Demo Abstract: Design and Implementation of a Web Service for LiteOS-based Sensor Networks

Masaaki Takahashi, Basit Hussain, and Bin Tang
Department of Electrical Engineering and Computer Science
Wichita State University, Wichita, KS 67260
Email: {mxtakahashi, bxhussain}@wichita.edu, bintang@cs.wichita.edu

ABSTRACT

We design and implement a web service for LiteOS-based wireless sensor networks (WSNs). LiteOS is a newly developed operating system for the sensor networks. Taking advantage of UNIX-like shell commands and C programming language supported by LiteOS, we implement a web service middleware which interacts with the front-end web application as well as the WSNs such that users can remotely request and view the sensor readings. Our web service system is equipped with secure membership, a visualizer for sensor readings, and accepts parameterized queries from multiple users simultaneously. We also address the issues of improving user perceived response time and energy efficiency of sensor nodes in the web service.

Categories and Subject Descriptors

H.3.5 [Information Storage and Retrieval]: Online Information Services - Web-based services

General Terms

Design, Experimentation

Keywords

Web Services, LiteOS

1. INTRODUCTION

Wireless Sensor Networks enables us to interact with the physical world and receive real time information of the physical phenomena. Currently the world is moving towards Service Oriented Architectures (SOAs) where different services can be provided over the Web. Web services, being an efficient programmatic way to exchange data over Internet, were recently proposed to support interoperable and evolvable sensor networks [4]. The study of the architecture of web services for sensor networks and a standard middleware is still in its infancy.

IPSN'09, April 13-16, 2009, San Francisco, CA, USA.

In this paper, we propose a web service architecture and implement a web service middleware for LiteOS-based wireless sensor networks. The layered architecture, shown in Figure 1, is comprised of an application layer, a web service middleware layer, and a WSN layer. The web service middleware interacts with the front-end web application as well as the WSNs such that users can remotely query and view the sensor readings of light, temperature, magnet, and acceleration.

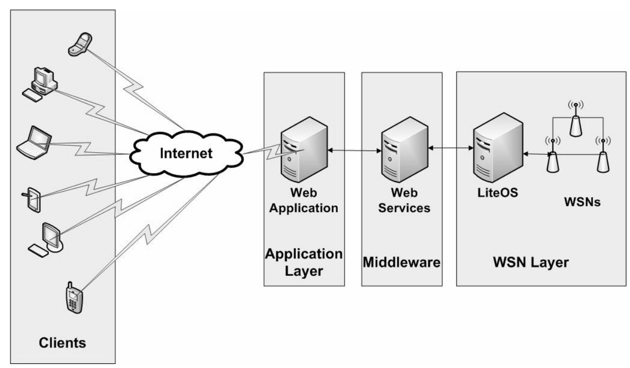

Figure 1: Web service architecture for LiteOS-based sensor networks.

2. WHY LITEOS

Web services for sensor networks have been developed in both research and commercial fields, such as Tiny Web Service [3, 4] and Arch Rock's web service [1]. They are both based on TinyOS [5]. TinyOS adopts NesC and the event-based programming model, which introduce a learning curve for 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. LiteOS includes three subsystems: LiteShell, LiteFS (File System), and LiteOS Kernel. These subsystems provide several desirable features for sensor network users and developers: (1) a hierarchical file system and a wireless shell interface for user interaction, (2) kernel support for dynamic loading and multithreaded execution, and (3) online debugging, dynamic memory, and file system assisted communication stacks. These features are handy for the design and implementation of web services for sensor networks.

3. DESIGN AND IMPLEMENTATION

Application layer is responsible for handling client requests and sending the appropriate requests to middleware. It is implemented using Microsoft ASP.NET and its interface provides three basic functions: a membership system for secure login, query customization, and a visualizer. The queries can be parameterized in terms of number of the sampling nodes, sampling intervals, and comparison type (e.g., comparing by node or network) of sensor nodes. The visualizer enables us to view the sensor readings by a simple chart, table, or grid. Figure 2 shows the light reading comparison of two sensor nodes using the visualizer, where we manually put a shadow on one sensor and remove a shadow from the top of the other. The communication between application layer and middleware is done by standard XML using SOAP and WSDL specification. Therefore our web service system can be easily accessed by users via standard interface.

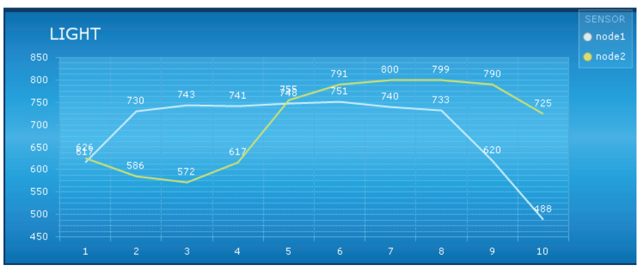

Figure 2: Light reading comparison of two sensor nodes.

For middleware, we use Apache Axis-2, a core engine for web services. For WSN layer, we use Crossbow IRIS motes. Compared with previous generation motes, IRIS motes demonstrate three times longer RF range, half lower sleep current, and double program memory(8KB), which makes them an ideal platform for web service development. A base station sits between middleware and WSN layer to facilitate the communication of these two layers.

Several challenges exist to make middleware communication with WSN layer: (1) how to handle multi-threading when multiple sensor responses are expected, (2) how to interact with LiteOS programmatically. For multi-threading, we use LiteOS's multithreaded kernel to run multiple applications concurrently. For programmatic purpose, we take advantage of LiteShell, a subsystem of LiteOS providing Unix-like command-line interface to sensor nodes. As shown in Figure 3, we develop an Event Handler in the middleware to handle the requests of the web services and to communicate directly with the Command Processor of LiteShell, which interprets user commands into internal forms and communicates with the sensor network.

Improving the user perceived query response time and the energy efficiency of sensors is considered in our web service system. We implement a data caching sublayer and a load-balancing mechanism in middleware layer.

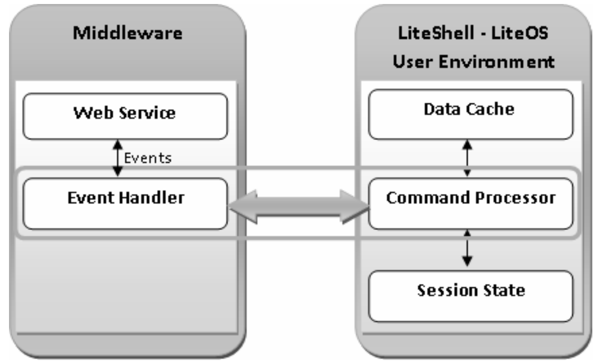

Figure 3: Interaction between middleware and LiteShell.

4. DEMONSTRATION AND FUTURE WORK

We demonstrate the following:

1. Middleware accepts parameterized requests through an front-end application.
2. Front-end application visualizes sensor readings including light, temperature, magnet, and acceleration.
3. Two users login and request sensor readings simultaneously.

Our work is still in its early stage. The current and future work include a medium access layer in LiteOS for sensor network web service, remote deployment of LiteOS programs, and multi-hop communication.

5. ACKNOWLEDGEMENTS

The authors would like to thank Dr. Qing Cao of UIUC for the valuable discussion and suggestions.

6. REFERENCES

[1] Arch Rock Corporation, *An IPv6 network stack for wireless sensor networks*, http://www.archrock.com/downloads/Arch_Rock_6LoWPAN_Whitepaper.pdf.

[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] Nissanka B. Priyantha and Aman Kansal and Michel Goraczko and Feng Zhao, *Tiny Web Services: Design and Implementation of Interoperability*, In Proceedings of ACM/IEEE IPSN, pages 567-568, 2008.

[4] Nissanka B. Priyantha and Aman Kansal and Michel Goraczko and Feng Zhao. *Tiny Web Services: Design and Implementation of Interoperable and Evolvable Sensor Networks*, In Proceedings of ACM SenSys, pages 253-266, 2008.

[5] TinyOS: www.tinyos.net.