International Journal of Computer and Electrical Engineering, Vol. 2, No. 3, June, 2010
1793-8163

# LiteOS based Extended Service Oriented Architecture for Wireless Sensor Networks

V.Vanitha, Dr.V.Palanisamy, N.Johnson and G.Aravindhbabu

***Abstract*-Over the last decade, embedded sensing systems have been successfully deployed in a range of application areas, from education and science to military, healthcare and industry. These systems are becoming more robust, capable and widely adopted. However, every particular applications requires complex integration work and therefore technical expertise, effort and time which prevents users from creating small tactical, ad-hoc applications using sensor networks. These limitations are already addressed by implementing service oriented architecture in the wireless sensor networks. In addition to that, many real time applications are very time critical and demand richer set of applications. To address this issue, we propose an extended service oriented architecture (ESOA) that provides customizable sensor network, consolidates services and manages applications.**

***Index Terms*—ESOA, WSN, SANET, LiteOS, Service oriented architecture**

## I. Introduction

A Wireless sensor network (WSN) is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it. The main objective of the wireless sensor networks is to observe an environment, collect information about the observed phenomena or events and deliver this information to the application.

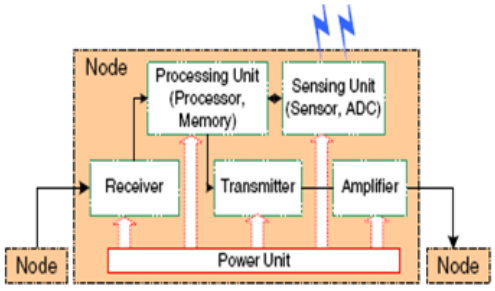


Fig. 1 Wireless Micro-node model. Each node has sensing unit, porcessing unit, power unit, and communication modules

The wireless micro sensor node consists of a sensing module, a processing element, and communication elements. The sensing module is an electrical part detecting physical variable from the environment. The processing unit (a tiny microprocessor) performs signal processing functions, i.e. integrating data and computation required in the processing of information. The communication elements consist of a receiver, a transmitter, and an amplifier if needed (see Fig. 1). Basically, all individual sensor nodes are operated by a limited battery, but a base station node as a final data collecting center can be modeled with an unlimited energy source. Nodes communicate wirelessly. Each node communicates directly (i.e., single hop) with a few other nodes within its radio communication range. A node may also transmit to distant nodes through multi-hop communication.

For years, wireless sensor network has been closed network deployed for a specific application with a specific set of characteristics. Typically, a single party (e.g., a government agency, a research institution, a private company) owns, maintains, and uses the sensor-actuator network (SANET). Most early SANET deployments have adopted ad-hoc, application-specific architectures. In recent years, the need to decouple SANETs from the applications using them led to the emergence of generic SANETs, an alternative design model where an application-independent query system is deployed on the SANET. In this model, the query system is designed to answer queries from any application. As SANETs evolve, they are expected to become open, ubiquitous, interoperable, multi-purpose infrastructures. This would translate into new requirements that existing architectures do not support. We argue that the next-generation SANETs require customizable architectures that would provide developers the ability to select individual software components from several SANETs and integrate them in new applications that achieve higher levels of efficiency and scalability.

Recently, the Service Oriented Architecture (SOA) has been considered as a good candidate to develop open, efficient, inter-operable, scalable and customizable WSN applications. In Service Oriented WSNs, node's sensing capability is exposed in the form of in-network services. Application development is simplified by providing standards for data representation, service interface description, and service discovery facilitation. By wrapping application functionality into a set of modular services, a programmer can then specify execution flow by simply connecting the appropriate services together. Some approaches are TinySOA [7], OASiS [5] and TinyWS [6]. In TinySOA, services are lightweight code units deployed directly on top of the operating system of nodes. Applications invoke services using a service-oriented query model. Queries are submitted to one of the established base stations or directly to individual nodes. OASiS also uses a passive discovery mechanism, but it is combined with an

Manuscript received Augest 11,2009.
V.Vanitha, N.Johnson, G.Aravindhbabu are with department of Computer Science and Engineering, Kumaraguru college of Technology, Coimbatore, Tamilnadu, India. (vanithajayaprakash@yahoo.com)
Dr.V.Palanisamy, Principal, Info Institute of Engineering, Coimbatore, Tamilnadu, India

object migration approach instead of using remote query mechanisms. Finally, TinyWS is a small web service platform that resides on the sensor nodes. It hosts the web services and has SOAP processing engine. The sensor nodes are service providers, the application devices are service requestors and a distributed UDDI acts as an overlay entity. We introduce ESOA, Extended SOA model, which inserts at the top of the typical SOA model two layers that provides the service consolidation and management.

TinyOS adopts NesC and the event-based programming model, which introduces a learning curve for the most traditional programmers. In our work we use LiteOS [2], a new operating system for sensor networks developed by UIUC. LiteOS supports C programming and provides Unix-like abstraction for wireless sensor networks, which greatly improved their compatibility with other development platforms and simplified the sensor network programming.

The rest of the paper is organized as follows. In section 2, we discuss about LITE OS. In section 3, we discuss the limitations of the current wsn architecture. In Section 4 we discuss the basic service oriented architecture and its advantages in wsn. In section 5 we present our extended service oriented middleware architecture. In section 6 we present the overview of research literature related to our work. The paper ends with the conclusion in section 7.

## II. Introduction to LiteOS

LiteOS provides a UNIX-like environment for sensor networks, networked embedded devices, and cyber physical systems. It provides a thread-based run-time execution environment for applications. This paper introduces the UNIX-like operating system, called LiteOS that fits on memory-constrained motes such as MicaZ. This operating system is multithreaded and comes bundled with a UNIX-like file system and a C++ compiler. While TinyOS and its extensions have significantly improved programmability of mote-class embedded devices via a robust, modular environment, NesC and the event-based programming model introduce a learning curve for the most developers outside the sensor networks circle. The purpose of LiteOS is to significantly reduce such a learning curve.

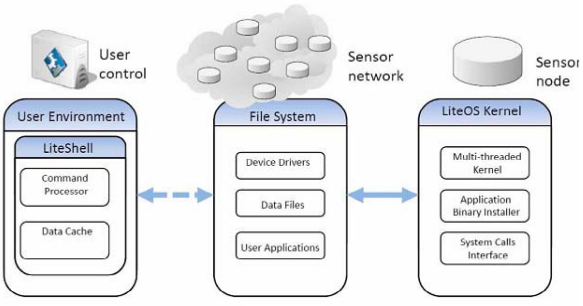


Fig. 2 LiteOS Architecture

### A. Architectural Overview

Fig.2 shows the overall architecture of the LiteOS operating system, partitioned into three subsystems: LiteShell, LiteFS, and the kernel. Implemented on a base station, the LiteShell subsystem interacts with sensor nodes only when a user is present. Therefore, LiteShell and LiteFS are connected with a dashed line in this figure. LiteOS provides a wireless *node mounting mechanism* (to use a UNIX term) through a file system called LiteFS. Much like connecting a USB drive, a LiteOS node mounts itself wirelessly to the root filesystem of a nearby base station. Moreover, analogously to connecting a USB device (which implies that the device has to be less than an USB cable - length away), the wireless mount works only for devices within wireless range. The mount mechanism comes handy, for example, in the lab, when a developer might want to interact temporarily with a set of nodes on a table-top before deployment. While not part of the current version, it is not conceptually difficult to extend this mechanism to a "remote mount service" to allow a network mount. Ideally, a network mount would allow mounting a device as long as a network path existed either via the Internet or via multi-hop wireless communication through the sensor network. Once mounted, a LiteOS node looks like a *file directory* from the base station. The shell, called LiteShell, supports UNIX commands, such as copy and move, executed on such directories. The external presentation of LiteShell is versatile. While the current version resembles closely a UNIX terminal in appearance, it can be wrapped in a graphical user interface (GUI), appearing as a "sensor network drive" under Windows or Linux.

### B. Advantages of LiteOS model

It is an interactive and reliable model which provides the benefits like robustness, integrity, availability. It helps to provide the simple operations for services when compared to TinyOS model. LiteOS provides the environment for Object-oriented languages which helps to have the advantages like reusability, modularity, extensibility.

## III. Limitations of Current Wsn Architectures

Current architectures for WSN have inherent limitations that may be summarized as follows:

### A. Tight coupling between WSNs and applications

A tight network–application coupling characterizes most current sensor network and WSN architectures. This coupling usually takes one of two forms:

- Network-dependent application development: Most current sensor applications are designed and implemented to be used on a specific sensor network or a specific type of sensor networks with specific characteristics and a specific querying interface. Often, application developers need to know network-specific information such as network topology, nodes' transmission range and processing/memory capabilities, etc.
- Application-dependent network design and deployment: In some cases, a decision is first made to use a specific software system to build a sensor application. The requirements of the software system must then be taken into consideration when designing and deploying the sensor network supporting that system.

### B. Costly optimization or suboptimal efficiency

In current WSNs, application-dependent optimization makes the sensor network unable to provide the same levels of performance to other applications. Also, several rounds of optimization may be needed as the application evolves or is replaced. The alternative of application independent optimization is often too generic and does not exploit optimization opportunities that a specific class of applications may offer. Typically, this leads to suboptimal efficiency.

### C. Limited reusability

Ideally, for a WSN to be cost effective, it would be necessary to amortize its deployment and maintenance cost by sharing its functionalities amongst a large group of users and applications. This reusability is generally not easily achievable in current sensor infrastructures due to the tight coupling between networks and applications.

### D. Low return on investment

This drawback follows from the previous one. The monolithic, application-specific design of current WSN makes it difficult to reuse most of an application's modules in developing another application. Often, intensive programming is required each time a new application has to be developed.

### E. Non-scalability

Most of today's WSN applications are designed and optimized for light loads, i.e., destined to be used by a small group of users. As a result, current WSNs often do not scale to support large numbers of simultaneous users.

## IV. Extended Service Oriented Architecture (ESOA)

Our proposed service oriented Architecture consists of service composition layer on top of basic service oriented architecture. Fig.3 depicts the architecture of extended service oriented architecture.

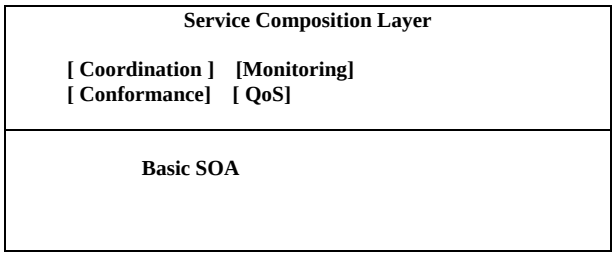


Fig. 3 ESOA Architecture

### A. Basic SOA

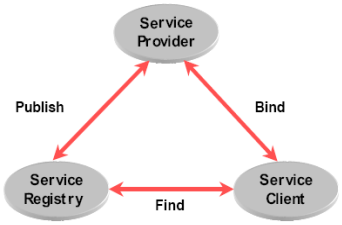


Fig. 4 Basic SOA

SOA is a logical way of designing a software system to provide services to either end-user applications or other services distributed in a network through published and discoverable interfaces. The basic SOA defines an interaction between software agents as an exchange of messages between service requesters (clients) and service providers. Clients are software agents that request the execution of a service. Providers are software agents that provide the service. Agents can be simultaneously both service clients and providers. Providers are responsible for publishing a description of the service(s) they provide. Clients must able to find the description(s) of the services they require and must be able to bind to them. The basic SOA is not an architecture only about services, it is a relationship of three kinds of participants: the *service provider*, the *service discovery agency*, and the *service requestor* (*client*). The interactions involve the publish, *find* and *bind* operations (see Fig.3). These roles and operations act upon the service artifacts: the service description and the service implementation. In a typical service based scenario a service provider hosts a network accessible software module (an implementation of a given service). The service provider defines a service description of the service and publishes it to a client or service discovery agency through which a service description is published and made discoverable. The service requestor uses a find operation to retrieve the service description typically from a discovery agency, i.e., a registry or repository like UDDI, and uses the service description to bind with the service provider and invoke the service or interact with service implementation. Service provider and service requestor roles are logical constructs and a service may exhibit characteristics of both.

This architectural approach is particularly applicable to heterogeneous wsn where multiple applications running on varied technologies and platforms need to communicate with each other. In this way, users can mix and match services to develop new applications with minimal programming effort.

### B. Service composition layer

The *service composition layer* in the ESOA encompasses necessary roles and functionality for the consolidation of multiple services into a single composite service. Resulting composite services may be used by *service aggregators* as components (i.e., basic services) in further service compositions or may be utilized as applications/solutions by service clients. Service aggregators thus become service providers by publishing the service descriptions of the composite service created by them. A service aggregator is a service provider that consolidates services that are provided by other service providers into a distinct value added service. Service aggregators develop specifications and/or code that permit the composite service to perform functions that include the following:

1) **Coordination:** controls the execution of the component services, and manages dataflow among them and to the output of the component service (e.g., by specifying workflow processes and using a workflow engine for run-time control of service execution).
2) **Monitoring:** allows subscribing to events or information produced by the component services, and publishes higher-level composite events (e.g., by

filtering, summarizing, and correlating component events).

3) **Conformance:** ensures the integrity of the composite service by matching its parameter types with those of its components, imposes constraints on the component services (e.g., to ensure enforcement of business rules), and performs data fusion activities.
4) **QoS composition:** leverages, aggregates, and bundles the component's QoS to derive the composite QoS, including the composite service's overall cost, performance, security, authentication, privacy, (transactional) integrity, reliability, scalability, and availability

### C. Advantages

It is realized that many leading organizations are moving towards a service-oriented architecture (SOA), an approach to architecting the IT infrastructure that eliminates redundancy and accelerates project delivery via consolidation and reuse of services (these services are often referred to as Web services). SOA allows an organization to effectively leverage existing assets rather than forcing them to create yet another redundant silo for each business need. This, in turn, also makes IT more efficient, allowing for shorter cycle times and quicker project delivery – further helping IT align with business.

By implementing a service oriented approach at all levels of WSN, the rapid development of applications as well as the thorough testing of sensor networks will be possible.

This service oriented approach will allow for the development of a WSN management system that will be able to handle the dynamic addition and removal of different sensors and applications as interoperable services.

### D. Scenario

Based on the above SOA architecture, we proposed to develop the health care application (as shown in Fig. 5) for patient monitoring system. The healthcare domain presents opportunities for a significant number of applications of wireless sensor technology

The following scenario explains the problem definition and the forthcoming explanation gives the implementation details regarding the service coordination layer operations.

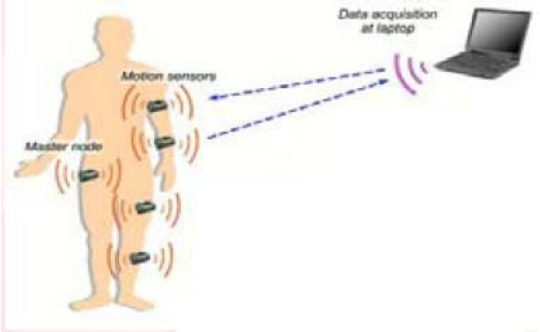


Fig. 5 Patient Monitoring System

Consider a patient who is admitted in hospital having chronic disease. It encompasses wide range of health problems including diabetes, asthma, heart diseases and sleep disorders. In many cases, chronic diseases require some kind of health monitoring, especially in the later stages of the disease progression. So, on later stages, we need continuous monitoring to handle the critical situation effectively. For this purpose, we have proposed ESOA concept in which the service composition layer helps to handle all the operations related to this critical patient monitoring system.

Under our implementation, the application mainly consists of three types of sensors which used to extract the data for pulse, blood pressure and oxygen saturation level. The data is considered to be either critical or non-critical based on the readings of each. For example, blood pressure monitors measure the pressure flowing through the blood vessels against the walls of the arteries. If blood flow is normal, then blood pressure is normal (average 120/80). If blood flow becomes restricted in some way, blood pressure goes up. If increased blood pressure goes undetected, the person is at risk of severe medical problems.

According to the readings, the response is given as a feedback in monitor either to alert a patient or nurse/doctors. The data should be transmitted at faster rate and the events should be handled immediately.

Feedback data includes patient profile, accident response, and medical care information and so on. Here we have proposed to use service provider which is a basic layer in ESOA to build service composition layer. Services included in this layer for chronic disease monitoring are:

1) Coordination: Data is extracted from each sensors and the workflow process is identified for proper data flow.
2) Monitoring: According to the filtered data, it is summarized and service is executed and the events (response) are handled either periodically/non-periodically. Correlation from each sensor's data might be found.
3) Conformance: Actual business rules are implemented under this service and it might be based on ECA(Event, Control & Action). Example: For pressure>190 medical care should be handled immediately. Object oriented approach will be used to ensure integrity.
4) QoS: There are potential opportunities for WSN in healthcare, but they are still in infancy. Due to monitoring gap and cost of wired diagnostic equipment it is difficult to monitor all patients. WSN helps to have reduced cost, faster data transmission and data processing. Reusability, modularity and configurability are of greater benefits. Privacy is achieved through password mechanisms.

## V. Related Work

One of the earlier works in taking the service-oriented approach in the design of a middleware system for wireless sensor networks was presented by Golatowski et al. [8]. In their work they introduce (at a very high level) a simple service-oriented model in which the responsibility for handling the services requests is assigned to an external entity. This external server works as a bridge between the requests received from the exterior and the internal network functionality. Nevertheless, no details on how the components interact between them, its characteristics or the

protocols used, are explained.

These are high-level programming abstractions that the service-oriented sensor and actuator (SOSANET) exposes to applications as a middleware layer. Typically, a middleware service is implemented as a module that runs off-network and that interacts with one or several nodes to provide its functionality. Most existing SOSANETs provide only middleware services. Examples include Atlas [9], and Sensation [10].

Atlas is a service-oriented sensor and actuator platform that enables programmable pervasive spaces. This platform is focused primarily on an OSGi-based service framework. Sensation is an open and generic service-oriented platform that enables standardized communication within individual infrastructures, between infrastructures and their sensors, but also among distributed infrastructures. Sensation allows developers to concentrate on the semantics of their infrastructure and to develop innovative concepts and implementations of context-aware systems.

In [11], the authors present WISE, a Web-Services framework for publishing, browsing, and analyzing real time sensor data. Providers register their sensors with a UDDI registry where any user can discover them over the Internet. Once the desired sensors are located, the user can employ two new communication protocols, namely SSCP and SSTP, to control the incoming sensor streams. As the data arrive at the user browser software, a corresponding plug-in such as a data animation module is activated to allow the user to ‘‘see’’ the data stream presented in an intelligible way.

ZigBee will play an important role in the adoption of Assistive Technology by enabling wireless low-power communication between devices and services that foster safe, healthy and independent living conditions for the disabled or elderly. In addition to Assistive Technology, ZigBee applications in the health and fitness domains can address several other market segments and needs.

Honeywell offers the 26PC SMT (Surface Mount Technology) Series pressure sensor. This small, low-cost, high-value pressure sensing solution takes the place of the health care professional having to place a stethoscope under the pressure cuff to hear the noise of the rushing blood. The sensor measures blood pressure faster and more accurately than manual devices, and is used directly with printed circuit boards (PCBs). The blood pressure monitor has an on-board processor to cycle through the test, record results and output the results to a digital read-out screen. Instead of watching the mercury fall in the manometer, the pressure sensor acts as the mechanism to provide a visual aid and noise monitor of the pulse of blood flowing through the arteries.

## VI. Conclusion

Today, the Wireless Sensor Networks use proprietary mechanisms in providing data access. They use non standard protocols and require gateways as a bridge between application devices and the sensor networks. To facilitate a wider use of sensor data and motivate more innovative applications, WSN should use common and standard protocols as most application devices and provide data access via frameworks that can be used by general application developers. To achieve this we have proposed ESOA. We are currently implementing this on a LiteOS, a new operating system for wsn.

## References

[1] Wang MM, cao JN, Li J et al, Middleware for wireless sensor networks: A survey, Journal of computer science and technology 23(3): 305-326 May 2008.

[2] Qing Cao and Tarek Abdelzaher and John Stankovic and Tian He, The LiteOS Operating System: Towards Unix-Like Abstractions for Wireless Sensor Networks, In Proceedings of ACM/IEEE IPSN, pages 233-244, 2008.

[3] M.P.Papazoglou D.Georgakopoulos, Service oriented computing , Communications of the ACM, October- 2003.

[4] Flavia Coimbra Delicato, P.F. Pires, L. Pirmez, L.F.R.D.C. Carmo, “A Flexible Web Service based Architecture for Wireless Sensor Networks”, in proceeding of the 23rd International Conference on Distributed Computing Systems workshops (ICDCSW2003), p. 730, 2003.

[5] M. Kushwaha, I. Amundson, X. Koutsoukos, S. Neema, and J. Sztipanovits. OASiS: A Programming Framewor for Service-Oriented Sensor Networks. In Proceedings of the 2nd IEEE/Create-Net/ICST International Conference on COMmunication System softWAre and MiddlewaRE (COMSWARE’ 07), Bangalore, India, January 2007. IEEE Computer Society Press.

[6] N. Y. Othman, R. H. Glitho, and F. Khendek. The Design and Implementation of a Web Service Framework for Individual Nodes in Sinkless Wireless Sensor Networks. In Proceedings of the IEEE International Conference on Computers and Communications (ISCC’07), pages 941–947, Aveiro, Portugal, July 2007. IEEE Computer Society Press.

[7] A. Rezgui and M. Eltoweissy. Service-oriented sensor actuator networks: Promises, challenges, and the road ahead. Computer Communications, 30:2627–2648, 2007.

[8] Golatowski F, Blumenthal J, Handy M, Haase M (2003) Service oriented software architecture for sensor networks. Intl.Workshop on Mobile Computing (IMC 2003),Rockstock,Germany, pp 93–98

[9] J. King, R. Bose, Y. Hen-I, S. Pickles, A. Helal, Atlas: a serviceoriented sensor platform: hardware and middleware to enable programmable pervasive spaces, in: Proceedings of the 31st IEEE Conference on Local Computer Networks, November 2006, pp. 630–638.

[10] T. Gross, T. Egla, N. Marquardt, Sens-ation: a service-oriented platform for developing sensor-based infrastructures, Int. J. Internet Protocol Technol. 1 (3) (2006) 159–167.

[11] K.A. Hua, R. Peng, G.L. Hamza-Lup, WISE: a Web-based intelligent sensor explorer framework for publishing, browsing, and analyzing sensor data over the Internet, in: Proceedings of the Fourth International Conference on Web Engineering (ICWE), July 2004, pp. 568–572.

[12] F. Golatowski, J. Blumenthal, M. Handy, M. Haase, H. Burchardt, D. Timmermann, Service-oriented software architecture for sensor networks, in: Proceedings of the International Workshop on Mobile Computing, 2003, pp. 93–98.

[13] ZigBee Wireless Sensor Applications for Health, Wellness and Fitness - March 2009. www.zigbee.org

[14] Blood Pressure Monitoring Using the 26PC SMT. www.honeywell.com/sensing, info.sc@honeywell.com