Review Article

Sensor Resource Sharing Approaches in Sensor-Cloud Infrastructure

Yujin Lim and Jaesung Park

1 Department of Information Media, University of Suwon, San 2-2, Wau-ri, Bongdam-eup, Hwaseong, Gyeonggi-do 445-743, Republic of Korea
2 Department of Information Security, University of Suwon, San 2-2, Wau-ri, Bongdam-eup, Hwaseong, Gyeonggi-do 445-743, Republic of Korea

Correspondence should be addressed to Jaesung Park; jaesungpark@suwon.ac.kr

Received 7 December 2013; Accepted 22 January 2014; Published 15 April 2014

Copyright © 2014 Y. Lim and J. Park. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Generally, wireless sensor networks have been designed to support predefined application services. The tight coupling between a network and application limits the usability of sensor data. To increase the utilities of wireless sensor networks, sensor-cloud infrastructure has been proposed to provide an open and flexible platform to gather, store, use, and share data from heterogeneous sensors. In this paper, we introduce sensor-cloud infrastructure and present not only comprehensive technical issues like sensor description and publish/subscribe middleware but also applications using sensor-cloud infrastructure. When the issues are tackled in the development of sensor-cloud system, the attractiveness of new applications and services will considerably be increased for end users.

1. Introduction

A wireless sensor network (WSN) consists of spatially distributed sensors to monitor physical or environmental conditions [1, 2]. It is generally invincible trend to monitor a region and fetch useful data about the surroundings. WSNs are utilized in various areas, such as healthcare, military, critical infrastructure surveillance, and environmental monitoring. Since sensors are limited in energy, processing power, memory, and communication bandwidth, a WSN usually has design constraints. Therefore, WSNs have been designed to support a specific application service in mind. The tight coupling between a network and application limits the usability of sensor data because only the targeted applications freely use and manage physical sensors as well as sensor data within a WSN while applications outside the WSN cannot use the sensors and data directly. Thus, sensor data used for specific applications cannot be easily shared among different groups of users, and different WSNs cannot be easily interconnected. If there was an infrastructure on which applications could share different kinds of physical sensors easily, many new applications and services could be provided. In the midst of these issues, the emergence of cloud computing is seen as one of efficient solutions.

Cloud computing has been evolved as the future generation computing paradigm. It is a model for enabling convenient, on demand network access to a shard pool of configurable computing resources [3]. The configurable resources involve networks, servers, storage, applications, and services. Cloud computing platform dynamically provisions, configures, and reconfigures the servers when needed by users. On the platform, users could switch to their application by connecting to the server and start working without any hassle, even though they do not know the locations of servers.

Sensor-cloud infrastructure (SCI) is the extended form of cloud computing to manage the sensors which are scattered throughout WSNs [4, 5]. Since physical sensors in WSNs are directly bound to their specific applications as well as services, users other than the relevant services cannot use these physical sensors when needed. It makes wastage of valuable
sensor resources that may be utilized when integrating with other applications' infrastructures. Therefore, these sensors need to be supervised by some special sensor management schemes. SCI provides the sensor management to improve sensor data usability by virtualizing the physical sensors on a cloud computing platform. On SCI, users can use and control physical sensors through virtual sensors. Users can destroy their virtual sensors quickly when they become unnecessary. Besides, virtual sensors can be monitored to maintain the quality of service (QoS).

Cloud computing services for IT resources provide users with virtual servers. Users can use the virtual servers with no concerns about the locations of actual servers or their detailed specifications. Similarly, SCI for sensor resources makes users use the sensors with no worrying about their locations and detailed specifications by using virtual sensors.

In this paper, we present a survey on SCI and address technical issues. Section 2 introduces architecture of SCI. Section 3 reviews sensor description methods and Section 4 analyzes the characteristics of middleware based on publish/subscribe data exchange model. The projects and applications based on SCI are described in Section 5. Finally, conclusion is drawn in Section 6.

2. Architecture of Sensor Cloud

In a WSN system, sensors are utilized by their own application for a specific purpose. The only application in the system freely handles the sensor data and the sensor itself. SCI enables the sensors to be shared and utilized by multiple applications by virtualizing the physical sensor on a cloud computing platform. Figure 1 shows architecture of SCI, which is divided mainly into three components. The first component is IT resource pool to provide IT resources such as processors, storage, network, and database. WSNs have issues regarding the limited resources. In other words, the issues include short communication range, security and privacy, reliable data delivery, power consideration, storage capacity, and processing capabilities. To solve the issues, cloud computing platform dynamically provisions, configures, and reconfigures IT resources. In other words, SCI achieves the highly scalable hardware and software resources on the platform. The second component is a pool of sensor resources. SCI virtualizes physical sensors as virtual sensors in order to share various physical sensors from different owners. Each virtual sensor is to use and control one or more physical sensors. A virtual sensor group is composed of one or more virtual sensors. To create virtual sensors and virtual sensor groups, service templates are prepared by SCI providers. The sensor resources include physical sensors, virtual sensors, virtual sensor groups, and service templates. The last component is end users. Users request virtual sensors or virtual sensor groups by selecting service templates. Dynamically grouped virtual sensors are provisioned automatically in response to the requests from users. Then, they use and control their virtual sensors/virtual sensor groups with standard functions. For example, they can activate or deactivate their virtual sensors, check their status, and set the frequency of data collection from them. When virtual sensors become unnecessary, users release them.

Each application requires a set of physical sensors in the sensor resource pool for its specific purpose. Thus, publish/subscribe mechanism is being employed for choosing the appropriate physical sensors [6]. As shown in Figure 2, there are three players in SCI, and the interactions between the players take place in the publish/subscribe manner. A sensor owner has its own physical sensors. To allow end users to use the physical sensors, the owner publishes the sensors with their metadata. The metadata include the types and locations for the physical sensors. The owner deletes the publication of the sensors when the owner wants to quit the sharing. End user has one or more applications or services that use sensor data. An end user subscribes to one or more physical sensors in order to retrieve sensor data from them for its applications. To do this, the user requests the use of virtual sensors or virtual sensor groups that satisfy the requirements. The user can control its virtual sensors directly.
Figure 2: Relationship among players in SCI.

or via a web browser. The user can monitor the status of the virtual sensors and use the virtual sensors with no detailed knowledge about physical sensors. The SCI provider manages SCI service. The provider manages and monitors sensor resources such as physical sensors, virtual sensors, virtual sensor groups, service templates, and user interface. It also provides middleware for the publish/subscribe mechanism.

The advantages of SCI are presented as follows. End users can freely control and use physical sensors by using virtual sensors without worrying about the implementation details such as location and specification of physical sensors. End users can also create the virtual sensors dynamically for new services, monitor the states of the virtual sensors, and release the sensors when they are unnecessary. Sensor owners can check the usage of their physical sensors and charge for the usage. The cost of IT resources and WSN infrastructure is reduced by integrating between cloud and WSN [7]. Even though SCI could provide flexible and cost efficient platform, there are requirements to be satisfied to fully take advantage of the SCI services. The IT resources and physical sensors should be prepared prior to operation of SCI [8], and continuous data connectivity is needed among SCI players [9].

3. Sensor Description

In WSNs, there are many different physical sensor types, from simple visual thermometers to complex electron microscopes and earth observing satellites. To allow end users to use physical sensors, a framework is needed to define the geometric, dynamic, and observational characteristics of sensors. On the framework, sensor owners represent the metadata for any physical sensor, such as the type of physical sensor, the location, and the accuracy. Besides, the framework should provide a mapping process between physical sensors and virtual sensors to describe how to translate commands coming from users to virtual sensors into commands for the corresponding physical sensors.

The Sensor Web Enablement (SWE) initiative of the OGC (Open Geospatial Consortium) standardizes the sensor model language (SensorML) to model and encode sensor metadata [10]. Initially, SensorML was developed for describing the geometric and radiometric properties of dynamic remote satellite sensors. In 2000, SensorML became a part of SWE initiative of the OGC. Approved as an international and open technical specification by OGC, SensorML provides standard models and an XML encoding for describing sensors and measurement processes. SensorML models detectors and sensors as processes, which convert real phenomena to data by defining metadata, inputs, outputs, parameters, and method. The metadata is not meant to provide detailed descriptions on hardware but to provide functional model for sensors. It includes identifier, classifier, constraints, capabilities, characteristics, contacts, and references, as well as inputs, outputs, parameters, and system location. The SWE also specified Observations and Measurement (O&M) standard to encode sensor data measured.

Even though SensorML is specified to efficiently use and manage huge amount of sensed data, the encoding and parsing of XML codes are not lightweight enough for resource constrained sensors. In addition, the flexibility of SensorML in modeling sensors causes the interoperability issue at the encoding level because there are multiple ways to encode sensor characteristics.

Since the SCI is composed of a lot of sensors and the number of applications using the SCI keeps increasing, the volume of data inside the SCI also increases accordingly. Furthermore, the devices, data formats, and measurement procedures become more and more heterogeneous. Therefore, as the scale and complexity of SCI increase, it becomes more important to manage not only sensors but also the data generated by them in an efficient manner. On the other hand, users want to use SCI in an intuitive way without involving technical details. Semantic sensor web and ontology are expected to satisfy both requirements of the SCI and users by assisting in managing, querying, and combining sensors and observation data [11]. There are a number of sensor
ontologies. They differ in their main purposes. For example, [12, 13] focused on the sensor data while [14, 15] concentrated on the role of stimuli, observed properties or processes. MMI focuses more on systems rather than measurements or sensor types [16]. Ontosensor covers wider ranges of sensor concept [17]. The surveys on the sensor ontology could be found in [18].

The W3C Semantic Sensor Network Incubator Group (SSN-XG) produced Semantic Sensor Network Ontology (SSNO) to describe the capabilities and properties of sensors, measurement processes used, and the resulting observations [19]. The SSN-XG has transitioned into the Semantic Sensor Networks Community Group. The SSNO is encoded in OWL (Web Ontology Language) [20] and aligned to DOLCE-UltraLite (DUL) [21].

Even though SensorML from OGC provides syntactic interoperability and SSNO provides semantic interoperability, it is difficult to integrate SSNO with SensorML. Starfish Fungus Language (StarFL) is proposed to relate SSNO and SWÆ [22]. StarFL focuses on the common features shared by a wide range of users and communities without supporting details of sensors. Unlike the Sensor Interface Descriptor approach which includes the description of sensor communication interfaces [23, 24], StarFL focuses on modeling sensor capabilities leaving aside the description of the actual sensor interface.

4. Publish/Subscribe Middleware for Sensor Cloud

SCI could support wide ranges of application services by providing various sensed data. However, sensors are different in their hardware and operating systems, and applications have their own specific requirements. Therefore, it is difficult for application developers to understand these specific details when they implement application services. Therefore, a middleware based on an appropriate communication method is required to glue the gap between the various details residing inside of SCI and specific requirements of applications so as to facilitate the development of various application services.

A user using SCI is provided with two interfaces. One interface is used to express the information the user is interested in, and the other one is used to receive the information from SCI. It means that users are not interested in the sensors providing the information and how the information is delivered to them if they receive information they want. Therefore, unlike the request/response data exchange model prevailed in the Internet where a sender and a receiver communicate synchronously by constructing an individual point-to-point connection between them, the information in SCI is exchanged asynchronously between an information producer and its consumer when specific events specified by the consumer are triggered. In addition, since SCI is a distributed system composed of a lot of different types of sensors, it needs a scalable information distribution method.

Considering these specific characteristics of SCI, a middleware based on publish/subscribe interaction scheme has drawn a lot of attention [25]. Usually, a publish/subscribe system operates by the interactions among three entities—a subscriber (end user), a publisher (sensor owner), and a broker (SCI provider). A subscriber registers an event or a pattern of events it wants to receive with a broker. A publisher produces the information and sends it to a broker. A broker maintains the subscription information and filters the information sent from publishers according to the rules that specified the subscribers. If an event is detected, the broker distributes the event to all the subscribers who subscribe to the event. Therefore, by taking into account the constraints in the SCI, it is important to implement efficient event filtering and delivery services for a publish/subscribe middleware.

The publish/subscribe middleware decouples the production and consumption of information in time and space. In other words, interacting parties are not necessarily participating in the information delivery at the same time. Therefore, they do not need to know the availability of the other parties. In addition, by decoupling space, interacting parties do not need to know each other (e.g., location, the number of publishers/subscribers, etc.). Therefore, the publish/subscribe middleware could increase the scalability in information delivery, which suits the design requirements of the SCI.

There have been a lot of proposals for publish/subscribe middleware in the literature. Mires is a topic-based publish/subscribe middleware for WSN [26]. It uses the message-oriented paradigm provided by TinyOS. Even though only the messages referring to the subscribed topic are delivered to a message sink, Mires is not energy efficient because it uses broadcast protocol to distribute subscriptions. In addition, publishers in Mires must know not only the event subscribers but also the paths the event must follow.

Since applications using SCI handle very complex situations, they need to interpret data from heterogeneous sensors at the same time so as to detect a composite event. However, since it is required for sensors to individually detect primitive events and cooperatively determine the relationship among these events, it is not easy to support composite events in publish/subscribe middleware. To overcome these difficulties, a PSWare is proposed to support both the primitive and composite events in a publish/subscribe manner [27]. The authors proposed an event definition language (EDL) to accurately define primitive events and describe the spatial and temporal relationship among the events. They also developed a compiler which compiles the applications programmed by EDL. In addition, an event detection protocol is proposed to detect composite events in an energy-efficient manner.

TinyDDS is proposed for a lightweight implementation of the DDS (Data Distribution Service) standardized for topic-based publish/subscribe middleware by OMG (Object Management Group) [28]. TinyDDS uses nesC to implement the standard interfaces for event subscription and publication for DDS, which are described in IDL (Interface Definition Language). A self-configuring routing protocol in TinyDDS delivers events adaptively according to current network states by autonomously balancing the tradeoffs among conflicting objectives. TinyDDS uses evolutionary multiobjective optimization mechanism (called MONSOON) in order to seek the optimal tradeoffs among operational objectives.
and adjusts parameters in its routing protocols accordingly to satisfy QoS requirements of an event delivery service. However, since TinyDDS is a topic-based publish/subscribe middleware, the granularities in event description may not be fine enough to satisfy the requirements of the applications using SCI.

Middleware applications based on overlay network are proposed in [29, 30]. In [29], a tree is constructed on top of the physical sensors to deliver subscriptions and published data. It is composed of a virtualization component and indexing component. The former maintains a naming structure by assigning each node a unique virtual address while the latter determines the notification paths for given subscriptions and publications. However, congestion could occur in a tree, which deteriorates the performance of the event delivery service. A content-based publish/subscribe middleware that employs a general message mapping and routing mechanism is proposed in [30]. Unlike the middleware in [29] that constructs a single tree, TinyMQ partitions sensors into multiple trees and maps a subscription/publish into multiple sensors in the different trees so as to facilitate load-balancing and increase availability. TinyMQ also allows cross-tree routing to relieve the burden of a broker. However, there could be considerable cost for tree maintenance, especially when sensors leave and join the tree dynamically.

Most of the existing publish/subscribe middleware applications support only stationary and interconnected sensors. Since they depend on the multipath routing protocols to deliver subscription and published data, they are not applicable to the environment where static sensors are sparsely deployed or sensors move around. A content-based publish/subscribe middleware supporting mobile sensors or static sensors sparsely deployed is proposed in [31]. The authors proposed to use mobile phones as data mules for relaying subscription and published data between a broker and sensors. They also proposed a method to adaptively update the locations of mobile mules so as not to deteriorate the scalability of publish/subscribe services. Despite the fact that most of the publish/subscribe middleware applications focus on the event delivery service among the players, the authors in [32] use the publish/subscribe mechanism to deliver mobile agents (MAs). Compared with other types of middleware, MA-based middleware could save the energy for communication between sensors because the size of mobile codes is much smaller than that of the data collected in a sensor. In addition, since small-sized mobile agents could be sent at any time according to the demands of a network, the MA-based approach provides flexibility in program installation and update. The authors in [32] implement a mobile agent manager on top of the publish/subscribe service. With this decoupling, the mobile agent manager could concentrate on details related to agents such as creating, moving, and executing codes without concerning the transmission details. However, since an agent represents an application, the failure of agent transfer results in the failure of the application. Therefore, the fidelity of the publish/subscribe mechanisms should be evaluated in an environment where there are constraints in network connectivity and available sensors.

5. Case Study

Since sensors are very specific to their applications, the numbers of organizations that can provide the sensor services are very limited. However, on top of SCI, it is possible to include sensors to realize a variety of applications and services.

IoTCloud develops a sensor-centric framework which supports an extensible set of sensor types and large numbers of geographically distributed smart objects [33]. As shown in Figure 3, the framework consists of four components: IoTCloud controller, distributed message broker, sensors, and clients. IoTCloud controller manages the other components and provides SOAP (Simple Object Access Protocol) web service [34] for sensor publication, discovery, subscription, and control. Message broker handles the low level details of message routing. Sensor is a component which generates a time state updates. IoTCloud controller, distributed message broker, and controllers are able to discover, subscribe to, and control sensors by using the web service API provided by IoTCloud controller. The IoTCloud framework is deployed on the FutureGrid [35]. FugureGrid is a part of the XSEDE (Extreme Science and Engineering Discovery Environment) [36]. The FutureGrid provides a capability that makes it possible for researchers to tackle complex research challenges in computer science related to the use and security of grids and clouds.

The project SensorCloud is a member of the project cluster Trusted Cloud which is funded by the German Federal Ministry of Economics and Technology [37]. Trusted Cloud aims to develop and evaluate innovative, robust, secure, and law conform cloud computing solutions. The goal of SensorCloud is to develop a cloud-based platform that integrates basically every kind of distributed internet-connected sensors and actuators. The project uses the smart home scenario to capture requirements when designing technical solutions. For the scenario, SensorCloud serves as a central hub to which sensors and actuators can be connected. The hub manages sensors and actuators and aggregates their data.
from different domains, places, and owners at the same time. Though serving the hub, SensorCloud provides an extensible and flexible service platform that is populated by applications from third-party developers. Bringing together sensor owners, end users, and cloud provider on the platform, the SensorCloud enables a marketplace for customer solutions that minimizes costs and hurdles for all involved players.

SAaaS (Sensing and Actuation as a Service) is to implement services to provide indexing and querying methods applied to things, that is, heterogeneous resources such as sensors and actuators [38]. The resources are aggregated according to a given thing-like semantics and provided to final users. For the services, SAaaS presents a new architecture which is composed of three components as shown in Figure 4: hypervisor, autonomic enforcer, and volunteer cloud manager. Hypervisor works at the level of a single sensor node, where it abstracts away either embedded sensors available on a device or standalone sensors belonging to WSN. It relays commands and data, abstraction of devices and capabilities, virtualization of abstracted resources, semantic labeling, and thing-enabled services. The abstraction has been developed according to SWE standard defined by OGC. Autonomic enforcer is tasked with enforcement of policies, local versus global policy tie-breaking, subscription management, and cooperation on overlay instantiation. Volunteer cloud manager is in charge of exposing the generated cloud, framing reward mechanisms and policies in synergy with SLA matching, to be mediated by QoS metrics and monitoring, as well as indexing duties to allow for efficient discovery of resources.

There are many other applications that are emerging based on SCI. Google Health provides a service for users to access their personal health and wellness information. Users are allowed to monitor their health records at collaborated cloud health service providers. However, now the service has been discontinued. Microsoft HealthVault is another personal health record provider. It helps users to gather, store, use, and share their health information, such as allergies, adverse drug reactions, and prescription record [39]. IBM India has considered SCI in the context of sensor web for smarter cities (SENSIT) to assist India with rainfall monitoring and flood forecasting [40]. Fujitsu provides Akisai as an agricultural cloud to analyze data from field sensors [41]. It combines data from a nationwide network of sensors and cameras in fields that keep track of factors including soil temperature, moisture, rainfall, and humidity. Farmers will benefit from detailed data on factors such as their profitability per crop and expenses.

Even though these case studies show the potential of the SCI, there are several issues need to be taken into consideration when designing an application using a SCI. Most of the applications require reliable and continuous data transfer. However, real radio propagation environments are prone to errors which make a reliable data transfer difficult and cause a connection failure. Therefore, application developers must consider these scenarios into consideration to avoid negative effects of unreliable communications. The other issues are related to security. For example, a web-based interface is used by many people with different authorization.

Therefore, the system should be able to provide different authorization roles for different types of users. In addition, since most of the data collected by sensors are related to privacy, the data exchanged and stored in the SCI must be protected appropriately. Therefore, privacy policies including secure key management and data integrity must be addressed.

6. Conclusion

In this paper, we introduced SCI and addressed technical issues such as sensor description and publish/subscribe middleware. To allow end users to use physical sensors, a sensor description framework is needed to define the characteristics of sensors. As the scale and complexity of SCI increase, ontologies and semantic approaches are considered to manage not only sensors but also the data generated by them in an efficient manner. SCI could support wide ranges of application services by providing various sensed data. The publish/subscribe middleware glues the gap between the various details residing inside of SCI and specific requirements of applications.

Integrating the sensors with cloud opens the doors to provide numerous applications and services. However, to maximize the potential of SCI, many research challenges including system design, energy efficiency, efficient information dissemination, collective intelligence harvesting, and interface standardization need to be resolved.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the GRRC program of Gyeonggi province. ([GRRC SUWON2013-B4), Research on Precision Location Tracking System for Real-Time Situation].
References

[1] K. Romer and F. Mattern, “The design space of wireless sensor networks,” IEEE Wireless Communications, vol. 11, no. 6, pp. 54–61, 2004.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” IEEE Communications Magazine, vol. 40, no. 8, pp. 102–114, 2002.

[3] S. K. Dash, J. P. Sahoo, S. Mohapatra, and S. P. Pati, “Sensor-cloud: assimilation of wireless sensor network and the cloud,” in Advances in Computer Science and Information Technology, Networks and Communications, vol. 84 of Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pp. 455–464, Springer, Berlin, Germany, 2012.

[4] A. Alamri, W. Shadab, M. M. Hassan, M. S. Hossain, A. Alelaawi, and M. A. Hossain, “A survey on sensor-cloud: architecture, applications, and approaches,” International Journal of Distributed Sensor Networks, vol. 2013, Article ID 917923, 18 pages, 2013.

[5] M. Yuriyama and T. Kushida, “Sensor-cloud infrastructure—physical sensor management with virtualized sensors on cloud computing,” in Proceedings of the 13th International Conference on Network-Based Information Systems (NBiS ’10), pp. 1–8, Takayama, Japan, September 2010.

[6] M. Gaynor, S. L. Moulton, M. Welsh, E. la Combe, A. Rowan, and J. Wynne, “Integrating wireless sensor networks with the grid,” IEEE Internet Computing, vol. 8, no. 4, pp. 32–39, 2004.

[7] K. Lee and D. Hughes, “System architecture directions for tangible cloud computing,” in Proceedings of the International Workshop on Information Security and Applications (IWSA’10), pp. 258–262, Qinhuangdao, China, October 2010.

[8] M. Yuriyama, T. Kushida, and M. Itakura, “A new model of accelerating service innovation with sensor-cloud Infrastructure,” in Proceedings of the Annual SRII Global Conference (SRII ’11), pp. 308–314, San Jose, Calif, USA, April 2011.

[9] B. Jit, J. Maniyeri, K. Gopalakrishnan et al., “Processing of wearable sensor data on the cloud—a step towards scaling of continuous monitoring of health and well-being,” in Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC ’10), pp. 3860–3863, Buenos Aires, Argentina, September 2010.

[10] M. Botts and A. Robin, “OpenGIS sensor model language (SensorML) implementation specification,” OpenGIS Implementation Specification OGC 07-000, The Open Geospatial Consortium, 2007.

[11] A. Sheth, C. Henson, and S. Sahoo, “Semantic sensor web,” IEEE Internet Computing, vol. 12, no. 4, pp. 78–83, 2008.

[12] S. Avancha, C. Patel, and A. Joshi, “Ontology-driven adaptive sensor networks,” in Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS ’04), pp. 194–202, Boston, Mass, USA, August 2004.

[13] M. Eid, R. Liscano, and A. El Saddik, “A universal ontology for sensor networks data,” in Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA ’07), pp. 27–29, Ostuni, Italy, June 2007.

[14] C. Stasch, K. Janowicz, A. Bröring, I. Reis, and W. Kuhn, “A stimulus-centric algebraic approach to sensors and observations,” in Proceedings of the International Conference of the Springer Geosensor Networks (GSN ’09), pp. 169–179, Oxford, UK, 2009.

[15] A. Devaraju, H. Neuhaus, K. Janowicz, and M. Compton, “Combining process and sensor ontologies to support geo-sensor data retrieval,” in Proceedings of the 6th International Conference of the AGILE Geographic Information Science (GIScience ’10), Zürich, Switzerland, 2010.

[16] L. Bermudez, J. Graybeal, and R. Arko, “A marine platforms ontology: experiences and lessons,” in Proceedings of the International Conference of the Springer Semantic Sensor Networks (SSN ’06), Athens, Ga, USA, 2006.

[17] D. Russomanno, C. Kothari, and O. Thomas, “Building a sensor ontology: a practical approach leveraging ISO and OGC models,” in Proceedings of the International Conference on Artificial Intelligence (ICAI’05), pp. 637–643, Las Vegas, Nev, USA, June 2005.

[18] M. Compton, C. Henson, L. Lefort, H. Neuhaus, and A. Sheth, “A survey of the semantic specification of sensors,” in Proceedings of the International Workshop of the CEUR Semantic Sensor Networks (SSN ’09), pp. 17–32, Washington, DC, USA, 2009.

[19] M. Compton, P. Barnagli, L. Bermudez et al., “The SSN ontology of the W3C semantic sensor network incubator group,” Web Semantics: Science, Services and Agents on the World Wide Web, vol. 17, pp. 25–32, 2012.

[20] W3C, “OWL 2 web ontology language document overview,” 2012, http://www.w3.org/TR/owl2-overview/.

[21] DUL.owl, http://www.loa.istc.cnr.it/ontologies/DUL.owl.

[22] C. Malewski, I. Simonis, A. Terhorst, and A. Bröring, “StarFL—a new metadata language for sensor descriptions—or: why do we need yet another metadata language for sensor descriptions?” in Proceedings of the International Workshop of the Sensor Web Enablement (SWE ’10), Banff, Canada, 2011.

[23] A. Bröring, P. Maué, K. Janowicz, D. Niist, and C. Malewski, “Semantically-enabled sensor plug & Play for the sensor web,” Sensors, vol. 11, no. 8, pp. 7568–7605, 2011.

[24] A. Bröring, S. Below, and T. Foerster, “Declarative sensor interface descriptors for the sensor web,” in Proceedings of the International Workshop on Pervasive Web Mapping (WebMGS ’10), Como, Italy, 2010.

[25] P. T. Eugster, P. A. Felber, R. Guerouati, and A.-M. Kermarrec, “The many faces of publish/subscribe,” ACM Computing Surveys, vol. 35, no. 2, pp. 114–131, 2003.

[26] E. Souto, G. Guimaraes, G. Vasconcelos et al., “Mires: a publish/subscribe middleware for sensor networks,” Personal and Ubiquitous Computing, vol. 10, no. 1, pp. 37–44, 2006.

[27] S. Lai, J. Cao, and Y. Zheng, “PSWare: a publish/subscribe middleware supporting composite event in wireless sensor network,” in Proceedings of the 7th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom ’09), pp. 1–6, Galveston, Tex, USA, March 2009.

[28] P. Boonma and J. Suzuki, “Self-configurable publish/subscribe middleware for wireless sensor networks,” in Proceedings of the 6th IEEE Consumer Communications and Networking Conference (CCNC ’09), pp. 1–8, Las Vegas, Nev, USA, January 2009.

[29] D. C. Tran and C. Pham, “A content-guided publish/subscribe mechanism for sensor networks without location information,” Computer Communications, vol. 33, no. 13, pp. 1515–1523, 2010.

[30] K. Shi, Z. Deng, and X. Qin, “TinyMQ: a content-based publish/subscribe middleware for wireless sensor networks,” in Proceedings of the International Conference of the IARIA Sensor
[31] X. Tong and E. C. H. Nagi, "A ubiquitous publish/subscribe platform for wireless sensor networks with mobile mules," in Proceedings of the International Conference of the IEEE Distributed Computing in Sensor Systems (DCOSS '12), pp. 99–108, Hangzhou, China, 2012.

[32] Z. Shen, R. Li, and J. Luo, "Mobile agent based middleware using publish/subscribe mechanism in wireless sensor networks," in Proceedings of the International Conference on Communication Software and Networks (ICCSN '09), pp. 111–115, Macau, China, February 2009.

[33] G. C. Fox, S. Kamburugamuve, and R. D. Hartman, "Architecture and measured characteristics of a cloud based internet of things," in Proceedings of the International Conference of the Collaboration Technologies and Systems (CTS '12), pp. 6–12, Denver, Colo, USA, 2012.

[34] S. Ran, "A model for web services discovery with QoS," ACM SIGecom Exchanges, vol. 4, no. 1, pp. 1–10, 2003.

[35] J. Diaz, G. von Laszewski, F. Wang, A. J. Younge, and G. Fox, "FutureGrid image repository: a generic catalog and storage system for heterogeneous virtual machine images," in Proceedings of the 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom '11), pp. 560–564, Athens, Greece, December 2011.

[36] A. Uyar and G. Fox, "Investigating the performance of audio/video service architecture I: single broker," in Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS '05), pp. 120–127, St. Louis, Mo, USA, May 2005.

[37] M. Eggert, R. Häußling, M. Henze et al., "SensorCloud: towards the interdisciplinary development of a trustworthy platform for globally interconnected sensors and actuators," Tech. Rep., RWTH Aachen University, Aachen, Germany, 2013.

[38] N. Mitton, S. Papavassiliou, A. Puliafito, and K. S. Trivedi, "Combining cloud and sensors in a smart city environment," EURASIP Journal on Wireless Communications and Networking, vol. 2012, no. 247, article 1, 10 pages, 2012.

[39] Microsoft HealthVault, https://www.healthvault.com/.

[40] Sensing troubled waters in a Smarter City, http://www-03.ibm.com/ibm/history.

[41] Fujitsu Akisai, http://www.fujitsu.com/global/news/pr/archives/month/2012/20120718-01.html.