Interaction Patterns for Bridging the Gap between Sensor Networks and the Sensor Web

Arne Bröring
ITC - Faculty of Geo-Information Science and Earth Observation
University of Twente
Enschede, Netherlands
broering@itc.nl

Theodor Foerster
Institute for Geoinformatics
University of Münster
Münster, Germany
theodor.foerster@uni-muenster.de

Simon Jirka
$52°North$
Initiative for Open Source Software
Münster, Germany
jirka@52north.org

***Abstract*—The Sensor Web Enablement (SWE) initiative of the Open Geospatial Consortium (OGC) defines standards for Web Service interfaces and data encodings usable as building blocks to implement a Sensor Web for geospatial applications. These standards encapsulate heterogeneous sensors installed in existing sensor networks for web-based discovery, scheduling and access. SWE has been applied in a multitude of projects in the recent years, showing its suitability in real world scenarios. However, there is still a fundamental challenge to be tackled. While SWE enables interoperability and is well-designed towards the upper application layer, the interaction between the Sensor Web and the underlying sensor network layer is not yet sufficiently described. This work identifies five fundamental interaction patterns between the Sensor Web and sensor networks by introducing an intermediary layer, prototypically implemented using Twitter. The patterns bridge the gap between the two distinct layers and are essential for enabling future sensor plug & play within the Sensor Web.**

***Keywords*-sensor web, sensor networks, interaction patterns, Twitter**

I. Introduction

Sensor technology improves continuously and its usage is rapidly growing. Devices are becoming smaller, cheaper, more reliable, more power efficient and more intelligent. It is applied in various applications ranging from home security, environmental monitoring, precision agriculture to early warning systems [1]. The kinds of sensors utilized in these applications may be stationary or mobile, either on land, water or in the air and could gather data in an in-situ or remote manner. Due to this variety, a coherent infrastructure has become necessary to integrate heterogeneous sensors in a platform independent and uniform way. The idea of the Sensor Web describes such an infrastructure for sharing, finding and accessing sensors and their data across different applications [2]. The Sensor Web is to sensors what the World Wide Web (WWW) is to general information sources - an infrastructure allowing users to easily share their sensor resources. It hides the underlying layers, the network communication details, and heterogeneous sensor hardware, from the applications built on top of it.

The Sensor Web Enablement (SWE) [3] initiative of the Open Geospatial Consortium (OGC)[1] standardizes Web Service interfaces and data encodings for the Sensor Web. In recent years, these SWE standards have demonstrated their practicability and suitability in various projects (e.g., [4], [5], [6], [7]) and applications (e.g., [8], [9], [10]). However, due to the missing interoperability between the two layers (sensor network and Sensor Web), it is currently not possible to dynamically install sensors on-the-fly and to enable plug & play of sensors with a minimum of human configuration efforts.

Dynamically installing sensors on the Sensor Web in a plug & play manner requires concepts for integrating sensors and sensor networks with the Sensor Web. Generally, the SWE standards focus on interacting with the upper application level. This is due to the fact that they are designed from an application-oriented perspective. As a result, the interaction between the Sensor Web and the underlying sensor network layer has not been sufficiently described yet. One reason for this gap between these two layers is, that both layers are designed by different organizations and with different approaches. The Sensor Web is based on the WWW and its related protocols. On the other hand, sensor network technologies are based on lower-level protocols such as Bluetooth [11], ZigBee [12], the IEEE 1451 standards family [13] or even proprietary protocols. From an application perspective, the SWE services encapsulate the sensor network and hide these lower-level protocols. Currently, the Sensor Web and sensor network layer are integrated by manually building proprietary bridges for each pair of Web Service and sensor type. This approach is cumbersome and leads to extensive adaption effort. Since the price of sensor devices is decreasing rapidly, the manual integration becomes the key cost factor in large-scale sensor network systems [14]. Finally, improving the interoperability of the two layers, contributes to the sustainability of future web-based sensor architectures.

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

978-1-4244-5328-3/10/$26.00 ©2010 IEEE

Coherent concepts and methods are missing which describe and facilitate an infrastructure to connect the two distinct layers by ensuring a high level of adaptivity for heterogeneous sensor types. This paper describes the conceptual foundation for an intermediary layer, which integrates the Sensor Web and the sensor networks. This intermediary layer is described by generic patterns for the different interactions between Sensor Web and sensor networks. These implementation independent patterns can be used as the basis for the development of infrastructure systems to connect sensor networks with the Sensor Web.

The remainder of this paper is organized as follows. Section II outlines related work. In Section III the concept of an intermediary layer and the identified interaction patterns connecting sensors and the Sensor Web is presented. Section IV describes an implementation of the developed concepts using Twitter. The paper ends with a conclusion and discusses future work.

II. Related Work

The OGC is an industry consortium defining interoperable services for the Geospatial Web. The SWE initiative [3] as part of OGC's specification program develops standards to integrate sensors into the Geospatial Web. In particular, SWE incorporates data models for describing sensors (SensorML [15]) as well as gathered sensor data (Observations & Measurements [16]). The main Web Service interfaces are the Sensor Observation Service (SOS), the Sensor Alert Service (SAS), and the Sensor Planning Service (SPS). The SOS [17] is designed for accessing real time as well as historic sensor data, and sensor metadata descriptions. While the SOS follows the pull-based communication paradigm, the SAS [18] is capable of pushing sensor data to subscribers. To control and task sensors the SPS [19] can be used. A common application of SPS is to define simple sensor parameters such as the sampling rate but also more complex tasks such as mission planning of satellite systems.

Research in the area of middleware for sensor networks is of particular interest. Until now, there has only been done little work on design patterns for sensor network middleware [20] which is an important contribution of this paper. A comprehensive survey on middleware for *wireless* sensor networks (WSN) is provided by Wang et al. [21]. However, middleware solutions for WSN are usually focusing on lower level functionality such as cost efficient message routing. Instead, this work focuses on lowering the efforts of integrating sensor networks and the Sensor Web by envisaging a plug & play of sensors.

A work which also focuses on fast and flexible integration of sensor networks and sensor data is the Global Sensor Network middleware [14]. Its central concept is the virtual sensor abstraction in combination with data access through plain SQL queries. Sgroi et al. [22] developed a set of well defined services and interface primitives for programming

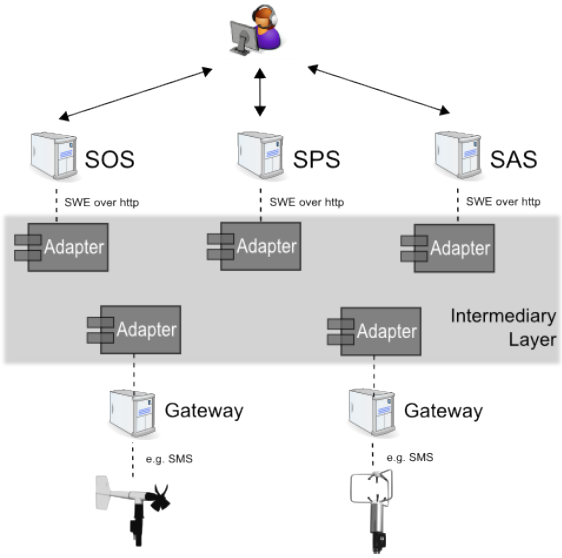

Figure 1. Architecture overview of intermediary layer

of sensor and actuator networks. However, the existing solutions do not yet address the seamless integration of sensors with standardized Web Service interfaces as defined by the OGC. Aim of this work is to leverage SWE technology and its benefits by facilitating the integration of new sensors into the Sensor Web. In future, it might be considered to use existing middleware systems as the basis of the proposed intermediary layer and reuse their functionality.

III. Interactions Between Sensor Networks and the Sensor Web

In the following, the intermediary layer concept is introduced. Subsequently, we derive basic interaction patterns from the core functionalities offered by the SWE services (Section II). The intermediary layer acts as a Broker [23] by transferring messages between sensors and services and establishes a publish/subscribe mechanism which is based on the Observer pattern [24].

Fig. 1 depicts an architecture overview of the intermediary layer. A client invokes a SWE service for example to task a sensor or to retrieve observations. The intermediary layer maintains associations to these services as well as associations to sensor gateways which supply access to connected sensors. The connections are established by adapters plugged into the intermediary layer.

Based on this architecture and independent of a certain technical realization, five interaction patterns can be identified to enable the core functionalities of the Sensor Web and support sensor plug & play. These patterns are *Service Registration*, *Sensor Registration*, *Resource Discovery*, *Data Publication* and *Sensor Tasking*. In the following subsections these patterns are described.

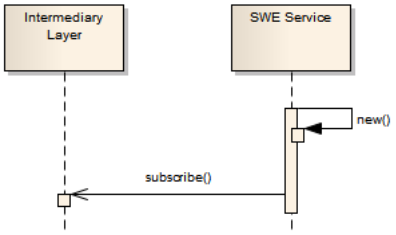

Figure 2. Service Registration pattern

A. Service Registration

This pattern describes the interaction of establishing a connection between a newly deployed SWE service and the intermediary layer.

A SWE service becomes available on the Sensor Web and is subscribed to the intermediary layer (see Fig. 2)[2].

Subscribing the service at the intermediary layer can be achieved in two alternative ways:

1. A SWE service subscribes for a specific sensor network resource. Such a resource might be a sensor but can also be a certain feature or a phenomenon observed by a sensor. This subscription requires the existence of unique identifiers for the available resources. In practice, Unified Resource Identifiers (URIs) can be utilized. However, catalogues enabling the look up of registered URIs and corresponding resources are just emerging [25], [26]. The intermediary layer has to keep track of the performed subscriptions.

2. A service subscribes itself as an anonymous observer at the intermediary layer without giving any further information. A registration for a certain resource is not required. In this case, messages are broadcasted to all registered observers.

In the first case, the mapping of subscriptions and the correct message delivery has to be managed by the intermediary layer. Hence, the approach is generally considered as a *tight coupling* of intermediary layer and Sensor Web layer. The second approach defines a *loosely coupling* between the two layers which effectively results in a light-weight approach. The advantage of the first approach is that communication can be reduced because messages are sent as a multicast only to those services for which the message is of interest. In principle, it is a matter of where to filter messages. Filtering the messages at the intermediary layer or at the specific SWE service has implications for the latency and performance and is also an issue of scalability.

The realization of the intermediary layer using Twitter as it is implemented here (Section IV) is an example for the second approach. Every service is registered as follower at all sensors and thus retrieves all sent messages.

A specialization of the Service Registration pattern is the Service Update. The purpose of this sub-pattern is to redefine subscriptions of an already registered service.

[2]The UML diagram in Fig. 2 shows the both steps, instantiation and registration, as initiated by the service itself. Implementations of this pattern might involve a service administrator which is responsible for these tasks. For the sake of simplicity of the pattern description this is considered as an autonomous act of the service.

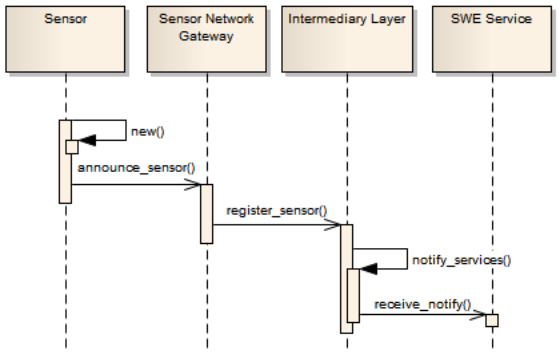

Figure 3. Sensor Registration pattern

B. Sensor Registration

The Sensor Registration pattern describes how a newly installed sensor is connected with the intermediary layer and its existence is published to subscribed services.

A sensor is added to an existing sensor network (see Fig. 3)[3]. This successful deployment is reported to the sensor gateway which publishes the existence of the new sensor to the intermediary layer and delivers its metadata as a link to an external resource, e.g., a SensorML document. Consequently, the intermediary layer notifies the registered services and forwards the metadata.

A specialization of this pattern is the Sensor Update sub-pattern. Instead of publishing a new sensor, this pattern is triggered to update the metadata description of a registered sensor. This could be for example the event of a firmware update where the sensor publishes its new functionality. Equivalent to a sensor registration, the update event is announced by the intermediary layer.

The two introduced patterns, Service Registration and Sensor Registration raise a problem: how can a service be notified about a sensor, which has not been in place, when the service registered? Alternative solutions are possible:

1. If the intermediary layer and the Sensor Web layer are loosely coupled, a solution is to repeatedly publish the metadata of registered sensors in regular intervals. Subscribed Services are subsequently notified and supplied with the information, whenever a sensor of interest is registered. However, this approach results in increased communication effort resulting from repeated metadata delivery.

[3]Wireless sensor network scenarios are working in an ad hoc fashion which means that the sensors register by themselves. In real-world use cases the registrations of sensors at the gateway are often performed by an operator. For reasons of simplicity the diagram shows the instantiation and announcement step as an autonomous act by the sensor.

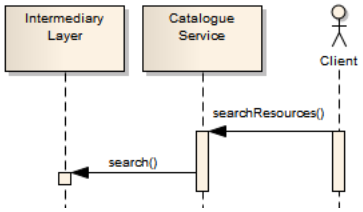

Figure 4. Resource Discovery pattern with persistent intermediary layer

2. If the two layers are tightly coupled, the intermediary layer is responsible for sending the sensor metadata. In response to a service registration, the intermediary layer transmits the metadata of the sensor. Therefore, it either queries the metadata from the sensor on demand and forwards it to the service or it triggers a repository to retrieve the sensor metadata. The repository could be a specific catalogue service (see Section III-C).

C. Resource Discovery

This pattern describes how a client can search and discover Sensor Web resources based on the intermediary layer.

A client may look for certain sets of sensor data (e.g., temperature data for Germany). This search may contain criteria such as particular phenomena, geographic regions or features of interest. Eventually, the client needs to discover SWE services which offer the gathered data or enable the submission of tasks to a sensor. For that purpose the Sensor Web incorporates catalogue services [26]. As other services, these catalogues can register at the intermediary layer. Based on the publish/subscribe mechanism established through the Registration patterns (Section III-B and III-A) all information is forwarded to the catalogue which can maintain its metadata repository. Consequently, the catalogue can immediately respond to client search queries without communicating with the intermediary layer.

However, depending on its particular realization, information may be inherently available and persistent within the intermediary layer. The realization using Twitter (see Section IV) is an example for such a persistent intermediary layer. Once posted, all sent messages are constantly available at the micro blog of the sensor or service. Those kinds of persistent intermediary layers may serve as the basis for catalogues to perform resource discovery (see Fig. 4).

D. Data Publication

The Data Publication pattern describes the interaction of publishing newly collected sensor data via the intermediary layer.

As depicted in Fig. 5, the sensor gathers data and delivers it to the sensor gateway. Subsequently, the gateway transmits the data to the intermediary layer. Before the data are delivered to the subscribed Web Services a translation of the data protocol is performed by specific adapters plugged

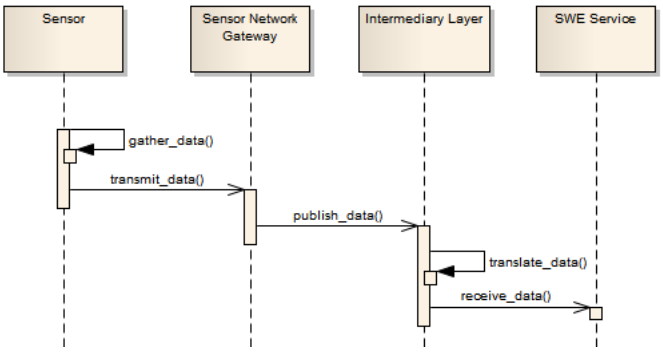

Figure 5. Data Publication pattern

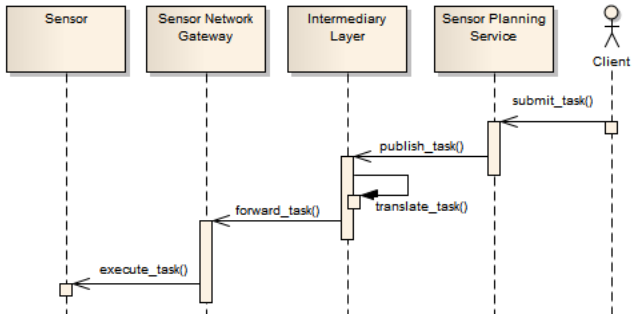

Figure 6. Sensor Tasking pattern

into the intermediary layer. A tight coupling of Sensor Web and intermediary layer results in a direct communication between the two layers. If the layers are loosely coupled an anonymous message broadcast has to be realized.

Once the data are published, a subscribed Sensor Observation Service is able to collect and store them. They are then available to clients via the standardized SOS interface. It can be accessed and retrieved in a pull-based manner. To provide the data in a push-based way, a Sensor Alert Service can be registered at the intermediary layer. The SAS receives the incoming data, filters it by certain predefined criteria and directly forwards it to interested clients.

E. Sensor Tasking

This pattern describes how tasks are submitted and transmitted from a client to a sensor via the intermediary layer.

To control a sensor in the Sensor Web, a client submits a sensor task to a Sensor Planning Service. An example of such a task description is a command for a sensor to change the sampling rate or to start measuring the temperature tomorrow at 4pm at a certain geographic location.

If the SPS is registered at the intermediary layer the Sensor Tasking pattern (see Fig. 6) applies. Once a task is received by the intermediary layer, it is translated to the concrete sensor protocol through specific sensor adapters. Subsequently, it is forwarded to the sensor gateway and eventually to the sensor.

IV. Implementation using Twitter

The described patterns are the basis for the publish/subscribe mechanism of the intermediary layer which can be implemented using different base technologies (e.g., instant messaging or message oriented middlewares). An evaluation and comparison of different approaches is current work in progress. Here, the implementation using Twitter[4] is presented.

For each service and sensor connected to the intermediary layer, an adapter realizes the identified interaction patterns. The adapters are specific for the communication via Twitter. For this research, we created a sensor adapter for the SunSPOT[5] and a service adapter for the SOS. The sensor adapter is running on the computing unit of the sensor gateway.

The sensor registration (Section III-B) involves the creation of a new Twitter profile for the sensor. The link to the sensor metadata (a SensorML document) is stored in the sensor's Twitter profile (as *Description URL*) so that it can be accessed by services at anytime. For service registration (Section III-A), a new Twitter profile is created for the service and it is registered as follower at registered sensor profiles. For data publication (Section III-D), a sensor adapter posts collected data compliant to a defined protocol as a tweet to its micro blog. The service adapters regularly check for new tweets on the micro blogs of the sensors to retrieve new data.

Sensor tasking (Section III-E) has not been implemented yet. However, to enable it, new sensors have to be registered as followers of interested services. Then, to publish a task an SPS posts a new tweet compliant to a predefined protocol. The tweet contains the ID of the sensor, which has to be tasked, as well as task parameters and corresponding values.

Building the intermediary layer on Twitter enables reusing functionality. For example, scalability and reliability are managed by Twitter. Also, in future, security mechanisms can be easily incorporated in the implemented approach, since authentication functionality is provided. However, there are also disadvantages in using Twitter. In general, the pull-based design of Twitter does not allow a true push-based architecture. Instead, the message retrieval has to be realized by regularly submitted API queries. Another disadvantage is the limited update rate of Twitter's search index which means that for example a data publication message posted by a sensor adapter is not instantly accessible by a service adapter.

Further on, there are functional limitations. Besides the restriction of the length of a tweet to 140 characters, a Twitter account (a) cannot submit more than 150 requests per hour and (b) cannot send more than 1.000 tweets a day. Restriction (a) results in a limited update rate for registered services. By using the method *statuses/friends_timeline* of the Twitter API a service adapter can maximally query 150 times an hour the recently posted tweets of all sensors it is following[6]. A more significant disadvantage is restriction (b). In the current implementation, a sensor posts one data value per tweet. Due to (b), this results in a maximum sampling rate of around 40 measurements per hour. In many sensor network applications, this would be unacceptable.

[4]http://www.twitter.com

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

V. Conclusions and Future Work

In this paper, we outline the need for mechanisms to close the gap between sensor networks and the Sensor Web. We propose to achieve this integration by introducing the concept of an intermediary layer. This layer incorporates five interaction patterns which are required to establish the integration.

First prototypical implementations of the intermediary layer and the presented patterns are published as an open source project as part of the $52°North$ Sensor Web community[7]. We have shown the universality of the patterns by implementing them in different ways based on instant messaging technologies or message oriented middleware. In this work, we outline how to implement the presented concepts based on Twitter. This allows reusing offered functionality (e.g., authentication and scalability management). However, the restrictions of Twitter (see Section IV) do not allow more complex use cases (e.g., satellite mission planning).

Typically, the SWE standards are applied to complex business processes such as incorporated in geoinformation systems or disaster management systems. Hence, the SWE services as well as the associated encodings are designed in a very flexible and generic way. So, traditionally the WS-* bindings are applied to the service models. However, the current work on version 2.0 of the SWE standards suite is in progress and will contain lightweight RESTful bindings for all services. This will allow a seamless integration of SWE with the Web of Things idea [27]. Then, they can be used as standardized RESTful APIs for heterogeneous sensor devices. In this case, the identified patterns serve as an abstract basis.

The prototypical implementations show the applicability of the approach and the identified interaction patterns, but also expose working packages for the future. Possible composition patterns and topologies for the intermediary layer have to be investigated and evaluated. Based on the described concepts, the technological mechanisms and the architecture for true sensor plug & play are required. Further, semantic challenges in the context of sensor plug & play [28] have to be tackled. Finally, the approach will be applied in

[6]To solve this issue Twitter offers the possibility for applying to *whitelist* certain accounts or IP addresses. A whitelisted entity can submit 20.000 requests per hour.

[7]http://www.52north.org/swe

real-world scenarios to demonstrate its benefits in sensor asset management.

REFERENCES

[1] D. Shepherd and S. Kumar, *Distributed Sensor Networks*. Chapman & Hall, 2005, ch. Microsensor Applications.

[2] S. Nittel, "A Survey of Geosensor Networks: Advances in Dynamic Environmental Monitoring," *Sensors*, vol. 9, pp. 5664 – 5678, 2009.

[3] M. Botts, G. Percivall, C. Reed, and J. Davidson, "OGC Sensor Web Enablement: Overview And High Level Architecture." Open Geospatial Consortium, Tech. Rep., 2007.

[4] L.-K. Chung, B. Baranski, Y.-M. Fang, Y.-H. Chang, T.-Y. Chou, and B. J. Lee, "A SOA based debris flow monitoring system - Architecture and proof-of-concept implementation," in *The 17th International Conference on Geoinformatics 2009*, Fairfax, USA, 2009.

[5] G. Schimak and D. Havlik, *ERCIM News 2009 - The Sensor Web*. European Research Consortium for Informatics and Mathematics, 2009, no. 76, ch. Sensors Anywhere - Sensor Web Enablement in Risk Management Applications, pp. 40 – 41.

[6] S. Jirka, A. Bröring, and C. Stasch, "Applying OGC Sensor Web Enablement to Risk Monitoring and Disaster Management," in *GSDI 11 World Conference, Rotterdam, Netherlands*, June 2009.

[7] C. Stasch, A. C. Walkowski, and S. Jirka, "A Geosensor Network Architecture for Disaster Management based on Open Standards." in *Digital Earth Summit on Geoinformatics 2008: Tools for Climate Change Research.*, M. Ehlers, K. Behncke, F. W. Gerstengabe, F. Hillen, L. Koppers, L. Stroink, and J. Wächter, Eds., 2008, pp. 54–59.

[8] A. Aasa, O. Järv, and R. Ahas, "Developing a model to determine the impacts of climate change on the geographical distribution of tourists," in *Sensing a Changing World 2008*, K. Lammert and L. Arend, Eds. Wageningen University, 2008, pp. 55 – 58.

[9] A. Bröring, E. H. Jürrens, S. Jirka, and C. Stasch, "Development of Sensor Web Applications with Open Source Software," in *First Open Source GIS UK Conference (OSGIS 2009), 22 June 2009, Nottingham, UK*, 2009.

[10] A. Bröring, S. Jirka, S. M. Benmoh, and E. H. Jürrens, "UWeather: A Web Portal for your Weather Data," *GISRUK 2009 - mashup challenge*, 2009.

[11] B. SIG, *Bluetooth Specification Version 3.0*, Bluetooth Special Interest Group Std., 2009.

[12] *ZigBee Specification*, ZigBee Standards Organization Std. 053 474r17, 2008.

[13] K. Lee, "IEEE 1451: A Standard in Support of Smart Transducer Networking," *Instrumentation and Measurement Technology Conference, 2000. IMTC 2000. Proceedings of the 17th IEEE Volume 2*, vol. 2, pp. 525 – 528, 2000.

[14] K. Aberer, M. Hauswirth, and A. Salehi, "Middleware support for the Internet of Things," *5. GI/ITG KuVS Fachgespraech - Drahtlose Sensornetze*, pp. 15 – 19, 2006.

[15] M. Botts, "OGC Implementation Specification 07-000: OpenGIS Sensor Model Language (SensorML)," 2007.

[16] S. Cox, "OGC Implementation Specification 07-022r1: Observations and Measurements - Part 1 - Observation schema," Open Geospatial Consortium, 2007.

[17] A. Na and M. Priest, "OGC Implementation Specification 06-009r6: OpenGIS Sensor Observation Service (SOS)," 2007.

[18] I. Simonis, "OGC Best Practices 06-028r3: OGC Sensor Alert Service Candidate Implementation Specification," Open Geospatial Consortium, Tech. Rep., 2006.

[19] ——, "OGC Implementation Specification 07-014r3: OpenGIS Sensor Planning Service," Open Geospatial Consortium, Tech. Rep., 2007.

[20] K. Tei, Y. Fukazawa, and S. Honiden, "Applying Design Patterns to Wireless Sensor Network Programming," in *Proceedings of 16th International Conference on Computer Communications and Networks, 2007 (ICCCN 2007)*, 2007, pp. 1099–1104.

[21] M.-M. Wang, J.-N. Cao, and J. Li, "Middleware for Wireless Sensor Networks: A Survey," *Journal Of Computer Science and Technology*, pp. 305 – 326, 2008.

[22] M. Sgroi, A. Wolisz, A. Sangiovanni-Vincentelli, and J. Rabaey, "A Service-Based Universal Application Interface for Ad Hoc Wireless Sensor and Actuator Networks," in *Ambient intelligence*, W. Weber, J. Rabaey, and E. Aarts, Eds. Springer Verlag, 2005.

[23] F. Buschmann, K. Henney, and D. C. Schmidt, *Pattern-Oriented Software Architecture*. Wiley, 2007.

[24] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Resusable Object-Oriented Software*. Addison-Wesley Professional, 1995.

[25] S. Jirka and A. Bröring, "OGC Discussion Paper 09-112 - Sensor Observable Registry," Open Geospatial Consortium, Tech. Rep., 2009.

[26] S. Jirka, A. Bröring, and C. Stasch, "Discovery Mechanisms for the Sensor Web," *Sensors*, vol. 9, 2009.

[27] D. Guinard and V. Trifa, "Towards the Web of Things: Web Mashups for Embedded Devices," in *Proceedings of the Second Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web*, Madrid, Spain, April 2009.

[28] A. Bröring, K. Janowicz, C. Stasch, and W. Kuhn, "Semantic Challenges for Sensor Plug and Play," in *Web & Wireless Geographical Information Systems (W2GIS 2009), 7 & 8 December 2009, Maynooth, Ireland*, ser. LNCS, J. Carswell, S. Fotheringham, and G. McArdle, Eds., no. 5886. Springer, 2009, pp. 72–86.