provide the data. In dependence on the similar *SenseBox*-project by Bröring et al. [10], the developed sensor platform is called *AgriSenseBox*.
In this thesis a location-aware sensor platform is developed, which is capable of sensing some of the indicators for plant growth, mentioned in the use case. Each sensor can be annotated with sensor descriptions; each measurement is annotated with meta data describing the measurement. A sensor platform in agricultural use can be exposed to rough physical conditions like extreme temperatures, floodings or impacts. It has to be resistant against those. In the field, configuration of a sensor platform is not possible, the platform should be pre-configurable and should consume a little amount of power, to decrease maintenance time.

1.3 Structure

This thesis is organized as follows; Section 2 provides background information on differences between IoT and WoT as well as fundamental models of sensor networks and combinations of WoT and sensor systems. In Section 3, the process of integrating a sensor platform into the Web of Things is presented. Section 4 describes the hard- and software which is used and/or developed to create a location-aware sensor platform. Section 5 applies the use case to the developed solution, discusses the outcomes and focuses on limitations. The thesis closes with an evaluation of the solution, a brief overview of future work and a conclusion.

2 Background

This section illuminates the background of this thesis. It is going into the *Internet of Things* and the *Web of Things*, including an introduction into *Representational state transfer* (REST), furthermore it addresses sensor networks and common methods of data designation.

2.1 IoT & WoT

The term *Internet of Things* has first been used in a paper by David Brock in 2001 [38]. It describes a concept in which the world of real, physical things is integrated into the virtual world of bits and bytes. The term *Web of Things* describes the evolution of the IoT [10]. It integrates web standards [23] and user-centering [38] into the concept.

2.1.1 Internet of Things

The term *Internet of Things* describes a mash-up of the real world and the virtual world. It embeds the technologies of identifying and locating as well as interacting with sensors and actuators. When necessary hardware got cheaper, the IoT increased relevance in industrial use-cases [38]. The IoT is build by using "autonomous smart objects" and resulting networks of those [40]. The CERP-IoT [12] (Cluster of European Research Projects on the Internet of Things) provides a definition stating that the IoT is "*dynamic*, *global*, *self configuring*" and "*inter operable*". Physical and virtual objects are "*identifiable*" and "*physical attributes* have *virtual personalities*" [12]. The result are everyday objects, which are embedding computers to bring information on-line [23]. IoT services are often embedded into custom solutions, which requires extensive time and expertise [23].

2.1.2 Web of Things

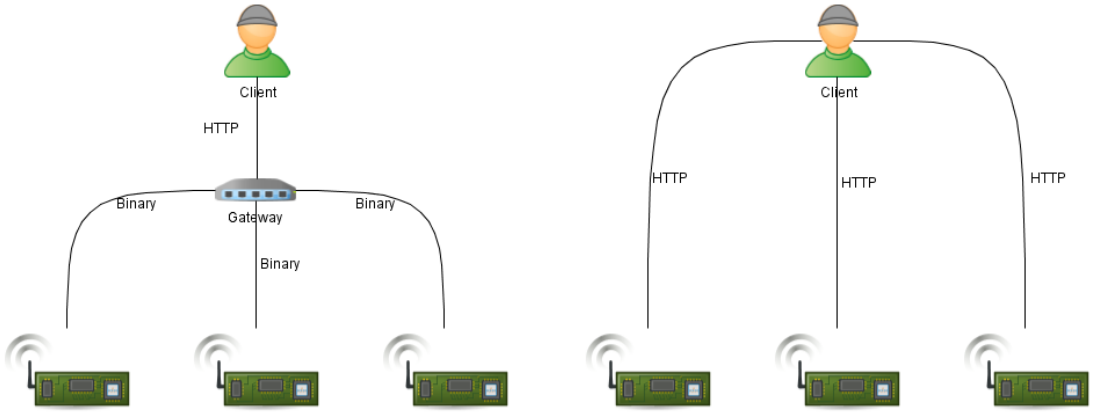
The *Web of Things* can be treated as the evolution of the IoT [10]. It extends the Internet of Things by using web-standards such as REST. The concept of the WoT is to make all "things" speak the same language, which leads to an renunciation from the expensive custom solutions of the IoT concept. In the concept of the WoT, tiny web servers are embedded into everyday objects, which were turned into "smart things" [24]. The WoT is intended to be user-friendly. Context of smart things can be displayed on web pages, which can even be used to control the configuration of the smart thing.

Representational State Transfer (REST) Representational state transfer is an architectural style of the web [17]. REST is using the *Hypertext Transfer Protocol* (HTTP) [16] and *Uniform Resource Identifier* URI [8]. Hypertext Transfer Protocol (HTTP) is an application protocol for data transportation. It is making use of requests and responses between a client and a server. A URIs consists of characters which identify a resource. The identifier contains the address (i.e. an IP-Address) or a placeholder for an address (i.e. a domain name) which specifies the host of an information. Uniform Resource Identifiers (URIs) identify a *resource* which is the "conceptual target of a hypertext reference" [17]. REST interactions are stateless. Each request contains all necessary information for the server to create a corresponding response. Representational state transfer (REST) includes the paradigm of *representation*. The server has to be capable of submitting different formats of the data element.

Methods of Data Access on Smart Devices The "RESTification" of a thing can be achieved in two methods, a direct –where each thing is directly connected to the web– and an indirect one –where a gateway is an intermediate layer between the web and the things [23]. The methods are depicted in figure 1a and 1b
According to Guinard & Trifa [23] *direct-RESTification* can be applied on web-enabled devices. Each device integrates a web server and uses a RESTful architecture to make data accessible. *Indirect-RESTification* or *gateway-RESTification* is used to integrate things which are not web-enabled by default. The things offer their data to a gateway which is an intermediary layer between the web and the things. The gateway is web-enabled and works as a web server using a RESTful architecture. The things exchange information with the gateway, which then transforms the information into web deliverable data.

2.2 Sensor Networks

Sensor networks consist of sensor nodes, which are capable of measuring factors of their environment. In a sensor network a sensor measures a value and sends this information to a final node, which is capable of processing the value [36]. According to Sohraby et al. [36] typical applications are *data collection*, *monitoring*, *surveillance* and *medical telemetry*. Methods of interaction of the sensor network with its environment are *sensing*, *control* and *activation*. Sensor networks consist of two to many sensors and use network topologies to exchange their data [15]. These topologies consist of at least three types of nodes. *End nodes* – which are typically referred as sensors –, *Routers* –which exchange information between devices– and a single *Coordinator* – which is



a: Access via a Gateway

b: Direct Access

Figure 1: Data access on smart devices

controlling the network [15]. At least two types of information exchange can be noted. One the one hand there is the single-hop method, where a sensor pushes its data directly to a processing node (Router or Coordinator), on the other hand, there is the multi-hop method, where a sensor pushes data to an other sensor which forwards this data to a processing node.
In a *Sensor Web* according to Delin & Jackson [14] this principles are extended by the matter, that sensors are intra-communicating in order to optimize the setup of the sensor web or to react on changing conditions. Sensors could even explore their environment by moving around [14].

In both concepts, data is sent from nodes (sensors) to primary nodes (gateways), which distribute the data to other nodes and processing devices. It is reasonable that primary nodes have to be more powerful than convenient nodes, as they have to be able of storing and forwarding data.

2.2.1 Combination of IoT / WoT and Sensor Networks

This part describes some already existing projects of the Internet of Things (IoT) and Web of Things (WoT) which are combining the principles of IoT and WoT with sensor networks.

The goal of the *CoolTown* project of Hewlett-Packard Laboratories, was to create a platform for ubiquitous computing. They made things accessible by Uniform Resource Locators (URLs) and grouped physically related things to be represented by

web servers [27, 26]. Another project is the commercial platform *Pachube*, which is a web-based service, used to manage and share real time data. Pachube is an IoT application, which is very similar to the WoT concept and uses an Application Programming Interface (API) to share the data collected by its users. Currently over 100000 data streams are active [34].
A third project has recently popped out of the nowhere. The platform koubachi[5] created a Wi-Fi sensor for indoor use, which can be attached to a plant. The sensor transfers measurements to a web service where the species of the plant is registered. The web service analyzes the data and hands out advices or alerts to optimize plant care [28]. One of the main differences of the koubachi project to this thesis can be found in genericness. Whilst koubachi concentrates on the implementation of soil-moisture, brightness and temperature sensors, the AgriSenseBox can be equipped with an indefinite variety of sensors. In addition, data can not only be pushed to one single web service but can be pulled from the AgriSenseBox by a plethora of different services.

SenseBox The objective of the SenseBox project was the creation of a generic sensor platform which is easy to deploy. The SenseBox is web enabled and uses a RESTful API to access the observed data. It uses a full scale computer, including web server and database, which is deployed in the field. Sensors are attached to an Arduino-device which transforms the measurements and transmits them to the computer. Measurements are processed by the computer, stored in the database and made available with the web server through a RESTful API [10]. Compared to the projects mentioned before, the SenseBox was especially designed to fit into the paradigms of the WoT.
In contrary to the solution provided in this thesis, the SenseBox requires a full scale computer to work. In addition it is offering a large amount of storage and advanced data management capabilities. The AgriSenseBox could be considered as a more lightweight offspring of the SenseBox concept, which does not require powerful hardware and a lower amount of power.

2.2.2 Data-Designation In Sensor Networks

This section will concentrate on possible methods of data designation and annotation which can be used in this thesis. It illustrates the XML-based provider-centric SensorML, the newer StarFL and gives an introduction into the user-centric Observation & Measurements.

[5] http://www.koubachi.com

OGC SensorML The Open Geospatial Consortium (OGC)[6] *Sensor Model Language* (SensorML) provides a framework which is capable of designating the "geometric, dynamic, and observational characteristics of sensors and sensor systems" [33]. SensorML is encoding this designations in XML. The model language is used to describe sensors, which can be dynamic or stationary. SensorML focuses on description of sensors, providing sensor information for observation discovery, processing and analysis of observations and process description. SensorML is flexible, and allows modelling of sensors in many ways. [33, 21]

StarFL The *Starfish Fungus Language* (StarFL) follows a more restrictive approach than the generic SensorML. It is modularized and consist of three modules, the *Static Module*, the *Dynamic Module* and the optional *Common Module*. The Static Module combines all information typical for a certain kind of sensor, like a datasheet; it can be used by many different sensors. The Dynamic Module models the alignment of the sensor to time and space; it is unique for each sensor and can contain a link to the Static Module. The objectives of StarFL are sensor and process description. [31]

Observation & Measurements The OGC *Observations & Measurements* (O&M) model is used for representing and exchanging results. An observation is an event, that occurs at a certain spatio-temporal position and generates a value for the observed phenomenon. In addition O&M is able to describe other properties of the measurement, for example the procedures which where used or the quality of a measurement. [20]

2.3 Precision Agriculture

In the following section existing applications of sensor networks in precision agriculture are presented.
In 2005 Baggio [5] proposed a setup for "wireless sensor networks in precision agriculture". Baggio used a gateway approach, without web services. Data is sent from the sensor to a router node which forwards data to the next router until the data reaches a coordinator device [5]. Allen et al. [2] use a sensor network to measure and monitor soil organisms and describe how to array sensor networks to measure soil dynamics. *AgroSense* [35] is a project, where a wireless sensor network was build to relay real time data of environmental properties. The project focused on application of enhanced data routing algorithms [35]. *SoilNet* [9] focuses on real time monitoring and estimation of

[6] http://www.opengeospatial.org/

soil water content. SoilNet has already started a large scale manufacturing of the sensors which are used in their wireless sensor network [9]. Liqiang et al. [30] built a wireless sensor network for crop monitoring by using IoT paradigms. In addition to common sensors, they attached a camera to determine crop growth [30].

All mentioned projects have in common, that they do not include a web server into the sensor platform which is deployed in the field. In addition, neither data nor sensors are annotated in a way which is sufficient for sensor and data recognition.

3 Applying the WoT Approach to the Sensor Network

In this chapter the approach for integrating a sensor platform into the Web of Things is depicted. Furthermore the term *sensor platform* –as used in this thesis– is elucidated as well as the requirements for a WoT integrated sensor platform are stated.

3.1 Terminology

In this thesis, *sensor* and *sensor platform* are major terms. In order to understand the concepts shown in this thesis, it is necessary to understand the meaning of each term.

The term *sensor* is understood as a device based upon an electric circuit which is capable of measuring a property and converts this measurement to a data signal [7]. This signal can be interpreted by an additional electronic component.

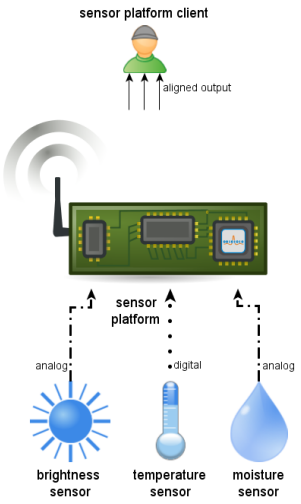


Figure 2: Sensor Platform

A *sensor platform* (as shown in figure 2) is a socket for one to many sensors. Sensors can be connected to it. It supplies power for the attached sensors and interprets the signals which are emitted by the sensors.

The sensor platform can have persistent storage capabilities to store the measured values, as well as the ability to decide and act based on the measurement. A sensor platform aggregates the measured information and converts the signals coming from the sensors into a common format. This aligned signal format can be read by a *sensor platform client*. In addition, the sensor platform can offer the possibility of remote configuration and remote control.

3.2 Requirements

The technical design of the sensor platform needs to be stamped by *low power consumption*, because the sensor platform might be used in an area where no power connection is available. In this, possibly remote, area a *robust case* is needed to protect the sensor

platform from external influences, like extreme temperatures or impacts. *Autonomous behaviour* of hard and software is required to minimize maintenance time. Each sensor platform has to be *simple to deploy* into an area of interest. As a matter of cause, the platform has to be capable of translating sensor data of different sensors into a common format. This data has to be accessible from remote. To fit the use case of precision agriculture, the platform has to be equipped with sensors, which are able to measure the location factors of a plant. Sensors must be capable of being integrated into the sensor platform with ease. A flexible method of mounting various sensors into the AgriSenseBox supports the *genericness* of the platform. This genericness supports the use of the platform in use cases different from precision agriculture.

The integration into the *Web of Things* makes it necessary that the platform is always connected to a network, which consequently results in higher power consumption on network interfaces as if the platform would only connect intermittently. In a *Web of Things*-integrated sensor network, the sensor platform comes with an integrated web server. Each sensor platform must have an unique address where data, which has been aggregated by the platform, can be requested. As the sensor platform is connected to a network, every client, which is also connected to this network, can access the measured data by accessing the given address. Unfortunately, the address does not lead to doubtless understanding of the dataset. Without detailed knowledge about the sensor platform that is currently accessed, the client still needs additional information about the dataset, because elements such as the unit of measurement or the accuracy of the measurement are unknown. It can be stated that a thing is useless as long as it remains unidentified. This confirms the need for a possibility to annotate the aggregated information. As a thing, the AgriSenseBox has to be capable of identifying itself and the connected sensors. Sensor data has to be annotated by describing the measurement and the sensor in an approved manner. These thoughts lead to the requirement of *Data Annotation*.

3.3 Design

The *AgriSenseBox* is supposed to be designed in a flexible easy-to-use manner. Each platform has a connection to a network which supports addressing of a network node, for instance the internet. These connections can be achieved with wired solutions like Local Area Networks (LANs) or wireless solutions such as Wireless Local Area Net-

works (WLANs) and Wireless Wide Area Networks (WWANs) which are commonly known as Universal Mobile Telecommunications System (UMTS) or Long Term Evolution (LTE). WWAN-solutions often cause special problems. These problems are mentioned in the limitations-part in section 5.2.1. Because of the mentioned problems, this thesis focused on the use of WLAN and LAN connections, only. The scope of access to the data can be regulated by choosing the network properties. Platforms which are integrated into a private LAN, whose data is not routed to the internet can only be queried in this private LAN. Platforms in a global company-network might be accessible from any place in this company-network, but not from the internet. Others are directly connected to the internet and can be only accessed if a connection to the internet is available. Each platform is directly connected to the network. No additional gateway is needed to access and transform the platform data, the chosen design paradigm corresponds to the concept of WoT which was presented in section 2.1.2. More information on routing and connecting things in networks can be found online, for example on the website of Juniper Networks[7].

The *AgriSenseBox* is designed to offer querying of data only. Therefore the platform is not pushing data to any service, instead it becomes a service itself. Pushing data to a web service requires knowledge of its address. As a sensor platform might last a long time, the address of the receiving service could change and has to be reconfigured on the sensor platform.

Data can be queried from the platform by accessing the address of the platform (which could also change). To receive data, the address of a sensor platform has to be known. There are at least two ways to do so, this is depicted in section 5.1, as well as a possibility to deal with changing sensor platform addresses. The address of the AgriSenseBox is be expressed by an URI. The sensor platform uses a fixed URI-scheme, which is depicted in section 3.3.4.
It is reasonable, that the content which is provided through this URI is the primary entry point for data retrieval from the WoT integrated sensor platform. The primary entry point serves as an index listing URIs to all available sensor resources and URIs to their descriptions. The data provided behind the entry-point is depicted in figure 12 and described in section 4.3.

[7] https://learningportal.juniper.net/juniper/user_activity_info.aspx?id=769

3.3.1 Use Case

To be used in an agricultural environment, the sensor platform requires interaction with the agriculturist. Of course, this requires additional amount of work, but the agriculturist is compensated with frequent information on the growth conditions of the crop. This supports in decision making and might accelerate acting. Accelerated acting leads to making early improvements of the growth conditions which can maximize crop productions or reduce the probability of crop shortfall (i.e. recognition of watering requirements when a drought is about to begin). In figure 3 the possibilities of interaction between agriculturist and sensor platform are depicted. The possible interactions are:

- Add sensor
- Temove sensor
- Alter sensor properties
- Deploy sensor platform
- Undeploy sensor platform
- Get data
- Maintain sensor network
- Decide
- Act

In order to *act* the agriculturist has to *decide* what he wants to do. To derive the correct decision he needs to *get data* from the sensor platform, which implies that the agriculturist has already *deployed a sensor platform* in his area of interest. Once deployed, a sensor platform is running autonomous in the field. Multiple sensor platforms are combined in a sensor network which can be maintained by the agriculturist. Amongst other things, *maintaining a sensor network* includes the inverse operation of deploying a sensor platform: *undeploy a sensor platform*. Each sensor platform needs to be equipped with sensors fitting the application scenario. It follows, that *add sensor* describes the process of adding a specific sensor to the platform and *remove sensor* describes the opposite. *Alter sensor properties* describes the operation of changing sensor-specific settings, the sampling interval for instance. Once the sensor platform is deployed, the platform tries to locate itself by using the GPS-signal and to connect to the attached network. When booted up properly, sensor data can be harvested from the sensor platform.

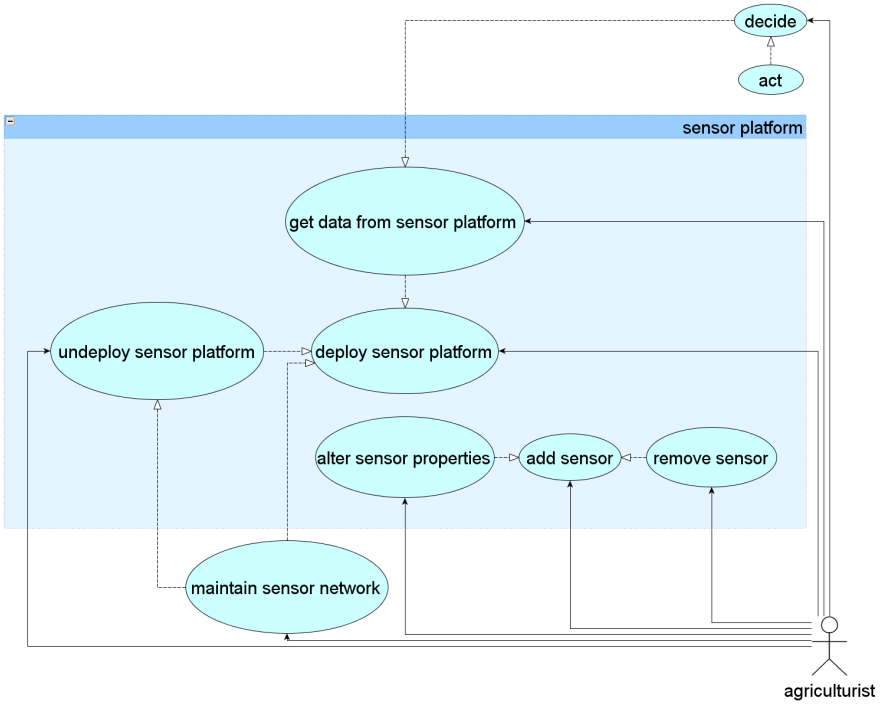


Figure 3: Use Case

3.3.2 Hardware Design

The platform needs possibilities to connect sensors to measure the location factors of the platforms domain. To locate the platform in the real world, the platform must be able to determine its geographic position. Connecting the AgriSenseBox to a network requires a networking interface. Operating the sensor platform requires power. These requirements lead to hardware design. The hardware of the AgriSenseBox can be divided into five units: Sensors, Network, Position, Power Supply, and a central Core which connects all units. These five units are represented by a *Main Unit* (MU), which is the central processing point of the sensor platform. It can control all units, except the power supply. A *Connectivity Unit* (CU), which is taking care of connecting the sensor platform to a network. Data is exchanged between Main Unit (MU) and Connectivity Unit (CU). A *Positioning Unit* (PU), which acquires the geographic and temporal position of the platform and a *Power Supply Unit* (PSU) as well as a set of Sensors. Figure 4 shows the interplay of the hardware components.

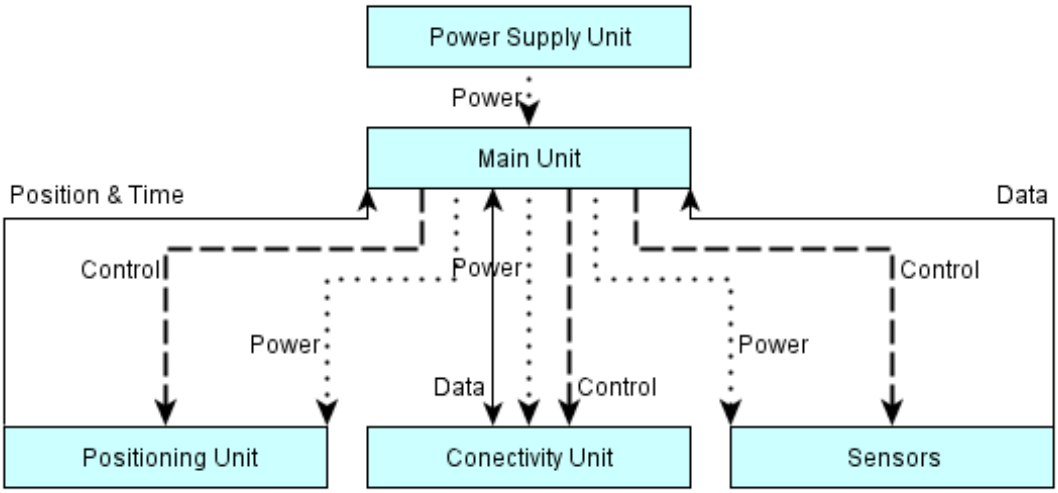


Figure 4: Interaction of the hardware units of the AgriSenseBox

3.3.3 Software Design

To match configuration and integration needs of the units[8], the software is divided into four components. The first component is a *Sensor Component*, which configures, integrates and manages the connected sensors. Second of all, a *Position and Time Component* which integrates the Positioning Unit and serves as a clock. Third of all, a *Web Service Component*, configuring the networking capabilities of the AgriSenseBox and providing a web server. The fourth component is the programming of the Main Unit which is designated as the *Main Component*. Figure 5 gives an overview of the used components.

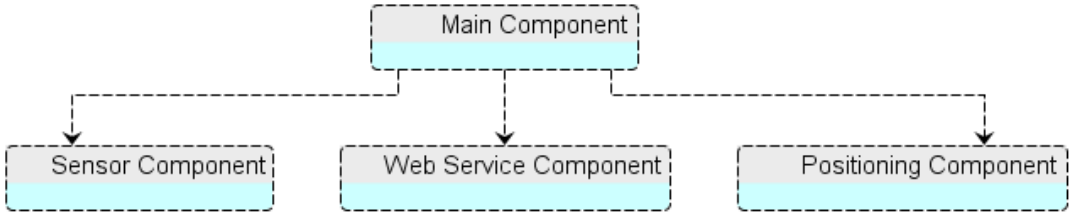


Figure 5: Software Components of the AgriSenseBox

[8]Except the power supply unit. It neither needs integration nor configuration

3.3.4 URI-Scheme

The sensor platform supports listing of all connected sensors and the retrieval of their sensor data. Connected sensors are listed at the entry point of the sensor platform. Sensor data is returned when accessing the corresponding resource. The pattern of the URI-Scheme is defined as:

```
http://<sensor platform address>/<sensor-id>/<method>
```

<sensor platform address> is the entry point of the sensor platform. It provides a collection of sensors which are attached to the platform.
<sensor−id> refers to an identifier for a specific sensor. When accessed this resource should list a collection of all available methods applicable for this sensor.
<method> stands for the method of interaction with the sensor. For example “data/” would list the measured data. At the current state of work, only listing of data is implemented. Therefore the sensor platform lists the sensor data as default, when the resource <sensor−id> is accessed. This listing has to be corrected when additional, more complex methods, like configuration or calibration, are added in the future.

4 Implementation of the AgriSenseBox Sensor Platform

To describe the prototypical implementation of the sensor platform, hard- and software configurations are described in this section. At the current state of work, the hardware is divided into units, which can be exchanged in a flexible way. In future, all hardware, except the power supply, could be fitted to one single *Printed Circuit Board* (PCB), to minimize the dimensions and number of plug and socket connections.

4.1 Hardware

This section describes the hardware chosen to build the prototypical sensor platform. It picks up the suggested five unit design of the previous chapter.

4.1.1 Main Unit

The *Main Unit* (MU) of the sensor platform is an *Arduino Mega 2560*[9] (figure 6) which provides an ATmega2560[10] microcontroller. The microcontroller is operating at 5 V, and offers 8 Kilobytes (kB) of RAM and 256 kB of flash memory which is used to store the programming of the sensor platform. The microcontroller works with a clock speed of 16 Mega Hertz (MHz), which is similar to the early central processing unit Intel i386DX, commonly known as "the 386" from 1985. The MU has 54 digital in- and outputs, and can operate with up to 16 analog inputs as well as with four Universal Asynchronous Receiver/Transmitterss (UARTs) [4]. Arduino offers some other cheaper models, unfortunately those do not provide sufficient RAM and ROM. The MU is the center of the AgriSenseBox. Almost all actions supported by the sensor platform are defined in the software of the MU. Still there might be components which cannot be configured from the MU.
Sensors are connected to the digital and analog inputs of this piece of hardware. The MU aggregates the different outputs of the sensors and converts them into the same data format, according to the definition of a sensor platform in section 3.1.

[9] http://www.arduino.cc
[10] http://www.atmel.com/devices/ATMEGA2560.aspx

Main Unit		
RAM	8	kB
FLASH	256	kB
Clock	16	MHz
Digital I/O	54	
Analog In	16	
UARTS	4	
Price	41.00	EUR

Table 1: Specification of the Main Unit

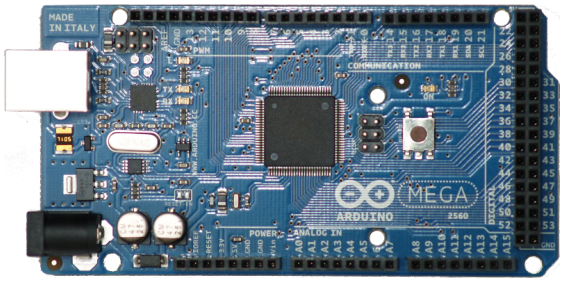


Figure 6: Arduino Mega 2560

4.1.2 Connectivity Unit

The second unit is the *Connectivity Unit* (CU). It is based on an *Arduino Ethernet Shield* (figure 7). This shield offers the possibility of a 10/100 Mbit s^{-1}(Megabits per Second) ethernet interface. The shield provides a Wiznet W5100[11] chipset which supports a network stack capable of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). Due to this network stack the shield can serve as a web server or post data to web services like Twitter[12]. Four simultaneous network sockets are possible. It does not support the new addressing scheme Internet Protocol version 6 (IPv6) [41]. The CU provides the hardware needed for the MU to work as a web server and is the interface of the sensor platform towards the network. The ethernet shield provides a slot for a micro SD card. With this slot it is possible to store measurements on a non-volatile memory card. The CU can be extended with a wireless bridge to make WLAN available on the sensor platform. This has been done with a *Vonets VAP11G*[13] wireless bridge. The CU should be optimized to offer native WLAN. This might be

[11] http://www.wiznet.co.kr/Sub_Modules/en/product/Product_Detail.asp?pid=1011
[12] http://www.twitter.com
[13] http://www.vonets.com

possible soon: A WLAN-Ethernet Shield is currently under development at Arduino [6].

Connectivity Unit		
Speed	10/100	Mbit s^{-1}
TCP	yes	
UDP	yes	
IPv4	yes	
IPv6	no	
Sockets	4	
Price	27.73	EUR

Table 2: Specification of the Connectivity Unit

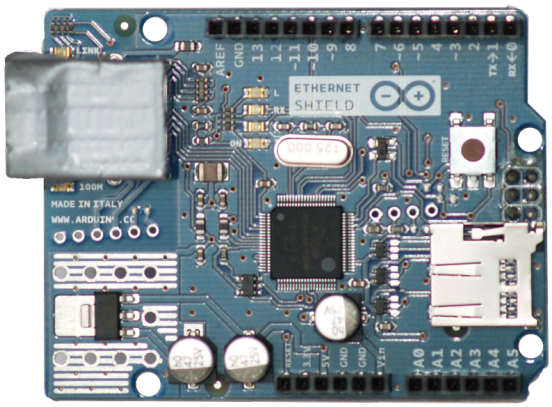


Figure 7: Arduino Ethernet-Shield with SD reader

4.1.3 Positioning Unit

The geographic position of the platform is determined by using a *Positioning Unit* (PU). The Positioning Unit (PU) consists of a *Globalsat EM-406*[14] Module (Figure 8). This module uses a Global Navigation Satellite System (GNSS) for positioning, in this case Global Positioning System (GPS). The EM-406 is a low-cost GPS engine, supporting the NMEA-0183[15] data protocol. According to the hardware specification of the module, it supports *Wide Area Augmentation System* (WAAS) and *European Geostationary Navigation Overlay Service* (EGNOS) to increase positioning accuracy. The datum of the acquired position is WGS-84 [39]. The second output of the positioning unit is

[14] http://www.globalsat.co.uk/product_pages/product_em406.htm
[15] http://www.nmea.org/

the current time in *Coordinated Universal Time* (UTC) also known as *Greenwich Mean Time* (GMT) or *Zulu-Time* (Z). The PU is an important component on a mobile sensor platform to get the spatio-temporal position of the platform. On a stationary platform it can be replaced by a clock and a hard-coded position. It is arguable that a GPS device is a collection of various sensors, because it is measuring more than one feature at a certain point of time. This thesis treats the GPS as one single device and not as an attached sensor, because it is a crucial part of the sensor platform.

Positioning Unit		
Datum	WGS-84	
Format	NMEA-0183	
	WAAS / EGNOS	
Price	38.49	EUR

Table 3: Specification of the Positioning Unit

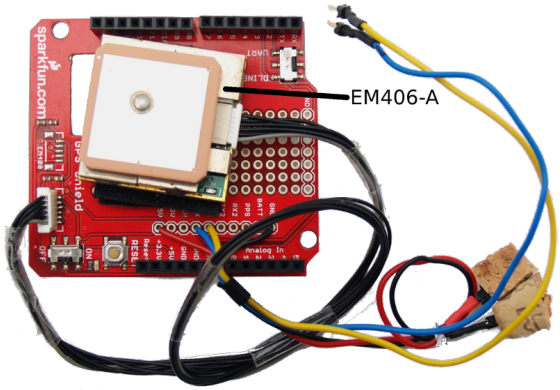


Figure 8: Positioning Unit of the AgriSenseBox

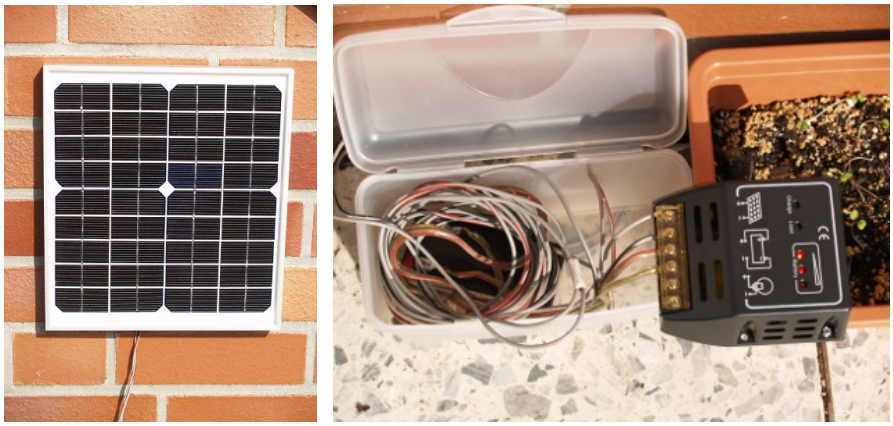
4.1.4 Power Supply Unit

The last component is the *Power Supply Unit* (PSU) of the device. To run the sensor platform autonomously a solar cell was chosen (figure 9a). Solar cells operate on principles of the photovoltaic effect and convert energy of the light into electric energy. The chosen cell has an output of 12 V and produces 10 Watt (W). During bright daylight, the cell produces more energy than needed to operate the platform. The excessive energy is buffered in a battery, which is powering the platform in times of insufficient (sun)light or at night. A charging controller (figure 9b) makes sure that the

battery is not overloaded or deep discharged, to guarantee a longer battery lifetime. If the battery is discharged, the sensor platform fails and powers off. The components of the Power Supply Unit (PSU) can be seen in 9.

Power Supply Unit		
Power	10	W
Voltage	12	V
Capacity	variant	
Price	48	EUR

Table 4: Specification of the Power Supply Unit



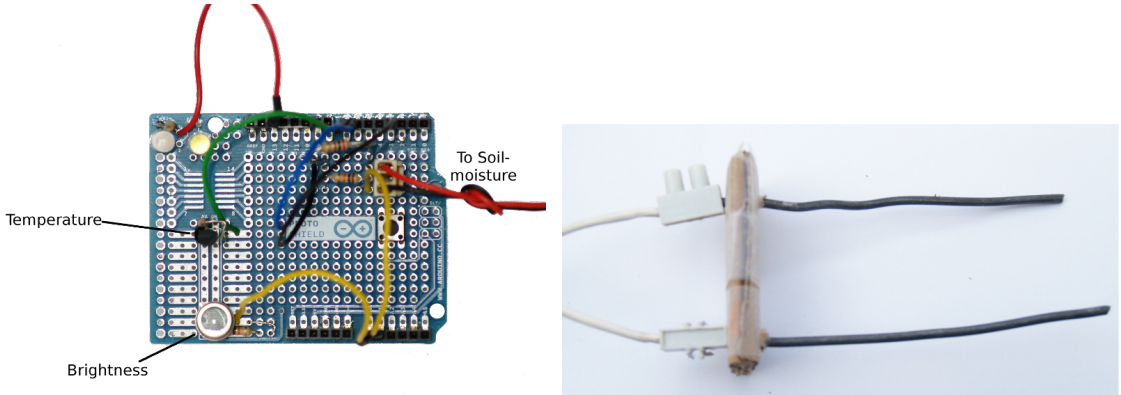
a: Solar panel b: Charge Controller

Figure 9: Power supply of the AgriSenseBox

4.1.5 Sensors

To match the use case, the sensor platform has to be equipped with sensors. A temperature, a light-intensity and a soil-moisture sensor have been chosen for this application. Temperature is measured by a digital *thermometer* (DS18B20 by MAXIM), light-intensity is measured with a *photoresistor* using the effect of photoconductivity, and the analogue inputs. The output of the photoresistor is a value between 0 and 1000. Soil-moisture is measured with a *self-made resistive-soil-moisture-sensor*, following the instructions given by [18]. The sensor-circuit consists of two electrodes and a resistor. The electrodes are shown in figure 10b. In combination this can be used as a voltage divider. With a voltage divider the soil-resistance can be deduced, which

depends on soil-moisture. Soil resistance is also depending on soil temperature [18]. The soil temperature is not measured with this platform. Therefore Soil-moisture data has to be considered as inaccurate in this case. As one electrode has a higher electric potential than the other and there is only direct current, the sensor is prone to electrolysis. All sensors are connected to a *Sensor Shield* (figure 10a) which is interfacing the sensors and the MU.



a: Sensor Shield with brightness and temperature sensors

b: Soil Moisture Sensor

Figure 10: Sensors of the AgriSenseBox

4.2 Software

This section describes the software, which was developed to control and operate the sensor platform. First it describes the implementation of the *AgriSenseBox* components, later it illustrates the RESTful service.
To develop the software running on the MU, the Integrated Development Environment (IDE) of Arduino was used. The IDE is available for the most operating systems on the arduino website. Source code of the platform is written in C or C++, compiled and uploaded to the microprocessor on the Main Unit. The community behind arduino has already created software libraries to support a lot of devices, which can be connected to an Arduino-based board. These libraries can be of use when adding a new sensor to a sensor platform.

The sourcecode developed in this thesis can be found in the following version control system:
`https://subversion.ifgi.de/thesis-demuth/`

4.2.1 Software Components

This part illuminates the four components of the *AgriSenseBox*, which were defined in the design of the sensor platform in section 3.3.3.

Sensor Component The Sensor Component uses the inheritance pattern. Each specialised sensor –like a thermometer– is a subclass of the abstract *Sensor* class. The *Sensor* class defines attributes and methods which are inherited by all subclasses. Attributes and methods provided by *Sensor* can be seen in figure 11. Some of them are depicted in the following.

> `pin` is an integer that describes the hardware input pin of the sensor platform where the signal of the sensor can be received.
> The attribute `observedProperty` is an array of characters that describes the URI to a specification of the measured property, for example an Uniform Resource Name (URN) `urn:x-ogc:def:property:OGC::Temperature`
> `previousSeconds` is an unsigned long which stores the timestamp of the latest measurement. Adding its value the unsigned integer `samplingTimeInterval` the timestamp of the next measurement can be calculated.
> The method `process()` has to be implemented in the *SpecialSensor* class. The implementation has to be capable of transforming the raw data submitted by the sensor to the input pin of the sensor platform, into a value. The measurement has to be taken when the current timestamp matches the timestamp calculated from `previousSeconds` and `samplingTimeInterval`.

Position and Time Component This component is responsible for determining the sensor platform's position in time and space. To do so, the component retrieves the current location and the current time from the GPS device. The GPS time is used for an internal clock, since the MU is not providing one. The time is synchronized in regular intervals. The component is based upon the *Arduino Time Library*[16] and the *Satellite Count Mod*[17] of the *Tiny GPS Library*[18].

[16] http://www.arduino.cc/playground/Code/time
[17] http://www.roguerobotics.com/wikidocs/code/arduino_tinygps_satellite_count_mod
[18] http://www.arduiniana.org/libraries/tinygps/

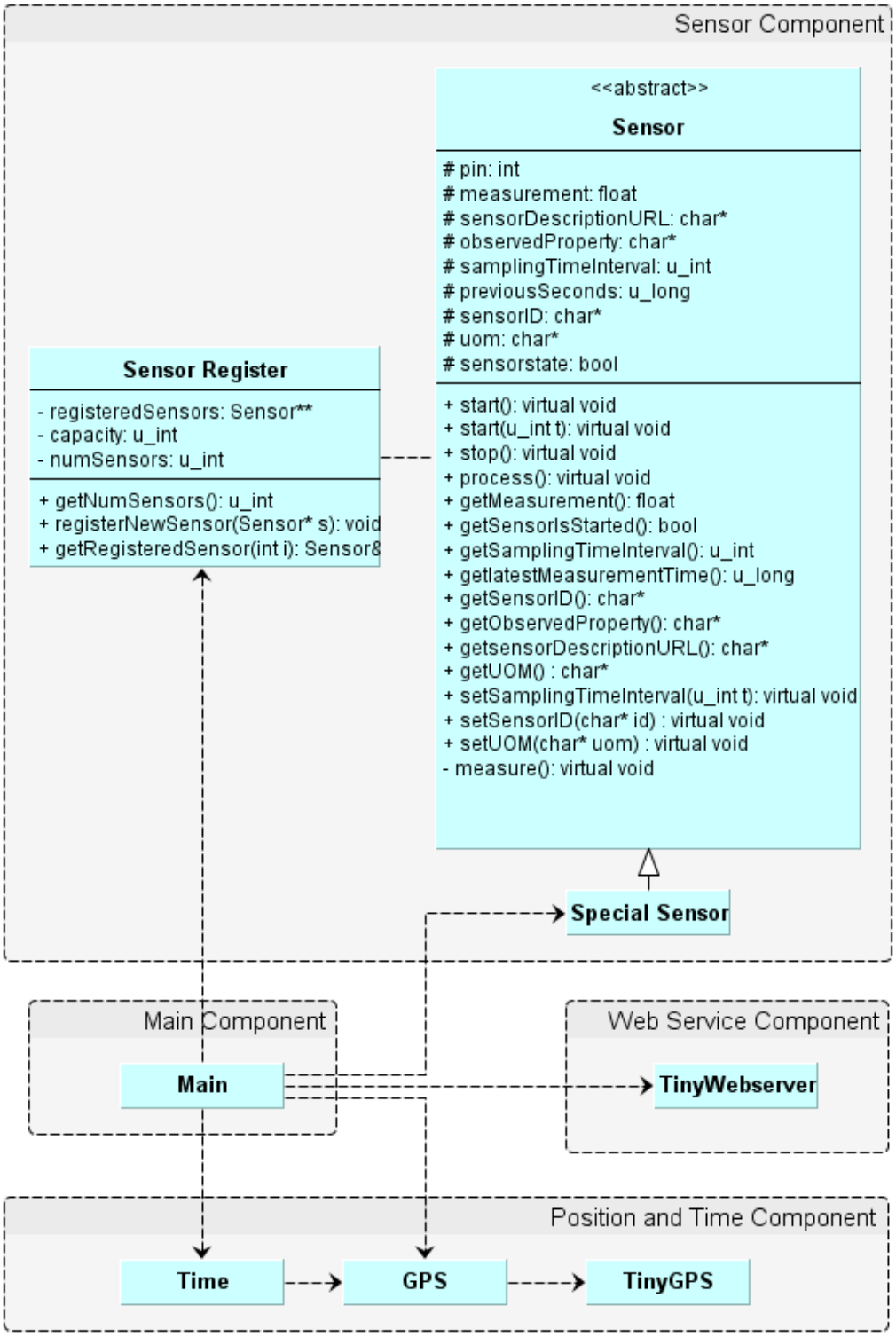


Figure 11: Class Diagram of the Sensor Platform

Web Service Component The Web Service Component is based upon the *Tiny Web Server Library*[19] to provide a RESTful Service and *EthernetDHCP* to acquire an IP-Address from a *Dynamic Host Configuration Protocol* (DHCP) service. The component is responsible for the communication with the client. The Internet Protocol (IP)-Address can be set manually, if no DHCP service is available. The *EthernetDHCP*-library is not necessary in this case.

Main Component The Main Component integrates the configuration of all other components. It is the component where the configuration of the whole AgriSenseBox can be changed. It implements the setup of the sensors as well as the configuration of the Web Service Component.

4.2.2 RESTful Service

The RESTful service is provided by the *Web Service Component* which was illustrated previously. The service enables the access on sensor resources by URI. This can be done with a web browser, cURL[20] or any other program supporting HTTP. The current implementation of the sensor platform only supports the GET-method. To access data, the address of the sensor platform has to be known. The URI scheme used by the sensor platform is depicted in section 3.3.4. When a resource is requested, the RESTful service maps the get-request against the URI scheme. If the requested resource fits a resource supported by the AgriSenseBox the corresponding data is delivered. If not, the client is redirected to the entry point of the AgriSenseBox.

4.3 Data Output

This section describes the output of the sensor platform. At the current state of work two different kinds of output are possible. On the one hand stands the entry point of the sensor platform, which lists all available sensor resources of the sensor platform, on the other hand is the data element which represents the latest measurement of a sensor resource. The attributes of each attached sensor can be discovered and evalu-

[19] http://www.webweavertech.com/ovidiu/weblog/archives/000484.html
[20] http://curl.haxx.se/

ated automatically by analyzing the information given in the entry point and the data element of the sensor platform.

Entry Point

The entry point displays a collection of sensors. It lists the ID of each sensor as well as a URI to a sensor description and a URI to the sensor resource.

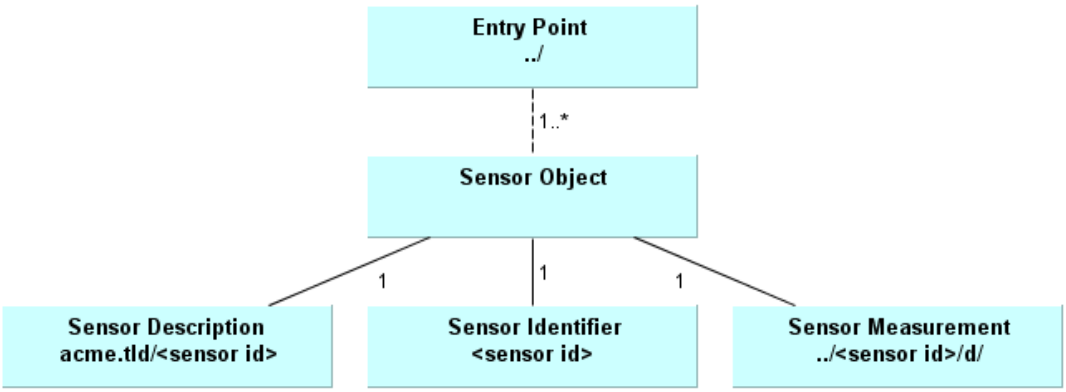


Figure 12: Data found behind the Entry Point

Listing 1 shows a sample output of the entry point. The output format is *JavaScript Object Notation* (JSON)[21]. JavaScript Object Notation (JSON) has the advantage of being more compact than XML. It is a lightweight text-based data exchange format; read- and writable for humans as well as parse- and generatable for machines [13].

The *sensorDescription* URI points to a file or a service which holds a file that is identifying the sensor. This identifier can be a static file in SensorML or StarFL or a dynamic web service. The web service creates a sensor description corresponding to the *sensorID* and possible additional parameters, for instance the file format. This service could be managed by the manufacturer of the sensor. With the *sensorDescription* URI the facts of each sensor can be discovered. It points towards a service which holds a sensor description. The *sensorResource* URI points towards the data element of the specific sensor.

[21] http://json.org/

Listing 1: Sample output of the entry point

```
{
    "sensors": [
        {
            "sensorID": "16807",
            "sensorResource": "http://192.168.178.34/16807",
            "sensorDescription": "http://www.dustindemuth.de/
                photocell.xml"
        },
        {
            "sensorID": "2856DBA4030000EF",
            "sensorResource": "http://192.168.178.34/2856DBA4030000EF
                ",
            "sensorDescription": "http://www.dustindemuth.de/ds18b20.
                xml"
        },
    ]
}
```

Data Element

The data element represents the sensor's latest measurement, encoded in a way aligned to the Observations & Measurements standard, which has been illuminated in section 2.2.2. The output is, like the output of the entry point, formatted in JSON instead of the typical XML-representation. This small change saves storage and bandwidth. The data element is generated by using a template which is stored on the sensor platform. A sample output of sensor data can be found in listing 2. Timestamps of the latest measurement and the next measurement which has to be performed are submitted in the HTTP-entity-headers *Last-Modified* and *Expires*. The format of the timestamps is conform to the HTTP standard [16]. This enables a service, which is querying the data, to detect the time when the next measurement will be available.

Listing 2: Sample output of sensor data


```
{
    "OM_Measurement": {
        "resultTime": {
            "TimeInstant": {
                "timePosition": "2012-02-26T12:49:19Z"
            }
        },
        "observedProperty": "urn:x-ogc:def:property:OGC::Temperature
            ",
        "procedure": "2856DBA4030000EF",
        "featureOfInterest": {
            "SF_SpatialSamplingFeature": {
                "type": "http://www.opengis.net/def/
                    samplingFeatureType/OGC-OM/2.0/SF_SamplingPoint",
                "sampledFeature": null,
                "shape": {
                    "type": "Point",
                    "coordinates": [
                        51.97695159,
                        7.56405019
                    ],
                    "crs": {
                        "type": "name",
                        "properties": {
                            "name": "http://www.opengis.net/def/crs/
                                EPSG/0/4326"
                        }
                    }
                }
            }
        },
        "result": {
            "uom": "degC",
            "value": 20.25
        }
    }
}
```

5 Discussion

This section discusses and evaluates results of a seven day indoor application of the proposed sensor platform. Later it illustrates limitations in connectivity, accuracy and other considerations which have to be taken into account.

5.1 Application

The AgriSenseBox has been tested in an indoor application (Figure 14). The sensors were analyzing soil moisture, air temperature and brightness at the location of a rosemary (bot.: *Rosmarinus officinalis*). Even though the testing location was located under a roof, a GPS signal could be received in most cases. The sensor platform was connected to the local WLAN. Power was supplied by wall power supply. The sensor platform was able to show the current state of a sensor's domain. When interacting with the domain (e.g. by shadowing) the output changed according to the interaction. The output of the sensor platform was monitored once per day for one week. It can be seen in table 5 and is visualized in figure 13.

Day	Bright.(‰)	Temp.(DegC)	Soil Moist.(%)	Comment
1	220	20.56	76	Bright Daylight, watered the day before
2	224	19.56	71	Bright Daylight
3	370	20.19	77	Moisturized by spraying
4	259	19.78	71	
5	10	19.12	64	Late Evening, soil still feels moist
6	358	19.50	50	
7	211	19.87	37	

Table 5: Seven day test-case with an indoor rosemary

The plot in figure 13 visualizes the three values which have been measured in a seven day series. It is visible, that the properties soil moisture and brightness are frequently changing, whilst temperature remains almost constant. Soil moisture is continuously decreasing since moisturizing on day one and day three. Brightness is varying, due to changing daylight. The measurement at day five was made at night, were no daylight was available. The temperature is constant at around 20 °C, because of the indoor use case. In an outdoor environment, the temperature profile for this week would be visible.

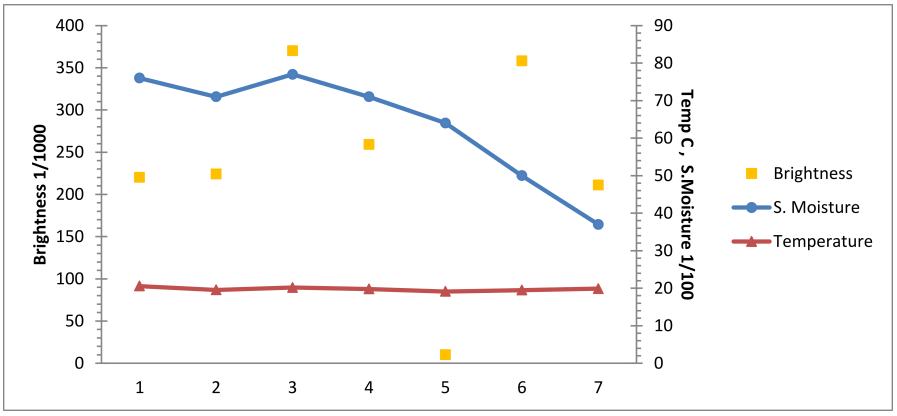


Figure 13: Plot of the test data

The Rosemary is a plant which likes dry conditions. The conditions of the rosemary could be optimized, by less watering, which makes the second moisturizing at day three dispensable. It might even harm the plant.

After deployment and successful startup, the sensor platform is ready to deliver data. It can receive up to four synchronous connections from clients. As already mentioned in section 3.3, the address of the platform has to be known. This might be tricky if the address administration is out of your control or addresses are dynamic and tend to change frequently. Finding the address can be managed in at least two ways: a) prepare a list of all sensor platform addresses when deploying the sensor platform (and take care of changing addresses), b) continuous scanning of the sensor network of interest, by iterating through all possible addresses and interpreting the response. The interpretation might be done by deciding, whether the output is a JSON-element containing a sensors-object or not. If it is, the just scanned object is a sensor platform. Depending on the amount of possible addresses in the network, method b) takes a long time, but offers the possibility to discover new sensor platforms (or rediscover platforms where the address has changed) automatically.
In both methods the address of the platform is added to the clients sensor platform address pool. The client connects to the entry point of the sensor platform and receives a list of sensor resources and sensor descriptions. It resolves the sensor description and stores the necessary parameters. If successful, the client knows which sensors are connected to the platform. If a sensor is of interest, the client accesses the corresponding sensor resource and receives the data element. It has to parse the O&M-JSON output of the data element to get the current measurement of the sensor. The client

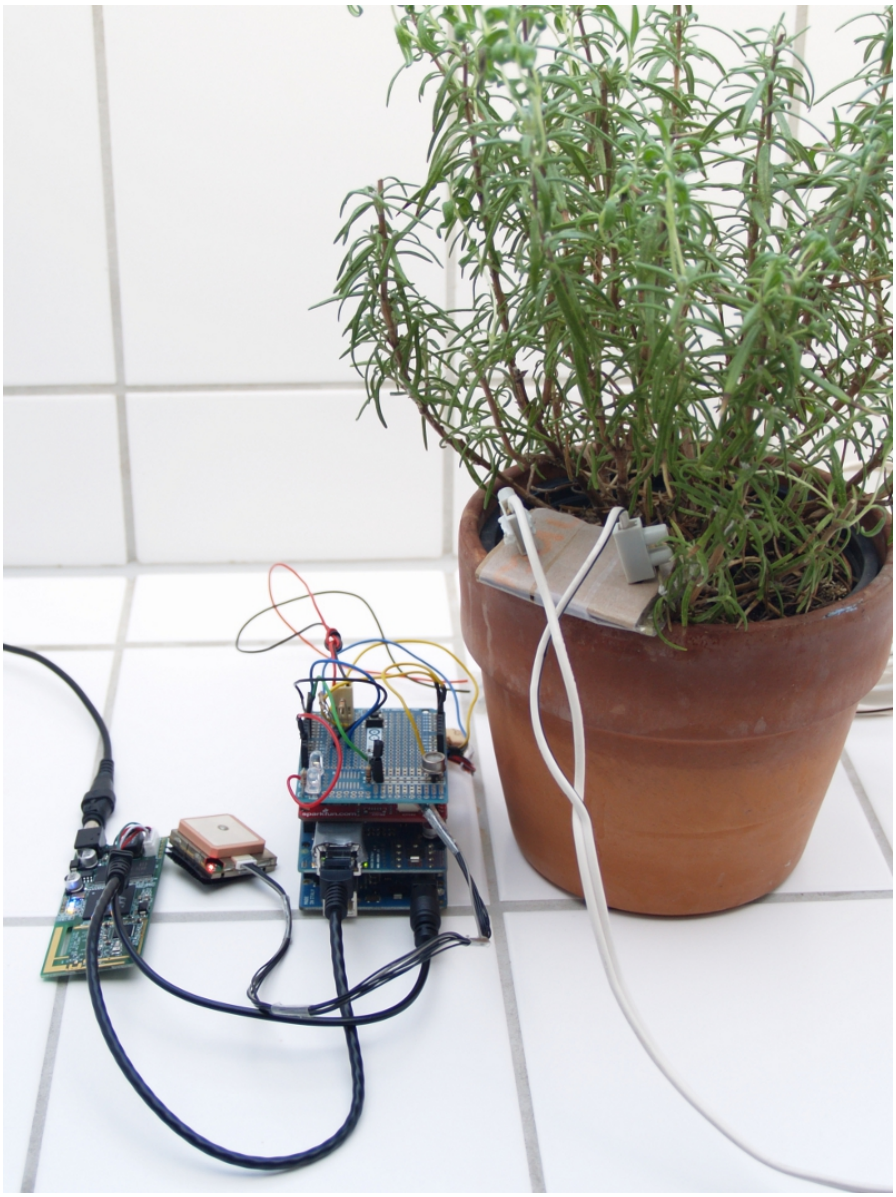


Figure 14: Indoor use of the sensor platform

has to process the timestamp which is submitted in the *Expires* header of the current data element, to receive information about the availability of the next measurement. After iterating all sensor resources which are of interest, the client should disconnect from the platform to free a socket. It has to reconnect to the platform, when the *Expires*-timestamp is reached to receive a the new measurement.

5.2 Limitations

The application of the proposed WoT integrated sensor platform is limited. Due to cost-reduction and technical possibilities restraints in connectivity and accuracy had to be accepted. This section gives a detailed overview on the consequences resulting from these design decisions.

5.2.1 Connectivity

The sensor platform is only equipped with an ethernet port. It is not WLAN-enabled from default. This can be solved with a WLAN-bridge, as stated in the definition of the CU in section 4.1.2. The additional hardware needs additional power. A measurement resulted in about 2.2 Watt higher power consumption than the platform would need without the bridge. The manufacturer of the bridge specified the power consumption with 1.5 Watt.
WLAN coverage is scarce in rural areas. Additional access-points have to be installed to ensure sufficient network coverage. This results in higher costs for constructing the sensor network. Each access-point requires a power supply.

A further possibility of integrating the sensor platform into a network is the use of WWAN. WWAN have the advantage of good network coverage, even in rural areas. The network is, in most cases, supplied by a mobile phone network provider. There is no need of adding additional access points. The disadvantage of this solution is the trend of using Network Adress Translation (NAT) in this mobile networks. In a NAT the mobile network of this cell assigns unique addresses to the devices which can only be reached from within this network, instead of an external reachable address. All of this internal addresses are mapped on a single external address. It is possible to reach the external network from the internal network, but it is not possible to reach a certain device within the network from the external network. This problem can be circum-

vented by using a Virtual Private Network (VPN) which creates a network tunnel to a third network. This network would assign an address to the sensor platform which can be resolved from external networks. This solution has been used in the SenseBox project [10] to which this thesis is aligned. This solution has higher hardware requirements and could not be integrated into the proposed sensor platform.

5.2.2 Accuracy

The sensors used in this project are very simple self-made constructions. They do not satisfy scientific accuracy and are not according to common standards for measurement. The soil-moisture and the brightness sensor are interpolating a measured voltage to a value between 0 and 100, respectively 0 and 1000. Precise sensors which would fit scientific needs would be to cost intensive for this study. Nevertheless the sensors mentioned above are providing data which is related to their domain. Changes in the domain (i.e. less light or more water) can be detected. In addition the sensors are –in more experienced variants– sufficient for an agricultural use-case if there is to decide whether an an area needs watering or more light. Creating accurate sensor-descriptions of the self-made-sensors is complex, as each sensor has to be calibrated in a testing environment, to determine its precision. However, the sensors are sufficient for a proof of concept.

5.2.3 Other Considerations

The sensor platform is limited in RAM, ROM and processing power, which requires a low memory footprint of the developed software. In addition persistency is not integrated into the current solution. The AgriSenseBox is not capable of storing measurements; a concept to do so has to be developed in the future. In this case stored data could be requested using the RESTful service.
Power consumption is another important point in sensor applications. It is expectable that power consumption can be minimized by using advanced sensors and a specialized more clever PCB-design of the sensor platform. The current modular approach is good for prototyping, but not usable in practice.

6 Evaluation

This section evaluates the proposed sensor platform and draws conclusions from the developed platform. To show perspectives for future works, a short overview on possible improvements is given.

The proposed "Web of Things Integrated Sensor Platform for Precision Agriculture" shows an alternative method of automated data retrieval from sensors. It enables access to sensor data by using a RESTful architecture. Meta data on sensors can be discovered by resolving and analyzing the sensor descriptions. Sensor data is annotated with important information which is describing the current measurement. The collected data can be accessed without limitations. The objectives, which where set in the introduction of this thesis have been achieved.
The sensor platform requires 4.5 Watt in wireless mode, 3.3 Watt in cabled mode. A solar panel and an adequate battery can match this power consumption. Nevertheless the power consumption can be considered as too high for outdoor applications. Indoor use, in large greenhouses for instance, would be a more fitting application for the platform. Indoor use would even solve the wireless network coverage problem, stated in section 5.2, since it is more easy to provide a well-covered WLAN or power infrastructure. The costs of equipping greenhouses with widespread network access can be considered as smaller than equipping outdoor areas, due to scale. Apart from WLAN, the convenient LAN can be taken into account. In cabled LAN, Power over Ethernet (PoE) can be used. PoE describes a technique where power is transmitted on the network cable. This can be used for devices with low power consumptions up to 15.4 W [25]. The sensor platform would only need one cable for both network and power, which would increase flexibility.

New sensors can be added to the platform with ease, but still require configuration. The platform has to be told which sensor is connected to which port or pin of the Main Unit. Each sensor type requires an own class definition to deal with incoming data. In addition, the URI of the sensor description has to be defined for each sensor type in the according class definition.
The costs of this platform add up to 185 Euro plus working materials (resistors, cables, etc.). A detailed list of costs can be found in table 6.

Item	Costs (EUR)
Arduino Mega	41.00
GPS	27.73
Ethernet Shield	38.49
WLAN-Bridge	24.69
Power Supply	48.00
Small Parts	5.00
Total	**184.91**

Table 6: Hardware costs of the prototype

6.1 Future Work

In future, additional components can and should be added to the sensor platform. At the current state of work, the sensor platform consists of different modules for controlling components like the Connectivity Unit and the Positioning Unit. These components might contain functions which are not necessary for the sensor platform. One unique sensor platform module, which combines the RESTful service and the position and time component could be more flexible, smaller and faster.
Storage of measurements could be implemented, to provide persistency of data. This includes methods of storing the data on SD-Card, as well as methods for querying the stored data and development of additions for the RESTful service to handle queries for the stored data.
Additional methods and structures for interaction with a sensor could be developed. The part on the "URI-Scheme" in section 3.3.4 mentioned that there is a namespace listing methods supported by the sensor. This listing is not implemented at the current state of work. This has to be done in future, to make interaction with the sensor possible. When doing so, concepts of access restriction to the control methods have to be developed as well.

6.2 Conclusion

This thesis has shown a way to build a "A Web of Things integrated Sensor Platform for Precision Agriculture" called the AgriSenseBox. The AgriSenseBox is a web-enabled, RESTful sensor platform that uses well documented web standards to make sensor data accessible. Attached sensors can be identified, due to links to sensor descriptions. The utilized sensors are sufficient for monitoring a domain, even though

there are limitations in accuracy. Apart from precision agriculture, the proposed platform can be used in other applications due to its genericness, for instance weather monitoring, home automation, air quality monitoring, pollution or radiation monitoring. Different sensors for various use cases can be attached with ease. The Main Unit of the sensor platform brings enough possibilities and processing power for upgrading, enabling it to fulfill even more complex use cases.

References

[1] Al Basset Almamou, A. ; Wrede, R. ; Kumar, P. ; Labiod, H. ; Schiller, J.: Performance evaluation of routing protocols in a Real-World WSN. In: *Information Infrastructure Symposium, 2009. GIIS '09. Global*, 2009, S. 1 –5

[2] Allen ; Michael, F. ; Vargas ; Rodrigo ; Graham ; Eric, A. ; Swenson ; William ; Hamilton ; Michael ; Taggart ; Michael ; Harmon ; Thomas, C. ; Rat'ko ; Alexander ; Rundel ; Phil ; Fulkerson ; Brian ; Estrin ; Deborah: Soil Sensor Technology: Life within a Pixel. In: *BioScience* 57 (2007), November, Nr. 10, 859–867. http://dx.doi.org/10.1641/B571008. – DOI 10.1641/B571008. – ISSN 0006–3568

[3] Anderson, Chris: *Why the Internet of Things finally makes sense*. online. http://www.theinternetofthings.eu/content/chris-anderson-why-internet-things-finally-makes-sense. Version: 2011, Checked: 2012-01-23

[4] ATMEL: *8-bit Atmel Microcontroller with 64K/128K/256K Bytes In-System Programmable Flash ATmega640/V ATmega1280/V ATmega1281/V ATmega2560/V ATmega2561/V Preliminary Summary*, http://www.atmel.com/Images/2549S.pdf

[5] Baggio, Aline: Wireless sensor networks in precision agriculture. In: *REALWSN 2005 proceedings*, 2005

[6] Banzi, Massimo: *Arduino WiFi Shield*. http://arduino.cc/blog/2011/09/17/arduino-launches-new-products-in-maker-faire/. Version: 2011, Checked: 2012-02-13

[7] Bermudez, L. ; Delory, E. ; O'Reilly, T. ; Rio Fernandez, J. del: Ocean observing systems demystified. In: *OCEANS 2009, MTS/IEEE Biloxi - Marine Technology for Our Future: Global and Local Challenges*, 2009, S. 1 –7

[8] Berners-Lee, T. ; Fielding, R. ; Masinter, L.: *Uniform Resource Identifier (URI): Generic Syntax*. RFC 3986 (Standard). http://www.ietf.org/rfc/rfc3986.txt. Version: Januar 2005 (Request for Comments)

[9] Bogena, Heye ; Huisman, Sander ; Rosenbaum, Ulrike ; Vereecken, Ansgar Weuthen A.: *SoilNet*. http://www2.fz-juelich.de/icg/icg-4/index.php?index=740. Version: 2009, Checked: 2012-03-12

[10] Bröring, A. ; Remke, A. ; Lasnia, D.: SenseBox - A Generic Sensor Platform for the Web of Things. In: *8th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous 2011)*, 2011 (Springer LNICST)

[11] Camilli, Alberto ; Cugnasca, Carlos E. ; Saraiva, Antonio M. ; Hirakawa, André R. ; Corrêa, Pedro L.: From wireless sensors to field mapping: Anatomy of an application for precision agriculture. In: *Computers and Electronics in Agriculture* 58 (2007), Nr. 1, 25 - 36. http://dx.doi.org/10.1016/j.compag.2007.01.019. – DOI 10.1016/j.compag.2007.01.019. – ISSN 0168–1699. – Precision Agriculture in Latin America

[12] CERP-IoT: Internet of Things - Strategic Research Roadmap / GRIFS-project. Version: 09 2009. http://www.grifs-project.eu/data/File/CERP-IoT%20SRA_IoT_v11.pdf, Checked: 2012-03-06. 2009. – Forschungsbericht

[13] Crockford, D.: *The application/json Media Type for JavaScript Object Notation (JSON)*. RFC 4627 (Informational). http://www.ietf.org/rfc/rfc4627.txt. Version: Juli 2006 (Request for Comments)

[14] Delin, K.A. ; Jackson, S.P.: The Sensor Web: a new instrument concept. In: *Proceedings of the SPIE International of Optical Engineering* Vol. 4284, 2001, 1-9

[15] Faludi, R.: *Building Wireless Sensor Networks: With ZigBee, XBee, Arduino, and Processing*. O'Reilly Media, 2010 (O'Reilly Series). http://books.google.com/books?id=xMC69vQJLZIC. – ISBN 9780596807733

[16] Fielding, R. ; Gettys, J. ; Mogul, J. ; Frystyk, H. ; Masinter, L. ; Leach, P. ; Berners-Lee, T.: *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616 (Draft Standard). http://www.ietf.org/rfc/rfc2616.txt. Version: Juni 1999 (Request for Comments). – Updated by RFCs 2817, 5785, 6266

[17] Fielding, Roy T. ; Taylor, Richard N.: Principled design of the modern Web architecture. In: *Proceedings of the 22nd international conference on Software engineering*. New York, NY, USA : ACM, 2000 (ICSE '00). – ISBN 1–58113–206–9, 407–416

[18] Frueh, Andrew: *GardenBot*. http://gardenbot.org/howTo/soilMoisture/. Version: 2012, Checked: 2012-02-13

[19] Gebhardt, H. ; Glaser, R. ; Radtke, U. ; Reuber, P.: *Geographie: Physische Geographie und Humangeographie*. Spektrum Akademischer Verlag, 2006 (Sav Geowissenschaften Series). http://books.google.co.in/books?id=1g6cQAAACAAJ. – ISBN 9783827415431

[20] Geonovum: *O&M*. http://geostandards.geonovum.nl/index.php/5.3.2_O%26M. Version: 2009, Checked: 2012-03-11

[21] Geonovum: *SensorML*. http://geostandards.geonovum.nl/index.php/5.3.3_SensorML. Version: 2009, Checked: 2012-03-11

[22] Goldberg, Edward M.: *RESTduino - Arduino hacking for the REST of us*. online. https://github.com/jjg/RESTduino. Version: 2011, Checked: 2012-01-23

[23] Guinard, Dominique ; Trifa, Vlad: Towards the Web of Things: Web Mashups for Embedded Devices. In: *Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences)*. Madrid, Spain, April 2009

[24] Guinard, Dominique ; Trifa, Vlad ; Mattern, Friedemann ; Wilde, Erik: From the Internet of Things to the Web of Things: Resource Oriented Architecture and Best Practices. Version: April 2011. http://www.vs.inf.ethz.ch/res/papers/dguinard-fromth-2010.pdf. In: Uckelmann, Dieter (Edt.) ; Harrison, Mark (Edt.) ; Michahelles, Florian (Edt.): *Architecting the Internet of Things*. New York Dordrecht Heidelberg London : Springer, April 2011. – ISBN 978–3–642–19156–5, Kapitel 5, 97–129

[25] LAN/MAN Standards Committee: *IEEE Standard for Information Technology — Telecommunications and Information Exchange Between Systems — Local and Metropolitan Area Networks-Specific Requirements Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications Amendment 3: Data Terminal Equipment (DTE) Power via the Media Dependent Interface (MDI) Enhancements*. http://standards.ieee.org/getieee802/download/802.3at-2009.pdf. Version: 2009

[26] Kindberg, Tim: *cooltown & ubicomp*. http://champignon.net/cooltown.php. Version: 2002, Checked: 2012-03-12

[27] Kindberg, Tim ; Barton, John: A Web-Based Nomadic Computing System / HP Laboratories Palo Alto. Version: 2000. http://www.hpl.hp.com/techreports/2000/HPL-2000-110.pdf. 2000. – Forschungsbericht

[28] Koubachi: *Koubachi*. http://www.koubachi.com. Version: 2012, Checked: 2012-03-12

[29] Li, Ke ; Zeng, Chunnian ; Liang, Hong: Better Power Management of Wireless Sensor Network. In: *Intelligent Human-Machine Systems and Cybernetics, 2009. IHMSC '09. International Conference on* Vol. 2, 2009, S. 103 –106

[30] Liqiang, Zhao ; Shouyi, Yin ; Leibo, Liu ; Zhen, Zhang ; Shaojun., Wei: A Crop Monitoring System Based on Wireless Sensor Network. In: *Procedia Environmental Sciences* 11, Part B (2011), Nr. 0, 558 - 565. http://dx.doi.org/10.1016/j.proenv.2011.12.088. – DOI 10.1016/j.proenv.2011.12.088. – ISSN 1878–0296. – <ce:title>2011 2nd International Conference on Challenges in Environmental Science and Computer Engineering (CESCE 2011)</ce:title>

[31] Malewski, Christian ; Simonis, Ingo ; Terhorst, Andrew ; Bröring, Arne: *StarFL – A new Metadata Language for Sensor Descriptions*. 2011

[32] Oerke, E.C. ; Gerhards, R. ; Menz, G. ; Sikora, R.A.: *Precision Crop Protection - the Challenge and Use of Heterogeneity*. Springer, 2010 http://www.springer.com/life+sciences/agriculture/book/978-90-481-9276-2. – ISBN 9789048192762

[33] Open Geospatial Consortium: *OpenGIS® Sensor Model Language (SensorML) Implementation Specification*. http://portal.opengeospatial.org/files/?artifact_id=21273. Version: 2007

[34] Pachube: *Pachube*. http://www.pachube.com. Version: 2012, Checked: 2012-03-12

[35] Siuli Roy, A.D. ; Bandyopadhyay, S.: Agro-sense: Precision agriculture using sensor-based wireless mesh networks. In: *Innovations in NGN: Future Network and Services, 2008. K-INGN 2008. First ITU-T Kaleidoscope Academic Conference*, 2008, S. 383 –388

[36] Sohraby, K. ; Minoli, D. ; Znati, T.F.: *Wireless sensor networks: technology, protocols, and applications*. Wiley-Interscience, 2007 http://books.google.de/books?id=aEPBTmUhyI0C. – ISBN 9780471743002

[37] Suryady, Z. ; Shaharil, M.H.M. ; Bakar, K.A. ; Khoshdelniat, R. ; Sinniah, G.R. ; Sarwar, U.: Performance evaluation of 6LoWPAN-based precision agriculture. In: *Information Networking (ICOIN), 2011 International Conference on*, 2011. – ISSN 1976–7684, S. 171 –176

[38] Uckelmann, Dieter ; Harrison, Mark ; Michahelles, Florian: An Architectural Approach Towards the Future Internet of Things. Version: 2011. http://www.springerlink.com/content/h82554jx34074622/. In: Uckelmann, Dieter (Edt.) ; Harrison, Mark (Edt.) ; Michahelles, Florian (Edt.): *Architecting the Internet of Things*. Springer Berlin Heidelberg, 2011. – ISBN 978–3–642–19157–2, 1-24

[39] USGlobalSat, Inc.: *GPS Engine Board EM-406a*, http://www.usglobalsat.com/store/download/46/em406a_ug.pdf

[40] Vasseur, J.P. ; Dunkels, A.: *Interconnecting Smart Objects with IP: The Next Internet*. Elsevier Science, 2010 (Morgan Kaufmann). http://books.google.com/books?id=mVji-YA5kgEC. – ISBN 9780123751652

[41] Wiznet: *W5100 Datasheet - Version 1.2.2*, http://www.wiznet.co.kr/UpLoad_Files/ReferenceFiles/W5100_Datasheet_v1.2.2.pdf

Appendix

List of Figures

List of Tables

List of Listings

List of Abbreviations

API	Application Programming Interface
CU	Connectivity Unit
DHCP	Dynamic Host Configuration Protocol
EGNOS	European Geostationary Navigation Overlay Service
GMT	Greenwich Mean Time
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IoT	Internet of Things
IP	Internet Protocol
IPv6	Internet Protocol version 6
JSON	JavaScript Object Notation
LAN	Local Area Network
LTE	Long Term Evolution
MIME	Internet Media Type
MU	Main Unit
NAT	Network Adress Translation
NMEA	National Marine Electronics Association
OGC	Open Geospatial Consortium
O&M	Observations & Measurements

PCB	Printed Circuit Board
PoE	Power over Ethernet
PSU	Power Supply Unit
PU	Positioning Unit
RAM	Random Access Memory
REST	Representational state transfer
ROM	Read-Only Memory
SensorML	Sensor Model Language
StarFL	Starfish Fungus Language
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver/Transmitters
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTC	Coordinated Universal Time
VPN	Virtual Private Network
WAAS	Wide Area Augmentation System
WGS	World Geodetic System
Wi-Fi	A brand name for Products using IEEE 802.11
WLAN	Wireless Local Area Network
WoT	Web of Things

WWAN Wireless Wide Area Network

XML Extensible Markup Language

Declaration of Authorship

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst habe, und dass ich keine anderen Quellen und Hilfsmittel als die angegebenen benutzt habe und dass die Stellen der Arbeit, die anderen Werken – auch elektronischen Medien – dem Wortlaut oder Sinn nach entnommen wurden, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht worden sind.

Münster, 19. März 2012

_____

Dustin Demuth