A Framework for Creating Healthcare Monitoring Applications Using Wireless Body Sensor Networks

†Sameer Iyengar, ‡Filippo Tempia Bonda, §Raffaele Gravina, §Antonio Guerrieri, §Giancarlo Fortino, and †Alberto Sangiovanni-Vincentelli

†University of California at Berkeley, ‡Telecom Italia, §University of Calabria
e-mail: †sameer,alberto@eecs.berkeley.edu, ‡filippo.tempia@telecomitalia.it, §raffaele.gravina,antoni.guerrieri@gmail.com;g.fortino@unical.it

ABSTRACT

To aid development of Body Sensor Network (BSN) applications, we define a framework that manages common tasks for healthcare monitoring applications. The framework allows for dynamic configuration and control of signal processing and sensing functions on the sensor nodes. Using this framework, we create a healthcare monitoring platform upon which we implement applications for posture recognition as well as physical therapy. We show how body sensor network resources can be dynamically allocated to support heterogeneous application requirements without hardware reconfiguration.

Categories and Subject Descriptors

D.2.11 [**Software Architectures**]

General Terms

Design

Keywords

Wireless sensor network, healthcare monitoring, dynamic sensor activation and reconfiguration

1. INTRODUCTION

Wireless sensor networks (WSNs) offer immense potential in a variety of industries. Research has produced many healthcare monitoring WSN applications built upon improvised system structures. As a result, creating WSN applications requires an unnecessary amount of redesign which increases development effort and time to market.

To aid the emerging field of Body Sensor Network (BSN) research, we propose an application development framework that manages hardware configuration and provides common functions used by applications that collect and store data as well as those that perform real-time processing. Many BSN frameworks have been proposed with similar general features [1], [2]. Using principles described in [3] and [4], we focus on developer usability, providing a robust, yet intuitive, software API that makes it easy to create a wide variety of applications.

This framework manages three main tasks: (1) discovery of sensors and their capabilities, (2) specification and adjustment of sensor parameters such as data rates, and (3) execution of local computation on the nodes.

2. FRAMEWORK DESCRIPTION

We consider networks in which nodes with varying sensing capabilities are placed on the human body. These nodes interact with a gateway which is attached to a PC or handheld device such as a PDA or cell phone.

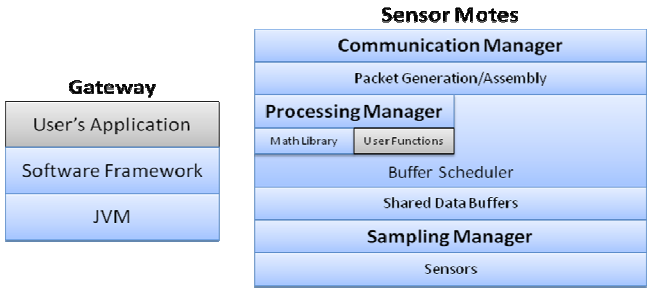

Figure 1. Framework Description

The framework provides an easy way for applications to request data from the sensors. Either raw data or function values can be requested and data rates can be changed throughout the application runtime. The components of the framework are shown in Figure 1. Developers implement applications using a software API. The API allows the application to interact with three main components on the sensor nodes: (1) the Communication Manager, (2) the Processing Manager, and (3) the Sampling Manager.

2.1 Communication Manager

The Communication Manager (CM) controls interaction between a particular mote and the gateway. When an application begins, the gateway broadcasts a service discovery packet. Each node responds with a list of available services. Services include: raw sensor data and local processing functions such as feature value computation or event detection. Based on the application requirements, the gateway can activate the services it requires. The gateway can request that data be sent in one of three ways: (1) regular intervals until requested to stop, (2) one time, or (3) when a value reaches a certain threshold.

In order to activate a particular service, the gateway sends a service activation packet specifying what kind of data is required and at which intervals. The CM passes these tasks to the Sampling Manager and/or Processing Manager. These components return values to the CM which then assembles packets and sends them to the gateway.

Common wireless communication issues are handled by the CM. Packet collision is avoided through the use of a TDMA protocol. Communication slots are assigned dynamically based on the current queue of requests. In addition, to save power and bandwidth, multiple requests and responses from a single node are aggregated into a single packet if possible. Integrity of large data

sets requiring multiple packets is preserved by re-requesting any missing packets.

2.2 Processing Manager

Because healthcare related sensors generate large amounts of data, real-time processing can reduce the amount of data communicated in the network. This module offers the ability to activate or deactivate local functions that run on the nodes and supplies functions with the current sensor signal values. The application can tune parameters such as the time intervals at which to compute the function and the length of the data segments to be used.

To aid developers in creating custom processing functions, a math library containing standard statistical functions is built into the framework. The application designer can define more complex functions in NesC using this library.

2.3 Sampling Manager

The Sampling Manager implements interfaces for sensor data acquisition and offers the flexibility to activate or deactivate acquisition channels at runtime. In particular, the Sampling Manager allows the application to control two parts of the sensing process. (1) The application can activate and deactivate entire sensors or independent channels in real time. For example, to reduce power consumption, the system can activate only the necessary axes of a three-axis accelerometer corresponding to the type of monitoring to be performed. (2) The application can control the sampling rate to allow for more detailed monitoring when an event has been detected. It is possible to save power by only sending packets as often as determined to be necessary.

The sampling manager creates one buffer for each of the active channels and populates them with the acquired data. These buffers are shared among the feature extractors. The buffer scheduler ensures that all the feature extractors read the same data at any given processing interval.

3. HEALTHCARE MONITORING PLATFORM

This section describes our implementation of a healthcare monitoring platform using the proposed framework.

3.1 Hardware

We use MoteIV Tmote Sky motes [6] with attached custom sensor boards. These sensor boards contain a 3-axis accelerometer and two dual-axis gyroscopes. To receive data, we use a base station with a CC2420 802.15.4 radio attached to an interactive gateway that is capable of running Java applications. As there are many compatible gateway devices, we focus on the PC and the cell phone. Currently, to connect to a PC we use a Tmote attached to the PC's USB port. To connect to a cellular phone we use an Intel PSI board [5] attached to the SD card slot of a Motorola E680i.

3.2 Software

On the gateway, we provide a Java library and API that developers can use to create applications. The API provides functions that allow the application to activate the services it requires from each node. Once services are activated, a thread polls for incoming data. Each time a packet is received, the thread calls a routine in the application to process the data.

3.3 Applications

To demonstrate the power and flexibility of the platform, we developed and tested two sample applications with very different requirements using this platform [7]. (1) A posture recognition application that performs real-time classification and reporting of the posture of a particular subject. (2) A physical therapy application that monitors and visually reports the range of motion of a patient.

4. CONCLUSION

We presented a framework that allows for easy development of healthcare monitoring applications. Using this framework, we successfully implemented a platform for healthcare monitoring that allows dynamic configuration and control of signal processing functions on the nodes. Our example applications in activity monitoring and rehabilitation show how body sensor network resources can be dynamically allocated based on application requirements.

It is still an open issue how to propagate new feature extractors to nodes without individually reprogramming the mote. Furthermore, to allow developers to explore various healthcare applications, we plan to add the ability to define application requirements in a configuration file and have the system automatically determine the necessary sensor activations. The platform will eventually be extended to medical sensors to evaluate its performance in a more complex heterogeneous context.

5. ACKNOWLEDGMENTS

The authors would like to thank Roozbeh Jafari, UT Dallas, and Trevor Pering, Intel Research, for their advice and contributions.

6. REFERENCES

[1] V. Shnayder, B. Chen, K. Lorincz, Thaddeus R. F. Fulford-Jones, and Matt Welsh. "Sensor Networks for Medical Care". Harvard University Technical Report TR-08-05, April 2005.

[2] W. Heinzelman, A. Murphy, H. Carvalho and M. Perillo,"Middleware to Support Sensor Network Applications," IEEE Network Magazine Special Issue. Jan. 2004.

[3] A. Bonivento, L. Carloni, A. Sangiovanni-Vincentelli. "Platform Based Design of Wireless Sensor Networks for Industrial Applications". Proceedings of Design Automation and Test in Europe (DATE), Munich, March, 2006.

[4] Y. Yu, B. Krishnamachari, and V.K. Prasanna. "Issues in Designing Middleware for Wireless Sensor Networks". IEEE Network Magazine, 2004.

[5] T. Pering, P. Zhang, R. Chaudri, Y. Anokwa, R. Want, "The PSI Board: Realizing a Phone-Centric Body Sensor Network" 4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN2007). Mar, 2007.

[6] MoteIV website: http://www.moteiv.com http://www.moteiv.com

[7] R. Gravina, A. Guerrieri, S. Iyengar, F. Tempia Bonda, R. Giannantonio, F.L. Bellifemine, T. Pering, M. Sgroi, G. Fortino and A. Sangiovanni-Vincentelli, "Demo abstract: SPINE framework for healthcare monitoring applications in Body Sensor Networks", In the Proc. of the 5th European conference on Wireless Sensor Networks 2008 (EWSN'08), Bologna, Italy, Jan 30 – Feb 1, 2008.