Future Network & MobileSummit 2010 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2010
ISBN: 978-1-905824-16-8

Showcase Paper

A Service Oriented Middleware Architecture for Wireless Sensor Networks

Hamidreza Abangar, Payam Barnaghi, Klaus Moessner, Anita Nnaemego, Karthik Balaskandan, Rahim Tafazolli
Centre for Communication Systems Research, University of Surrey, Guildford, UK
Tel: +44 (0)1483 300800, Fax: +44 (0)1483 300803, {h.abangar, p.barnaghi, k.moessner, a.nnaemego, k.balaskandan, r.tafazolli}@surrey.ac.uk

Abstract: Increasingly distributed sensor networks and in particular Wireless Sensor Network (WSN) platforms are available on the market, these platforms usually implement protocols such as 6LowPan or other IEEE 802.15 based protocols. This leads to a certain degree of heterogeneity which in turn limits the potential interoperability between the individual nodes and platforms. This paper argues for a middleware platform that closes this interoperability gap. It discusses the range of potential technologies that could be deployed and evaluates their advantages and disadvantages. Finally it provides a SOA based middle approach along with the description of a proof of concept implementation.

Keywords: Sensor Networks, Middleware, Service Oriented Architecture.

1. Introduction

Sensor nodes usually consist of small computational devices which have limited processing capabilities and power resources. These devices are often provided by different vendors and have various hardware and software platforms. The variation of specifications and operations in sensor nodes introduces a major issue in providing interoperable environments in heterogeneous sensor networks. Introducing sensor middleware to the sensor network architecture, as common interfaces, enables the systems to utilise sensor nodes in heterogeneous environments without being involved in diversity and heterogeneity of sensor node platforms. The sensor middleware can be implemented as a Web service and defines standard interfaces to interact with client applications. The middleware will handle the requests of sensor nodes, provides a temporary data storage for streaming sensor data, and decides on how and when to query and access the data from sensor nodes. In this scenario a middleware component which can be run on more powerful machines, plays the main role in interaction with clients and network services. The middleware will use intelligent mechanisms to decide if the data can be supplied from the temporary storage, or if communication with the sensor node is required to acquire a new measurement and/or observation. This enables power savings and provides efficient request processing and delivery in dealing with sensor data. Sensor gateways communicate with sensor nodes through a wrapper which act like a plug-in for each type of sensors. Figure 1 illustrates use of a middleware component (i.e. gateway) in a distributed sensor network framework.

In this paper we describe a Service Oriented Architecture (SOA) for a middleware component which mediates data exchange between heterogonous sensor platform and Web applications and services in a unified way. The paper describes the common architecture

and demonstrates the implementation of the design on the *SunSPOT*[1] sensor platform. The rest of the paper is organised as follows. Section 2 discusses middleware technologies and different approaches in designing middleware components in sensor networks. Section 3 describes the general architecture and different components for our proposed middleware solution and discusses the main advantages and technical novelty of the proposed solution. Section 4 describes the demonstration and implementation of the proposed architecture and shows how it can be used in integrating sensor data into Web applications. Section 5 discusses the future work and concludes the paper.

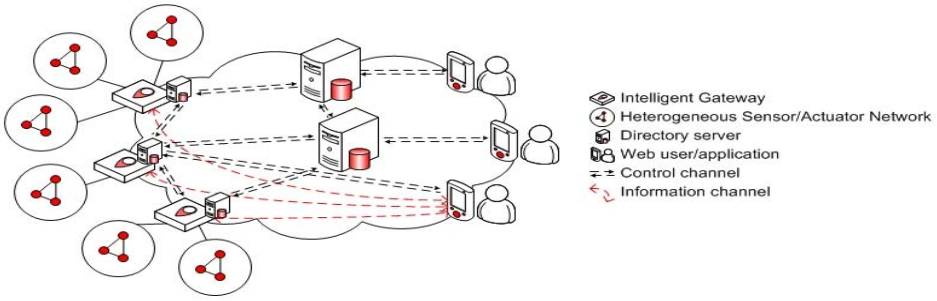

Figure1. Middleware component in a distributed sensor network framework

2. Wireless Sensor Networks and Middleware Technologies

Wireless Sensor Networks (WSNs) usually consist of large number of tiny sensor nodes that are programmed to monitor and report physical world phenomenon like temperature, light, humidity, pressure, etc. Sensor nodes generally consist of low power microcontroller, memory, sensors, radio transmitter, and power source. Large WSN's can be programmed to perform extensive monitoring and sensing tasks where timely and continuous observations are often required by other high-level applications [1].

Table 1: Common sensor node platforms

Sensor node	Air Interface	Platform
BTnode rev3	Chipcon CC1000 (433-915 MHz) and Bluetooth (2.4 GHz)	BTnut and TinyOS support
IMote 2.0	TI CC2420 802.15.4 and ZigBee compliant radio	Microsoft .NET Micro, Linux, TinyOS Support
Iris	Atmel AT86RF230 802.15.4 and ZigBee compliant radio	TinyOS, MoteWorks Support
Mica	RFM TR1000 radio 50 kbit/s	TinyOS Support
MicaZ	TI CC2420 802.15.4 and ZigBee compliant radio	TinyOS, SOS, MantisOS and Nano-RK Support
Rene	916 MHz radio with bandwidth of 10 kbit/s	TinyOS Support
SunSPOT	802.15.4	Squawk J2ME Virtual Machine
TelosB	250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver	Contiki, TinyOS, SOS and MantisOS Support
XYZ	CC2420 Zigbee compliant radio from Chipcon	SOS Operating System Support
FireFly	Chipcon CC2420	Nano-RK RTOS Support

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

Wireless senor nodes often suffer from limited resources like processing power, memory, and power. Wide range of sensor nodes have been introduced to the market during the last decade, some of them are depicted in Table 1. The current sensor nodes are mainly equipped with IEEE 802.15.4 [2] radio interface to enable communication between them. IEEE 802.15.4 is a low power, low range radio system covering physical layer and media access layer. Although a great amount of effort has been poured into the design and development of applications for WSN's interfacing and programming the sensor nodes is still a complex and error-prone task. To deal with this issue, a middleware software can hide the complexity of underlying layers and provides "programming abstraction" to simplify the application development process. Middleware components integrate WSN's with the user application while hiding the complexity and heterogeneity of the underlying hardware and network platforms. While most of the solutions have been specifically tailored for individual applications there are some solutions that adopt "programming abstractions" which have been successfully used for many years in distributed computing; e.g. [3], [4].

2.1 Middleware solutions for WSN's

This section describes some of the existing middleware solutions for sensor networks. We describe some of the core attributes and the design principles that are adopted in introducing different solutions for middleware components in WSN's.

MIRES [5] utilises a publish/subscribe paradigm with active messages. The supplier publishes a set of available services for clients to subscribe to. The subscription is used to query and obtain data from the network when a certain condition is met. This, to some extent, addresses traffic rate and power inefficiency issues that could happen using a query/response architecture which can overwhelm the system with high number of requests. The communication model adopted is asynchronous, event-driven and message-oriented using multi-hop algorithms.

Middleware Linking Applications and Networks (MiLAN) [6] utilises an architecture that extends to the network protocol stack. It is operates on top of various physical networks and adapts to different network protocols. This is accomplished through the conversion of MiLAN commands to network-specific commands thereby giving it the capability to organise and manage a network.

Mate [7] adopts a Virtual Machine (VM) designed to run on TinyOS [8]. It comprises 24-byte long instructions with simple APIs for sensor node interactions. Mate implements an ad hoc routing algorithm and provides for user defined algorithms. Similar to traditional virtual machines, Mate provides a level of abstractions to the underlying physical implementation of sensor nodes. A disadvantage is high overhead associated with the instruction interpretation making it suitable for sleeping applications [9].

Impala [10] embraces a modular programming concept based on mobile agents. Network updates are carried out on a component/modular basis thereby having a positive effect on power conservation due to the small amount of overhead to be transmitted. It is also an event-driven middleware with an adaptable application approach. Conversely, the makeup of its code instructions hinders hardware heterogeneity which is a key tenet of sensor networks [9].

The concept of Virtual Relational Database system is implemented in Cougar [11]. It uses a SQL-like query language to retrieve information from the sensor network. Cougar is a loosely-coupled architecture for aggregation and in-network computation.

The middleware solutions discussed above use different approaches namely the publish-subscribe method, macro-programming, virtual machine, modular programming (mobile agents) and database models. Although these solutions provide mainly communication

services between high-level applications and queries between heterogeneous WSN resources, they only provide limited flexibility and interoperability in terms of interaction with high-level consumer application and end-users. Updating the interfaces and providing automated machine-to-machine interactions in these types of solutions is often constrained due to restricted interaction models and pre-defined interfaces and operations. Another solution is adapting SOA architectures to WSN's and providing flexible design and enabling machine-to-machine interaction for middleware by introducing them as service components.

2.2 Adoption of Service Oriented Architecture in WSN's

SOA is an architectural design pattern that enables applications to be developed using loosely coupled and interoperable services. In SOA the whole business logic of the application is fulfilled by interweaving services available on the network no matter where they are, what platform they are run on, or in which language they are developed. Although SOA is not tied to any specific technology, Web Service technology appears to be the most popular approach. Since emergence of Web Service, a plethora of complimentary specifications have been proposed to cover areas of security, reliability, and transaction-based messaging. Devices Profile for Web Services (DPWS) [12] pulls together a subset of these specifications to meet varied service requirements. Since DPWS has been designed with a small footprint, it specifically targets resource constrained devices. The following provides a brief description of current trends in employing SOA and Web Service based technologies in WSN's.

IrisNet [13] and Tenet [14] are among solutions that have adopted SOA in developing middleware solution for WSN's. Tenet [14] simplifies application development for tiered sensor networks. It benefits from generic motes in the lower tier and masters, relatively unconstrained 32-bit platform nodes, in the upper tier. Tenet provides a SOA based solutions that although is flexible to accommodate some applications, but still the application level gateway play an important role in the proposed solution. Although these solutions expose the functionality of the WSN in a more accessible way to the Internet through application level gateways, they mainly suffer from single point of failure and scalability issues common to gateway approaches. Additionally, no functionality to enable direct and seamless interaction between wireless sensors and Internet has been supported.

2.3 Sensor Web frameworks

This type of solutions generally aims at making the heterogeneous sensors (and actuators), and sensor reading repository discoverable and accessible for the Internet applications and users over the Web. They generally provide a mash-up application that allows visualising the data.

SensorMap [15] provides a set of tools that data owners can use to easily publish their data and a GUI that can be used by users to make queries over live data. SensorMap transparently provides mechanisms to archive, index the sensor data, process queries, and aggregates the sensor data [16]. The SenorMap GUI is a mash-up application that lets users submit queries on available sensors and overlays the aggregated results on a map.

The framework introduced in [17] facilitates access to both real time and historical sensed data, though of variety of access methods. It addresses the scalability issue by introducing a distributed sensor register.

Although these solutions provide either an SOA based API or common interfaces that make sensing data accessible for the users, but still the role of general heavy application level gateway and single point of failure is very crucial in them. There is no direct

interaction between sensor nodes and the Internet applications and users, and the sensing data and actuator functionality have been represented by a SOA based facade which is provided by application level gateways.

2.4 Devices Profile for Web Services (DPWS) based solutions

Devices profile for Web Services is a lightweight subset of Web Services technologies that has been developed to bring plug-and-play features to resource limited devices. It aims to facilitate finding devices and their shared services in a network. Figure 2 describes the Web Services technologies used in DPWS.

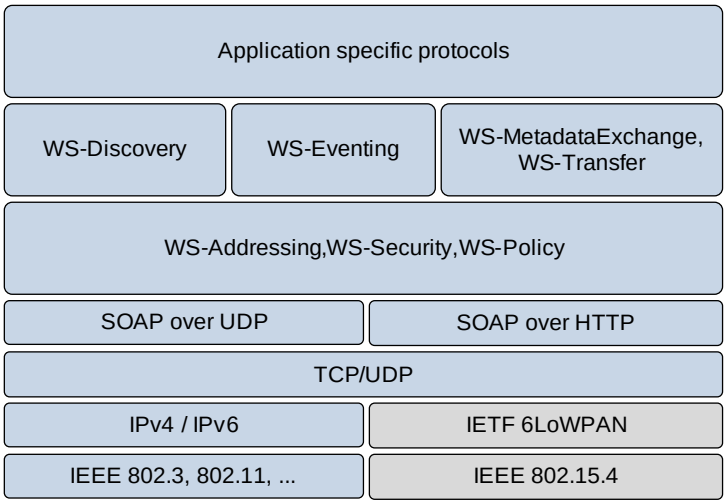

Figure 2. Devices Profile for Web Services

The following Web Service technologies are employed in DPWS [12]. WS-Discovery enables plug-and-play feature for sensor and actuator nodes. It enables sensor nodes to announce their presence and their offered service in the network. The offered services can be located by clients upon service request. The discovery process is based on service type and scope. A combination of multicast/unicast message exchange enables the discovery process. WS-Eventing enables asynchronous message exchange between sensor nodes. The subscriber can simply subscribe to service event source and receive the notification using push delivery mechanism. WS-Addressing provides transport neutral addressing information in SOAP envelopes. WS-Policy is used to express the capabilities of the services offered by nodes and their constraints. WS-MetadataExchange enables the service clients to dynamically get the metadata information about the offered services like description of the service (WSDL). WS-Transfer defines a mechanism for obtaining XML-based representations of entities using the Web service infrastructure. Entity can be either a resource or a resource factory that can create a new resource from an XML representation. WS-Security: lays out a set of rules to enable a secure, message exchange through message integrity, message confidentiality, and single message authentication.

Some recent works such as [18] and [19] have discussed the possibility and suitability of DPWS for WSN's. Some of the proposed solutions eliminate the use of SOAP and HTTP protocols; instead the solution is based on application-specific-formats that are used in the proposed Tiny DPWS protocol stack. Although the proposed application-specific-formats reduces the size of the transmitted messages in the network, but it hinders the extensibility of the solutions. For any new service to be offered by sensor nodes, a new

application-specific-format should be defined in order to make it work in the proposed infrastructure.

3. Middleware Architecture for Sensor Networks

We are proposing a SOA-based solution for the middleware architecture in WSN's. The middleware is implemented based on the DPWS architecture. Figure 3 shows the different layers and core components as well as the modifications to the original DPWS. The proposed solution comprises of the software deployed in sensor nodes, and the software running in the gateway.

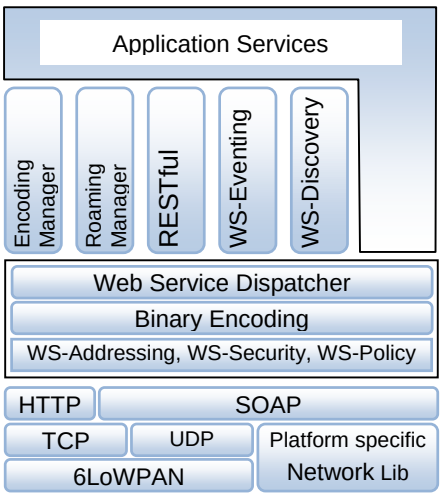

Figure 3. Service Architecture at node level

The architecture of the middleware component is inspired by Web Service standards to accommodate the desired interoperability and reusability. Given the inherent resource limitations of WSN's, optimisation techniques have been employed to reduce the overhead imposed by traditional Web Service technologies. In order to mitigate huge overhead imposed by large XML messages, binary encoding technique has been used. The messages are encoded in binary format before transmission. The proposed solution benefits from functionality supported in DPWS and RESTful functionality for less powerful sensor nodes.

Depending on the network condition directory-less, directory-based or a hybrid architecture can be used. In directory-less mode there is no service registry in the network and the nodes perform ad-hoc service discovery using multicast communication. While the presence of a service registry in directory-based networks reduces the number of multicast messages in the network at the cost of messaging required to maintain an up-to-date service registry.

Web services can ease development and deployment of most applications compared to other legacy solutions, but this approach still suffers from some complexities. REST can be seen as a solution to both reaping the benefits of using web services and not to suffer from the complexities inherent in web services, as discussed above. The less powerful nodes can alternatively offer their functionality in the RESTful interface. This component makes it possible even for sensor nodes equipped with less memory and processing power to participate in service interaction. WS-eventing eliminates the need for periodic calling the desired service and the user can simply subscribe to the service eventing interface. WS-eventing notifies the clients when the requested service/data has changed according to the request definition. This can significantly save the limited network bandwidth. Due to use of binary encoding, string tables between communication parties need to be synchronised. The

component deals with this and enables encoding and decoding of messages with new schemes. A roaming manager enables the sensor nodes to seamlessly leave one gateway and to attach to another one without losing the connection to previous service clients. Figure 4 demonstrates the interactions between the middleware and other components in a simplified scenario.

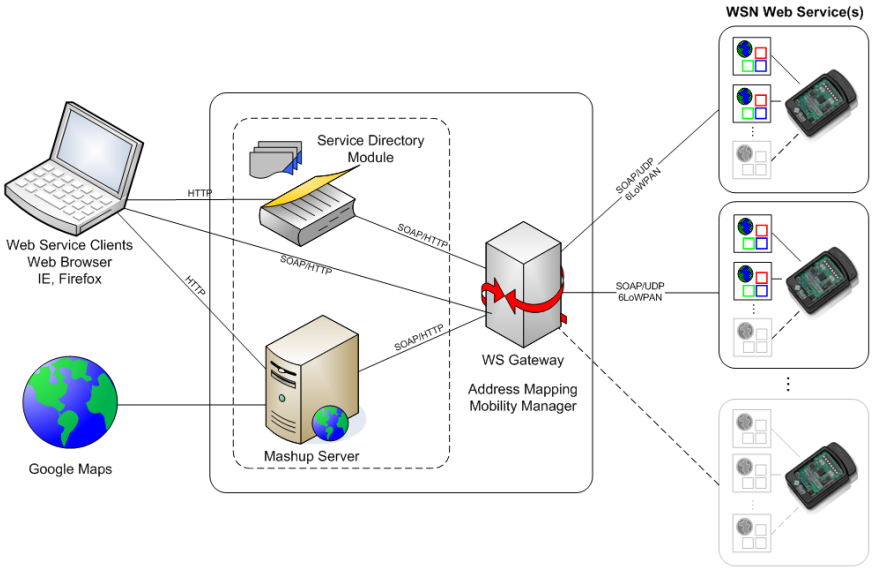

Figure 4. The middleware and interaction with other components

The address mapping module provides the mapping between WSN and Internet applications. Due to the fact that some sensor platforms still do not support IPv4/IPv6 addressing, this module employs stateless address mapping to facilitate communication between these networks. When a node leaves one platform type and attaches another one, the open service transaction should not get interrupted and this handover should happen in a seamless manner. The latter is handled by the Mobility Manager. This module accommodates this requirement by establishing the connection with the other involved gateway and shares the context of service transactions.

Protocols such a 6LoWPAN [20] enables sensor nodes to directly be accessible through the Internet, but some platforms are not equipped with this protocol. To accommodate heterogeneous sensor nodes, the proposed protocol stack works on top of 6LoWPAN as well as other platform specific networking libraries. The proposed solution has been developed for the SunSPOT platform and porting to Contiki and TinyOS platforms is in progress. Figures 5 and 6 show a Web 2.0 application implemented using the SOA-based middleware component using SunSPOT sensor nodes.

Figure 5. Google Maps mash-up application using Web Services for WSN's

The demonstration setup includes a WSN running sensor Web Services and a mobile workstation that runs the WSAN-Explorer application. The WSAN includes 5 SunSPOT nodes and a base-station connected to a workstation, which provides the IEEE 802.15.4 network access for the workstation. The mash-up server displays the current location and nearby sensors with their location are marked on the map. The list of sensors and their endpoint references are fetched from the Directory-Module.

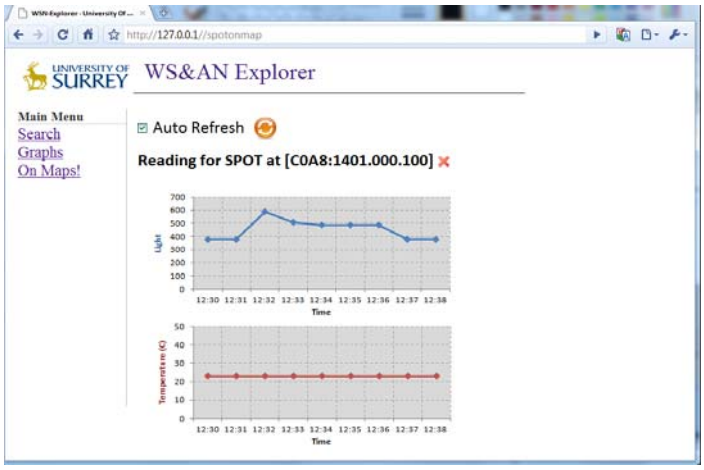

Figure 6. Presenting sensor information in a graph view

4. Conclusions

This paper presents an overview of how heterogeneous wireless sensor networks (WSNs) can be integrated using middleware technologies; it discusses design considerations and the required communication protocols. The heterogeneous nature of WSNs dictates a need for a flexible and autonomous middleware thereby shielding application developers from the intricacies of the underlying network. The proposed middleware strives to present a homogeneous platform in an interoperable manner bearing in mind the resource limited nature of WSNs in addition to the adoption of energy-aware mechanisms paramount to extending the lifetime of the system as a whole.

Advantages of service-oriented middleware include reduction of complexity by offering well-defined, service-specific interfaces to the rest of the system. Also they are capable of maintaining acceptable performance levels regardless of network changes, device failures, mobility, obstacles in the path and interference. Service Oriented middleware aims at maintaining QoS requirements specified by the applications. Considering the restrictions

imposed on sensor resources, SOA efficiently controls energy consumption of sensors by decreasing the number of transmitted messages to sink using multi-hop communication, data aggregation and fusion techniques [2]. While the proposed approach introduces extreme flexibility and interoperability between different platforms, there are some hurdles in the use of SOA. In particular limitations of SOA have to be taken into account, in particular the fact that applications could run slower and may require more processing power if non-native/binary forms of Remote Procedure Call are used.

Acknowledgement

This paper describes work undertaken in the context of the SENSEI project, Integrating the Physical with the Digital World of the Network of the Future (www.sensei-project.eu). SENSEI is a Large Scale Collaborative Project supported by the European 7th Framework Programme, contract number: 215923.

References

[1] S. Liang, A. Croitoru, and C. Tao, "A distributed geospatial infrastructure for Sensor Web," Computers and Geosciences, vol. 31, pp. 221-231, 2005.
[2] I. C. Society, "Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)," IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003), 2006.
[3] T. Liu and M. Martonosi, "Impala: a middleware system for managing autonomic, parallel sensor systems," pp. 107-118, 2003.
[4] H. Attiya, and J. Welch, Distributed computing: fundamentals, simulations, and advanced topics: Wiley-Interscience, 2004.
[5] Souto, E., Guimarães, G., Vasconcelos, G., Vieira, M., Rosa, N., Ferraz, C., and Kelner, J. "Mires: a publish/subscribe middleware for sensor networks", Personal Ubiquitous Computing, vol. 10, no. 1, pp. 37-44, 2005.
[6] W.B Heinzelman, A.L. Murphy, H.S. Carvalho, and M.A. Perillo, "Middleware to support sensor network applications," IEEE Network, vol.18, no.1, pp. 6-14, 2004.
[7] P. Levis, and D. Culler, "Mate: A Tiny Virtual Machine for Sensor Networks", In Proceedings of the 10th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS-X), ACM Press, pp. 85–95, 2002.
[8] TinyOS, http://www.tinyos.net/
[9] S. Hadim, and N. Mohamed, "Middleware Challenges and Approaches for Wireless Sensor Networks", IEEE Distributed Systems Online, vol. 7, no. 3, 2006.
[10] Liu, T., Sadler, C. M., Zhang, P., and Martonosi, M. "Implementing software on resource-constrained mobile sensors: experiences with Impala and ZebraNet", In Proceedings of the 2nd international Conference on Mobile Systems, Applications, and Services, 2004.
[11] Y. Yao, and J. Gehrke, "The Cougar Approach to In-network Query Processing in Sensor Networks", In Proceedings of the ACM SIGMOD International Conference on Management of Data, 2002.
[12] S. Chan, C. Kaler, T. Kuehnel, A. Regnier, B. Roe, D. Sather, J. Schlimmer, H. Sekine, D. Walter, and J. Weast, "Devices profile for web services", 2006.
[13] P. Gibbons, B. Karp, Y. Ke, S. Nath, S. Seshan, I. Res, and P. Pittsburgh, "Irisnet: An architecture for a worldwide sensor web," IEEE Pervasive Computing, vol. 2, pp. 22-33, 2003.
[14] O. Gnawali, K. Jang, J. Paek, M. Vieira, R. Govindan, B. Greenstein, A. Joki, D. Estrin, and E. Kohler, "The tenet architecture for tiered sensor networks," In Proceedings of the 4th international conference on Embedded networked sensor systems , pp. 153-166, 2006.
[15] S. Nath, J. Liu, and F. Zhao, "Sensormap for wide-area sensor webs", Computer, vol. 40, pp. 90-93, 2007.
[16] A. Santanche, S. Nath, J. Liu, B. Priyantha, and F. Zhao, "Senseweb: Browsing the physical world in real time," Demo Abstract, ACM/IEEE IPSN06, 2006.
[17] I. Rhead, M. Merabti, H. Mokhtar, and P. Fergus, "Worldwide Sensor Web Framework Overview", In Proceedings of the 9th Annual Postgraduate Symposium, The Convergence of Telecommunications, Networking and Broadcasting, Liverpool John Moores University, 2008.

[18] K. S. Ioakeim, V. G. John, and D. H. George, "Integrating Wireless Sensor Networks into Enterprise Information Systems by Using Web Services," In Proceedings of the 2009 Third International Conference on Sensor Technologies and Applications - IEEE Computer Society, 2009.
[19] G. Moritz, E. Zeeb, F. Golatowski, D. Timmermann, and R. Stoll, "Web Services to improve interoperability of home healthcare devices," In Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009. 3rd International Conference on, 2009, pp. 1-4.
[20] IPv6 over Low power WPAN (6lowpan), http://www.ietf.org/dyn/wg/charter/6lowpan-charter.html
[21] K. K. Khedo, R.K. Subramanian, "A Service-Oriented Component-Based Middleware Architecture for Wireless Sensor Networks", IJCSNS International Journal of Computer Science and Network Security, vol.9 no.3, pp. 174-182, 2009.